Doktori értekezés

Algoritmusok és döntéstámogató rendszerek kidolgozása gyártási folyamatok és terepi szolgáltatások erőforrásainak optimális ütemezésére folyamatszintézissel

DOI:10.18136/PE.2021.802

Szerző:

Frits Márton

Témavezetők:

Dr. Bertók Botond és Dr. Fábián Csaba

Pannon Egyetem

Műszaki Informatikai Kar

Informatikai Tudományok Doktori Iskola

2021

Algoritmusok és döntéstámogató rendszerek kidolgozása gyártási folyamatok és terepi szolgáltatások erőforrásainak optimális ütemezésére folyamatszintézissel

Az értekezés doktori (PhD) fokozat elnyerése érdekében készült a Pannon Egyetem Informatikai Tudományok Doktori Iskolája keretében

Írta: Frits Márton

Témavezetők: Dr. Bertók Ákos Botond, Dr. Fábián Csaba

Elfogadásra javaslom (igen / nem)

…….
Dr. Bertók Ákos Botod
(témavezető)

A jelölt a doktori szigorlaton..... %-ot ért el,
Veszprém, 2020.03.15

…….
(a Szigorlati Bizottság elnöke)

Az értekezést bírálóként elfogadásra javaslom:

Bíráló neve.....igen /nem

…….
(aláírás)

Bíráló neve..... igen /nem

…….
(aláírás)

A jelölt az értekezés nyilvános vitáján ..... %-ot ért el.

Veszprém,

…….
(a Bíráló Bizottság elnöke)

A doktori (PhD) oklevél minősítése….....

Veszprém,

…….
(az EDHT elnöke)

Tartalomjegyzék

Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani mindazoknak, akik lehetővé tették, hogy ez a dolgozat elkészüljön. Elsősorban témavezetőmnek, Dr. Bertók Botondnak tartozom köszönettel, aki éveken át irányította munkámat és értékes tanácsokkal látott el. Köszönöm a családomnak, akik támogattak és nélkülözni tudtak az elmúlt időszakban, amikor a publikációk és eme dolgozat megírásával töltöttem az időmet.

Továbbá szeretnék köszönetet mondani kollégáimnak a Rendszer- és Számítástudományi Tanszéken, akik valamilyen módon hozzájárultak munkám elvégzéséhez, valamint családomnak és barátaimnak a kitartó biztatásért.

Kivonat

Napjainkban egyre nagyobb az igény a szűkös erőforrások minél hatékonyabb felhasználására, legyen az nyersanyag, pénz vagy idő. Iparvállalatok és infrastruktúra szolgáltatók esetén összetett tervezés szükséges a gyártási és üzemeltetési folyamatok gördülékeny működéséhez, amely rendszerint vállalatirányítási rendszerek közreműködésével valósul meg. Annak összetettsége miatt a napi működés megtervezése már a kisebb vállalatok esetén sem megvalósítható informatikai döntéstámogatás nélkül. A hatékony működés biztosítása érdekében szükséges a tervezési folyamat szoftveres támogatása.

A piacon számos szoftver érhető el, amelyek általános megoldást kínálnak egy adott problémára, például gyártás ütemezésére, jármű-hozzárendelésre vagy fuvarszervezésre. Ezek a rendszerek általában nem elég rugalmasak, előre meghatározott üzleti folyamatok mentén zárt paraméterkészlettel működnek, így egy esetleges bevezetés során a szervezetnek is igazodnia kell a szoftverben meghatározott működéshez, de így is elveszíthetnek olyan egyedi szempontokat, ami a hatékonyságukra alapvető hatással van. Felmerül ezért az igény az egyedi és az iparág specifikus követelmények kezelésére, amely csak egy megfelelően rugalmas optimalizáló eljárás alkalmazásával valósulhat meg.

A kutatási témám megválasztásánál számomra fontos szempont volt a kutatási eredmények gyakorlati és üzleti hasznosíthatósága. A korábbi tapasztalatok alapján észrevehető volt, hogy az erőforrás-hozzárendelési és ütemezési feladatok terén még számos gyakorlati probléma vár megoldásra. Kutatásaim során az időkorlátos folyamathálózat-szintézis (TCPNS) optimalizáló módszertan felhasználási lehetőségeit vizsgáltam többféle ütemezési, valamint folyamat- és kapacitástervezési probléma együttes megoldására. Az ütemezési feladatok matematikai modellezés alapú megoldásához modell sablonokat dolgoztam ki különféle tárolási stratégiák, valamint folytonosan skálázható korlátos és elosztott erőforráskapacitások kezeléséhez. Kifejlesztettem egy TCPNS alapú garantált optimumot biztossító ütemezési eljárást, és többféle relaxációját nagyméretű feladatok többszintű, praktikusan gyors megoldásához. A dolgozatban bemutatásra kerülnek a kidolgozott modellezési módszerek és eljárások, amelyek működését valós gyártásütemezési és fuvarszervezési problémák megoldásával igazolom.

Abstract

Nowadays, there is a growing need to use scarce resources as efficiently as possible, be it raw material, money, or time. In industrial companies and infrastructure providers, complex planning is required for the smooth operation of production and operational processes, which is usually implemented with the help of enterprise resource planning systems. Due to its complexity, the planning of daily operations is no longer possible in smaller companies without IT decision support. Software support for the design process is required to ensure efficient operation.

There is much software available on the market that offers a general solution to a specific problem, such as production scheduling, vehicle assignment, or vehicle routing management. These systems are usually not flexible enough to work with a closed set of parameters and predefined business processes, so during a possible implementation, the organization must also adapt to the operation defined in the software but may still lose individual aspects that have a fundamental impact on their efficiency. Therefore, there is a need to address unique and industry-specific requirements, which can only be achieved using a sufficiently flexible optimization procedure.

An important aspect for me when choosing my research topic was the practical and business usability of the research results. Based on experience, it has been noticed that there are still many practical problems to be solved in resource allocation and scheduling problems. In my research, I investigated the possibilities of using the Time-constrained Process Network Synthesis (TCPNS) optimization methodology to solve numerous scheduling, process, and capacity planning problems together. To solve scheduling problems based on mathematical modeling, I developed model templates for handling various storage policies and continuously scalable limited and distributed resource capacities. I have developed a scheduling procedure based on TCPNS that ensures optimum and multiple relaxations for multi-level, practically fast solving large-scale problems. The dissertation presents the developed modeling methods and techniques, which I verify by solving real production scheduling and transport organization problems.

Abstrakt

Heutzutage wächst der Bedarf, knappe Ressourcen möglichst effizient einzusetzen, sei es Rohstoff, Geld oder Zeit. Bei Industrieunternehmen und Infrastrukturanbietern ist für den reibungslosen Ablauf von Produktions- und Betriebsabläufen eine komplexe Planung erforderlich, die in der Regel mit Hilfe von Unternehmenssteuerungssystemen umgesetzt wird. Aufgrund seiner Komplexität ist die Planung der Alltagstätigkeiten bei kleineren Unternehmen ohne IT-Entscheidungshilfesystem nicht mehr möglich. Für einen effizienten Betrieb ist eine Softwareunterstützung des Planungsprozess erforderlich.

Auf dem Markt gibt es eine Reihe von Software, die eine allgemeine Lösung für ein bestimmtes Problem bieten, wie beispielsweise die Produktionsplanung, die Fahrzeugzuordnung oder die Transportsystemplanung. Diese Systeme sind in der Regel nicht flexibel genug, um mit einem geschlossenen Parametersatz aufgrund im voraus definierter Geschäftsprozesse zu arbeiten, sodass sich die Organisation bei einer möglichen Einführung auch an den in der Software definierten Betrieb anpassen muss, dennoch können einzigartige Aspekte mit grundlegender Wirkung auf ihre Leistungsfähigkeit verloren gehen. Daher besteht der Bedarf, auf einzigartige und branchenspezifische Anforderungen einzugehen, die nur durch ein ausreichend flexibles Optimierungsverfahren erreicht werden kann.

Ein wichtiger Aspekt bei der Wahl meines Forschungsthemas war die praktische und wirtschaftliche Verwertbarkeit der Forschungsergebnisse. Aufgrund der bisherigen Erfahrungen wurde festgestellt, dass im Bereich der Ressourcenallokation und der Planungsaufgaben noch eine Reihe praktischer Probleme zu lösen sind. Im Rahmen meiner Forschung habe ich die Möglichkeiten untersucht, um mit der Optimierungsmethodik der zeitlich begrenzten Prozessnetzwerksynthese (TCPNS) mehrere Terminierungs-, Prozess- und Kapazitätsplanungsprobleme gemeinsam zu lösen. Zur Lösung von Terminierungsaufgaben auf Basis mathematischer Modellierung habe ich Modellvorlagen (Modellschablonen)für den Umgang mit verschiedenen Speicherstrategien sowie kontinuierlich skalierbaren begrenzten bzw. verteilten Ressourcenkapazitäten entwickelt. Ich habe ein TCPNS-basiertes garantiertes Optimierungs-Scheduling-Verfahren und dessen mehrfache Entspannung für die eine mehrstufige, praktisch schnelle Lösung umfangreicher Aufgaben entwickelt. Die Dissertation stellt die entwickelten Modellierungsmethoden und -verfahren vor, deren Funktionsweise ich durch die Lösung realer Produktionsplanungs- und Transportorganisationsprobleme beweise.

1. fejezet

Bevezetés

Általános megközelítéssel egy ütemterv létrehozásának folyamatát, vagyis annak eldöntését, hogy az egyes feladatokat, eseményeket vagy műveleteket hogyan kell időben elrendezni és hogyan kell ezekhez a megfelelő erőforrásokat hozzárendelni, ütemezésnek nevezzük. A kifejezés hallatán az emberek gyakran valami nagyobb méretű ipari tevékenységre, mint például a gyártás ütemezésére gondolnak, pedig a hétköznapokban is gyakran találkozunk ütemtervvel, például busz menetrendekkel, vagy saját magunk is végzünk ütemezést, amikor a napi vagy heti teendőinket tervezzük meg. Bár az idővel való gazdálkodás egyrészről egy mindennapi élethez köthető alapképesség, mégis az ipar különböző területein felmerülő új igények, valamint a feladat komplexitásából fakadóan ez a több évtizede kutatott terület máig szolgáltat újabb kihívásokat a kutatók számára. A gyakorlatban felmerülő ütemezési és erőforrás-hozzárendelési problémák összetettsége már egy kisebb vállalat esetén is hamar eléri azt a szintet, amikor már elengedhetetlen a rövid- és hosszútávú tervezés számítógépes támogatása.

Számos vállalatirányítási rendszer érhető el a piacon, amelyek egy vállalat működésének több területéhez is biztosítanak általános ütemezési és erőforrás-hozzárendelési megoldásokat, mint például: gyártásütemezés, munkaerő hozzárendelés vagy fuvarszervezés. Viszont ezen rendszerek által használt optimalizáló modellek utólag nem szerkeszthetők, ezért a speciális feltételek kezelése és az egyedi szempontok figyelembe vétele nem minden esetben valósítható meg. Gyakran még az üzleti folyamatokat is fixen, a kódban implementálva kezelik, amelyek bár a telepítés során részben konfigurálhatóak, mégis gyakran a szervezet által elvárt működés nem képezhető le megfelelően a számítógépes rendszerben.

A legmodernebb modellezési technológiák és optimalizáló keretrendszerek már támogatják a folyamatok dinamikus kezelését és az optimalizálási modellek automatikus generálását, miközben garantálják a feladat optimális megoldását is. Ahhoz, hogy egy vállalatirányítási rendszer támogassa a folyamatok konfigurálását és ehhez illeszkedően az erőforrások automatikus ütemezését, szükséges bevezetni a szerkeszthető modelleken alapúló működést, valamint integrálni kell egy a modellt értelmezni képes ütemező keretrendszert is.

Munkánk során általában az ilyen erőforrás ütemezéshez kapcsolódó modellezési, modellgenerálási és döntéstámogatási kérdésekkel foglalkozunk, amelyek közül a dolgozatban két speciá-

lis változat kerül ismertetésre: (i.) az adott telephelyen megvalósuló ipari folyamat ütemezése, beleértve a folyamatok és tárolók optimalizálását is, valamint a (ii.) földrajzilag kiterjedt infrastruktúrák fejlesztési és karbantartási feladatainak ütemezése.

1.1. Ütemezési feladatok

Az ütemezési feladatokhoz kapcsolódó kutatások kiemelt figyelmet kaptak az elmúlt néhány évtizedben. A témakör jelentőségét az adja, hogy az élet minden területén megjelenik, így ha csak közvetetten is, de mindenki találkozott már ütemezési problémákkal. A napi tevékenységeink megtervezése során is be kell osztani erőforrásainkat a kitűzött feladatok elvégzése érdekében, legyen az idő, pénz vagy egyéb a végrehajtáshoz szükséges korlátos erőforrás. Egy jól bevált tervezési módszer képes jelentősen javítani a munkavégzésünk hatékonyságát. Ipari környezetben a rövid és hosszú távú tervezés elengedhetetlen a megfelelő hatékonyság és szolgáltatási szint fenntartása érdekében. Egy heurisztikus, vagyis a korábbi tapasztalatokon alapuló ütemezési eljárás hatékony lehet egy kisebb szervezet feladatainak ütemezésére, bár a probléma mérete és komplexitása hamar eléri azt a kritikus szintet, ahol már szükséges a szoftveres támogatás. A nagyobb vállalatok esetén viszont a paraméterek és feltételek nagy száma már megköveteli egy számítógépes rendszer használatát.

1.1.1. Irodalmi áttekintés

Az ütemezéshez kapcsolódóan számos tudományos publikáció születik minden évben, amelyeket nehéz kimerítően rendszerezni, de a dolgozat fő témáját adó gyártórendszerek ütemezésén felül kiemelnék két területet, amelyek manapság kiemelkedő jelentőséggel bírnak.

A felhőalapú szolgáltatások témaköre az elmúlt évtizedben egyre fontosabb terület, amelyre jellemző az igény szerinti erőforrás hozzárendelés [1], a használat mértékétől függő fizetés, internet-alapú hozzáférés biztosítása a megosztott számítási erőforrásokhoz (hardver és szoftver) [2], amelyek egy önkiszolgáló, dinamikusan méretezhető formában állnak rendelkezésre. Egyik leginkább aktuális téma a feladatütemezés [3], amely kritikus felhőszolgáltatás teljesítménye szempontjából [4]. A nem megfelelő ütemezés miatt felmerülhetnek a kihasználatlan (underloaded) [5] és a túlhasznált (overloaded) [6] erőforrások dilemmái, ami pedig a felhőalapú erőforrások pazarlásához vagy a szolgáltatás teljesítményének romlásához vezet.

Másik kiemelkedő terület a villamosenergia-termelés és -elosztás, amely jelentős fejlődésen ment keresztül az elmúlt évtizedekben, a hagyományos központosított termelésből az elosztott, kisméretű, termelő-fogyasztó modell felé haladva, amely az elosztóhálózathoz kapcsolódik [7]. Ezt a fejlődést egy megbízható információs és kommunikációs infrastruktúra kifejlesztése tette lehetővé, de bizonyos kihívásokat is előidézett, amelyeknek az intelligens hálózat megjelenésével sikerült megfelelni [8]. Az intelligens hálózat által meghatározott technológiai keret megbízhatóbb, hatékonyabb és gazdaságosabb működést tesz lehetővé, amely képes megújuló energiaforrások [9] és energiatároló rendszerek [10] fokozott kihasználására. Egyrészt a rendszerüzemeltetőknek most rengeteg bejövő információjuk és rendelkezésre álló vezérlési lehetőségeik vannak a kritikus há-

lózati állapotváltozók vezérlése érdekében [11]. Másrészt foglalkozniuk kell a hálózatüzemeltetés modern kihívásaival [12], amelyek az elosztott termelésből és tárolásból, valamint a megújuló energiaforrások sztochasztikusságából adódnak.

Az ütemezési problémák széles skálája eredeztethető a szakaszos folyamatok optimalizálása problémakörből. Az irodalomban számos tanulmány jelenet meg, amely részletes áttekintést ad, hogy milyen paraméterek mentén lehetséges az ütemezési feladatok osztályozása [13, 14], amelyek közül a három legfontosabbat tekintjük most át. A szakaszos folyamatok működését általában egy recept határozza meg, amely definiálja a termék előállításához szükséges feladatok sorrendjét.

A feladatok közötti precedenciák általános esetben egy gráf segítségével reprezentálhatók, de sok esettanulmányban és irodalmi példában a feladatok egyszerűen csak sorban követik egymást. Az ilyen szekvenciális folyamatok esetén is kétféle receptet különböztetünk meg: több termékes (multiproduct) folyamatok esetén minden termék gyártása ugyanazon szekvenciális lépések alapján történik, míg a több célú (multipurpose) folyamatok esetén mindegyik terméket ugyanazon gyártási lépések különböző sorozata állítja elő.

Az ideiglenes köztes tárolókra vonatkozó stratégia a szakaszos folyamatok másik fontos paramétere, amely jelentősen befolyásolhatja mind a probléma összetettségét, mind az optimális megoldást. A szakirodalomban számos jól ismert tárolási stratégia található meg, amelyek közül a végtelen köztes tároló (Unlimited Storage Policy, UIS), a köztes tároló nélküli (Non-Intermediate Storage, NIS) és várakozás nélküli (Zero Wait, ZW) stratégiák a legelterjedtebbek. Allahverdi egy részletes áttekintést publikált a ZW stratégiát alkalmazó shop ütemezési problémákról [15].

A feladatok közötti átváltás ütemezési szempontból egy kardinális kérdés, amely kezelésére az általános ütemezési algoritmusok kiterjesztésére van szükség. A váltási műveletet befolyásolhatják az alkalmazott technológiából fakadó korlátok is, mint például Branaud és társai által vizsgált minőség-alapú váltás (quality-based changeovers, QBC) [16], ahol az ütemezés során figyelembe kell venni, hogy egy berendezés tisztítása a feladatok között elkerülhető, ha a második termékből adott mennyiségnél többet gyártunk.

Végül meg kell említeni az optimalizálás célját. A szakaszos működésű folyamatok ütemezésénél a két leggyakoribb cél a teljes feldolgozási idő minimalizálása, az úgynevezett makespan [17]; és a profit vagy a teljesítőképesség (throughput) adott időhorizonton való maximalizálása. A gyakorlati problémák során viszont számos más szempont vagy korlátozás befolyásolhatja az ütemezés célját, mint például a szennyvíztermelés minimalizálása [18] vagy a hővisszanyerés maximalizálása [19].

A legtöbb ütemezési módszer a problémát vegyes-egész lineáris programozási (MILP) vagy vegyes-egész nemlineáris programozási (MINLP) modellként fogalmazza meg [20], azonban a probléma kombinatorikus jellege és számítási bonyolultsága miatt a nagyméretű ipari problémák gyors megoldására genetikus algoritmusokat is alkalmaznak [21]. A probléma kombinatorikus jellege ábrázolható gráfon is, amelyre példa az S-gráf [22], az időzített automaták [23] és a dolgozatban is használt időkorlátos folyamathálózat szintézis [24, 25, 26]. A matematikai programozási technikák kritikus pontja a bináris változók használata, amelyek meghatározzák a modell hatékonyságát, méretét és alkalmazhatóságát. A publikált matematikai programozási megközelítések két fő osztálya különböztethető meg: idő és precedencia alapú modellek.

Idő- vagy időintervallum-alapú modelleknél az időhorizont előre meghatározott számú időpont vagy időrés alapján kerül diszkretizálásra. A tipikus bináris változó a $y(i, j, n)$, amely megadja, hogy az i feladat a j berendezésben kerül-e végrehajtásra az n időrésben, így a bináris változók száma nagymértékben függ az időrések számától. A korábbi modellek a diszkretizálás során az időhorizontot több kisebb azonos méretű időintervallumra osztották [27]. Ezeket a modelleket diszkrét idejű modelleknek nevezzük, amelyek jellemzője, hogy nagyszámú intervallumra van szükség a megfelelő megoldás eléréséhez. Később bináris változók számának csökkentése érdekében folytonos idejű megközelítéseket fejlesztettek ki mind a szekvenciális [28], mind a hálózati alapú [29] ütemezési problémákra, ahol az időrések hossza már eltérő is lehet.

A precedencia alapú MILP modellek kategorizálhatók úgy, mint általános precedencia [30]; közvetlen precedencia [31] és berendezés-specifikus közvetlen precedencia [32] modellek a bináris változók meghatározása alapján. Ezek a bináris változók azt adják meg, hogy a i. batch-et közvetlenül vagy közvetve megelőzi-e a j. batch egy adott vagy az összes berendezés esetén. A precedencia alapú megközelítések nem diszkretizálják az időhorizontot, vagyis nem szükséges előre meghatározni az időpontok számát. Viszont ezeknek a modelleknek a mérete érzékeny a batch-ek számára és az erőforrás korlátokra, valamint az implementálásuk is nehezebb.

A gráf reprezentáció jobban illeszkedik egy ütemezési probléma kombinatorikus jellegéhez, mint a matematikai programozás egyenletei, ezért a gráfalapú módszerek hatékonyan kihasználhatják a probléma sajátosságait [33]. Az S-gráf keretrendszert eredetileg NIS tárolási stratégiát alkalmazó gyártási folyamatok makespan minimalizálására fejlesztették ki [22]. Az S-gráf módszertan matematikai modellje egy olyan irányított gráf, amelyben a csomópontok a folyamat feladatait és termékeit reprezentálják. Az előbbit az úgynevezett receptgráf tartalmazza, amely egy branch & bound alapú optimalizálási algoritmus bemenete. Az ütemező élek az optimalizálás során meghozott ütemezési döntések alapján kerülnek hozzáadásra a gráfhoz. Az eljárás eredménye egy ütemezési gráf, amely egyértelműen meghatározza az optimális ütemezést. A keretrendszert Majozi és társai [34] terjesztették ki az áteresztőképesség (throughput) maximalizálási problémákra. Az S-gráf keretrendszer legnagyobb előnye, hogy szigorú matematikai alapja révén globálisan optimális megoldást nyújt a problémára, és soha nem szolgáltat megvalósíthatatlan vagy nem optimális megoldásokat. Továbbá az S-gráf algoritmusainak működéséhez nincs szükség az időrések számának definiálására, amely a legtöbb gyakorlati probléma esetén nem, vagy csak nehezen határozható meg. Hegyháti bevezette az eS-gráf modellt, amely az S-gráf általánosítása, ahol a csomópontok és a feladatok közötti egy-egy kapcsolatra vonatkozó megkötés enyhült, lehetővé téve az ütemezési problémák sokkal szélesebb körének kezelését [35].

Egy másfajta megközelítés az automaták használata, ahol például Alur és társai az időzített automatákat az ω-automaták általánosításaként írják le [36], amelyek az ábécé alapján végtelen időzített szavakat fogadnak el. Bár az automatákat gyakran használják analitikai célokra, de bebizonyították, hogy a rendelkezésre állás elemzésével optimalizálásra is alkalmazhatók [37]. Dávid és társai egy lineárisan árazott (linearly priced) időzített automata alapú modellt dolgoztak ki egy buszütemezési és hozzárendelési problémához [38].

Manapság gyakori megközelítés a metaheurisztikák használata, amely képes belátható időn belül jó megoldást szolgáltatni, mint például a genetikus programozás [39, 40], set partitioning

[41] vagy az ant colony alapú optimalizálás [42].

A folyamatok ütemezésére kifejlesztett MILP modellek hosszú múltja ellenére a diszkrét feladatok párhuzamos feldolgozása, valamint a műveletek volumenének folytonos skálázása az anyagmérleg vagy a kapcsolódó anyag áramok alapján, még továbbra is megoldásra váró feladatok. Fontos megjegyezni továbbá, hogy a fent említett gráf alapú módszerek esetén a gráfok a megvalósítható melett megvalósíthatatlan ütemezéseket is leírhatnak, viszont egyik sem képes egyetlen gráfban ábrázolni az összes lehetséges ütemezést, vagyis egy úgynevezett szuperstruktúrát [43]. Korábbi munkáink alapján felmerült a lehetősége, hogy az ütemezési feladatok időkorlátos folyamathálózat szintézisként (TCPNS) való modellezése révén ezen problémák kezelhetővé vállnak. A TCPNS alkalmazása során egy minden lehetséges ütemezést magába foglaló P-gráf stukrúrát kell felépíteni, amely megoldása szolgáltatja a megfelelő feladat és erőforrás összerendeléseket és az egyes aktivitások végrehajtásának pontos idejét, vagyis az ütemezést.

1.1.2. Időkorlátos folyamathálózat szintézis

A 90-es évek elején Friedler és Fan bevezette a folyamathálózat szintézist (Process Network Synthesis, PNS) [44], amelyről már bebizonyosodott, hogy egy hatékony megközelítés a gyártási folyamatok modellezére és optimalizálására, amelyet az úgynevezett folyamat gráffal vagy P-gráffal valósít meg [45]. A P-gráf keretrendszerben külön algoritmusok kerültek kifejlesztésre a maximális struktúra és a megoldási struktúrák generálására. A maximális struktúra egy szigorú szuperstruktúra, amely bizonyíthatóan legalább egy optimális hálózatot tartalmaz az eredeti, P-gráffal megfogalmazott gyakorlati problémára; míg a megoldási struktúrák pontosan azok a folyamathálózatok, amelyeket kombinatorikusan megvalósíthatók a megfogalmazott axiómák szerint [46]. Az elmúlt negyedszázad során a P-gráf keretrendszert különböző mérnöki optimalizálási problémákhoz igazították, ideértve az erőforrás-elosztást [47], ütemezést [24], diszkrét eseményszimulációt [48] és több periódusos optimalizálást [49].

A dolgozatban bevezetésre kerülő ütemező és fuvarszervező eljárások az időkorlátos folyamathálózat szintézisen (Time Constrained Process Network Synthesis, TCPNS) alapulnak, amelyet Kalauz és társai dolgoztak ki 2012-ben [50]. A PNS keretrendszer időparaméterekkel történő bővítésével már modell szinten kezelhető a műveletek végrehajtási ideje, a nyersanyagok rendelkezésre állása, valamint a termékek előállításának határideje. A megfogalmazott tézisek a TCPNS keretrendszeren alapuló modelleket és eljárásokat mutatnak be ütemezési és fuvarszervezési feladatok megoldására. Ezek működésének megértéséhez szükséges a PNS keretrendszer, az általa használt P-gráf modellező eszköz, valamint a TCPNS matematikai modelljének ismerete. Korábbi munkák során több összefoglaló is készült a PNS és P-gráf témakörében ([11]), ezért ez a fejezet csak azokat a modellparamétereket és feltételeket mutatja be, amelyek szükségesek az idő kezelésének bevezetéséhez, valamint a későbbi fejezetekben ismertetett eljárások megértéséhez.

Tekintsünk egy $(\mathsf{M}, \mathsf{P}, \mathsf{R}, \mathsf{O}, \alpha, \beta)$ szintézis problémát, amely meghatározza az O lehetséges műveletek halmazát, ezen műveletek potenciális be és kimeneteit tartalmazó M halmazt, ami tartalmazza az erőforrások R és a termékek P halmazát is. A [11] cikkben használt jelöléseket az 1.1.-1.3. táblázatok foglalják össze a szintézis, a mennyiség korlátos, valamint a lineáris

paraméteres mennyiség korlátos szintézis problémák leírásához.

1.1. táblázat. A PNS jelölései

$(\mathsf{M}, \mathsf{P}, \mathsf{R}, \mathsf{O}, \alpha, \beta)$	szintézis probléma
M	az aktivitások lehetséges bemeneteinek és kimeneteinek halmaza
P	a termékek (célok) halmaza
R	az erőforrások halmaza
O	a lehetséges aktivitások halmaza
o_i	az i. aktivitás
m_j	a j. bemenet vagy kimenet
$\alpha(o_i)$	az o_i aktivitás bemenete
$\beta(o_i)$	az o_i aktivitás kimenete

1.2. táblázat. A mennyiség korlátos PNS jelölései

$(\mathsf{M}, \mathsf{O}, L_p, U_p, U_c, fcm, fco, fio)$	mennyiség korlátos szintézis probléma
$\psi(\mathsf{o})$	az o halmazban szereplő összes aktivitás előfeltételeinek és következményeinek uniója
Lp	alsó korlát a teljes struktúra nettó termelésére
Up	felső korlát a teljes struktúra nettó termelésére
$Lp(m_j)$	alsó korlát az m_j anyag nettó termelésére
$Up(m_j)$	felső korlát az m_j anyag nettó termelésére
Uc	felső korlát a nettó fogyasztásra
$Uc(m_j)$	felső korlát az m_j anyag nettó fogyasztásra
fcm	az erőforrások és termékek értéke a mennyiségük alapján
fco	az aktivitások költsége
fio	az eredmény és az elvárt mennyiség közötti különbség

A fix részt tartalmazó lineáris költségfüggvényű paraméteres PNS probléma idő korlátok kezelésével való bővítése négy új paraméter bevezetését tette szükségessé. A $tf(o_i)$ a fix, a $tp(o_i)$ pedig az arányos része annak a függvénynek, amely meghatározza az aktivitás végrehajtási idejét annak volumene alapján. A fix időállandó megadja a műveletvégzés minimális idejét, amely független a volumentől. Az arányos együttható pedig megadja, hogy mennyivel növekszik a végrehajtási idő a volumen egységnyi növelése mellett. Az $Ut(m_j)$ paraméter meghatározza az egyes anyagokra vonatkozó határidőket:

$$Ut(m_j) = \begin{cases} \geq 0, & \forall m_j \in \mathsf{P} \\ \max\limits_{m_j \in \mathsf{P}} \{Ut(m_j)\}, & \text{különben} \end{cases} \tag{1.1}$$

míg a $Lt(m_j)$ megadja az anyagok legkorábbi rendelkezésre állását:

$$Lt(m_j) = \begin{cases} \geq 0, & \forall m_j \in \mathsf{R} \\ 0, \text{különben} & \end{cases}. \tag{1.2}$$

1.3. táblázat. A mennyiség korlátos PNS lineáris modelljének jelölései

$(\mathsf{M}, \mathsf{O}, \mathsf{A}, l, u, cp, cf, cm, L_p, U_p, U_c)$	lineáris paraméteres mennyiség korlátos szintézis probléma
$a(m_j, o_i)$	az aktivitások és entitások közötti kapcsolatot leíró függvény
$x(o_i)$	az o_i aktivitás volumene
$l(o_i)$	az o_i aktivitás alsó korlátja
$u(o_i)$	az o_i aktivitás felső korlátja
$cp(o_i)$	az aktivitás proporcionális költsége
$cf(o_i)$	az aktivitás fix költsége
$cm(m_j)$	az erőforrás költsége vagy a céltermék értéke
$y(o_i)$	bináris változó, amely megadja, hogy az aktivitás része-e a megoldásnak

Minden $o_i \in \mathsf{O}$ aktivitáshoz egy $y(o_i)$ bináris változó kerül bevezetésre az 1.1. - 1.3. táblázatokban bemutatott változók és paraméterek mellett, így az alábbi MILP modell megadja a lineáris mennyiség korlátos szintézisprobléma optimális megoldását $(\mathsf{M}, \mathsf{O}, \mathsf{A}, l, u, cp, cf, cm, L_p, U_p, U_c)$:

$$\forall o_i \in \mathsf{O} : y(o_i) \in \{0,1\} \tag{1.3}$$

$$\forall o_i \in \mathsf{O} : x(o_i) \in \mathcal{R}_0^+ \tag{1.4}$$

$$\forall m_j \in \psi(\mathsf{O}) \cap \mathsf{R} : -Uc(m_j) \leq \sum_{o_i \in \mathsf{o}} a(m_j, o_i) x(o_i) \leq 0 \tag{1.5}$$

$$\forall m_j \in \psi(\mathsf{O}) \cap \mathsf{P} : Lp(m_j) \leq \sum_{o_i \in \mathsf{o}} a(m_j, o_i) x(o_i) \leq Up(m_j) \tag{1.6}$$

$$\forall m_j \in \psi(\mathsf{O}) \setminus \mathsf{R} \setminus \mathsf{P} : 0 \leq \sum_{o_i \in \mathsf{o}} a(m_j, o_i) x(o_i) \leq Up(m_j) \tag{1.7}$$

$$\forall o_i \in \mathsf{O} : l(o_i) y(o_i) \leq x(o_i) \leq u(o_i) y(o_i) \tag{1.8}$$

A problémában definiált bármely tevékenység esetén $x(o_i)$ csak akkor vehet fel nem-nulla értéket, ha a kapcsolódó $y(o_i)$ bináris változó értéke egy. Az $x(o_i)$ értéke csak az (1.6.)-(1.8.) feltételekben meghatározott korlátok közötti lehet. A megoldásban szereplő műveletek költsége az (1.9) összefüggés szerint a fix $cf(o_i)$ és az $x(o_i)$ értékétől függő proporcionális $cp(o_i)$ költség alapján kerül meghatározásra.

$$\sum_{o_i \in \mathsf{O}} \left(cf(o_i) y(o_i) + x(o_i) \left(cp(o_i) - \sum_{m_j \in \psi(\{o_i\})} a(m_j, o_i) cm(m_j) \right) \right) \rightarrow min. \tag{1.9}$$

A mennyiség korlátos szintézis probléma esetén a cél annak meghatározása, hogy mely hálózat teljesíti az (1.3.) - (1.8.) feltételeket, ahol az (1.9.) célfüggvény értéke minimális.

Az időkorlátos folyamathálózat szintézis esetén az időparaméterek kezeléséhez új feltételek és változók bevezetésére volt szükség. A $t(m_j)$ változó megadja egy anyag első rendelkezésre állásának időpontját, valamint a $t(o_i)$ pedig az o_i tevékenység megkezdésének időpontját. Ezekre a változókra a korábban bevezetett korlátozó paraméterek érvényesek:

$$\forall m_j \in \mathsf{M} : Lt(m_j) \leq t(m_j) \leq Ut(m_j) \tag{1.10}$$

$$o_i = (\alpha(o_i), \beta(o_i)) \in \mathsf{o}, \forall m_j \in \alpha(o_i) : t(o_i) \geq t(m_j) \tag{1.11}$$

az (1.11) leírja, hogy a o_i tevékenység $t(o_i)$ kezdési időpontja nem lehet korábbi, mint bármely m_j előfeltételének $t(m_j)$ az előállási ideje;

$$o_i = (\alpha(o_i), \beta(o_i)) \in \mathsf{o}, \forall m_j \in \beta(o_i) : t(m_j) \geq t(o_i) + tf(o_i) + x(o_i)tp(o_i) \tag{1.12}$$

az o_i tevékenység bármely kimenetének $t(m_j)$ rendelkezésre állási ideje nem lehet korábbi, mint a $t(o_i)$ kezdési idő és a $tf(o_i)+tp(o_i)x(o_i)$ végrehajtási idő összege alapján meghatározott időpont.

Az időkorlátos szintézis probléma esetén a cél annak a hálózatnak a meghatározása, amely teljesíti az (1.3.) - (1.8.) és az (1.10.)-(1.12.) feltételeket, ahol a célfüggvény értéke minimális. A célfüggvényben a költség paramétereken felül megjelenhetnek az időparaméterek is, amelyeket a feladattól függően többféleképpen alkalmazhatunk. Egyik leggyakoribb célfüggvény, ahol az elvégzett feladatok összértékének és a céltermékek előállítási idejének különbségét maximalizáljuk; lásd 1.13. egyenlet. Az előállítási idő minimalizálásával a jobb megoldásokban kisebbek lesznek a feladatok közötti várakozások.

$$\sum_{m_j \in \mathsf{P}} v(m_j) - \sum_{m_j \in \mathsf{P}} t(m_j) \rightarrow max. \tag{1.13}$$

A termék $v(m_j)$ értékének meghatározása általában egy komplex, adott feladat típusra szabott formula alapján történik, amely magába foglalhatja a nyersanyagok és tevékenységek költségét, a feladatok prioritását és határidejét stb., ezért nem a feladat költsége, hanem a feladat (üzleti) értéke kifejezést használjuk.

Az időkorlátos hálózat szintézis MILP modelljében az $y(o_i)$ bináris változója megadja, hogy az o_i tevékenység része-e a megoldás struktúrának, ekkor az $y(o_i) = 1$, különben $y(o_i) = 0$. A megoldási folyamatban egy tevékenység struktúrába történő bevonásáról vagy kizárásáról való döntést megelőzően becslést végzünk a költség alsó korlátjára és az alternatív szcenáriók időtartalmára minden egyes lépésben. A TCPNS relaxált modelljének felírásához bevezetésre került a tb paraméter, amely megegyezik az egyes végtermékekhez meghatározott határidők közül a legkésőbbivel:

$$tb = \max_{m_j \in \mathsf{P}} \{Ut(m_j)\} \tag{1.14}$$

a tb segítségével felírható a TCPNS relaxált modellje:

$$o_i = (\alpha(o_i), \beta(o_i)) \in \mathsf{o}, \forall m_j \in \beta(o_i) :$$

$$t(m_j) \geq t(o_i) + x(o_i)tp(o_i) + y(o_i)(tb + tf(o_i)) - tb \tag{1.15}$$

A becsléshez az $y(o_i)$ döntési változó relaxálható a 0-1 intervallumon, de ekkor is kapcsolatban van a tevékenység megengedett kapacitásával (ha egy tevékenység nem valósul meg, akkor csak nulla lehet a kapacitása), amely összefüggés az 1.18. egyenlettel adható meg:

$$y(o_i) = 0 \rightarrow t(m_j) \geq t(o_i) - tb \tag{1.16}$$

$$y(o_i) = 1 \rightarrow t(m_j) \geq t(o_i) + x(o_i)tp(o_i) + tf(o_i) \tag{1.17}$$

$$x(o_i) \geq u(o_i)y(o_i) \tag{1.18}$$

Az 1.15. egyenlőtlenség megegyezik az 1.16. és az 1.17. egyenletekkel attól függően, hogy az $y(o_i) = 0$ vagy $y(o_i) = 1$. Az 1.15. egyenlet korlátai csak akkor vannak hatással, ha az $y(o_i)$ értéke közel van az egyhez a relaxált modellben.

1.1.3. Ütemezési feladat megoldása folyamatszintézis problémaként

A dolgozatban bevezetésre kerülő ütemező és járatszervező eljárások alapja az időkorlátos folyamathálózat szintézis (TCPNS), amely eredetileg üzleti folyamatok és ellátási láncok tervezésére került kifejlesztésre a PNS keretrendszer időparaméterekkel való kiterjesztésével [50]. A PNS esetén egy grafikus modell, az úgynevezett P-gráf segítségével írható le egy folyamat, amely egy irányított páros gráf, anyag és művelet csomópontokkal. Tervezés során felírásra kerül a folyamat maximális struktúrája, amely tartalmazza a feladat összes lehetséges megoldását. A PNS megoldó algoritmusai segítségével meghatározható az adott célfüggvény szerinti optimális vagy N legjobb megoldás. A megoldás struktúrák előállítása során az egyes műveletekről eldöntésre kerül, hogy részei-e az adott struktúrának, és ha igen milyen méretezéssel.

Üzleti folyamatok, illetve ellátási láncok esetén nem csak a folyamat struktúrája, hanem az abban lévő tevékenységek végrehajtásának időpontja is releváns. Ha vannak olyan entitások, amelyekhez többféleképpen el lehet jutni, akkor kérdés, hogy az alternatív ágaknak szinkronba kell-e lenniük. Így a megoldás során már nem csak az a kérdés, hogy az adott művelet benne van-e a megoldásban, hanem hogy kell-e szinkronizálni egy másik művelettel. Mivel nem csak a folyamatok teljes lefutásának az ideje érdekes, ami tartalmazhat várakozási időket, hanem az egyes tevékenységek tényleges végrehajtása, ezért szükség volt az időparaméterek modell szintű kezelésére, amely eredményeként jött létre a TCPNS. Az új paraméterek révén kezelhetővé vált a feladatok megkezdésének és végrehajtásának ideje, a nyersanyagok rendelkezésre állási ideje, valamint a termékek határideje. A TCPNS esetén a megoldásban már az egyes műveletek pontos kezdési időpontja is meghatározásra kerül. Viszont a maximális struktúra felépítése során plusz műveletek szükségesek a szinkronizáláshoz, de ez a strukturális bővítés automatizálható [50].

A TCPNS esetén felmerült a kérdés, hogy alkalmazható-e ütemezési feladatok megoldására? Néhány ismert ütemezési megközelítéssel, mint a precedencia alapú modellekkel vagy az S-gráffal

összehasonlítva megállapítható, hogy a TCPNS modell ötvözi ezek előnyeit. A folyamatok ütemezésére kifejlesztett MILP modellek hosszú múltja ellenére a feladatok párhuzamos feldolgozása, valamint a műveletek volumenének folytonos skálázása az anyagmérleg vagy a kapcsolódó anyag áramok alapján, még továbbra is megoldásra váró feladatok.

A matematikai programozás alkalmazásának legnagyobb nehézsége a minimális komplexitású matematikai modell meghatározásában rejlik, ami az eredeti problémának legalább egy optimális megoldását tartalmazza. Időpont- vagy intervallum modellek esetén az időhorizont előre meghatározott számú időponttal vagy intervallummal kerül diszkretizálásra. Viszont nincs olyan megközelítés, amely meghatározza a globálisan optimális megoldáshoz szükséges időpontok számát [51]. A precedencia alapú MILP megfogalmazások esetén nem szükséges az időpontok számának a priori pontos meghatározása, de a modell mérete nagyon érzékeny a batch-ek számára, és bizonyos korlátozásokat nehéz vele megvalósítani [52]. A TCPNS matematikai modellje is egy precedencia alapú modell, viszont a grafikus gráf reprezentáció megkönnyíti a fejlesztést, valamint az automatikus modell generálás biztosítja a nagyméretű feladatok megoldását is.

Az S-gráf keretrendszerben [22] az optimalizálás matematikai modellje egy irányított gráf. A feladatok időbeli sorrendjét két élhalmaz fejezi ki: recept-élek és ütemezés-élek. A recept-élek által definiált recept bemenetként szolgál a branch & bound alapú optimalizálási algoritmusnak. Az ütemezési döntéseket ábrázoló ütemezési élek az optimalizálás során kerülnek beépítésre a gráfba. Az S-gráf keretrendszer biztosítja, hogy a probléma megoldása globálisan optimális legyen, és hogy soha ne álljanak elő megvalósíthatatlan megoldások. A megfelelően generált S-gráf tekinthető az ütemezési probléma szuperstruktúrájának, ha az tartalmazza legalább egy optimális megoldását a feladatnak [43]. Hasonlóan a TCPNS esetén is, a folyamat szintézis alapú megközelítésből fakadóan egy olyan maximális struktúrát kell meghatározni, amely tartalmazza az összes lehetséges ütemezést.

Ezek alapján felmerül a kérdés, hogy egy ütemezési problémához felépíthető-e egy olyan P-gráf szuperstruktúra, amely tartalmazza az összes lehetséges ütemezést, továbbá a megoldása révén megadja az egyes műveletek pontos kezdési idejét, valamint a végrehajtáshoz szükséges erőforrás-feladat összerendeléseket. A dolgozat első tézise erre a kérdésre adja meg a választ.

1.2. Terepi munkavégzés ütemezése

Terepi szolgáltatásnak nevezzük azokat a szolgáltatásokat, amelyeket csak a megadott helyszínen lehet elvégezni, nem pedig a szolgáltató saját telephelyén. A szakembereknek először el kell jutniuk a megfelelő lokációra, ahol elvégezhetik a feladatot, amely lehet egyes berendezések vagy rendszerek telepítése, javítása vagy karbantartása. A legtöbb ember a terepi szolgáltatás hallatán a kábel TV vagy az elektromos hálózat szakemberei által nyújtott szolgáltatásokra asszociál, de a szektor fejlődésével az élet számos területén megjelenik. A kapcsolódó optimalizálási kérdések a tudományos publikációkban is megjelennek, mint például a mobil egészségügyi ellátás [53], a vagyonőrzés [54] vagy a műszaki szolgáltatások [55] területén.

A magas színvonalú szolgáltatás biztosítása érdekében folyamatos karbantartásra és továbbfejlesztésre van szükség. Következésképpen számos karbantartási és fejlesztési feladatot kell elvé-

gezni a megfelelő tanúsítvánnyal, szakértelemmel és helyi ismeretekkel rendelkező szakembereknek. A feladatok ütemezése és a szerelő csapatokhoz való rendelése, majd a végrehajtás koordinálása egy komplex tervezési feladat. Minden feladatot hozzá kell rendelni egy szerelő csapathoz, amely megfelel a feladat végrehajtásához szükséges követelményeknek, mint például a szakemberek kompetenciái vagy a megvalósítás tárgyi feltételei.

A 3. tézis során egy kétszintű hibrid eljárás kerül bevezetésre, amely alkalmas a terepi munkák ütemezésére, nagyméretű feladatok esetén is. A modellezése során kérdésként merült fel, hogy ütemezési vagy fuvarszervezési problémaként kezeljük a terepi munkák kiosztását, amely felvetés a következő fejezetben kerül megvizsgálásra.

1.2.1. Ütemezési és fuvarszervezési problémák

A terepi munkavégzés optimalizálása esetén a feladatok kiosztása, valamint időbeni ütemezése mellett figyelembe kell venni a feladatok elhelyezkedését és az utazási időket is. Felmerül a kérdés, hogy ütemezési vagy fuvarszervezési problémaként kell-e kezelnünk, ezért érdemes megvizsgálni mindkét megközelítést. A terepi munkák kiosztásának egyik fő jellemzője, hogy a kiosztandó feladatok száma jóval magasabb, mint a rendelkezésre álló szerelő csapatoké, amelyből következik, hogy egy csapat számára több, jellemzően különböző helyszíneken lévő feladat kerül kiosztásra. A munkaidő hasznos kihasználását jelentősen befolyásolja a csapathoz rendelt feladatok végrehajtási sorrendje, valamint ezek egymástól való távolsága. Ezért egy fuvarszervezési probléma is felmerül a feladat-hozzárendelési és ütemezési probléma mellett, így már egy úgynevezett munkaerő-ütemezési és -útvonaltervezési problémához (Workforce Scheduling and Routing Problem, WSRP) jutunk. Castillo-Salazar részletes áttekintést adott a WSRP közös jellemzőiről és megoldási módszereiről [56]. Feltételezzük, hogy egy szerelő csapat által bejárt útvonal különböző helyszínek sorozatából áll, tehát nem tartoznak ide azok a problémák, amelyek esetén a munkaerőnek egy épületen belül kell több munkaállomáson feladatot ellátnia, mint például egy gyártórendszer esetén. Az elvégzendő feladatokra jellemző továbbá, hogy előre meghatározott időbeli korlátok között kell elvégezni, így az útválasztás mellett ütemezésre is szükség van. Feltételezhető, hogy a munkáltató elvárja, hogy az alkalmazottak a munkaidő minél nagyobb részét töltsék hasznos munkával, ezért az utazási idők csökkentése egy fontos célként jelenik meg ([57]; [58]). A munkaerő ütemezéséhez hasonlóan a feladatok végrehajtásához szükséges készségek és egyéb feltételek kezelése is fontos szempont a WSRP-k esetén ([55]). Tehát a WSRP rendelkezik egy feladatütemezési és egy útvonalkeresési komponenssel, amelyek közötti kapcsolat a következő példa segítségével kerül szemléltetésre.

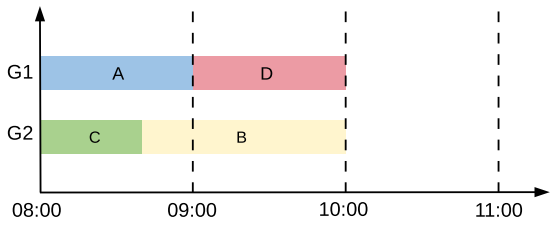
Ütemezési problémák esetén N számú feladatot kell kiosztani a rendelkezésre álló M számú végrehajtó számára. Egy adott probléma leírása tartalmazhat további, a hozzárendelésre vonatkozó szabályokat, valamint idő és mennyiségi korlátokat. Tekintsük a Példa#1 problémát, amely során négy feladatot ($T1 - T4$) kell kiosztani két szerelő csapat számára ($G1, G2$). Egy feladatot csak akkor lehet egy csapathoz rendelni, ha az rendelkezik a megfelelő képességgel, amely Példa#1 esetén piros, zöld és kék képesség lehet. A feladatokat és a kapcsolódó paramétereket, mint a szükséges képességet, az előre definiált időablakot, a munkavégzés várható idejét, valamint a

munkavégzés költségét az 1.4. táblázat foglalja össze. A példában egy feladat elvégzésének üzleti értéke egységesen 1000.

1.4. táblázat. Példa#1: A feladatok paraméterei

Feladat	Elvárt képesség	Időablak	Normaidő [perc]	Költség
A	piros	08:00-11:00	60	500
B	kék	08:00-11:00	80	200
C	zöld	08:00-11:00	40	300
D	piros	09:00-10:00	60	100

A szerelő csapatok fő jellemzője a képesség. A Példa#1 esetén $G1$ csapat az összes, $G2$ pedig csak a kék és zöld képességgel rendelkezik. Az ütemezés célja az elvégzett feladatok üzleti értékének maximalizálása a végrehajtás költségének és a feladatok befejezési idejének a minimalizálása mellett. A Példa#1 ütemezési probléma optimális megoldása látható az 1.1. ábrán, ahol $G1$ csapat az A majd a D feladatot, míg $G2$ csapat a C feladatot követően a B feladatot hajtja végre



1.1. ábra. A Példa#1 ütemezési probléma optimális megoldásának ábrázolása Gantt diagramon

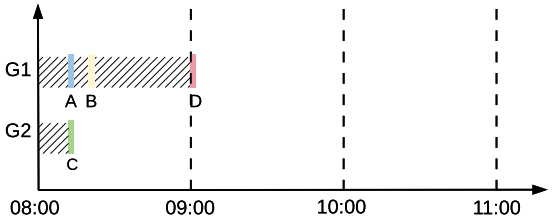
Amíg az ütemezési problémák esetén a feladat végrehajtási ideje egy meghatározó tényező, addig a feladatok közötti váltási idők kevésbé relevánsak. Ellenben egy jármű útvonaltervezési probléma (Vehicle Routing Problem, VRP) esetén a cél a különböző helyszínekre való eljutáshoz szükséges idő és pénz minimalizálása. A VRP a jól ismert utazó ügynök probléma (Travelling Salesman Problem, TSP) általánosítása. A VRP feladatot leíró Példa#2 a korábban bemutatott négy feladatot és két szerelő csapatot tartalmazza azzal a módosítással, hogy a feladatok végrehajtási ideje egységesen 1 perc, valamint a feladatok közötti utazási idők az 1.5. táblázat alapján kerülnek meghatározásra.

A Példa#2 VRP probléma optimális megoldása látható az 1.2. ábrán, ahol a $G1$ csapat rendre az A, B és D feladatokat, a $G2$ csapat pedig csak a C feladatot hajtja végre. A Gantt diagramon a csíkozott sávok a két egymást követő feladat közötti utazásokat jelölik.

A korábban ismertetett WSRP esetén a feladatokat a probléma definícióban szereplő szabályrendszernek megfelelően kell hozzárendelni a munkaerőhöz, ütemezni a végrehajtásukat, amely során kezelni kell a feladatok közötti utazásokat. A WSRP illusztrálására tekintsük Példa#3

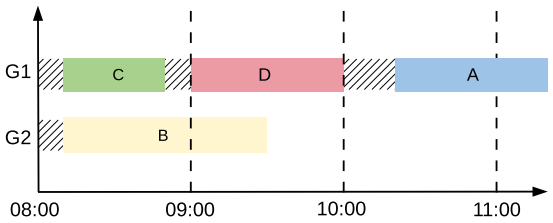
1.5. táblázat. Példa#2 probléma: Utazási időkkel kifejezett távolság mátrix [perc]

	A	B	C	D
A	0	10	10	20
B	10	0	20	10
C	10	20	0	10
D	20	10	10	0



1.2. ábra. A Példa#2 VRP probléma optimális megoldásának ábrázolása Gantt diagramon

problémát, amely az eredeti Példa#1 probléma távolság mátrixal történő bővítésével kapunk. A Példa#3 WSRP probléma optimális megoldása az 1.3. ábrán látható, ahol $G1$ csapat rendre C, D és A feladatokat, $G2$ csapat pedig a fennmaradó B feladatot hajtja végre. A Gantt diagramon a csíkozott sávok itt is a két egymást követő feladat közötti utazásokat jelölik.



1.3. ábra. A Példa#3 WSRP probléma optimális megoldásának ábrázolása Gantt diagramon

A Példa#1 és a Példa#2 problémák összehasonlítása alapján megállapítható, hogy egy ütemezési probléma tekinthető egy olyan fuvarszervezési problémának, amelyben a távolságmátrix csak nulla értéket tartalmaz. Azonban minden helyszínen jelentős feldolgozási idő szükséges. Hasonlóképpen, egy VRP tekinthető egy ütemezési problémának, amely esetén a feladat végrehajtási ideje (feldolgozása) kicsi, de a feladatok (helyszínek) között jelentős váltási (utazási) idő van. Mivel az utazási idők jelentősen csökkentik a hasznos munkaidőt, úgy, mint a WSRP (Példa#3) probléma esetén, az ütemezés csak akkor lehet hatékony, ha a feladat végrehajtási sorrend meghatározásakor a feladat végrehajtásának idejét és az utazási időket is figyelembe vesszük. Tehát egy WSRP megfogalmazható akár VRP-ként, feldolgozási időkkel a helyszíneken,

vagy ütemezési problémaként, váltási időkkel, az egymást követő feladatok helyszínétől függően. Mivel az előre meghatározott feldolgozási időkkel rendelkező helyfüggő utazási idők könnyebben értelmezhetők, mint a feladatsorrendtől függő feldolgozási idők, a VRP alapú megközelítés egy természetesebb megfogalmazást eredményez. Ezen megfontolások alapján került kidolgozásra a fejezet során ismertetett hibrid ütemező eljárás.

1.2.2. Irodalmi áttekintés

Az előző fejezetben megállapításra került, hogy a terepi munkák kiosztásának modellezése során érdemes a VRP alapú megközelítést alkalmazni, ezért ebben a fejezetben a VRP feladatok főbb típusai és kapcsolódó megoldási módszerek kerülnek áttekintésre.

A fuvarszervezési probléma (Vehicle Routing Problem, VRP) a kombinatorikus optimalizálás egy jól ismert NP-nehéz problémája, amelyet Dantzig és Ramser fogalmaztak meg először 1959-ben [59]. A VRP az elmúlt néhány évtizedben is egy kiemelt kutatási terület volt, mivel számos gyakorlati probléma vezethető rá vissza. A gyakorlati jelentőségét jól mutatja, hogy minden olyan esetben VRP feladatot kell megoldanunk, amikor a megrendelt árukat szeretnénk eljuttatni az ügyfelek számára, a lehető legtöbb haszonnal, ezért optimalizáljuk a rendelkezésre álló járművek útvonalait. A rövidebb utak, kevesebb szükséges jármű vagy a pontosabb kiszállítás gazdasági hatásai azonnal érezhetők; ezért a VRP az egyik legtöbbet vizsgált kombinatorikus optimalizálási probléma.

Az elmúlt három évtizedben a VRP problémáról és annak változatairól szóló tudományos publikációk száma jelentősen megugrott. Számos, a VRP-hez kapcsolódó taxonómia és irodalmi áttekintés jelent meg [60, 61, 62]. Laporte egy részletes áttekintést írt a fuvarszervezési problémához kapcsolódó akadémia kutatások elmúlt 50 évéről [63], Eksioglu pedig bevezetett egy rendszertant [64] az irodalomban fellelhető VRP problémák csoportosítására.

A fuvarszervézési problémának számos változata ismert és szinte mindegyik valamilyen valós alkalmazáson alapul. Az egyik ilyen a kapacitás korlátos fuvarszervezési probléma (Capacitated VRP, CVRP), ahol általában a jármű raktere vagy a megtehető út hossza a korlátos. Egy tipikus fuvarszervezési probléma megfogalmazható egész vagy vegyes-egész programozási feladatként. Az egzakt megoldást adó módszerek hátránya az egész vagy bináris változó, ami jelentősen megnehezíti a probléma megoldását, ezáltal csökkentve a megoldható probléma méretét. Ezért különböző relaxációs technikákra van szükség. Laporte és társa például a problémát egész változós programozási feladatként fogalmazták meg, majd megoldásnál használt branch & bound eljárás során a kapacitáskorlátokat először lazították, amelyek csak akkor kerülnek újra bevezetésre, ha a változók sértik a feltételeket [65].

A vágó síkok használata szintén egy gyakori megközelítés. Balinski és társai egy általánosan alkalmazható szállítási problémát írtak fel egész változókat tartalmazó programozási modellként, ahol a Gomory-féle vágó síkokat használták a számítási teljesítmény javítására [66]. Baldacci és társai egy új egész változós programozási modellt vezettek be a CVRP problémákhoz, amely two-commodity network flow megközelítésen alapul, ahol az új formula lineáris programozási relaxációjából származtatott alsó korlát meghatározása a vágósík technikát alkalmazó fázisban

új egyenlőtlenségek hozzáadásával vált hatékonyabbá [67].

A valós ipari problémák általában annyira komplexek és nagyméretűek, hogy egzakt módszerekkel csak ritkán vagy egyáltalán nem oldhatók meg. Ezért gyakran alkalmaznak heurisztikus megközelítéseket, amelyek népszerűek a fuvarszervezési problémák esetén is. Szeto és társai a CVRP problémák megoldására egy artificial bee colony modellezésén alapuló heurisztikus eljárást vezettek be, ahol az általános bee colony heurisztikájának egy továbbfejlesztett változatát javasolták a megoldás minőségének javítása érdekében [68]. Akpinar és társai egy hibrid algoritmust mutattak be, amely egy large neighbourhood search algoritmust hajt végre, az ant colony optimalizáló algoritmus megoldáskonstrukciós mechanizmusával kombinálva [69]. A valós idejű döntéstámogató rendszerekben megoldandó nagyméretű problémák megoldásához a számítási idő kiszámíthatósága szintén kulcsfontosságú szempont a megfelelő módszer kiválasztása során [70].

A CVRP problémának is számos speciális esete létezik, ahol például a távolság vagy a rakodási terület korlátozott. A távolság korlátozott fuvarszervezési probléma (DCVRP) esetén a flottára egy globális távolság vagy időbeni korlátozás vonatkozik. Az új feltételek kezelésére általában speciális megoldó eljárásokat fejlesztenek.

A branch & bound alapú megközelítést az elmúlt évtizedekben széles körben alkalmazták a CVRP és speciális változatainak megoldására. Sok esetben, hasonlóan a DCVRP-hez, ezek az algoritmusok továbbra is a legfejlettebb technikát képviselik az egzakt megoldási módszer tekintetében [71]. A heurisztikus megközelítések közül gyakran használják még a jól ismert tabu search módszer és módosított változatait [72].

A CVRP másik speciális esete a disztribúció két legfontosabb problémájának kombinációja, az úgynevezett két-dimenziós rakodással kiterjesztett fuvarszervezési probléma. Ez a probléma ötvözi az áru járművekbe történő berakodását és a járművek fuvarszervezését, amely során a cél az ügyfelek igényeinek teljesítése. A rakodási problémát Fuellerer és társai különböző heurisztikák segítségével oldották meg, majd egy ant colony modellezésén alapuló optimalizálási eljárást vezettek be a teljes probléma megoldására [73]. Gendreau és társai pedig egy tabu search algoritmust javasoltak, amelyben a rakodási fázist heurisztikával, alsó korlátok és egy módosított branch & bound típusú eljárással oldották meg [74].

A VRP probléma másik meghatározó típusa a fuvarszervezés időablakokkal (VRP with Time Windows, VRPTW), ahol az ügyfelek kiszolgálása egy előre definiált időablakban történik. A dolgozat 4. fejezetében tárgyalt terepi munkák ütemezésének problémája is ebbe a feladat osztályba tartozik. A VRPTW problémához számos egzakt és heurisztikus megoldás, valamint irodalmi áttekintés [75] érhető el. Bard és társai egy branch & bound alapú egzakt módszert dolgoztak ki, ahol a megfelelő vágások megtalálásához egy separation probléma megoldására van szükség, amelyhez egy heurisztikus eljárást dolgoztak ki [76]. Kohl és társai egy új optimalizálási módszert vezettek be VRPTW feladatok megoldására, amely a feltételek halmazának Lagrangian relaxációján alapul, amely megköveteli minden ügyfél kiszolgálását [77]. A fő problémában az optimális Lagrangian-szórzókat keresik, míg a részfeladatokban egy legrövidebb út keresési problémát oldanak meg időablakokkal.

A heurisztikus megközelítések gyakoriak a VRPTW problémák esetén is. Cordeau és társai

egy úgynevezett unified tabu search heurisztikát mutattak be [78, 79]. Gambardella és társai ant colony alapú megközelítéseket prezentálnak egy multiple-colony kezelését megvalósító keretrendszerben [80]. Bent és társai egy kétlépcsős hibrid algoritmust mutattak be VRPTW problémára megoldására [81]. Az algoritmus először a simulated annealing módszerével minimalizálja a járművek számát. Ez után minimalizálja az utazási költségeket azáltal, hogy olyan neighborhood search eljárást használ, amely képes nagyszámú ügyfél átütemezésére. Ibaraki és társai pedig egy local search algoritmust javasolnak, az úgynevezett cyclic-exchange neighborhood keresést, amely során az ügyfeleket rendelik a járművekhez, majd megkeresik az ügyfelek megrendeléseit a kiválasztott járműhöz [82]. Zang és társai egy raklap rakodási korláttal bővített VRPTW problémát oldottak meg egy olyan hibrid megoldással, amely kombinálja a tabu search és az artificial bee colony heurisztikus eljárásokat [83].

A VRP következő speciális esete a fuvarszervezés átvétellel és szállítással (VRP with Pickup and Delivery, VRPPD). Minden szállítási igényt meghatároz egy átvételi és a megfelelő kézbesítési pont. A Clarke és társai által ajánlott saving algorithm [84] alapján Gronalt és társai olyan algoritmust fejlesztettek, ami prioritási számokon alapul, amelyek értelmezhetők megtakarításként is [85]. VRPPD esetén jellemzően közvetlen szállítás feltételez a két pont között, de létezik a feladatnak a közbenső átrakodási pontokkal bővített változata (cross-docking). Nikolopoulou és társai összehasonlították a két problémát [86], majd egy local search alapú optimalizálási keretrendszert dolgoztak ki, amelyet a meglévő és új benchmark adathalmazokon teszteltek. Arra a következtetésre jutottak, hogy a környezettől és a korlátoktól függően a cross-docking stratégia általában hatékonyabb a közvetlen szállításnál. Ancele és társai pedig egy simulated annealing eljárás alapú metaheurisztikát dolgoztak ki [87]. Bár a cross-docking megoldások iránti érdeklődés növekszik, csak nagyon kevés tanulmány található a többszöri átrakodás lehetőségről. Maknoon és Laporte egy olyan matematikai megfogalmazást dolgoztak ki, ahol legalább egy átrakodási pontot érinteni kell [88]. Az általuk javasolt adaptive large neighbourhood search heurisztika hatékonyságát a CPLEX megoldó szoftverrel való összehasonlítással bizonyították.

Mindazonáltal az elmúlt években a módszertani és a számítógépes technológiák fejlődésének eredményeként egyre nagyobb tudományos figyelmet szenteltek a VRP feladatok új változatainak, beleértve a komplexebb korlátozási feltételeket és célkitűzéseket is. Ezt a tendenciát ösztönzik a valós, bonyolult karakterisztikájú VRP feladatok, amelyeket gazdag VRP feladatoknak (Rich VRP, RVRP) hívunk. Bár már évek óta tanulmányozzák, a feladatok összetettsége és sokrétűsége miatt a kapcsolódó kutatások még mindig nagyon aktívak. Caceres-Cruz és társai egy részletes áttekintést publikáltak az RVRP-ről, amelyben összefoglalták a lehetséges probléma típusokat, a gyakran előforduló követelményeket, valamint a különböző megoldási módszereket [89]. Az RVRP feladatok esetén a heurisztikus vagy hibrid megoldások jellemzőbbek a paraméterek nagy száma miatt. Sorensen és társai egy új local search algoritmust mutattak be, amelyet multiple neighbourhood keresésnek neveztek, amely képes kiküszöbölni az egyetlen szomszédot kezelő változatra jellemző rövidlátó viselkedést, ezért hatékonyabb eljárásnak bizonyult [90]. Sorensenék szerint ezt egy jól adaptálható metaheurisztikának kell tekinteni, ami különösen alkalmas a való életben felmerülő gyakorlati problémák megoldására. Rieck és társai olyan megoldást dolgoztak ki, ahol a problémát egy two-index vehicle-flow modellként fogalmazták meg, amely integrálja

a valós körülmények közötti útválasztást és a járművek rakodóhelyekhez való hozzárendelését [91] . Ezt a modellt a Solomon és társai által javasolt megközelítéssel kombinálva vezették be a deterministic nearest neighbor keresésnek nevezett eljárást [92]. Ozkan és társai kapacitásos RVRP feladatotok megoldását vizsgálták heterogén járműpark esetén, amelyre egy tabu search alapú megoldást javasoltak [93].

Az RVRP gyakorlati szempontból nagyon fontos probléma, mert az iparban a VRP-típusú feladatokat a valós korlátok speciális kombinációi jellemzik. Méretük és karakterisztikájuk miatt nehéz általános megoldást nyújtani az RVRP problémák megoldására, ezért a kutatók arra törekszenek, hogy jól alkalmazkodó hibrid megoldásokat dolgozzanak ki, amelyek megfelelnek az adott problémában megfogalmazott követelményeknek. A dolgozat 3. tézisében egy ilyen hibrid megoldás kerül bemutatásra az RVRP osztályba tartozó terepi munkavégzés ütemezési probléma megoldására, amely valós ipari követelmények alapján került megfogalmazásra. Az eljárás két modell iteratív megoldásából áll, ahol először egy LP alapú diszkrét intervallumokon dolgozó rexált modell megoldásával kerülnek felosztásra a kapacitások, majd az így generált részproblémák időkorlátos folyamathálózat szintézissel (TCPNS) kerülnek ütemezésre.

1.3. Célkitűzés

A korábbi munkáink során azt tapasztaltuk, hogy az optimalizálás területén felmerülő iparág specifikus követelmények hatékony kezelésére speciális, személyre szabott megoldások szükségesek, amelyek megfelelően rugalmas ütemező módszerek használatát követelik meg. Viszont az irodalmi elemezések alapján megállapítható, hogy még mindig számos olyan probléma várat megoldásra, amelyek kezelése gyakorlati szempontból nélkülözhetetlen.

Ezek alapján célul tűztük ki olyan modellezési technikák és modell generáló eljárások kidolgozását, amelyek alkalmazásával

- ütemezési feladatok oldahatóak meg időkorlátos folyamatszintézis problémaként,
- kezelhetőek a gyártórendszerekhez köthető tárolási stratégiák, valamint
- optimalizálhatók a terepi munkavégzés feladatai, beleértve a körjárattervezést és ütemezést.

További fontosnak tartottuk az egyes eljárások implementálását is, amely lehetővé teszi ipari döntéstámogató szoftverekhez való integrálást, ezzel elősegítve a kidolgozott módszerek tesztelhetőségét és felhasználását.

2. fejezet

Ütemezési feladatok megoldása folyamatszintézis problémaként

A PNS keretrendszerrel történő optimalizálás során a mennyiségi korlátok figyelembevételével kerül meghatározásra a garantáltan optimális megoldás. A PNS nem kezel olyan időparamétereket, mint a termékek határideje vagy a műveletek végrehajtási ideje, ezért az időkritikus problémák esetén nem, vagy csak korlátozottan alkalmazható. Korábbi kutatások során oldottak meg járatütemezési és útvonaltervezési feladatokat P-gráffal, ahol az időt korlátos nyersanyagként modellezték, így szabályozható volt az adott időszakban megvalósítható feladatok száma. Járatszervezés esetén az előre definiált menetrend alapján lehetett a megfelelő P-gráf struktúrát előállítani. [47].

Az 1.1.2 fejezetben részletesen ismertetett időkorlátos PNS viszont már modell szinten kezeli az anyagokhoz és aktivitásokhoz kapcsolódó időparamétereket, ezért felmerült a lehetősége az ütemezési problémák P-gráffal való modellezésének. A probléma potenciális megoldásainak egy P-gráfban való felírásával egy olyan szuperstruktúrát kapunk, amely garantáltan tartalmazza az optimális ütemezést is. A megoldás során már a mennyiségi és időkorlátok együttes kezelésével előáll az eredeti probléma optimális megoldása. Ebben a fejezetben bemutatásra kerül, hogy miként építhető fel a szuperstruktúra egy ütemezési probléma alapján. A bevezetett új módszer alkalmazásával további modellezési technikák kerültek kidolgozásra, mint például korlátos tároló kapacitás kezelésére vagy a feladatok több berendezésen való megosztására.

2.1. A P-gráf struktúra generálása ütemezési feladatokhoz

A TCPNS már modell szinten kezel olyan időparamétereket, mint a műveletek végrehajtási ideje, a termékek előállításának határideje vagy a nyersanyagok legkorábbi rendelkezésre állása. Viszont az ütemezési problémák megoldásához egy kötött, de jól definiálható szisztéma szerint kell felépíteni a szuperstruktúrát ahhoz, hogy az eredményül kapott megoldás struktúra a feladat egy valós ütemezését reprezentálja. A modell építésére vonatkozó irányelvek, valamint annak auto-

matizálásához szükséges lépések kerülnek bemutatásra ebben a fejezetben.

Tekintsük át, hogy általánosan mit értünk ütemezési probléma alatt. Tegyük fel, hogy van m számú berendezésünk $E_j (j = 1, ..., m)$, amelyeknek fel kell dolgoznia n darab feladatot $T_i (i = = 1, ..., n)$. Az ütemezés során az egyes feladatok egy vagy több géphez kerülnek hozzárendelésre egy vagy több jól meghatározott időintervallumban, amely során a feladat végrehajtása megtörténik. A probléma definíciója további korlátozásokat tartalmazhat arra vonatkozóan, hogy mely berendezések képesek elvégezni a feladatot és mennyi idő alatt. Bizonyos problémák esetén a feladatok között precedenciák is meg vannak határozva, amelyet általában receptgráfnak nevezett irányított gráffal írunk le.

Egy ütemezési probléma specifikációja tartalmazza az idő- és mennyiségi paramétereket, valamint azokat a korlátozásokat, amelyeknek a valóságban is megvalósítható ütemezéseknek meg kell felelniük. Egy adott problémának több lehetséges megoldása létezik, amelyek közül meg kell határozni, hogy melyik az optimális. A szuperstruktúra alapú modellek esetén az egzakt megoldó algoritmusok explicit vagy implicit módon megvizsgálják az összes lehetséges megoldást, ezzel biztosítva, hogy az eredmény optimális. A TCPNS keretrendszer bemenete, az eredeti PNS-hez hasonlóan, egy P-gráf struktúra, ami egyben a feladat matematikai modellje is, amely megoldásával előáll a legjobb vagy akár az N legjobb megoldása a feladatnak. A következő fejezetekben ismertetésre kerülnek a modell építés lépései, valamint generáló algoritmus formális leírása is.

2.1.1. P-gráf struktúra generálásának lépései

A modell generáló eljárás bemenete egy ütemezési probléma, amely megadja az elvégzendő feladatokat, a rendelkezésre álló berendezéseket, definiálja ezek lehetséges összerendeléseit, a feladatok közötti precedenciákat, valamint a kapcsolódó mennyiségi és időbeni korlátokat. Az eljárás kimenete pedig egy P-gráf, amely strukturálisan tartalmazza az összes lehetséges feladat-berendezés összerendelést minden lehetséges sorrendben, vagyis az összes potenciális ütemezést. Ezt nevezzük a probléma maximális struktúrájának. Itt fontos megjegyezni, hogy a TCPNS esetén használt P-gráf elemei megegyeznek a PNS keretrendszerben használttal, vagyis megkülönböztetünk nyersanyagokat (erőforrásokat), termékeket (eredményeket), köztes anyagokat (állapotokat) és a műveleteket leíró csomópontokat. Ezen csomópontok összekötése a páros gráf definíciójának és a PNS axiómáinak megfelelően történik.

Az ütemezési feladatok megoldása során használt P-gráf modell generálásának az alapötlete az, hogy az egyes berendezéseket olyan erőforrásként modellezük, amely szükséges bemenete (feltétele) az általa elvégezhető feladat végrehajtást leíró csomópontoknak. A struktúrának továbbá biztosítania kell, hogy ez az erőforrás minden lehetséges feladat végrehajtáshoz eljusson, minden lehetséges sorrendben, ezzel reprezentálva a feladatok végrehajtási sorrendjét. A szintézis feladatok megoldása során jellemzően folytonos változókat használunk, tehát a megoldásban a megadott korlátok között szabadon vehetnek fel értéket, addig egy TCPNS problémánál a berendezések esetén token szerű működést követelünk meg. Ez azt jeleneti, hogy a berendezést leíró erőforrás a megoldás során minden időpillanatban a gráfnak csak egy csomópontján fordulhat elő, és mindig pontosan egységnyi mennyiséggel. Így biztosítható, hogy egyidőben egy berendezés

csak egy feladatot hajtson végre. A gráfban ezek a tokenek egy utat járnak be, természetesen a mennyiségi és időbeni korlátok betartása mellett. Ehhez szükség van olyan műveletekre, amelyek modellezik a berendezés és feladat összerendelését, valamint a feladatok közötti átváltásokat. Egy műveletnek ismert a kezdési időpontja és a végrehajtási ideje, amely alapján meghatározható a kimenetének legkorábbi rendelkezésre állási ideje. Így a token által bejárt út során érintett műveletekhez meghatározott kezdési idők ábrázolhatók egy időtengelyen, amely megadja az egyes berendezések ütemezését.

Egy ütemezési probléma maximális struktúrájának generálása a következő három fő lépésből áll:

1. Minden berendezéshez egy külön részgráfot generálunk, amely a berendezést leíró erőforrásból kiindulva egy-egy gráfággal írja le a feladatok végrehajtását.

2. A részgráfokat olyan új élekkel és műveletekkel bővítjük, amelyek biztosítják, hogy a token több feladathoz is képes legyen eljutni az összes lehetséges sorrendben.

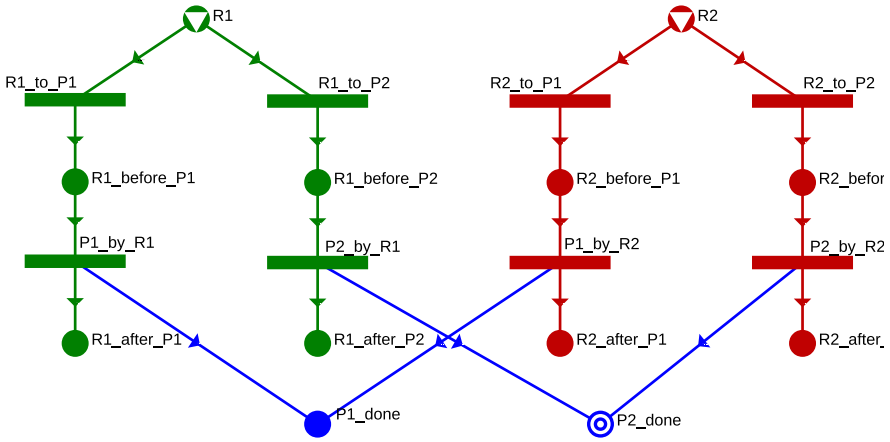
3. A gráfba további új élek és műveletek bevezetésével biztosítjuk a recepgráf által definiált precedenciákat, tehát egy feladat csak akkor kezdődhessen meg, ha minden előfeltétele teljesült.

A struktúra generálás lépéseinek szemléltetésére tekintsünk egy olyan példát, amelyben két feladatot P1-et és P2-öt szeretnék ütemezni, ebben a sorrendben, és a végrehajtásra két berendezés, vagy a PNS analógiánál maradva két erőforrás, R1 és R2 áll rendelkezésre, amelyek mindkét feladatot képesek elvégezni. Az egyszerűség kedvéért az időparaméterek később kerülnek ismertetésre, jelenleg a struktúra építése kerül fókuszba.

A generálás során elsőként az egyes berendezésekhez külön részgráfok kerülnek felépítésre, amelyek modellezik az összes lehetséges feladat végrehajtási sorrendet. Ehhez felveszünk minden berendezéshez egy nyersanyagot (R1 és R2), amely mennyiségre vonatkozó felső korlátját 1-re állítjuk. Ezután generáljuk a termékeket leíró csomópontokat, amelyek ütemezési probléma esetén az elvégzett feladatok lesznek. A recepttől függően, ha az adott feladatra épül egy másik feladat, akkor köztes anyagként (P1_done), különben pedig termékként (P2_done) vesszük fel. A struktúrának biztosítania kell, hogy az ütemezés során minden feladat csak egyszer kerüljön végrehajtásra, amelyet a termékek mennyiségére vonatkozó felső korlát 1-re állításával tehetünk meg. Ha egy feladat végrehajtása kötelező, tehát bele kell kerülnie az ütemezésbe, akkor az alsó korlátját is 1-re állítjuk, viszont ha a végrehajtás opcionális, akkor pedig 0-ra. Itt fontos megjegyezni, hogy ha a kötelezően ütemezendő feladat nem rendelhető hozzá egy berendezéshez sem, akkor a problémának nem lesz megoldása.

Az így létrehozott erőforrásokból kiindulva kell előállítani azokat a gráf ágakat, amelyek a feladat végrehajtásokat modellezik. Az ütemezési problémában közvetlenül definiálva van, vagy az összerendelés feltételei alapján összegyűjthető, hogy egy berendezés melyik feladatokat képes elvégezni. A végrehajtási folyamat első lépése a berendezés feladathoz rendelése, amelyet a R1toP1, R1toP2, R2toP1 és R2toP2 aktivitások valósítanak meg. Az összerendelés eredményeként a berendezést leíró token az erőforrásból a feladat végrehajtását megelőző állapotba

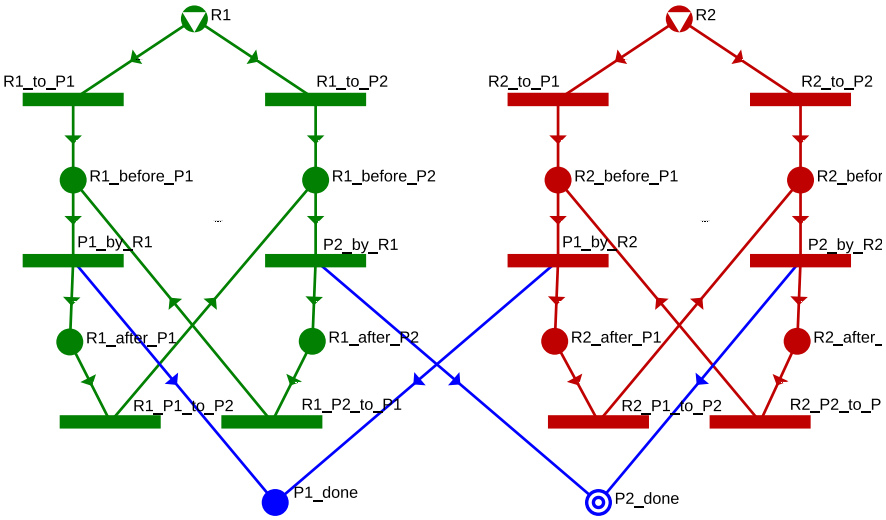
kerül (R1_before_P1, R1_before_P2, R2_before_P1 és R2_before_P2), amely bemenete lesz a P1byR1, P2byR1, P2byR1 és P2byR2 aktivitásoknak, amik már a tényleges feladat végrehajtást reprezentálják. A feladat elvégzése során előáll a termék (P1_done és P2_done), valamint a művelet következményeként a berendezés felszabdul és a többi feladat számára elérhető lesz (R1_after_P1, R1_after_P2 ,R2_after_P1 és R2_after_P2). Ennél a lépésnél látható, hogy a felépített ágak révén a berendezés el tud jutni P1 és P2 feladathoz, de annak végrehajtása után tovább lépésre nincs lehetőség a modell jelenlegi állapotában, amelyet a 2.1. ábra mutat be.



2.1. ábra. Általános P-gráf struktúra ütemezési feladatok megoldására: A generálás első lépésének eredménye

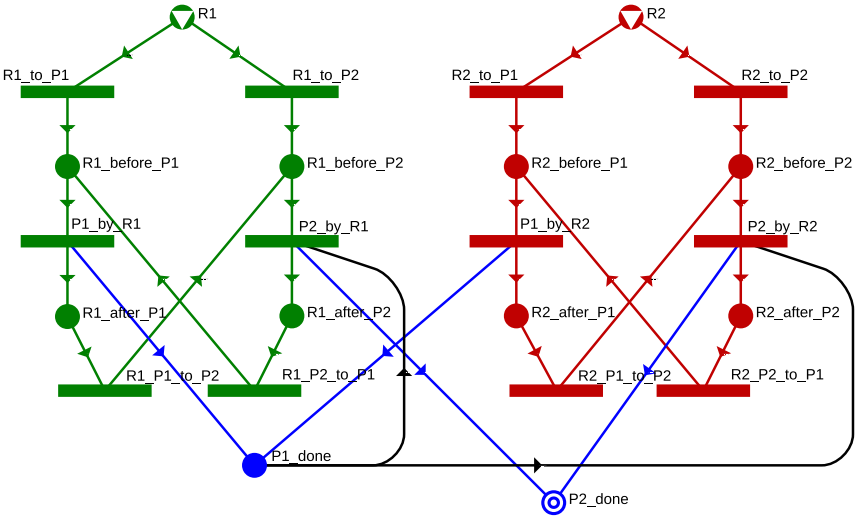
Mivel a maximális struktúra definíciója alapján a feladatokat minden lehetséges sorrendben végre kell tudni hajtani, ezért biztosítani kell, hogy ha egy feladat végrehajtása befejeződött, akkor a berendezés képes legyen egy újabb feladatot elkezdeni. Ez azt jeleneti, hogy a művelet eredményeként létrejövő állapotból (Rx_after_Py) a token képes eljutni egy másik feladat megkezdése előtti állapotba (Rx_before_Pz). Ehhez új műveletek hozzáadása szükséges, amelyek megvalósítják ezt az állapot átmenetet (R1P1toP2, R1P2toP1, R2P1toP2 és R2P2toP1). Az átmenetek révén már a P-gráf modell tartalmazza a feladatok sorrendjének összes kombinációját, amely állapotot a 2.2. ábra mutatja be.

Az ütemezési probléma által definiált recept meghatározza az egyes feladatok egymásra épülését. Ha a feladatok függetlenek, akkor az eddigi lépések során felépített P-gráf már megfelelően modellezi az ütemezési problémát. Ha viszont a feladatok egymástól függenek, akkor biztosítani kell, hogy minden feladat csak akkor kezdődjön el, ha az összes előfeltétele teljesült. Jelenlegi példában a végrehajtási sorrend P1-P2, tehát P2 végrehajtása csak akkor kezdődhet meg, ha P1 feladat befejeződött. Ezt egy új él hozzáadásával tudjuk biztosítani, amely a P1 végrehajtása után létrejövő P1_done anyagot bemenetként köti össze minden csomóponttal, amely P2 feladat végrehajtását reprezentálja, tehát P2byR1 és P2byR2 aktivitásokkal. Így biztosított a receptgráf által definiált precedencia sorrend.



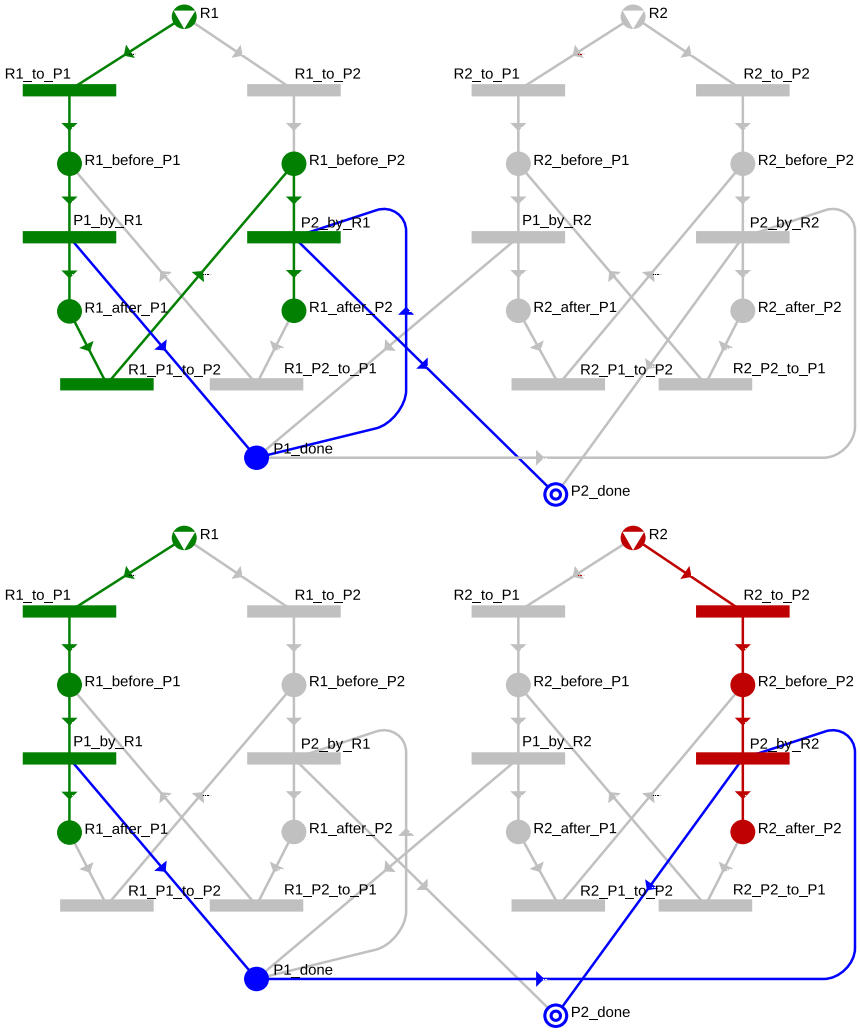
2.2. ábra. Általános P-gráf struktúra ütemezési feladatok megoldására: A generálás második lépésének eredménye

A 2.3. ábra bemutatja a példához generált maximális struktúrát.

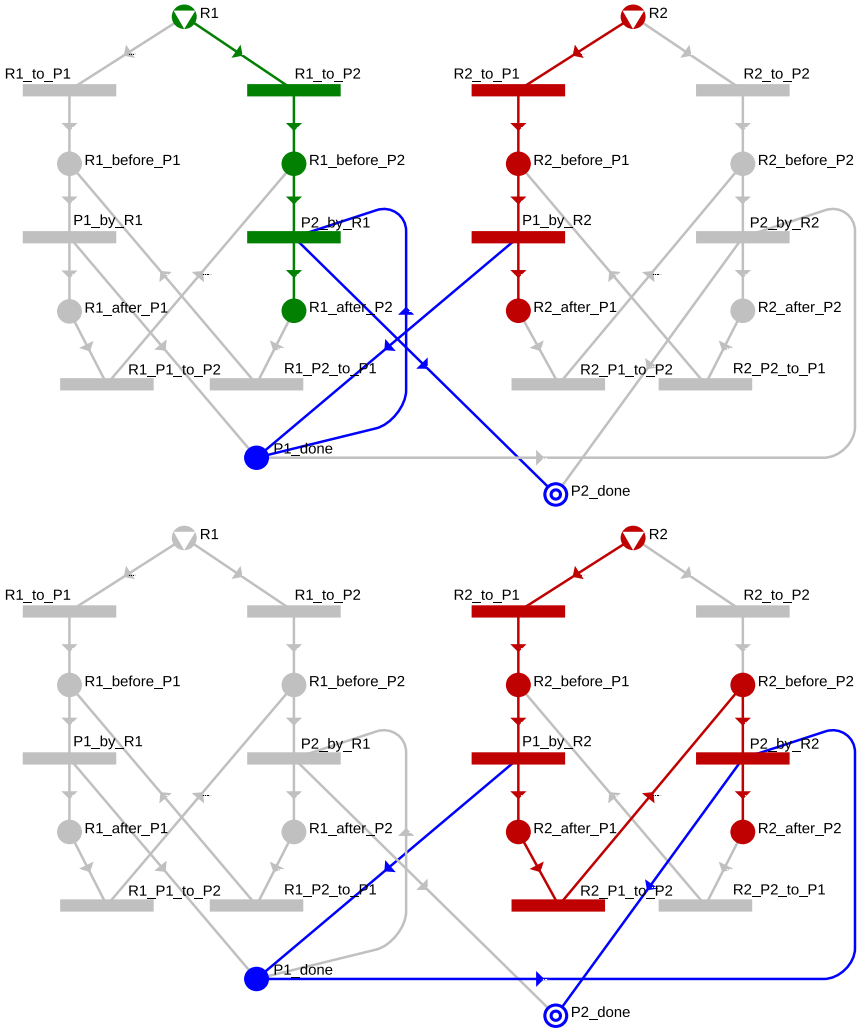


2.3. ábra. Általános P-gráf struktúra ütemezési feladatok megoldására: A maximális struktúra

A maximális struktúra négy lehetséges ütemezést tartalmaz: (1) R1 berendezés végzi mindkét feladatot, (2) R2 berendezés végzi mindkét feladatot, (3) R1 berendezés P1 és R2 berendezés P2 feladatot hajtja végre, valamint (4) R1 berendezés P2 és R2 berendezés P1 feladat hajtja végre. Ezeket az eseteket mutatják be a 2.4. és 2.5. ábrák.



2.4. ábra. Általános P-gráf struktúra ütemezési feladatok megoldására: A lehetséges ütemezések a maximális struktúrában (R1-P1 és R1-P2; R1-P1 és R2-P2)

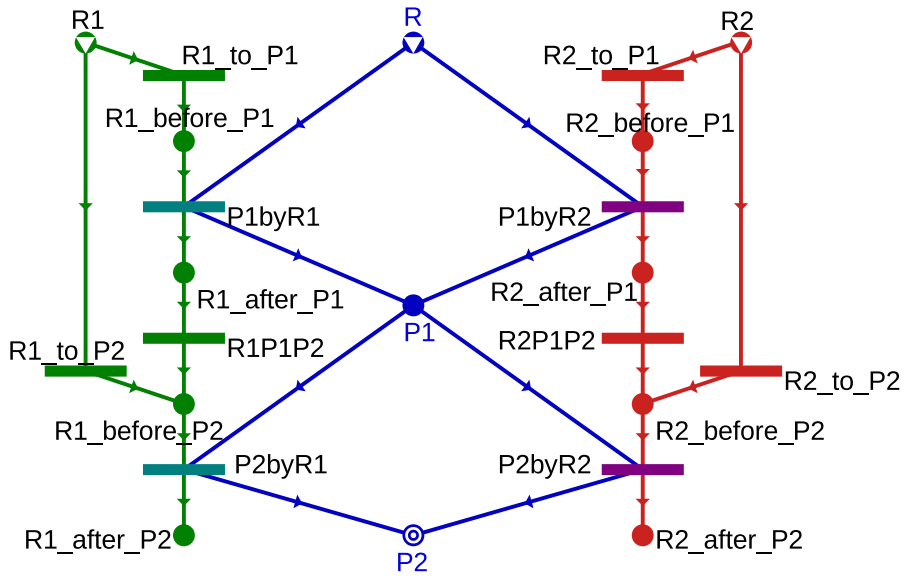


2.5. ábra. Általános P-gráf struktúra ütemezési feladatok megoldására: A lehetséges ütemezések a maximális struktúrában (R1-P2 és R2-P1; R2-P1 és R2-P2)

A bemutatott lépések alapján generált P-gráf modell tartalmazza az összes lehetséges megoldását az ütemezési feladatnak. A modell képes kezelni olyan speciális eseteket, amikor megengedett egy feladat részleges végrehajtása, vagy akár több berendezés közötti megosztása is. A mennyiségi és idő paraméterek megfelelő beállításával lehet szabályozni, hogy a lehetséges ütemezések között milyen esetek fordulhatnak elő. A következő fejezetek ezeket a paraméter beállításokat, és azok hatásait mutatják be.

2.1.2. Az anyagmennyiségek kezelése

Az ütemezési probléma definiálja a feladatok során előállítandó termékek volumenét, amely ténylegesen csak akkor állítható elő, ha elegendő bemenet áll rendelkezésre. Ezért egy döntéstámogató rendszerben működő optimalizáló modulnak ellenőrizni kell a készletszintet és az ütemezés során figyelembe kell vennie a rendelkezésre álló alapanyagok mennyiségét. A mennyiségek és kapcsolódó korlátok megfelelő kezelése érdekében az ütemezési feladathoz felépített P-gráfban közvetlenül modellezni kell a folyamat során kezelendő anyagokat. Ilyen például a 2.6. ábrán látható R nyersanyag, $P1$ közbenső termék, illetve $P2$ végtermék. Ezek mennyisége korlátozott, valamint a termelt vagy elfogyasztott mennyiség arányos a potenciális tevékenységek volumenével. A szemléltető példában 900 darab $P2$-t kell előállítani, amihez először 900 R nyersanyag felhasználásával 900 $P1$ köztes termék előállítása szükséges, amelyből már gyártható az elvárt mennyiségű $P2$ termék. Ennél a modellnél minden tevékenységnek volumene fix, vagyis ha egy aktivitás része az ütemezésnek, akkor pontosan 900 egységnyi bemeneti anyagot használ fel és 900 egységnyi kimeneti anyagot előállít elő. Ha nem része a megoldásnak, akkor pedig a bemeneti és kimeneti anyagáram is nulla.

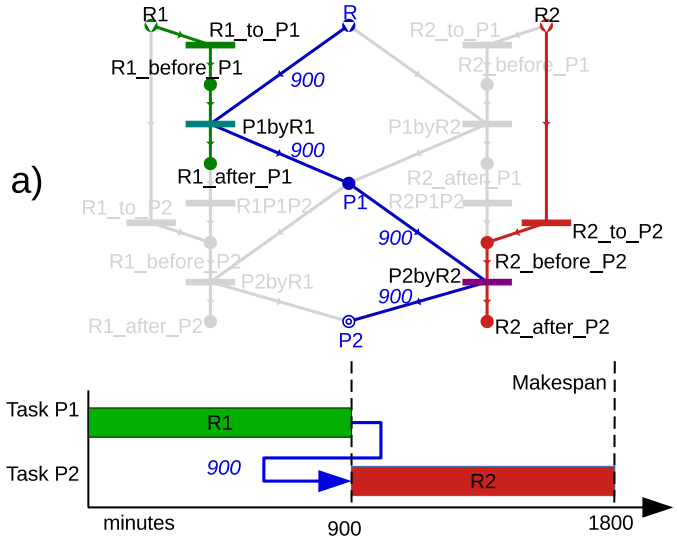


2.6. ábra. A lehetséges anyag mozgások szemléltetése a maximális struktúrában

A berendezések által végzett tevékenységek feldolgozási ideje meghatározható fix és arányos

időállandóval is. A művelet fix ideje a végrehajtáshoz szükséges minimális időt fejezi ki, függetlenül a tevékenység során végzett feladat volumenétől, míg az arányos idő meghatározza, hogy a feldolgozási idő milyen arányban változik a volumen függvényében. A példában az $R1$ berendezésnek 1 perc szükséges ahhoz, hogy egy R anyagból egy $P1$ anyagot állítson elő, és 2 percet igényel minden egyes $P1$ anyag $P2$ végtermékké alakítása.

Ezzel szemben az $R2$ berendezés 2 percet igényel minden R nyersanyag $P1$ közbenső termékké és 1 percet minden $P1$ köztes termék $P2$ végtermékké való feldolgozásához. Így 900 egységnyi anyag feldolgozási ideje 900 és 1800 perc, amelyet a tevékenységek fix idejének állítunk be. A beállatott időparaméterek alapján a problémának négyféle megoldása létezik, amelyek közül az optimális a 2.7. ábrán látható. A további lehetséges megoldásokat a mellékletben szereplő A.6.1-A.6.3. ábrák részletezik. Az ábrákon a zöld részgráf az $R1$, a piros pedig az $R2$ berendezés aktivitásainak felelnek meg, míg a kék szín az anyagáramokat jelölik. A maximális struktúra azon részei, amelyek nem szerepelnek az ütemezésben, világosszürkével került megjelenítésre.



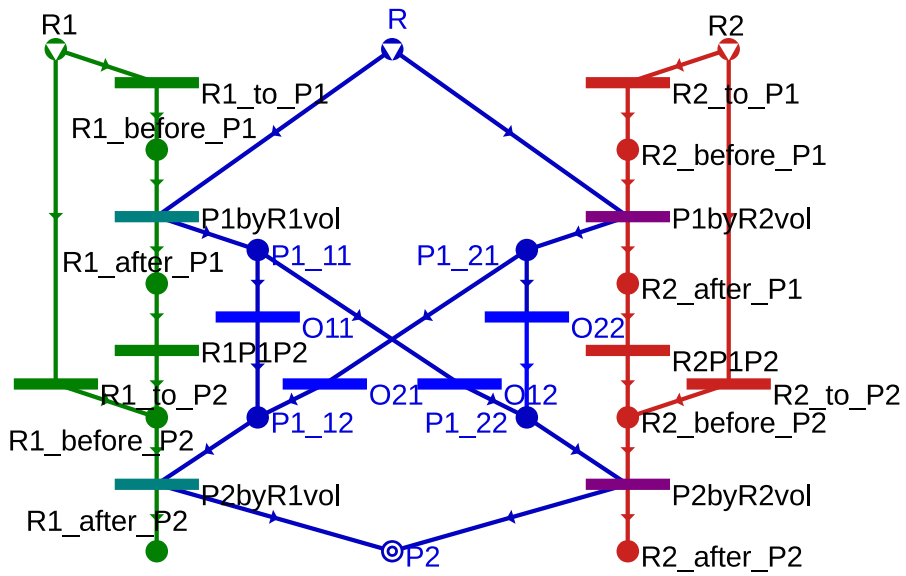
2.7. ábra. Az optimális ütemezés megjelenítése anyagáramokkal

2.1.3. Részterhelés kezelése

Ha a technológia és a gyártási folyamat lehetővé teszi, akkor az egy feladathoz tartozó nagy volumenű megrendelés több berendezéssel történő végrehajtása egy hatékonyabb ütemezést eredményezhet. Ha ezt az esetet szeretnénk modellezni, akkor fenti példában a $P1$ köztes terméket és a $P2$ végterméket előállító műveletek volumene nem kerül rögzítésre, hanem 0 és 900 között méretezhető lesz. Míg az előző modellben egy művelet során a megrendelésben meghatározott mennyiségű termék állt elő, addig most egy művelettel egy termék kerül előállításra, amely műveletet akár 900-szor lehet ismételni. Így a felhasznált és előállított anyagok mennyisége is egységnyi. A kívánt mennyiségű termék előállításának biztosítása érdekében a $P2$ végtermék mennyiségének alsó korlátját 900-ra kell állítani. A művelet végrehajtási ideje attól függ, hogy hány terméket

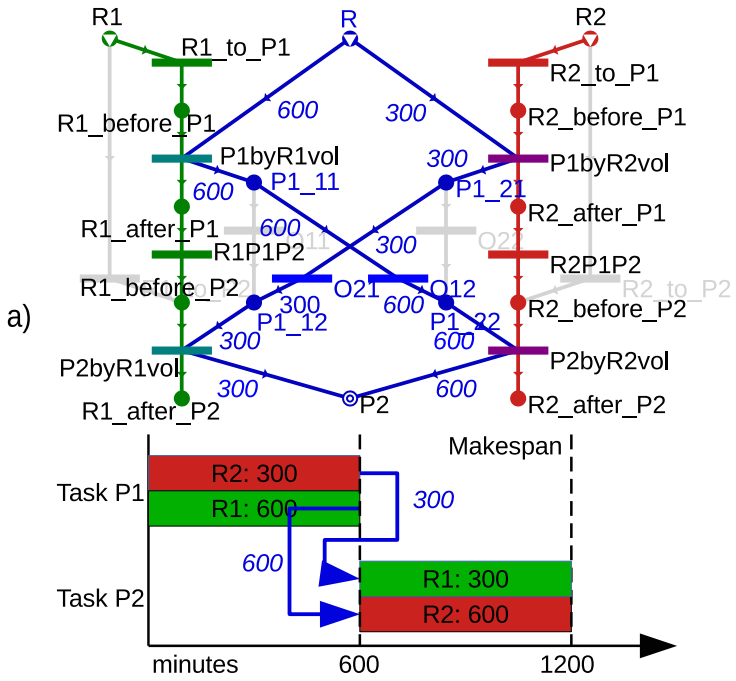
állít elő a berendezés az adott művelet során, tehát az arányos időállandót kell beállítani az egy termék gyártási idejére vonatkozóan, vagyis a jelenlegi példában 1 és 2 percet a korábban leírt paraméterek alapján.

Az egyes feladatok mennyiségének megosztása mellett a modellnek az egyes berendezések között tartalmaznia kell szinkronizálási pontokat is. Mivel a P-gráfban minden anyaghoz egyetlen időváltozó van hozzárendelve, ezért egy köztes anyag több állapotát megkülönböztetjük, például mikor az $R1$ berendezés elvégezte $P1$ feladatot ($P1\_11$) vagy mielőtt az $R2$ berendezés felhasználja $P2$-t a következő feladat végrehajtása során ($P1\_22$); lásd a 2.8. ábrát. Ezeket az állapotokat összekötő műveletek ($O11$, $O21$, $O12$, $O22$) úgy értelmezhetők, hogy vagy megtartják a köztes terméket az $R1$ ($O11$) berendezésbe, vagy a köztes anyagot az $R1$ berendezésből az R2-be töltik át ($O12$), vagy a köztes anyagot megtartják az $R2$ berendezésben ($O22$) és végül a köztes anyagot az $R2$ berendezésből $R1$-be ($O22$) töltik át.

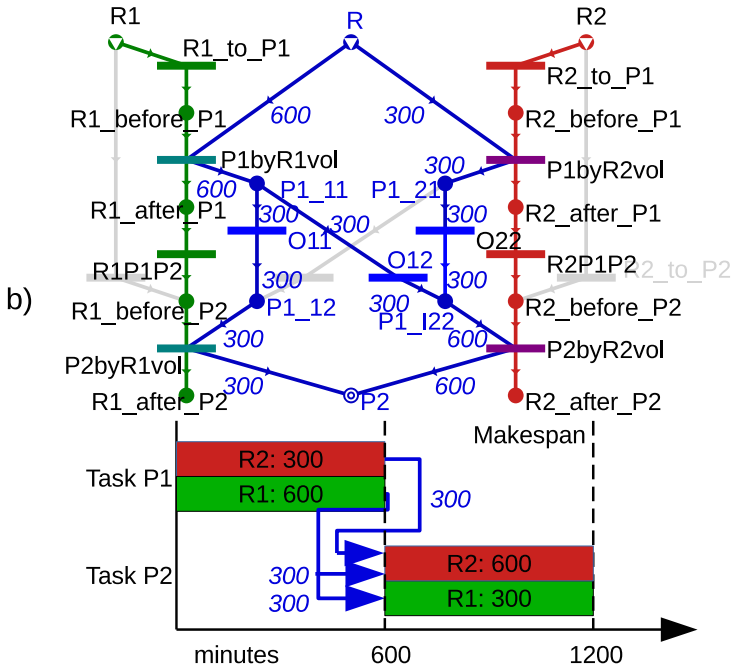


2.8. ábra. Példa: Lehetséges átváltások a feladatok aszinkron párhuzamos végrehajtásához a maximális struktúrában

A példa feladatnak 11 lehetséges megoldása van, amelyek közül a két optimális, vagyis a minimális makespannel rendelkező ütemezést a 2.9. és 2.10. ábrák mutatják be, a további lehetséges ütemezések megtalálhatóak a mellékletben az A.6.4- A.6.8. ábrákon. Az f), i), j) és k) ütemezések az A.6.5., A.6.7. és az A.6.8. ábrákon megegyeznek a 2.7 és az A.6.1. - A.6.3 ábrák a), b), c) és d) ütemezésével. Az öt legjobb a), b), c), d) és e) ütemezésben azonban a 2.9. - 2.10. és a A.6.4. - A.6.5. ábrákon legalább egy feladat végrehajtása megoszlik majd szinkronizálására kerül az $R1$ és $R2$ berendezés között, amint az a Gantt diagramokban ellenőrizhető.



2.9. ábra. Példa: Az a) optimális ütemezés a feladatok aszinkron párhuzamos végrehajtásával



2.10. ábra. Példa: A b) optimális ütemezés a feladatok aszinkron párhuzamos végrehajtásával

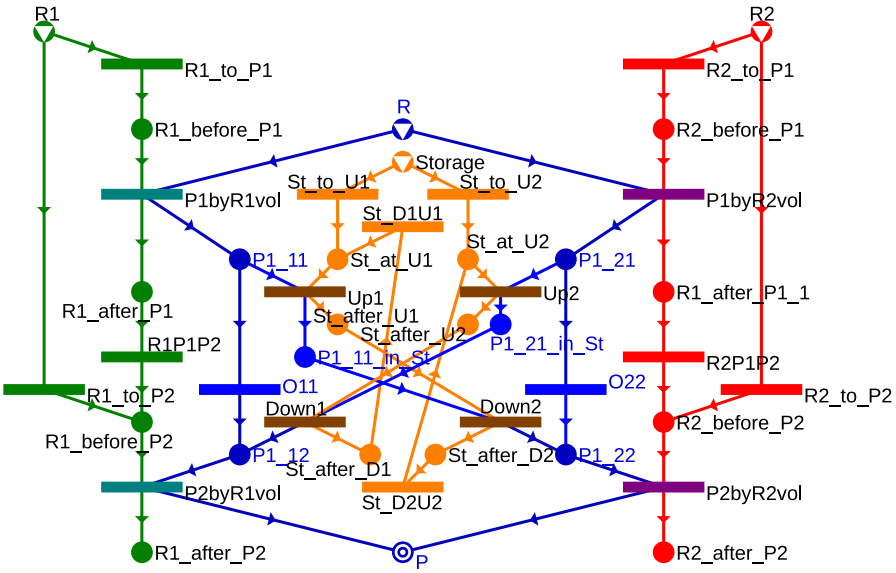
2.1.4. Korlátos tároló kapacitás modellezése

Az előző fejezetben látható megoldások alapján megállapítható, hogy a négy legígéretesebb ütemezés közül háromban, pontosabban a 2.9.,2.10. és az A.6.4. ábrákon látható a), b) és d) megoldások esetén rendre 600, 300 és 600 darab köztes terméket kell áttölteni az egyik berendezésből a másikba. Ha azonban a művelethez csak egy korlátozott kapacitású köztes tároló áll rendelkezésre, akkor a művelet nem lesz minden esetben megvalósítható. Ezért az ilyen problémákra csak úgy lehet a gyakorlatban is kivitelezhető ütemezést megadni, ha a modellben kezeljük a tárolót és annak aktuális kapacitását. A tárolókat is a gyártó berendezésekhez hasonlóan nyersanyagként modellezzük, amelyet hozzárendelhetünk egy-egy feladat végrehajtásához, majd az egyes feltöltési, letöltési vagy áttöltési műveletek során szabályozza a mozgatott anyagok mennyiségét. A 2.11. ábrán narancs színnel láthatók a tárolót és a kapcsolódó műveleteket leíró csomópontok.

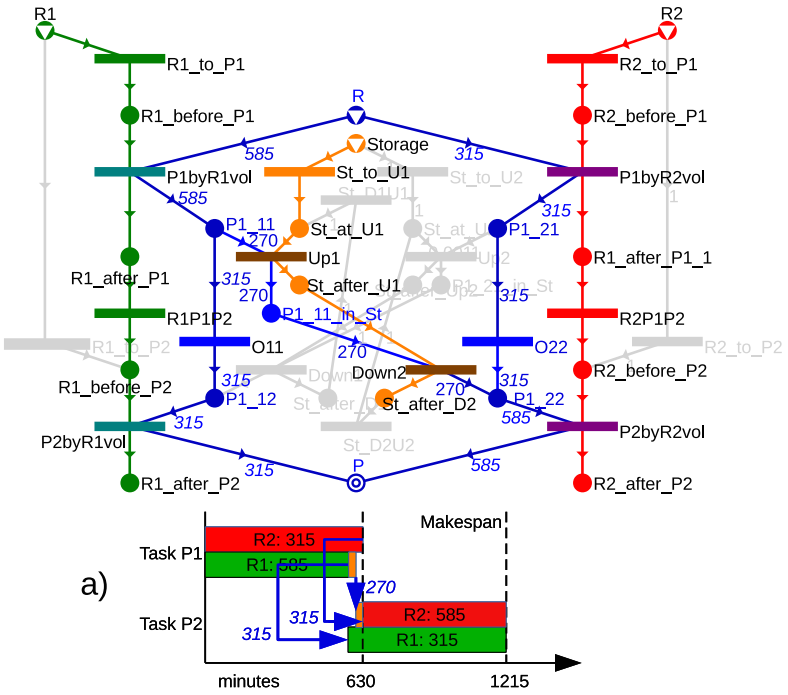
A példában leírt folyamat első lépésében $P1$ feladatot kell végrehajtani, amely során keletkező anyagot el kell tárolni a rendelkezésre álló korlátos köztes tárolóban. A köztes anyag tárolása egy feltöltési művelettel valósul meg. A feltöltés előtt hozzá kell rendelni a tárolót azokhoz a berendezésekhez, amelyek a $P1$ feladatot végre tudják hajtani. Mivel $R1$ és $R2$ berendezés is képes erre, ezért két művelettel (St_to_U1 és St_to_U2) kell bővíteni a gráfot. A hozzárendelés után már elvégezhető a feltöltés. Ha például az $R1$ berendezés hajtja végre $P1$ feladatot, akkor annak eredményét az $Up1$ művelet segítségével helyezhetjük el a tárolóba. Mivel a második lépésben $P2$ feladatot is mindkét berendezés el tudja végezni, ezért lehetőséget kell biztosítani a tárolóban lévő anyag $R1$ és $R2$ berendezésbe való betöltését, amelyet rendre a $Down1$ és $Down2$ berendezések révén lehet megvalósítani. A letöltések után, szintén lehetőséget kell biztosítani a tároló újbóli feltöltéséhez, amelyet a St_D2U2 és St_D1U1 műveletek valósítanak meg. Nézzünk egy példát, ahol a köztes tároló kapacitása 500 darab $P1$ feladat során előálló köztes anyagnak felel meg. Így a 2.9 ábrán látható a) és az A.6.4 ábrán d) ütemezése már nem lesz megvalósítható.

A tárolási kapacitás korlátozása mellett a $P1$ eredményeként előálló köztes termék egy darabjának fel- illetve letöltésére 5 másodperces töltési idő is bevezetésre került. Ezzel a módosítással már a 2.10. ábrán látható b) megoldás lesz az optimális ütemezés, ahol a töltési idő alatt további 15 darab $P1$ kerül legyártásra, amelyeket az R1 berendezésben tárolunk; lásd 2.12. ábra.

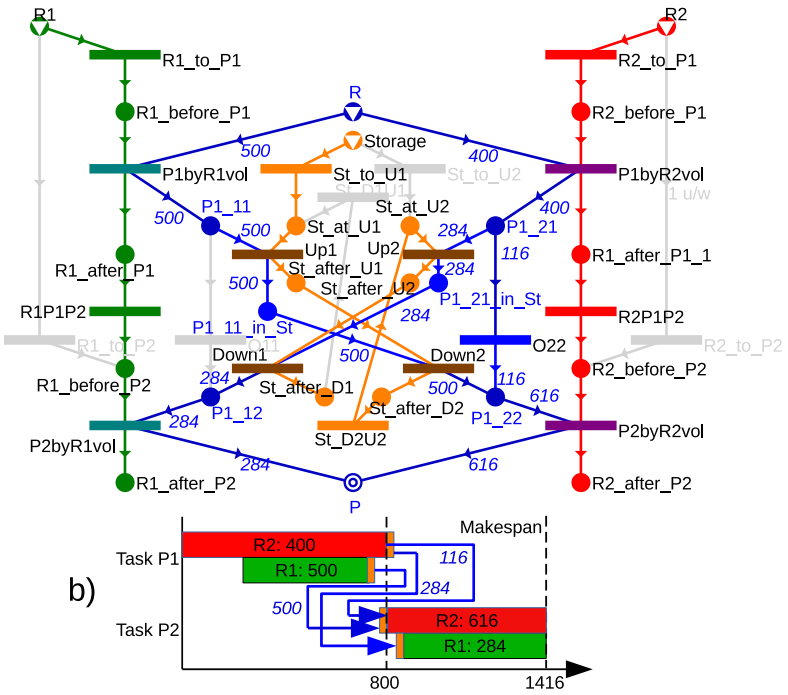
Az A.6.4. ábrán látható c) ütemezés a második legjobb ütemezéssé válik tárhely felhasználása nélkül. Az ütemezés módosításával a 2.9. ábrán látható a) ütemezés a harmadik legjobb megoldás, ahol az $R1$ berendezésből $R2$ berendezésbe való áttöltés során a mennyiség 600-ról 500-ra korlátozva lett, és ezt következi az $R2$-ből $R1$-be való áttöltés, miközben a hiányzó 100 darab $P1$ köztes terméket $R2$ berendezés gyártja és tárolja, ez látható a 2.13. ábrán.



2.11. ábra. Example 1: Véges köztes tároló modellezése a P-gráf struktúrában



2.12. ábra. Example 1: Az optimális megoldás korlátos tároló kapacitás esetén



2.13. ábra. Example 1: A harmadik legjobb megoldás korlátos tároló kapacitás esetén

2.2. P-gráf struktúra generálásának formális leírása

A P-gráf modell generálása során feltételezzük, hogy az ütemezési probléma definiálja az elvégzendő feladatokat és azok precedenciáit, a berendezéseket, a lehetséges összerendeléseket, valamint a kapcsolódó mennyiségi és idő paramétereket. Egy gyakorlati probléma megoldása további korlátozások és paraméterek bevezetését követelheti meg, mint például a 2.3. egyedi lenyomatos szalvéta gyártás esetén a feladat átváltási idők, vagy a 3. fejezetben ismertetett különböző tárolási stratégiák. Az ütemezési problémákat leíró P-gráf alapszerkezetét azonos módon kell felépíteni, amely során használt lépések jól algoritmizálhatók, így a modell generálás automatizálható. Ebben a fejezetben megadásra kerül a P-gráf modell generálásának formális leírása, amely alapján az bármilyen programozási nyelven hatékonyan megvalósítható.

A modell generáló eljárás bemenete egy ütemezési probléma, amely a következő paraméterekkel rendelkezik:

- T - az elvégzendő feladatok halmaza
- E - a rendelkezésre álló berendezések halmaza
- $t_j \in T$- az j. feladat
- $e_i \in E$ - az i berendezés
- $A(t_j) \subseteq E$ - azon berendezéseknek a halmaza, amelyek képesek t_j feladatot végrehajtani

– $P(t_j)$ - azon feladatoknak a halmaza, amelyek t_j előfeltételei

– $ts(t_j)$ - a t_j feladat legkorábbi megkezdésének ideje

– $te(t_j)$ - a t_j feladat végrehajtásának határideje

– $s(t_j)$ - a t_j feladat volumene

– $t(t_j, e_i)$ - a t_j feladat egységnyi volumenre vonatkozó végrehajtási ideje az e_i berendezés esetén

– $v(t_j)$ - a t_j feladat értéke

– $te(e_i)$ - az e_i berendezés lekorábbi rendelkezésre állásának időpontja

– $ts(e_i)$ - az e_i berendezés legkésőbbi rendelkezésre állásának időpontja

– $cf(e_i)$ - az e_i berendezés fix költsége

– $cp(e_i)$ - az e_i berendezés arányos költsége

– $ht(t_j)$ - bináris változó, amely 1 értéke esetén a t_j feladat ütemezése kötelező

Az eljárást formálisan az 1. Algoritmus írja le, amely bemenete egy ütemezési probléma, ami alapján generálja az ekvivalens időkorlátos folyamathálózat szintézis feladat P-gráf modelljét. A könnyebb átláthatóság érdekében az 1. Algoritmus egy blokk diagramon is ábrázolásra került, amely a 2.14. ábrán látható. Az eljárás során feltételezzük, hogy az ütemezett feladatokat teljesen el kell végezni az elvárt volumennek megfelelően. A részleges terhelés kezeléséhez szükséges módosítások később kerülnek ismertetésre.

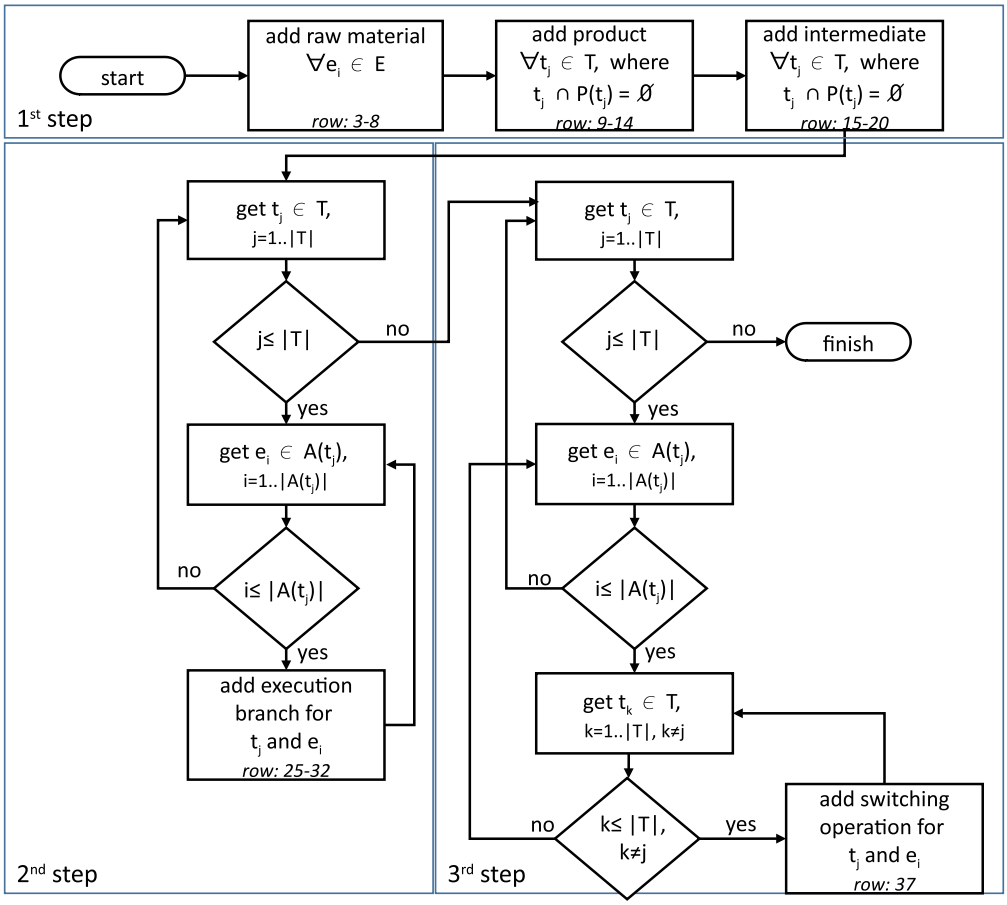
Az eljárás során először minden berendezéshez (erőforráshoz) egy nyersanyag típusú csomópont kerül felvételre (2-8.sor), amely mennyiségre vonatkozó felső korlátját 1-re, legkorábbi és legkésőbbi elérhetőségét pedig a megadott paraméterek szerint állítjuk be. A következő lépésben a feladatokhoz tartozó anyagpontok felvétele történik. Azon feladatokhoz, amelyek nem előfeltételei egy másik feladatnak, egy-egy termék csomópontot veszünk fel (9-15.sor). A felső korlátot egyre állítjuk, míg alsó korlát a $ht(t_j)$ függvénytől függően nulla vagy egy értéket vesz fel. Ha a feladat egy másiknak előfeltétele, akkor köztes termékként vesszük fel (16-22.sor).

Ez követően az egyes feladatok végrehajtási folyamata kerül modellezésre (24-36.sor). Ehhez minden t_j feladathoz az $A(t_j)$ függvény segítségével meghatározzuk azokat az e_i berendezéseket, amelyek képesek t_j végrehajtására, majd minden ilyen párhoz egy a feladat elvégzésének folyamatát leíró részgráfot hozunk létre. Ehhez fel kell venni a művelet végzése előtti $e_i\_before\_t_j$ és utáni $e_i\_after\_t_j$ állapotokat (27. sor), valamint az ezeket összekötő műveleteket. Egy művelet definiálása során annak nevét, bemeneteinek és kimeneteinek halmazát adjuk meg. Az $e_i\_to\_t_j$ az e_i berendezéseket köti össze a művelet végzése előtti állapottal, ezáltal megtörténik a feladat és a berendezés összerendelése. A $t_j\_by\_e_i$ művelet (o_k) pedig magát a feladat végrehajtást írja le, amely összeköti a művelet végzése előtti és utáni állapotot. Az egyszerűbb leírás érdekében ezt az aktivitást már ennél a lépésnél úgy hozzuk létre, hogy a precedenciáknak megfelelő köztes

Algoritmus 1: P-gráf modell generálása ütemezési feladatokhoz

input : $(T, E, A, P, t_s, t_e, s, t, v, t_{ee}, t_{es}, cf, cp, ht)$: ütemezési probléma
output: $(\mathcal{M}, \mathcal{P}, \mathcal{R}, \mathcal{O}, a, L, U, l, u, c, L_t, U_t, t_f, t_p)$: paraméteres TCPNS probléma

begin
  $\mathcal{R} := \emptyset; \mathcal{P} := \emptyset; \mathcal{M} := \emptyset;$
  foreach $e_i \in E$ **do**
    $\mathcal{R} := \mathcal{R} \cup \{r_k = e_i\};$
    $U(r_k) = 1;$
    $L_t(r_k) = te(e_i);$
    $U_t(r_k) = ts(e_i);$
  end
  foreach $t_j \in T$ ***where*** $\forall t_i \in T, t_j \cap P(t_i) = \emptyset$ **do**
    $\mathcal{P} := \mathcal{P} \cup \{p_k = (t_j)\};$
    $U(p_k) = 1;$
    $L(p_k) = ht(t_j);$
    $U_t(p_k) = te(t_j);$
  end
  foreach $t_j \in T$ ***where*** $\exists t_i \in T, t_j \cap P(t_i) \neq \emptyset$ **do**
    $\mathcal{M} := \mathcal{M} \cup \{m_k = (t_j)\};$
    $U(m_k) = 1;$
    $L(m_k) = ht(t_j);$
    $U_t(m_k) = te(t_j);$
  end
  $\mathcal{M} := \mathcal{M} \cup \mathcal{P} \cup \mathcal{R};$
  $\mathcal{O} := \emptyset;$
  foreach $t_j \in T$ **do**
    foreach $e_i \in A(t_j)$ **do**
      $\mathcal{M} := \mathcal{M} \cup \{e_i\_before\_t_j, e_i\_after\_t_j\};$
      $\mathcal{O} := \mathcal{O} \cup \{(e_i\_to\_t_j; \{e_i\}; \{e_i\_before\_t_j\})$
       $o_k = (t_j\_by\_e_i; \{e_i\_before\_t_j, P(t_j)\}; \{e_i\_after\_t_j, t_j\})\};$
      $u(o_k) = 1$
      $cf(o_k) = cf(e_i);$
      $L_t(o_k) = te(t_j);$
      $U_t(o_k) = ts(t_j);$
      $t_f(o_k) = t_f(t_j, e_i);$
    end
  end
  foreach $t_j \in T$ **do**
    foreach $e_i \in A(t_j)$ **do**
      foreach $t_k \in T$ ***where*** $k \neq j$ **do**
        $\mathcal{O} := \mathcal{O} \cup \{(e_i\_t_j\_to\_t_k; e_i\_after\_t_j; e_i\_before\_t_k)\};$
      end
    end
  end
end



2.14. ábra. Az 1. Algoritmus ábrázolása blokk diagramon

termékeket is felvesszük bemenetként a $P(t_j)$ függvény alapján. Az o_k műveletre beállítjuk a költség paramétereket, valamint a mennyiségi és időkorlátokat (29-33.sor).

Az utolsó lépésben azon műveletek kerülnek hozzáadásra, amelyek biztosítják a berendezés számára, hogy megkezdhessen egy újabb feladatot (38-44.sor), ezért minden t_j és e_i párhoz tartozó végrehajtás utáni állapotoból ($e_i\_after\_t_j$) átjárás biztosítanak minden olyan t_k feladathoz, amelyet e_i képes végrehajtani és nem előfeltétele t_j feladatnak.

Ezen lépések mentén felépített P-gráf modellezi a bemenetként átadott ütemezési feladatot abban az esetben, ha részterhelés nem engedélyezett, vagyis egy feladat végrehajtása csak akkor lehetséges, ha a kiválasztott berendezés képes a teljes feladatot az elvárt volumennel elvégezni. Egyedi feltételek kezelése esetén is érdemes a bemutatott eljárásból kiindulni és szükség esetén az egyes lépéseket módosítani vagy bővíteni. Például a részterhelés kezelése esetén a struktúra megegyezik az eddig felvázolttal, viszont az egyes paraméterek beállítása már eltérő. Részterhelés esetén a köztes és végtermékek mennyiségének felső és alsó korlátja is az elvárt volumen alapján kerül beállításra $U(m_k) = s(tj)$ és $L(m_k) = s(t_j) * ht(t_y)$. A művelet végrehajtást leíró csomópont ($t_j\_by\_e_i$) esetén pedig a felső korlát megegyezik a volumennel $u(o_k) = s(t_j)$. Ezekkel a

módosításokkal már a modell képes kezelni azokat az eseteket, amikor egy feladat csak részben kerül végrehajtásra, de akár több berendezés számára is kiosztásra kerülhet.

A következő fejezetben ismertetett gyártás ütemezési feladatnál is a P-gráf modell felépítése az 1. Algoritmusnak megfelelően történik, viszont ennél a problémánál a feladatok közötti átállások ideje is releváns tényező, ezért a váltásokat leíró műveleteknél is szükséges a végrehajtás idő beállítása.

2.3. Esettanulmány: Egyedi lenyomtatos szalvéták gyártása

Az egyetem és egy szalvétákat gyártó cég együttműködése során egy valós gyártás ütemezési probléma fogalmazódott meg, amely a TCPNS keretrendszerrel került modellezésre, majd implementálásra. A gyártó cég korábbi működésére nem volt jellemző a számítógéppel támogatott tervezés, a gyártási terv összeállítását papíron, korábbi tapasztalatok alapján végezték. A cég fejlesztése során új célként fogalmazódott meg a termelési kapacitás növelése (további berendezések vásárlásával vagy több műszak bevezetésével), amely már szükségessé tette a szoftveres támogatás alkalmazását. A fejezetben bemutatásra kerül az egyedi lenyomatos szalvéta gyártási probléma, majd annak megoldására felírt TCPNS modell, valamint egy illusztratív példán keresztül az ütemező eljárás működése is igazolásra kerül.

2.3.1. Az egyedi lenyomatos szalvéta gyártási probléma

Az együttműködés során megismerésre kerültek a szalvéták és a gyártó berendezések paraméterei, valamint a gyártás folyamata. Bár a papírtekercsből a csomagolt szalvéta előállításához több művelet szükséges, a gyakorlatban a gyártó berendezések egy komplex lépésben képesek elvégezni a méretre vágást, nyomtatást és hajtogatást, amely eredményeként előálló szalvéták csomagolását már manuálisan végzik. Így a gyártás receptje egy két-lépéses szekvenciális gráffal írható le. Viszont a probléma nehézsége nem a lépések vagy a berendezések számából fakad, hanem a megrendelések közötti átállásokhoz szükséges idő kezeléséből, amely nagyban függ a két egymást követő megrendelés paramétereitől.

Az átállási időt befolyásolja a nyersanyagok típusa, a festék színe és a nyomtatott minta is. Egy kevésbé hatékony ütemezés esetén az átállásokhoz szükséges idők nagyságrendileg összevethetőek a gyártási idővel, ami drasztikusan csökkenti a hatékonyságot. A 2.1. táblázatban láthatóak az átállási idők az eltérő paraméterek függvényében. Az első esetben, ha a papír típusa, a minta és a festék is eltérő a két egymást követő megrendelés esetén, akkor átállási idő 45 perc. A többi esetben csak egy vagy két paraméter eltérés van, így ennek megfelelően az átálláshoz szükség idő is kevesebb.

2.1. táblázat. Átállási idők a két egymást követő megrendelés eltérő paraméterei alapján

Papír típusa	Nyomtatott minta	Festék színe	Váltási idő [perc]
x	x	x	45
	x	x	20
	x		10
x	x		30
x			25

Az átállások során nagy mennyiségű selejt is keletkezhet, amíg az operátor átállítja a gépet. Amikor a papír típusa eltérő, akkor a régit el kell távolítani, az újat pedig be kell fűzni, amely jelentős időveszteséggel és selejttel jár. Amikor a nyomtatott minták különbözők, akkor a klisét tartalmazó nyomtató hengert kell cserélni, amely egy gyorsabb művelet. Viszont, ha egy új klisét kell felragasztani a hengerre, akkor a váltás többlet időt igényel. A festék cseréje esetén a hengereket el kell mosni, kivétel ha a következő minta fekete, mert akkor a mosás elhagyható. Megállapítható, hogy a megfelelő gyártási sorrend esetén az átállásokhoz szükséges idő jelentősen csökkenthető, amely hatékonyabb gyártást eredményez.

A szalvéták paramétereit és ezek lehetséges értékeit mutatja be a 2.2. táblázat. A szalvéta alapanyagát a papír színe és rétegeinek száma határozza meg. A jelenlegi gyártás ütemezési probléma esetén megkülönböztetünk 1,2 és 3 rétegű fehér, valamint 2 rétegű bézs szalvétákat. A gyártás során nyersanyagként használt papírok 120kg-os tekercsekben állnak rendelkezésre. A gyártás során a megrendelésnek megfelelő papírtekercseket választják ki, majd vágják és nyomtatják. A technológiai korlátok miatt maximum két színnel tudnak nyomtatni egy szalvétára, de gyártanak üres szalvétákat is.

2.2. táblázat. A szalvéták paraméterei

Paraméter	Lehetséges értékek
Papír színe	fehér, bézs
Rétegek száma	1,2 or 3
Hajtás típusa	$\frac{1}{4}$ vagy $\frac{1}{8}$
Súly	2.0691g, 3.7026g or 4.9005g

Két gyártó berendezés, Guszti és KisVakond segítségével valósul meg a gyártás, amelyek paramétereit a 2.3 táblázat mutatja be. (A berendezések elnevezése az eszközökön található felíratok alapján történt, amelyet feltehetően a korábbi tulajdonos ragasztott rájuk.)

2.3. táblázat. Gyártó berendezések paraméterei

Név	Hajtás	Klisék száma	Sebesség (szál/s)
KisVakond	$\frac{1}{4}$	1	0.3
Guszti	$\frac{1}{4}, \frac{1}{8}$	2	0.25

A 2.3. táblázat alapján az $\frac{1}{8}$-os hajtással csak Guszti képes szalvétát gyártani, ami következtében a keresési tér nagymértékben csökken. Az ütemezés során figyelembe kell venni a gyártási sebességet is, ami a nagyobb volumenű megrendelések esetén jelentősen befolyásolja az ütemezést.

Egy megrendelés megadja a konkrét terméket (papír színe és típusa, minta, festék színe és csomagolás), a szalvéták darabszámát, valamint a gyártásra vonatkozó határidőt. Alapesetben a vállalt határidő 2 hét, de nagyobb prioritású ügyfelek esetén rövidebb határidő is jellemző.

A fent részletezett paraméterek alapján látható, hogy egy jó ütemezés jelentős hatással van a gyártó berendezések kihasználtságára és az effektív munkaidőre, vagyis a gyártás hatékonyságára. A megoldás során a problémát P-gráffal modelleztük a TCPNS keretrendszer segítségével.

2.3.2. Az egyedi lenyomtatos szalvéta gyártás modellezése

A gyártórendszerek modellezése esetén egyik kardinális kérdés az egyes fázisok során előálló anyagok kezelése. Az irodalomban több tárolási stratégia is ismert, mint például a véges vagy végtelen köztes tárolót feltételező stratégia, de vannak olyan folyamatok is, ahol nem áll rendelkezésre köztes tároló, vagy akár az előálló köztes anyag azonnali feldolgozása szükséges. A 2.1.1. fejezetben ismertetett modell megfelel a végtelen köztes tárolót feltételező esetnek. További tárolási stratégiák kezelése a 3. fejezetben kerül bemutatásra.

A jelenlegi szalvétagyártási probléma esetén a gyártó berendezés egy lépésben képes elvégezni a vágást, a nyomtatást és a hajtogatást, majd a csomagolást manuálisan végzik. Így a gyártási folyamat egyes lépéseit nem lehet különböző eszközökre szétosztani, ezért nem szükséges az egyes lépések során előálló félkész termékek tárolása. Vagyis a 2.1.1. fejezetben leírt modell alkalmazható a problémára.

A modellezés első lépéseként a P-gráf csomópontjai és a probléma elemeit kell megfeleltetni egymásnak; az eredmény a 2.4. táblázatban látható.

2.4. táblázat. A P-gráf csomópontjai és az ütemezési probléma elemeinek megfeleltetése

P-gráf csomópontjai	Az ütemezési probléma elemei
Erőforrás	Gyártó berendezés, Papír tekercsek
Céltermék	Végrehajtott feladatok
Műveletek	Gyártás, Átváltás

A gyártó berendezéseket és a készleten lévő papírtekercseket erőforrásként kell kezelni. Az alapanyag mennyiségének kezelése lehetővé teszi, hogy a megrendeléseket az aktuális készlet-

szintnek megfelelően ütemezzük, majd az elvégzett feladatoknak megfelelően a raktárkészletet csökkentsük. Négyféle típusú papírt különböztetünk meg szín és a rétegek száma alapján: 1,2 vagy 3 rétegű fehér, illetve 2 rétegű bézs. A modellben minden papírtípushoz külön erőforráscsomópont kerül létrehozásra. A legyártható szalvéták maximális számát a szalvéta és a papírtekercsek súlya alapján lehet meghatározni. Így egy megrendelés előállításának előfeltétele nemcsak a szabad eszköz, hanem a megrendelésben meghatározott mennyiségű szalvétának megfelelő papír is. A pontosabb készletszint számítás érdekében a papírfogyasztás hasonló módon kezelhető minden olyan átállásnál, ahol hulladék keletkezik.

Két megrendelés gyártása közötti átállási idő úgy modellezhető, hogy minden átállást leíró művelethez egy fix végrehajtási idő kerül beállításra, amelyet a két egymást követő megrendelés paraméterei alapján kell meghatározni. A szalvéta gyártási problémához tartozó modell építésének szemléltetésére a 2.5. táblázatban szereplő megrendeléseknek megfelelő P-gráf generálásának lépései kerülnek bemutatásra.

2.5. táblázat. A megrendelések és paramétereik

Megrendelés id.	Minta	Papír típus*	Hajtás	Festék szín	Darab	Határidő
O1	A	W2L	1/4	black	36.000	16:00
O2	B	B2L	1/4	red	21.600	16:00
O3	C	W2L	1/4	red	24.000	16:00
O4	D	B2L	1/8	red	36.000	16:00
O5	E	W1L	1/8	black	28.800	16:00
O6	F	W1L	1/8	black	14.400	16:00

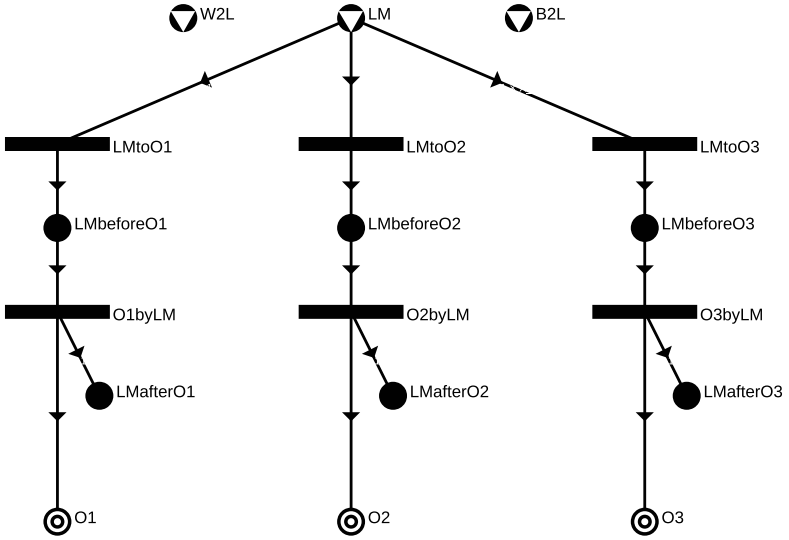
* B2L - bézs 2 rétegű, WxL - Fehér x rétegű

Az ütemezési probléma megoldásához meg kell határozni a célfüggvényt, amelyben a TCPNS esetén a költség- és időparaméterek lineáris kombinációja. A motivációs példában a cél a tervezési időszak alatt teljesített megrendelések összértékének maximalizálása, a teljesítési idők minimalizálása mellett. Az egyes megrendelések értéke a nyereség, valamint a megrendelő prioritása alapján kerül meghatározásra. A prioritás exponenciálisan növekszik a várható szállítási határidőig hátralévő idő függvényében. A megrendelések teljesítési idejének minimalizálása másodlagos cél, így a célfüggvényben a kapcsolódó együtthatók lényegesen alacsonyabbak, mint a megrendelések értékei.

A megrendelések paraméterei közül a hajtás típusa határozza meg, hogy melyik gyártó berendezés tudja azt legyártani. Következésképpen a KisVakond csak az O1, O2 és O3 megrendeléseket, míg Gustav az O1, O2, ..., O6 megrendelések bármelyikét teljesíteni tudja. A cél az optimális ütemezés meghatározása a fenti feltételeknek megfelelően. A következőkben a KisVakondhoz tartozó modell építésének lépései kerülnek részletes ismertetésre, majd bemutatásra kerül az egész probléma maximális struktúrája és optimális megoldása is.

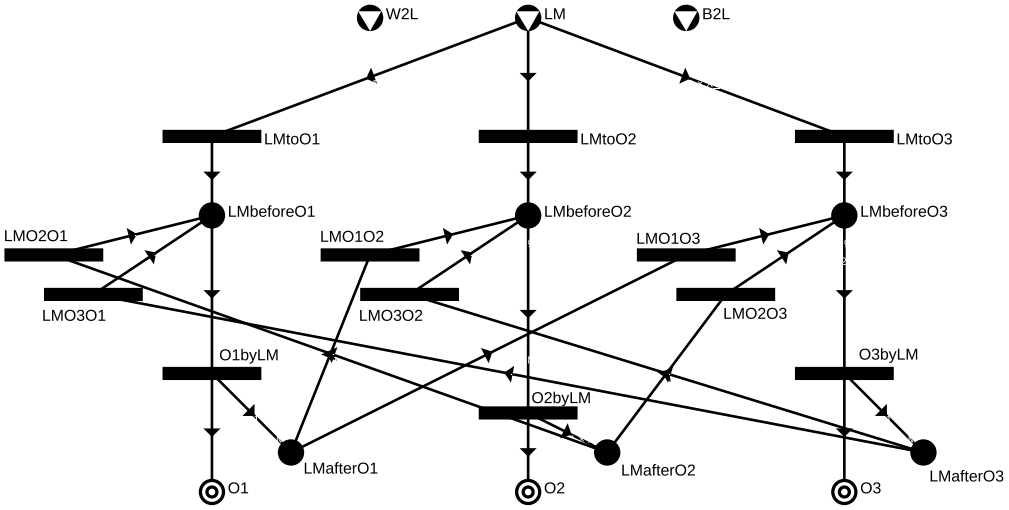
A 2.1.1. fejezetben leírt modell generáló eljárásnak megfelelően a papírtekercsek ($W2L$, $B2L$)

és a gyártó berendezések (LM) erőforrás csomópontként kerülnek modellezésre. Mivel a példában nincs precedencia meghatározva a feladatok között, így minden megrendeléshez ($O1, O2, O3$) egy új termék kerül bevezetésre. A következő lépésben a feladatok végrehajtási folyamatát leíró részgráf ($E_i to T_j, E_i before T_j, T_j by E_i, E_i after T_j$) kerül generálásra minden egyes gyártó berendezés és az általa elvégezhető feladat számára. A P-gráf ezen állapota a 2.15. ábrán látható.



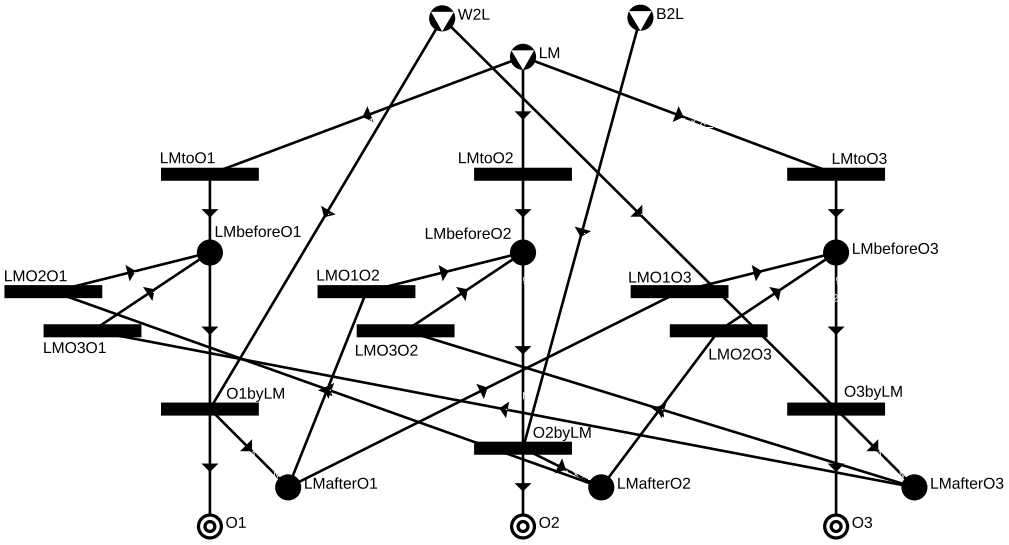
2.15. ábra. Esettanulmány: A végrehajtási folyamatot leíró részgráfok hozzáadása utáni állapot

A következő lépésben történik az átállásokat megvalósító műveletek hozzáadása a gráfhoz, amely során azt vizsgáljuk, hogy egy t_j feladat elvégzése után a gyártó berendezés melyik másik feladatot képes még végrehajtani. Minden lehetséges átmenethez egy új művelet felvételére van szükség, amely összeköti a feladat végrehajtás utáni állapotot ($E_i\_after\_T_j$) a következő lehetséges t_k feladat előtti állapottal ($E_i\_after\_T_k$). Mivel a feladatok között nincs meghatározott precedencia, ezért minden lehetséges átállást hozzá kell adni a gráfhoz, mert egy feladat elvégzése után bármely másik feladat elindítható; lásd: $LMO1O2$, $LMO1O3$, $LMO2O1$, $LMO2O3$, $LMO3O1$ és $LMO3O2$ a 2.16. ábrán. Például az $LM0102$ csomópont egy olyan átállást ír le, amely lehetővé teszi a berendezés (KisVakond) számára, hogy a $O1$ befejezése után elkezdhesse a $O2$ végrehajtását. Az összes lehetséges végrehajtási sorrend kezeléséhez három feladat esetén hat átmenet szükséges. Ha a feladatok között precedencia sorrend van meghatározva, akkor az átállásoknál ezt figyelembe kell venni, vagyis csak olyan feladatra lehet átállni, amely nem előfeltétele a befejezett feladatnak.



2.16. ábra. Esettanulmány: Átállásokkal bővített P-gráf modell

Az utolsó lépés a papírtekercsek kezelése, amely magában foglalja a megfelelő $W2L$ és $B2L$ alapanyag összekötését a $O1byLM$, $O2byLM$ és $O3byLM$ gyártási műveletekkel; lásd a 2.17. ábrát. A felhasznált papír mennyisége megegyezik az előállított szalvéták számával.

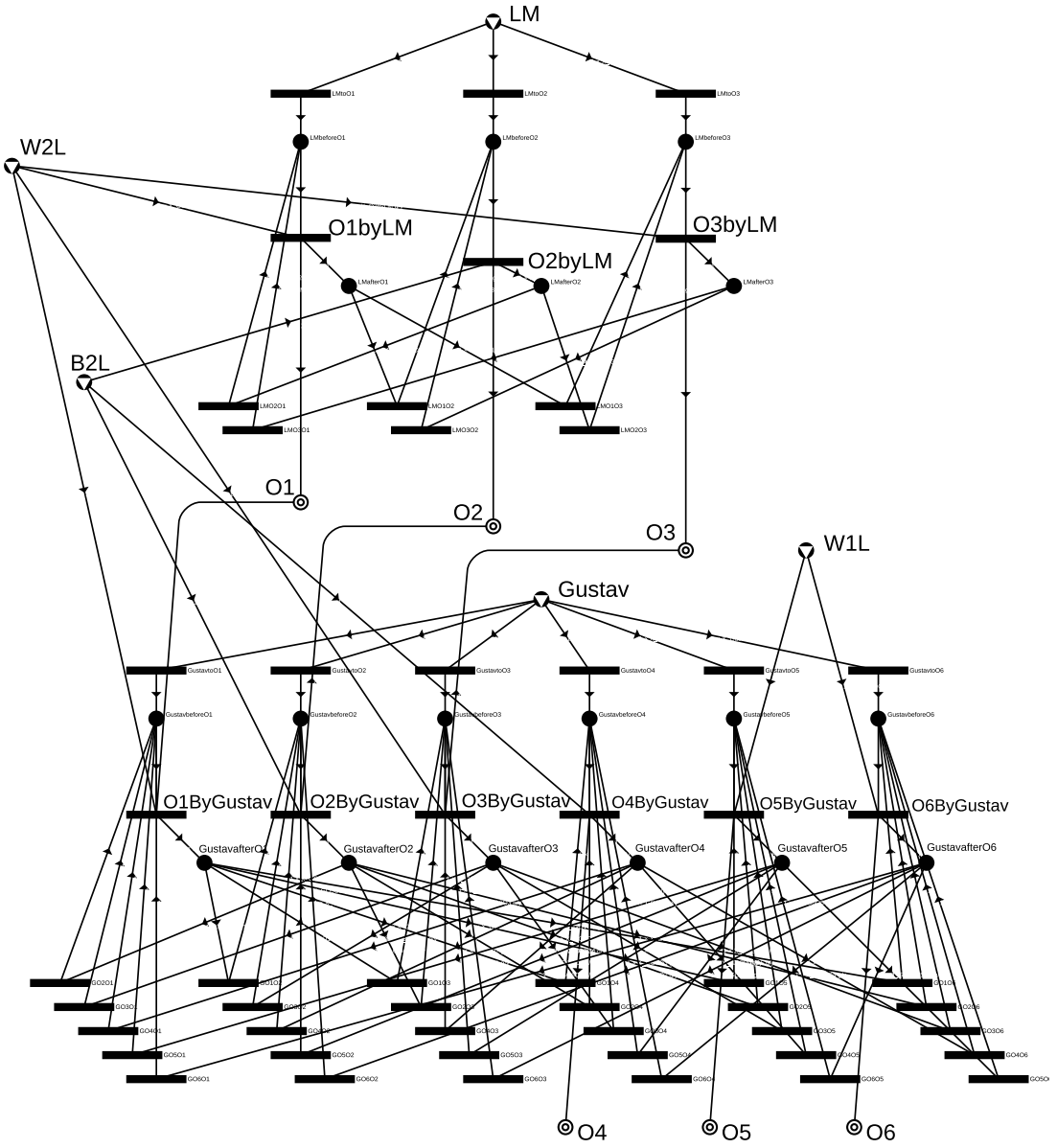


2.17. ábra. Esettanulmány: A papírtekercsek kezelésével bővített P-gráf modell

Ha a modellben a selejtet is kezelnie kell, akkor minden átállási műveletnél, ahol különböző alapanyagot igénylő megrendelések követik egymást, a következő megrendelésben meghatározott alapanyag egy új bemenetként jelenik meg. A hulladék mennyiségét az ütemezés becsüli, de a papír hulladék pontos mennyiségét a végrehajtás során rögzítik.

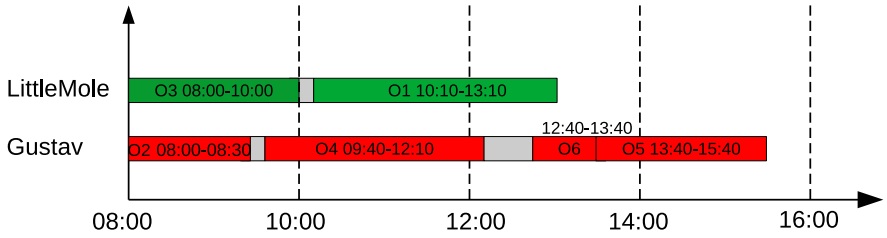
Az esettanulmányban definiált megrendelések két gyártó berendezésen való ütemezését leíró

maximális struktúrát a 2.18. ábra mutatja be.



2.18. ábra. Esettanulmány: A szalvéta gyártás ütemezését leíró maximális struktúra

Az optimalizálás célja, hogy minél több megrendelés a lehető leggyorsabban teljesítésre kerüljön, amely alapján az optimális ütemezés a 2.19. ábrán lévő Gantt diagramon és a 2.6. táblázatban látható.



2.19. ábra. Az optimális ütemezés ábrázolása Gantt diagramon

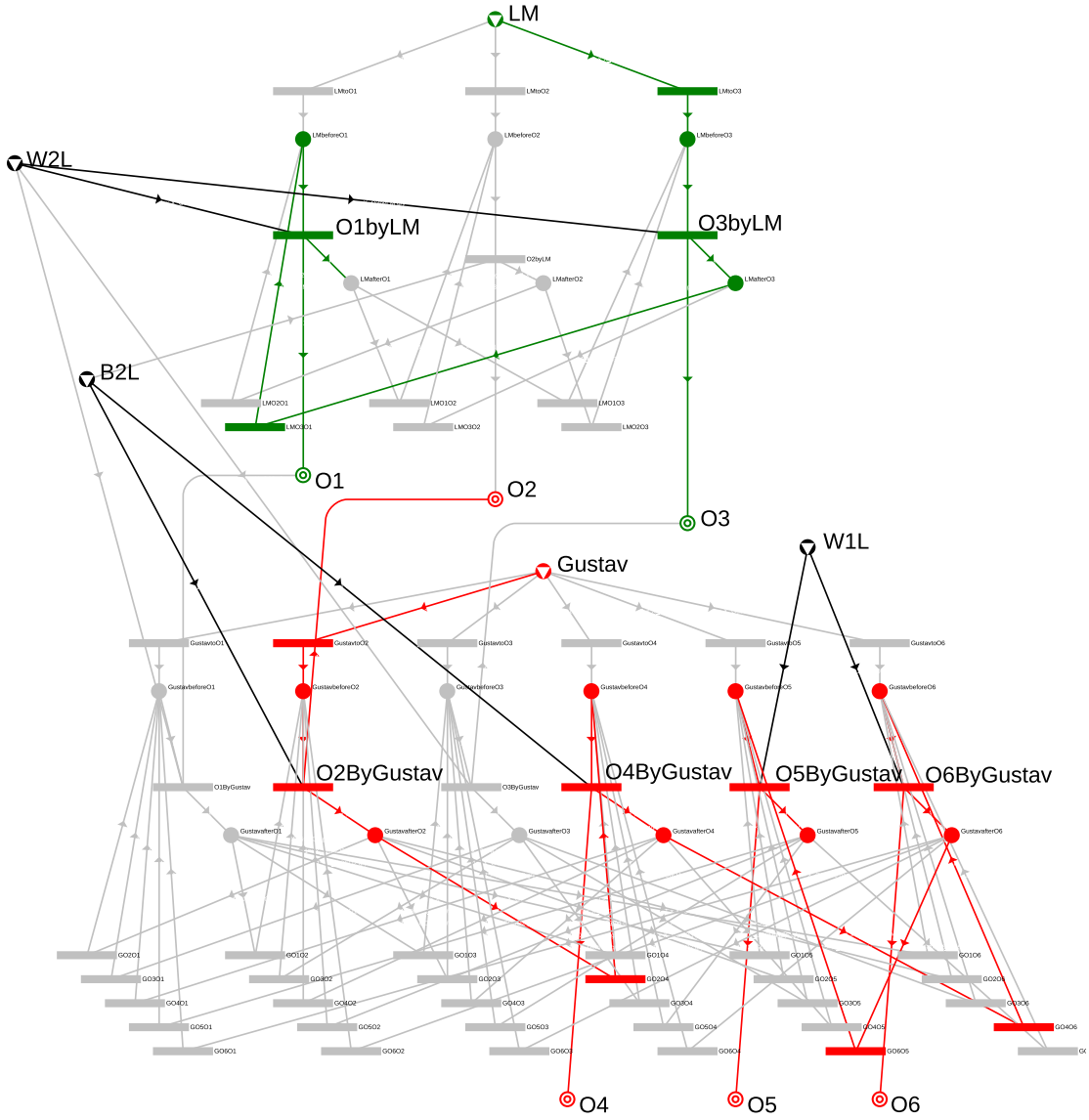
2.6. táblázat. Az optimális ütemezés táblázatos formában

Berendezés	Megrendelés Id	Kezdési idő	Befejezési idő	Átállási idő
KisVakond	O3	08:00	10:00	10
KisVakond	O1	10:10	13:10	-
Guszti	O2	08:00	09:30	10
Guszti	O4	09:40	12:10	30
Guszti	O6	12:40	13:40	0
Guszti	O5	13:40	15:40	-

A 2.20 ábra az optimális megoldáshoz tartozó megoldás struktúrát jeleníti meg a maximális struktúrában. Az ábrán jól látható a korábban leírt működés, amely szerint a folyamat a gyártó berendezéshez tartozó nyersanyagból indul és a feladat végrehajtást leíró csomópontokon halad végig, így meghatározva a gyártási sorrendet, illetve a kapcsolódó időváltozók révén az optimális ütemezést.

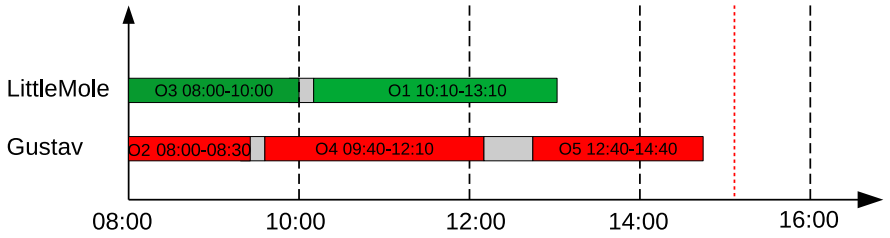
A 6 darab megrendelés alapján az volt várható, hogy a gyártó berendezések számára 3-3 feladat kerül kiosztásra. Az optimális ütemezés viszont azt mutatja, hogy ha a megrendelések gyors teljesítése is cél, akkor KisVakondnak $O3$ és $O1$, Guszti számára pedig $O2, O4, O6$ és $O5$ feladatot kell kiosztani, ebben a sorrendben. Ennek oka az átállási időkben keresendő, mert ha csak a végrehajtási időket vennénk figyelembe az ütemezés során, akkor az $O2$ megrendelést is KisVakond végezné el. Viszont a jelenlegi célfüggvény esetén, ha az $O2$ megrendelést KisVakondhoz rendelnénk, akkor az $O1$,$O2$ és $O3$ megrendelések bármely sorrendje hosszabb gyártási időt eredményezne, mint amikor Guszti végezné $O2, O4, O6, O5$ megrendeléseket ebben a sorrendben. Ebből a példából is látható, hogy már kisméretű feladat esetén is nehézkes a manuális tervezés, ezért szükséges a gyártás tervezés szoftveres támogatása.

A példában mind a 6 megrendelés sikeresen teljesült a határidőig, viszont ha ez nem kivitelezhető, akkor a modelltől függően két eset van; ha nem lehetséges a feladatok részleges teljesítése, akkor a határidőig nem végrehajtható megrendelések nem kerülnek bele a megoldásba. Viszont részleges teljesítés esetén egy feladat részben, kisebb volumennel is elvégezhető, akár több berendezésen is. A kétféle működés főként a megadott korlátok, valamint a fix és arányos idő beállításától függ, amely akár megrendelésenként változhat.

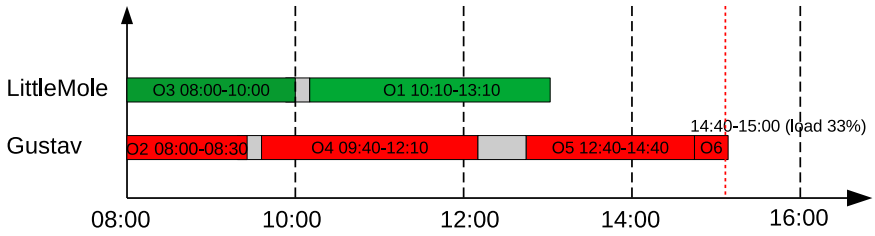


2.20. ábra. Az optimális megoldás szemléltetése a maximális struktúrában

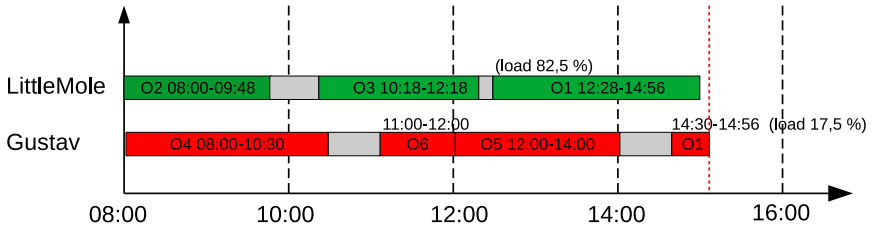
Módosítsuk a 2.5. a táblázatban szereplő megrendelések határidejét 15:00 órára, és vizsgáljuk meg a következő eseteket: a) nincs részleges teljesítés, b) az $O6$ megrendelés esetén lehetséges részleges teljesítés, és c) a részleges teljesítés minden megrendelés esetén lehetséges.



2.21. ábra. a) Az optimális megoldás 15:00 órás határidővel és részleges teljesítés nélkül



2.22. ábra. b) Az optimális megoldás 15:00 órás határidővel és részleges teljesítés csak $O6$ megrendelésnél lehetséges



2.23. ábra. c) Az optimális megoldás 15:00 órás határidővel és részleges teljesítéssel

A módosított példában a határidő lerövidítése miatt már nem ütemezhető minden megrendelés. Ezért a legalacsonyabb értékű $O6$ megrendelés nem került ütemezésre, amint azt a 2.21. ábrán látható. Ha a hatékonyabb gyártás érdekében a $O6$ megrendelésnél megengedett a részleges teljesítés, akkor az eredmény a 2.22. ábrának megfelelően alakul, ahol az $O6$ újra része a megoldásnak, de az elvárt volumennek csak 33%-a került végrehajtásra.

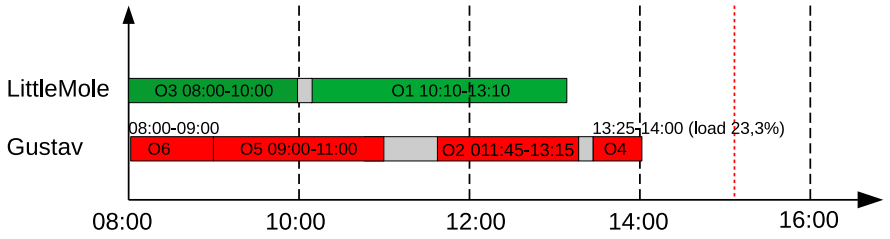
A 2.23. ábrán látható eredmény a TCPNS keretrendszer egyik fő előnyét, vagyis a feladatok több berendezésen való osztott végrehajtásának lehetőségét mutatja. A modell lehetővé teszi több gyártó berendezés számára, hogy ugyanazon feladaton dolgozzanak az elvárt volumen több részletben való teljesítésével. Ez látható a c) esetben, ahol a $O1$ megrendelés elvárt volumenét KisVakond és Guszti együtt állítják elő 82,5 % és 17,5 % arányban. Így viszont már a rövidebb határidő mellett is az összes megrendelés maradéktalanul teljesül.

A c) esethez generált P-gráf strukturálisan megegyezik az eredeti feladat modelljével, csak az időparamétereket és a műveletekre vonatkozó korlátokat kell úgy megadni, hogy a részleges

teljesítés megjelenhessen az eredményben. A mellékletben szereplő A.6.9. ábra bemutatja a c) esethez tartozó optimális megoldás struktúráját.

A szalvéta gyártási probléma esetén további feltétel volt az anyagkészletek kezelése. Csak azokat a feladatokat lehet ütemezni, amelyek gyártásához elegendő alapanyag áll rendelkezésre. Az új feltétel nehézsége abból adódik, hogy minden újabb megrendelés ütemezése során ellenőrizni kell, hogy az új gyártási sorrend továbbra is kivitelezhető a rendelkezésre álló készletek alapján.

A P-gráf modell építése során a különböző típusú papírtekercseket is nyersanyagként vesszük fel, amelyeket a gyártási műveletek a megrendelés volumenének függvényében fogyasztanak el. A készlet szint kezeléséhez a megfelelő felsőkorlátot kell beállításani ezeknek a nyersanyagoknak az elérhető mennyiségére. A módosítsuk a c) esetet úgy, hogy a bézs színű papírból 30.000 szalvétára elegendő áll rendelkezésre, amely eredményeként az optimális gyártási terv a 2.24. ábra szerint változik.



2.24. ábra. d) Az optimális megoldás 15:00 órás határidővel, részleges teljesítéssel és limitált mennyiségű bézs színű papírral

A $O2$ és $O4$ megrendelésekhez van szükség bézs színű papírra, amely korlátozott mennyisége miatt nem lehetséges az összes megrendelés teljesítése. A jelenlegi célfüggvény mellett, az optimális ütemezés szerint a $O4$ megrendelés csak 23,3 %-ban kerül teljesítésre.

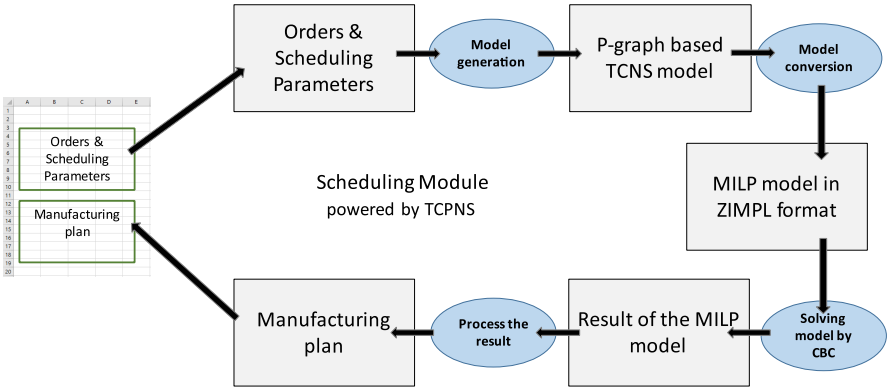
A szemléltető példa rávilágít arra, hogy már egy ilyen kisméretű példa esetében is az optimális ütemezés előállításához szükséges a számítógépes támogatás. Olyan szoftver modul implementálására van szükség, amely egy, az optimális megoldást garantáló módszertan felhasználásával, automatikusan képes elvégezni a gyártás ütemezését. A TCPNS keretrendszer teljesíti ezeket a feltételeket, így a szalvéta gyártás ütemezésére a bemutatott modellező és optimalizálási eljárásokat implementáltuk.

2.3.3. Szoftveres megvalósítás

Az előző fejezetben bemutatott P-gráf modell megfelelő alapot nyújt a szalvéta gyártás ütemezéséhez. Látható azonban, hogy még viszonylag kevés megrendelés esetén is nagyszámú csomópontot kell felvenni annak érdekében, hogy megkapjuk az összes lehetséges ütemezést magába foglaló maximális struktúrát. Belátható, hogy nagyobb, valós problémák esetén a modell manuális létrehozása, például a P-graph Studio (University of Pannonia) használatával szinte lehetetlen. Viszont a TCPNS modell generálása során alkalmazott lépések automatizálhatók, így az aktuális megrendelések alapján a P-gráf modell szoftveresen előállítható. A szalvéta gyártó céggel

folytatott együttműködés során implementálásra került egy szoftver modul, amely képes a szükséges adatok beolvasására, a P-gráf modell előállítására és megoldására, valamint az eredmény értelmezésére. A modul bemenete és kimenete is egy MS-Excel (Microsoft Corp.) fájl, amely tartalmazza a megrendeléseket, berendezéseket és a releváns paramétereket, valamint az ütemezés eredményeként ide kerül kiírásra az optimális gyártási terv is.

A feldolgozás során az ütemező modul képes létrehozni a megfelelő P-gráf modellt az aktuális megrendelések alapján, amelyet a megoldás során egy ZIMPL [96] formátumban leírt MILP modellé konvertálunk. A ZIMPL egy leírónyelv, amely egy probléma matematikai modelljét lineáris vagy nemlineáris (vegyes) egész matematikai programozási modellé alakítja, ami megkönnyíti a modell validálását és a hibák javítását, valamint támogatja az így megfogalmazott modell standard .lp formátumba való átalakítását. Az így generált .lp fájlt a COIN-OR Branch-and-Cut MIP Solverrel került megoldásra [97]. A MILP modell optimális megoldásában szereplő változók értékei szerint a gyártási ütemterv előállítható és visszaírásra kerül az Excel fájlba.



2.25. ábra. A probléma megoldásának szoftveres megvalósítása

A valós adatokkal végzett tesztek átlagosan 12%-os javulást mutattak a korábban alkalmazott kézi ütemezéshez képest. A tervezett gyártókapacitás-bővítéssel a szoftveres támogatás előnye várhatóan még jelentősebb lesz, mivel a berendezések nagyobb száma esetén a kézi ütemezés eredménye és a TCPNS által biztosított optimális megoldás közötti különbség valószínűsíthetően tovább növekszik.

2.4. A fejezet rövid összefoglalása

A fejezetben bevezetésre került egy új ütemezési módszer, amely segítségével időkorlátos folyamathálózat szintézis (TCPNS) problémaként oldhatók meg ütemezési problémák. A megoldás során egy P-gráf modellt építünk, amely tartalmazza a probléma összes lehetséges megoldását, így az optimális ütemezést is. Kidolgozásra került egy olyan struktúra generáló eljárás, amely lépései garantálják a szuperstruktúra előállítását egy általános ütemezési feladat esetén.

Ugyanakkor az alapmodell tovább fejlesztésével olyan lehetőségek váltak kezelhetővé az ütemezés során, mint az anyagmennyiségek folytonos kezelése, a részterhelés vagy egy feladat több berendezésre való arányos kiosztásának lehetősége, valamint a korlátos tároló kapacitás modellezése. A fejezet során részletes ismertetésre kerülnek a kapcsolódó modellek és működésük egy-egy illusztratív példán keresztül kerül bemutatásra.

Továbbá, az új ütemező eljárás működését egy valós ipari feladaton, az egyedi lenyomatos szalvéta gyártási probléma megoldásán keresztül is szemléltettem, ahol a váltási idők megfelelő kezelése révén 12%-al növekedett a gyártó berendezések kihasználtsága, amely hatékonyabb gyártást eredményezett.

2.4.1. A fejezethez tartozó tézis

Kidolgoztam egy új időkorlátos folyamat hálózat szintézisen (TCPNS) alapuló módszert ütemezési és folyamathálózat szintézis feladatok együttes megoldására. [24], [25], [98], [99], [100], [101]

(a) Meghatároztam az ütemezési feladatok leírását időkorlátos folyamat-hálózat szintézis feladatként

(b) Megmutattam, hogy az általam javasolt TCPNS formalizmus lehetővé teszi az ütemezés és folyamat szintézis együttes megvalósítását, beleértve a folytonos értékű erőforrás korlátok kezelését, feladatok tetszőleges arányban való megosztását több berendezés között, a tevékenységek volumenétől folytonos függvény szerint változó műveletvégzési idők figyelembe vételét az ütemezés során.

(c) Algoritmust dolgoztam ki olyan szuperstruktúra generálására, mely tartalmazza egy ütemezési feladat összes lehetséges megoldását, így garantáltan tartalmazza az optimális megoldást is.

(d) Az új ütemező módszer alkalmazhatóságát egy valós ipari probléma, az egyedi lenyomtatos szalvéta gyártás ütemezési feladat megoldásával igazoltam.

2.4.2. A fejezet témaköréhez kapcsolódó publikáció

Nemzetközi folyóiratcikk

- Frits Márton és Bertók Botond: Process scheduling by synthesizing time constrained process-networks, kiadvány: Computer Aided Chemical Engineering, vol 33, oldal 1345-1350 (2014) (IF=0.54) [24]
- Frits Márton és Bertók Botond: Scheduling custom printed napkin manufacturing by P-graphs, kiadvány: Computers & Chemical Engineering, vol 141, oldal 107017 (2020) (IF=3.845)[25]

3. fejezet

Tárolási stratégiák kezelése folyamatszintézis modellekben

Egy gyártási folyamat során a termék előállítása általában több lépésben történik. A lépések eredményeként előálló köztes termékek kezelését a tárolási stratégia határozza meg, amely nagyban befolyásolja az ütemezést. Szabályozza, hogy lehet-e tárolni a köztes termékeket, és ha igen, akkor milyen korlátok betartása mellet. Az irodalomban a következő tárolási stratégiák a leggyakoribbak:

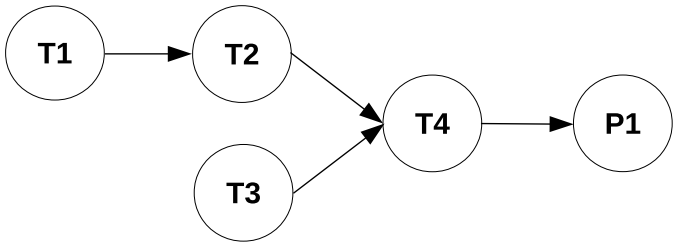
- Az UIS (Unlimited Intermediate Storage) tárolási stratégia esetén végtelen tároló kapacitás áll rendelkezésre, így a köztes termékek tárolása mindig megvalósítható. Ezért feltételezzük, hogy egy feladat befejezése után a berendezés közvetlenül megkezdheti a következő feladat végrehajtását. Ebből az következik, hogy a köztes tárolók modellezésére nincs szükség, mert nincsenek hatással az ütemezésre.
- A NIS (Non-Intermediate Storage) tárolási stratégia esetén a köztes termékek tárolására nincs lehetőség, így egy feladat elvégzése után a berendezésben kell a tárolást megoldani. Ebből az következik, hogy a berendezés csak akkor kezdheti meg a következő feladat feldolgozását, ha a benne tárolt köztes termék már áttöltésre került egy másik berendezésbe. Ebben az esetben az ütemező modellnek már kezelnie kell az áttöltési műveleteket is, amely eredményeként felszabadul az adott berendezés és hozzárendelhető másik feladathoz.
- A FIS (Finite Intermediate Storage) tárolási stratégia esetén a gyártási folyamathoz véges kapacitású tároló áll rendelkezésre, amelyek csak előre meghatározott berendezések között használhatók. Az ütemezés során biztosítani kell a köztes termékek tárolását, így a berendezés gyártási volumene a szabad tároló kapacitás függvénye, amely a folyamat során változik. A tároló kapacitás megfelelő kezelésének érdekében a tároló egységeket is modellezni kell, amely az tárolandó anyagok mennyiségének pontos meghatározását feltételezi, amely a 2.1.4. fejezetben már részletesen ismertetésre került.

– A ZW (Zero Wait) tárolási stratégia esetén a köztes termékek nem tárolhatók sem az azokat előállító gyártó berendezésbe, sem pedig egy külön tárolóban, ezért az ilyen köztes termékeket az előállításuk után át kell tölteni a következő berendezésbe, majd a feldolgozást azonnal meg is kell kezdeni.

– A MIS (Mixed Intermediate Storage) tárolási stratégia az előző négy stratégia egy folyamaton belüli alkalmazását jelenti, amely során minden egymást követő lépésnél külön meghatározható az elvárt tárolási eljárás.

Az iparban felmerülő gyártás ütemezési problémák általában visszavezethetők a fenti esetek valamelyikére. Például a 2.3. fejezetben bemutatott szalvéta gyártási probléma esetén nem szükséges a köztes tárolók modellezése, így megfeleltethető az UIS stratégiának. A 2. fejezetben bevezetett ütemezési feladatok megoldására kidolgozott TCPNS modell bővítésével modellezhetővé vállnak a fenti tárolási stratégiák, amelyek ebben a fejezetben egy szemléltető példán keresztül kerülnek bemutatásra.

A példa folyamat receptjét a 3.1. ábra mutatja be. A $P1$ termék előállításához a $T1, T2, T3$ és $T4$ lépésekre van szükség, ahol $T4$ feladatnak $T2$ és $T3$ lépések az előfeltételei, valamint $T2$ csak $T1$ végrehajtása után kezdődhet meg. Négy gyártó berendezés áll rendelkezésre ($E1$- $E4$), de ezek csak bizonyos lépéseket tudnak elvégezni, továbbá az összerendeléstől függően változik a feladat végrehajtási ideje is, amelyeket a 3.1. táblázat mutat be.



3.1. ábra. A példa feladat receptje

3.1. táblázat. A lehetséges feladat és berendezés összerendelések és végrehajtási idők

	E1	E2	E3	E4
T1	5	7	x	x
T2	x	8	12	x
T3	10	x	x	x
T4	x	x	x	10

Az ütemezés célja az elvégzett feladatok értékének maximalizálása a lehető leggyorsabb végrehajtás mellett. Mivel a fejezet fő témája a tárolási stratégiák modellezésének szemléltetése, ezért feltételezzük, hogy minden feladat azonnal megkezdhető, a határidőn belül az összes feladat ütemezhető és elegendő nyersanyag áll rendelkezésre a gyártáshoz. Viszont a gyártó berendezések

elérhetőségére annyi megkötéssel élünk, hogy az $E2$-es berendezés csak a 6. perctől lesz elérhető. A következő fejezetekben bemutatásra kerülnek a tárolási stratégiák kezelését megvalósító modellek.

3.1. Végtelen köztes tárolók modellezése

Az UIS (Unlimited Storage Policy) stratégia esetén feltételezzük, hogy végtelen tároló kapacitás áll rendelkezésre, ezért a folyamat lépéseinek eredményeként előálló köztes termékek tárolása mennyiségtől függetlenül megoldható. Az ütemezésben ez úgy jelenik meg, hogy egy feladat befejezése után a gyártó berendezés azonnal megkezdheti a következő feladat végrehajtását. Az UIS stratégiát alkalmazó gyártási folyamatok modellezése során egy olyan P-gráf modellt kell építenünk, amely tartalmazza az összes lehetséges ütemezést a probléma definíciójában megfogalmazott feltételek mellett. Ez magába foglalja a mennyiségi és időbeni korlátozásokat, a recept által előírt végrehajtási sorrendet, illetve a tárolási stratégiát. Ebben a fejezetben bemutatásra kerül az UIS stratégia ütemezési feladatokhoz felírt TCPNS modellben való kezelése.

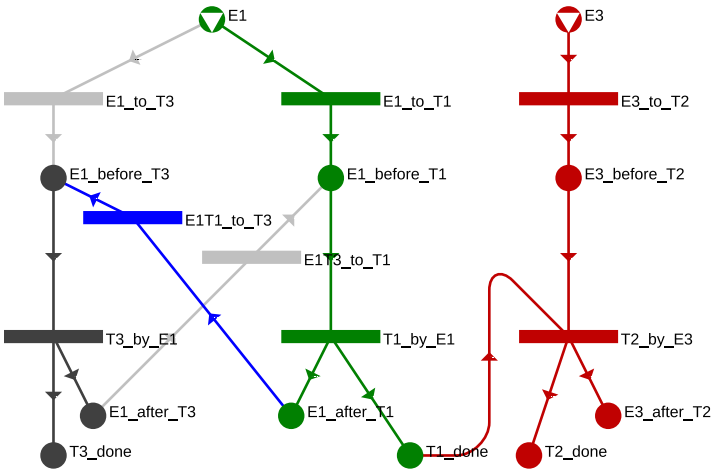
3.1.1. A végtelen köztes tárolók modellezése

A dolgozatban tárgyalt tárolási stratégiák közül az UIS stratégia a legengedékenyebb, mivel nem korlátozza a gyártás folyamán keletkező köztes termékeket sem mennyiségben, sem pedig időben. Ezért UIS stratégia esetén a köztes tárolók kezelésére nem szükséges új feltételek vagy műveletek bevezetése, így ez a gyártás ütemezési probléma megegyezik a 2. fejezetben tárgyalt általános ütemezési problémával. Ebből következik, hogy a 2.1.1. fejezetben részletesen ismertetett modell generáló lépések alkalmazásával a problémának megfelelő modell előállítható, így ezeket most nem részletezem.

Tekintsünk egy egyszerű példát az UIS stratégia modellezésének szemléltetésére. A bevezető részben definiált probléma receptje alapján (3.1. ábra) $T1$ feladat megelőzi $T2$ feladatot, viszont $T3$ ezektől függetlenül megoldható. Vegyük a problémának egy szűkebb részét, ahol csak $E1$ és $E3$ áll rendelkezésre $T1$, $T2$ és $T3$ feladatok elvégzésére. A 3.1. táblázat alapján $E1$ berendezés $T1$ és $T3$ feladatot hajthatja végre, amíg $E3$ berendezés csak $T2$ feladat elvégzésére képes. Tekintsünk egy olyan ütemezést, amelynél $T1 \rightarrow T2 \rightarrow T3$ a végrehajtási sorrend. A 3.2. ábra ezt az esetet mutatja be, ahol az egyes feladatok végrehajtásában érintett csomópontok és élek rendre zölddel, bordóval és sötétszürkével, valamint a tárolási stratégia kezelését megvalósító részgráf pedig kékkel lett jelölve.

Látható, hogy a $T1$ elvégzése után az eddigi szabályoknak megfelelően létrejönnek az $T1\_done$ és $E1\_after\_T1$ állapotok, ahol az előbbi engedélyezi a $T2$ feladat megkezdését, míg az utóbbi lehetővé teszi az $E1$ számára további feladatok végrehajtását. A két állapot egymástól teljesen független, így az $E1$ már dolgozhat $T3$ feladaton, amíg $T2$ feladat a megfelelő berendezésre várakozik. Ez csak úgy lehetséges, ha a $T1$ feladat során létrejött köztes termék tárolását biztosítani tudjuk, amely végtelen tárolási kapacitást feltételezve mindig kivitelezhető. A szemléltető példa alapján is látható, hogy a korábban bemutatott modellezési eljárás segítségével a felvázolt

probléma megoldható.



3.2. ábra. Az UIS stratégia modellezésének szemléltetése

3.1.2. A modell generálás lépései UIS stratégia esetén

Az UIS tárolási stratégia esetén alkalmazható TCPNS modell generálásának lépései megegyeznek a 2.1.1. fejezetben már részletesen bemutatott lépésekkel, ezért ez a fejezet csak egy rövidebb áttekintést ad az eljárás 3 lépéséről:

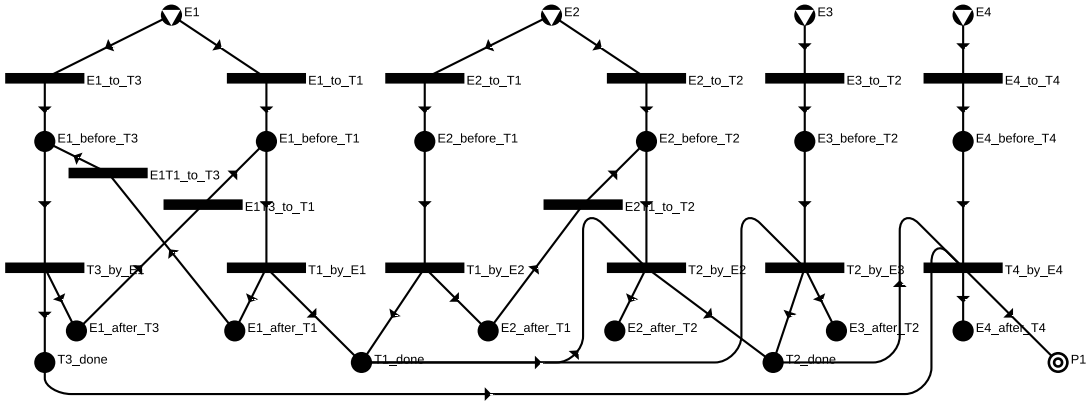
- **1.lépés:** felvételre kerülnek a berendezéseknek megfelelő erőforrások (E_i), a feladatoknak megfelelő köztes és végtermékek (T_j_done), valamint a feladatok végrehajtási folyamatát leíró gráf ágak minden lehetséges berendezés-feladat esetén. Egy ág magába foglalja a berendezés feladathoz való hozzárendelését ($E_i\_to\_T_j$) és a feladat végrehajtását ($T_j\_by\_E_i$). Ezen műveletek összekötése a végrehajtás előtti ($E_i\_before\_T_j$) és utáni ($E_i\_after\_T_j$) állapotok segítségével valósul meg.
- **2.lépés:** a feladatok közötti átváltási műveletekkel bővül a gráf ($E_iT_j\_to\_T_k$). Ennél a lépésnél azokat a berendezéseket vesszük figyelembe, amelyek egynél több feladatot is el tudnak végezni, ezeknél biztosítani kell, hogy a feladatokat minden lehetséges sorrendben képesek legyenek végrehajtani. A lehetséges sorrendek számát a recept korlátozza, ezért csak olyan átmeneteket vegyünk fel a gráfba, amelyek a precedencia sorrend alapján értelmezhetők, vagyis egy feladat végrehajtása után nem folytathatjuk a feldolgozást annak valamelyik előfeltételével.
- **3.lépés:** a receptben meghatározott precedenciáknak megfelelő végrehajtási sorrend biztosítása érdekében további élekkel bővítjük a gráfot. Ennek eredményeként minden feladatot leíró köztes termék (Tj_done) előfeltétele lesz a receptben meghatározott következő feladat végrehajtásának ($T_k\_by\_E_i$).

Ezen lépések mentén generált P-gráf struktúra az UIS stratégiát alkalmazó gyártási ütemezési feladat maximális struktúrája, amely megoldása révén megkapjuk az optimális ütemezést. A modellgeneráló eljárás formális leírása megegyezik a 2.2. fejezetben tárgyalt algoritmussal, így az ebben a fejezetben külön nem kerül részletezésre.

3.1.3. A példa feladat megoldása

Az előző fejezetben bemutatott lépések alapján bemutatásra kerül a bevezető részben definiált gyártás ütemezési feladat P-gráf modelljének generálása UIS tárolási stratégia esetén.

Az első lépésben felvételre kerülnek az erőforrások ($E1$- $E4$), a feladatoknak megfelelő köztes ($T1\_done$ - $T4\_done$) és végtermékek ($P1$), valamint a 3.1. táblázatnak megfelelően a feladat végrehajtást leíró gráf ágak. Például az $E1$ berendezés esetén a $T1$ és $T3$ feladatok megoldásához szükséges két ág kerül létrehozásra. A második lépésben a feladatok közötti átváltási műveletekkel bővül a gráf. A példában $E1$ berendezés a $T1$ és $T3$ feladatokat is végre tudja hajtani, amelyek nincsenek relációba a recept alapján, így mindkét végrehajtási sorrend valós, ezért mindkét váltási műveletre szükség van. Másik eset az $E3$ berendezés, amely képes $T1$ és $T2$ teljesítésére, viszont a receptben definiált reláció miatt csak $T1 \rightarrow T2$ átállás értelmezhető. A példában három új művelet felvétele szükséges ($E1T1\_to\_T3, E1T3\_to\_T1$ és $E2T1\_to\_T2$), amelyek révén a P-gráf modell már tartalmazza az összes lehetséges végrehajtási sorrendet.



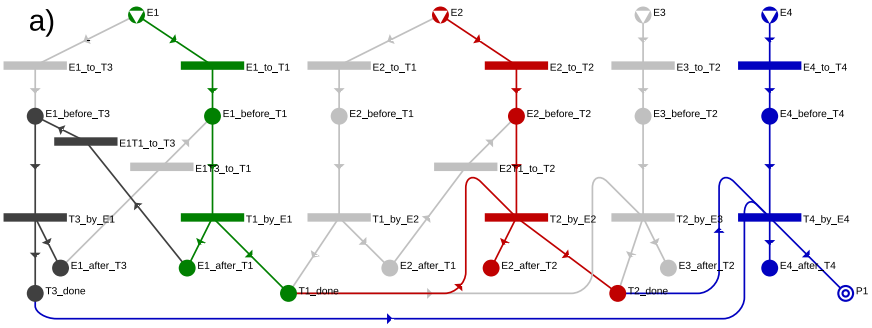
3.3. ábra. A példa feladat maximális struktúrája UIS stratégia esetén

A harmadik lépésben a receptben meghatározott precedenciáknak megfelelő végrehajtási sorrend biztosítása érdekében további élekkel bővítjük a gráfot. A példa feladatban ez két esetben releváns: (1) $T2$ feladat feldolgozása csak akkor kezdődhet, amikor $T1$ befejeződött, ezért a $T1\_done$ új bemenetként kerül bekötésre a $T2$-t megvalósító műveletekhez ($T2\_by\_E2$ és $T2\_by\_E3$), (2) $T4$ feladat végrehajtása csak $T2$ és $T3$ után kezdődhet, így $T2\_done$ és $T3\_done$ is bemenete lesz $T4\_by\_E4$ műveletnek. A probléma maximális struktúrája a 3.3. ábrán látható.

3.2. táblázat. A példa feladat lehetséges ütemezései

	1.feladat	2.feladat	3.feladat	4.feladat
a)	$T1 - E1$	$T2 - E2$	$T3 - E1$	$T4 - E4$
b)	$T1 - E1$	$T2 - E3$	$T3 - E1$	$T4 - E4$
c)	$T1 - E2$	$T2 - E2$	$T3 - E1$	$T4 - E4$
d)	$T1 - E2$	$T2 - E3$	$T3 - E1$	$T4 - E4$
e)	$T3 - E1$	$T1 - E1$	$T2 - E2$	$T4 - E4$
f)	$T3 - E1$	$T1 - E1$	$T2 - E3$	$T4 - E4$

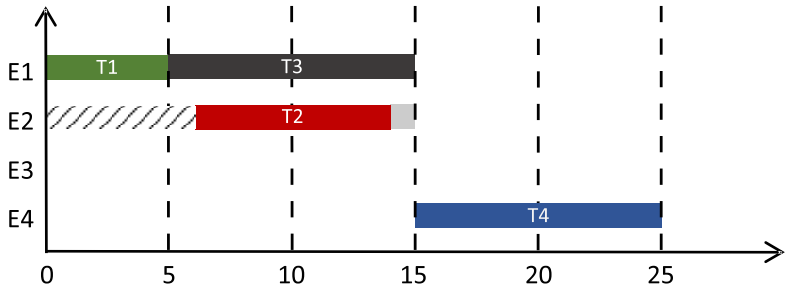
A maximális struktúrának tartalmaznia kell az összes lehetséges ütemezést, amelyek számát a 3.1. táblázat és a recept alapján lehet meghatározni. Mivel a $T4$ feladat mindig utoljára az $E4$ berendezés által kerül végrehajtásra, ezért elég csak az első három feladat kombinációit figyelembe venni. A recept alapján a lehetséges kombinációk száma tovább csökken, mivel $T2$ nem előzheti meg $T1$ feladatot. Továbbá mivel $T3$ nincs relációban $T1$ és $T2$ feladatokkal, ezért azoktól függetlenül is végrehajtható, így strukturális szempontból például a $T1E1 \rightarrow T2E2 \rightarrow T3E1 \rightarrow T4E4$ valamint $T1E1 \rightarrow T3E1 \rightarrow T2E2 \rightarrow T4E4$ ütemezések között nincs különbség. Ezek alapján 6 különböző megoldás struktúrát különböztetünk meg, amelyekhez kapcsolódó hozzárendelések a 3.2. táblázatban kerültek felsorolásra, amelyeket a 3.4., valamint a mellékletben szereplő B.6.10. - B.6.12. ábrákon struktúrálisan is megjelenítésre kerültek.



3.4. ábra. Az a) megoldás strukturális ábrázolása

A P-gráf alapján generált MILP modell megoldásával megkapjuk a feladat optimális ütemezését, amely megfelel a 3.4. ábrán szereplő megoldás struktúrának. A időváltozók értékei alapján pedig felírható a 3.5. ábrán látható Gantt diagram.

Az optimális megoldásában az $E1$ berendezés közvetlenül egymás után hajtja végre a $T1$ és $T3$ feladatokat. A $T2$ feladat a recept alapján a $T1$ után következik, de mivel az $E2$ csak a 6. percben válik elérhetővé, ezért várakoznia kell, viszont UIS stratégia esetén ez nincs hatással az $E1$ berendezés működésére, vagyis $T3$ feladat végrehajtására. A $T4$ feladatnak $T3$ és $T2$ is előfeltétele, így az csak a 15. percben kezdhető meg az $E4$ berendezésben. UIS tárolási stratégia



3.5. ábra. A példa feladat optimális megoldásának Gantt diagramon való megjelenítése UIS stratégia esetén

esetén az optimális ütemezés során a $P1$ termék a 25. percben áll elő. A példa feladat megoldása során bizonyítást nyert, hogy a korábban bemutatott általános ütemező modell használatával megoldhatók az UIS stratégiát alkalmazó gyártás ütemezési feladatok.

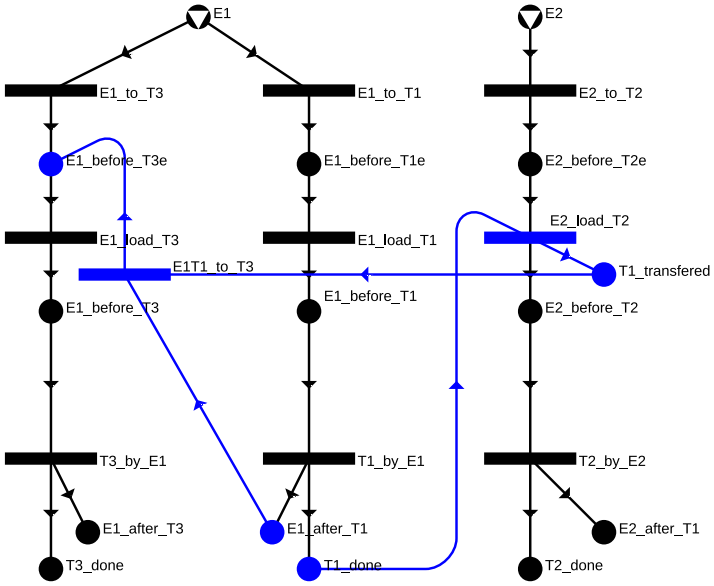
3.2. Köztes tároló nélküli ütemezés

A NIS (Non-Intermediate Storage Policy) stratégia esetén nincs lehetőség a gyártás lépései során előálló köztes termékek tárolására. Az ütemezésben ez úgy jelenik meg, hogy egy művelet elvégzése után az anyagot addig tároljuk a berendezésbe, amíg az nem tölthető át a következő lépést megvalósító berendezésbe. A korábban használt modell nem teljesíti ezt a feltételt, ezért azt ki kell egészíteni az áttöltések kezelésével, amely révén már a berendezések a NIS stratégiának megfelelően szinkronizálhatóvá vállnak. A probléma megoldása során a cél egy olyan P-gráf modell előállítása, amely tartalmazza az összes lehetséges ütemezést a probléma definíciójában megfogalmazott feltételek mellett, amely magába foglalja a mennyiségi és időbeni korlátozásokat, a recept által előírt és köztes tárolók hiányában is kivitelezhető végrehajtási sorrendet. Ebben a fejezetben bemutatásra kerül a NIS stratégiát megkövetelő gyártás ütemezési feladatok megoldására alkalmazható TCPNS modell.

3.2.1. A köztes tároló nélküli ütemezés modellezése

A korábban már ismertetett UIS stratégiához képest a NIS stratégia esetén feltételezzük, hogy a gyártás során nem áll rendelkezésre köztes tároló, ezért a folyamat egy lépésének eredményeként előálló köztes termék addig a berendezésbe várakozik, amíg nem tölthető át a következő feladatot végrehajtó berendezésbe. Ez csak úgy kezelhető megfelelően, ha a modellt kiterjesztjük az áttöltési művelettel, amely eredményeként a berendezés felszabadul és megkezdheti egy másik feladat végrehajtását. Az általános ütemezési feladat megoldására felírt modellben ez a fajta szinkronizáció nincs kezelve, ezért szükség van a P-gráf modell bővítésére. A modell generáló eljárás ismertetése előtt három kisebb szemléltető példán keresztül kerül bemutatásra, hogy milyen esetek lefedése szükséges a NIS stratégia megfelelő kezelésére.

Eset#1 Áttöltések kezelése: Tekintsünk először a bevezető részben definiált probléma egy szűkebb részét, ahol csak $T1E1 \rightarrow T2E2 \rightarrow T3E1$ ütemezésre fokuszálunk. Ha megfigyeljük a végrehajtás sorrendjét, akkor látható, hogy $T1$ feladat után csak akkor kezdődhet meg a $T3$-as feladat megvalósítása, amikor már az $E1$ berendezésből áttöltésre került a köztes termék az $E2$ berendezésbe. Ez egy általános eset, amely kezelését bemutató P-gráf modell látható a 3.6. ábrán.

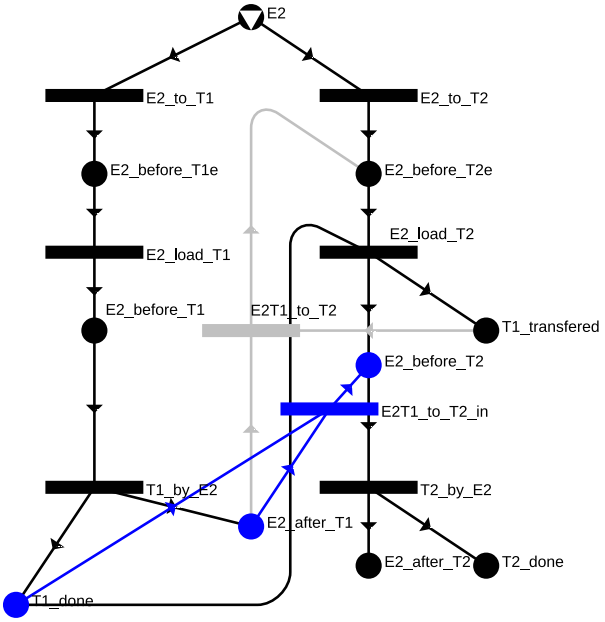


3.6. ábra. A NIS stratégia személtetése - Áttöltések kezelése

Az egyik jelentős változás, hogy a végrehajtást leíró ágakban megjelenik a feltöltési művelet ($E_i\_load\_Tj$) és megkülönböztetjük a végrehajtás előtti üres ($E_i\_before\_T_je$) és a feltöltött ($E_i\_before\_Tj$) állapotokat . A feltöltés során megtörténik a köztes anyag betöltése a berendezésbe, így az előző feladatot megvalósító berendezés felszabadul. A példa ütemezésnek megfelelően először $E1$ berendezés fogja végrehajtani $T1$ feladatot, amely eredményeként előáll a feladat befejezését jelző ($T1\_done$), valamint a végrehajtás utáni ($E1\_after\_T1$) állapot. Az UIS stratégia esetén ez a két állapot teljesen független volt, az átváltási művelet révén az $E1$ berendezés egyből eljuthatott az $E1\_before\_T3$ állapotba és meg is kezdhette $T3$ feladat végrehajtását. A 3.6. ábrán kék színnel van jelölve a NIS stratégia által előírt követelmények kezelését megvalósító részgráf. Látható, hogy a feladat váltásnak ($E1T1\_to\_T3$) a korábbitól eltérően két feltétele van, a feladat vége utáni állapot ($E1\_after\_T1$) mellett megjelenik egy $T1\_transfered$ állapot is, amely azt jelzi, hogy a $T1$ feladat eredménye áttöltésre került a következő berendezésbe. Ez az állapot úgy jön létre, hogy a $T1\_done$ már nem a $T2$ feladat elvégzésének előfeltétele, hanem a $T2$ feladat $E2$ berendezésbe való feltöltését végző műveleté ($E2\_load\_T2$). Így továbbra is biztosítható, hogy a $T2$ feladat végrehajtása csak $T1$ után kezdődhessen meg, viszont elkülöníthető a feltöltés és a feladat végrehajtás lépése. A feltöltéshez határozhatunk meg időt is, amely így megjelenik az ütemezésben. Ha az áttöltés befejeződött, akkor a $T1\_transfered$ állapot

létrejöttével már az $E1$ berendezés képes eljutni a $T3$ végrehajtása előtti üres állapotba. Mivel $E1\_load\_T3$ műveletnek nincs más előfeltétele, így a $T3$ végrehajtása megkezdődhez. Látható, hogy az Eset#1 példa modellezése NIS tárolási stratégia megkövetelése mellett a feltöltési művelet és a feladat váltáshoz bevezetett új előfeltétel bevezetésével már megfelelően modellezhető.

Eset#2 Áttöltés nélküli feladat váltás: Tekintsünk ugyancsak a bevezető részben definiált probléma egy szűkebb részét, ahol $T1E2 \rightarrow T2E2$ ütemezésre fokuszálunk. Ebben az esetben az $E2$ berendezés a $T1$ után közvetlenül a $T2$ feladatot is végrehajtja. A két feladat a receptben definiált sorrend alapján is egymás után következik. Könnyen belátható, hogy az első esetben leírt modell ilyenkor nem használható, mivel nincs áttöltés, amely révén előáll a $T1\_transfered$ állapot, így a szürkével jelölt feladat váltás megakadna. Ilyen esetekre alkalmazható megoldást mutatja be a 3.7. ábra, amelyen az átváltási művelet kékkel került feltüntetésre.

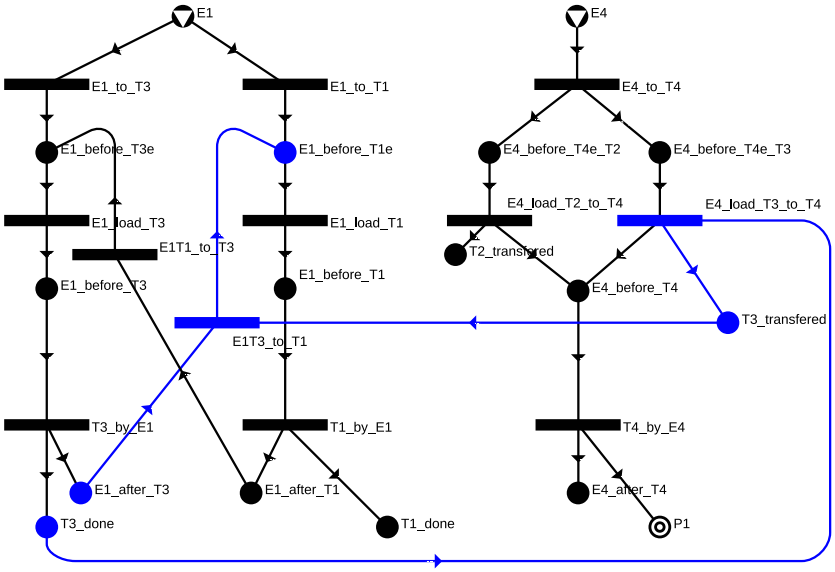


3.7. ábra. A NIS stratégia személtetése - Áttöltés nélküli feladat váltás

A példa gráfon látható, hogy az előző esetben tárgyalt váltási művelet helyett egy másik, a berendezésen belüli feladat váltást megvalósító művelet kerül hozzáadásra ($E2T1\_to\_T2\_In$). A különbség, hogy ennél váltási műveletnél $T1\_done$ közvetlenül kerül felhasználásra, tehát nincs szükség egy másik berendezésen végzett feltöltésre, valamint a művelet eredménye sem az üres, hanem közvetlenül a végrehajtás előtti tele állapot. Így miután $E2$ berendezés végrehajtotta $T1$ feladatot, akkor minden feltétel adott, hogy közvetlenül megkezdje $T2$ végrehajtását is. Érdemes megjegyezni, hogy a korábban bevezetett váltási műveletre ebben az esetben nincs szükség, mert ha egy másik berendezés hajtja végre $T1$ feladatot, akkor a $E2\_to\_T2$ műveleten keresztül történik az $E2$ erőforrás hozzárendelése a $T2$ feladathoz. Látható, hogy az új típusú váltási

művelet révén az áttöltés nélküli végrehajtás is kezelhető a NIS tárolási stratégiának megfelelően.

Eset#3 Többszörös előfeltétel kezelése: A harmadik esetben is tekintsük a bevezető részben definiált probléma egy szűkebb részét, ahol csak $T3E1 \rightarrow T1E1$ ütemezésre fókuszálunk. A példa hasonló, mint az első esetben, ahol szintén $E1$ berendezés hajtotta végre $T1$ és $T3$ feladatokat, viszont a fordított sorrend egy újabb esetre világít rá. Ha megvizsgáljuk a példát, akkor nyilvánvaló, hogy $T1$ végrehajtása csak akkor kezdődhet meg, ha $T3$ eredménye áttöltésre került a $T4$ feladatot megvalósító $E4$ berendezésbe. Az eset érdekessége, hogy $T4$ feladatnak két előfeltétele van, vagyis $T2$ és $T3$ feladatot is be kell fejezni a $T4$ feladat megkezdése előtt. Viszont nagy valószínűséggel a két feladat nem egyszerre fejeződik be, ezért az ilyen esetekben a feltöltéseket külön kell kezelni. Ezt mutatja be a 3.8. ábra.



3.8. ábra. A NIS stratégia személtetése - Többszörös előfeltétel kezelése

Többszörös előfeltétel esetén a végrehajtást leíró ág a hozzárendelési művelet után elágazik, és az előfeltételekkel megegyező számú (N) feltöltés előtti üres állapot és feltöltési művelet kerül létrehozásra. A művelet eredményeként létrejön az adott előfeltételnek megfelelő $Tj\_transfered$ és a feltöltés utáni tele állapot, de utóbbi csak $1/N$ részben. Így biztosítható, hogy a feltöltések függetlenül is elvégezhetők legyenek, de csak az összes előfeltétel befejezése és áttöltése után kezdődhessen meg a végrehajtás. A szemléltető példában miután $E1$ befejezte $T3$ feladatot, akkor létrejönnek a $T3\_done$ és $E1\_after\_T3$ állapotok. Az első esetben bevezetett feladat váltási művelet egyik feltétele a $T3\_transfered$ állapot, amely jelzi, hogy az áttöltés megtörtént, az $E1$ berendezés üres. Az $T4$ feladatnak a $T2$ és $T3$ feladat is előfeltétele, ezért ennek megfelelően kétfelé ágazik a gráf. Az $E4\_load\_T3\_to\_T4$ művelet előállítja a szükséges $T3\_transfered$ állapotot és emellett a $0.5*E4\_before\_T4$ állapotot is. Megjegyzés: $T2$ végrehajtása nem szerepel

az ábrán, de feltételezzük, hogy valamely berendezés számára kiosztásra került, így a folyamat során előáll a $T2\_done$ állapot, ami felhasználásával már elvégezhető a másik feltöltési művelet is, amely eredményeként már a $E4\_before\_T4$ állapot teljes lesz, így megkezdődhet $T4$ megvalósítása is.

NIS tárolási stratégia esetén egy olyan modell generáló eljárásra van szükség, amely a bemutatott három esetnek megfelelően képes automatikusan generálni a megfelelő P-gráf struktúrát az ütemezési probléma definíciója alapján.

3.2.2. A modell generálás lépései NIS stratégia esetén

A NIS stratégiát megkövetelő gyártási folyamatra felírt P-gráf modell generálása során biztosítani kell az előző fejezetben bemutatott három eset megfelelő kezelését. Ennél a feladatnál is az általános ütemezési feladatok megoldására felírt modellből (lásd: 2. fejezet) érdemes kiindulni, amelyet bővíteni kell a feladat váltások megfelelő kezelésével. Ez alapján a P-gráf modell generálása a következő lépések révén valósul meg:

- **1.lépés:** felvételre kerülnek az erőforrások (E_i), a feladatoknak megfelelő köztes és végtermékek (Tj_done), valamint a feladatok végrehajtási folyamatát leíró gráf ágak minden lehetséges berendezés esetén. Az áttöltések megfelelő kezelése érdekében elemezni kell a receptet és a feladatok előfeltételeinek számától függően kell az ágakat generálni.

  Ha a T_j feladatnak legfeljebb egy előfeltétele van, akkor az ág a berendezés feladathoz való hozzárendelését ($E_i\_to\_T_j$), a feladat feltöltését ($E_i\_load\_T_j$) és a feladat végrehajtását ($T_j\_by\_E_i$) leíró műveleteket foglalja magába, amelyek összekötése rendre a végrehajtás előtti üres ($E_i\_before\_T_je$) és tele ($E_i\_before\_T_j$), valamint a végrehajtás utáni ($E_i\_after\_T_j$) állapotok segítségével valósul meg.

  Ha a T_j feladatnak legalább kettő előfeltétele van, akkor az ágnak az előfeltételek számával megegyező számú párhuzamos feltöltési feladatot kell tartalmaznia. Legyen $P(T_j)$, azoknak a feladatoknak a halmaza, amelyek T_j feladatnak közvetlen előfeltételei. Ilyenkor a feladathoz való hozzárendelés ($E_i\_to\_T_j$) eredményeként minden $T_k \in P(T_j)$ feladathoz létrehozunk egy $E_i\_before\_T_je\_T_k$ feltöltés előtti állapotot, majd ehhez kapcsolódóan egy $E_i\_load\_T_k\_to\_T_j$ feltöltési műveletet. Minden ilyen feltöltési művelet előállít egy $T_k\_transfered$ és $1/N$ résznyi $E_i\_before\_T_j$ állapotot, ahol $N = |P(T_j)|$.

- **2.lépés:** a feladatok közötti átváltási műveletekkel bővül a gráf ($E_iT_j\_to\_T_k$). Ennél a lépésnél azokat a berendezéseket vesszük figyelembe, amelyek egynél több feladatot is el tudnak végezni, ezeknél biztosítani kell, hogy a feladatokat minden lehetséges sorrendben képesek legyenek végrehajtani. Miden olyan T_j ,T_k feladatra, amelyeket Ei berendezés el tud végezni és a recept alapján értelmezhető a $T_j \rightarrow T_k$ végrehajtási sorrend, akkor egy új feladat váltási műveletet ($E_iT_j\_to\_T_k$) veszünk fel a gráfba, amelynek két bemenete van: T_j feladat E_i berendezés általi befejezését jelző állapota ($E_i\_after\_T_j$), és a köztes termék áttöltését jelző állapot ($T_j\_transfered$). A váltási művelet eredményeként az E_i berendezés

az $E_i\_before\_T_ke$, vagyis a feltöltés előtti üres állapotba kerül, ahonnan eljuthat a T_k feladat végrehajtásáig.

További feladat váltási műveletek szükségesek abban az esetben, ha az E_i által végrehajtható T_j és T_k műveletek a receptben közvetlenül követik egymást. Az új művelet ($E_iT_j\_to\_T_k\_in$) bemenetei a $E_i\_after\_T_j$ és T_j_done állapotok, kimenete pedig a $E_i\_before\_T_k$ feltöltés utáni állapot.

– **3.lépés:** a receptben meghatározott precedenciáknak biztosítása, amely során minden feladatot leíró köztes termék (Tj_done) előfeltétele lesz a receptben meghatározott következő feladat feltöltési műveletének ($T_k\_load\_E_i$).

Ezen lépések mentén generált P-gráf struktúra az NIS stratégiát alkalmazó gyártási ütemezési feladat maximális struktúrája, amely megoldása révén megkapjuk az optimális ütemezést.

A modell generálás lépéseinek formális leírását mutatja be a 2. és 3. Algoritmus, amelynél a 2.2. fejezetben bevezetett jelöléseket alkalmazom. Az eljárás bemenete egy NIS tárolási stratégiát megkövetelő gyártás ütemezési probléma, amelyhez generálásra kerül a vele ekvivalens időkorlátos hálózat szintézis feladat.

A 2. Algoritmus által leírt eljárásból a könnyebb átláthatóság érdekében kiemelésre került a feladat végrehajtást leíró ágak generálása a 3. Algoritmusba. Az eljárás az 1-25. sorig részben megegyezik az általános ütemezési problémánál (és UIS stratégiánál is) használt inicializáló lépésekkel, amely során felvételre kerülnek a berendezések, mint erőforrások, valamint a feladatok, mint köztes vagy végtermékek. Az eltérés az új áttöltést jelző ($t_j\_transfered$) állapotok felvétele az egyes feladatokhoz. Ezután a 26. sorban következik a 3. Algoritmusban kiemelt eljárás, amely eredményeként modellezésre kerülnek a feladat végrehajtások. Az utolsó lépésben (27-38.) pedig hozzáadásra kerülnek azok a váltási műveletek, amelyek lehetővé teszik, hogy egy berendezés több feladatot is képes legyen végrehajtani. Két esetet különböztetünk meg aszerint, hogy a t_j feladat végrehajtása után a következő lehetséges t_k feladatnak közvetlen előfeltétele-e. Ha nem (29-32. sor), akkor hozzáadunk egy új műveletet ($e_i\_t_j\_to\_t_k$), amelynek két bemenete a t_j feladat végrehajtása utáni ($e_i\_after\_t_j$) és a t_j feladat áttöltését jelző ($t_j\_transfered$) állapot. A művelet eredményeként a berendezés eljut a t_k feladat végrehajtásának feltöltés előtti üres ($e_i\_before\_t_ke$) állapotába.

A 3. Algoritmus valósítja meg a feladat végrehajtást modellező ágak generálását és a 2. generáló lépést. Minden olyan e_i berendezéshez, amely képes t_j végrehajtására, generáljuk a feltöltést és megvalósítási műveletek tartalmazó részgráfot. Két esetet különböztetünk meg aszerint, hogy a t_j feladatnak hány előfeltétele van. Ha legfeljebb egy előfeltétele van (4-12. sor), akkor az 1. Algoritmushoz hasonlóan felvételre kerül a feltöltés előtti üres ($e_i\_before\_t_je$), feltöltés utáni tele ($e_i\_before\_t_j$), valamint a végrehajtás utáni ($e_i\_after\_t_j$) állapotok. Majd az ezek közötti átmenet biztosító műveletek, mint a berendezés feladathoz való hozzárendelése ($e_i\_to\_t_j$), a berendezés feltöltése ($e_i\_load\_t_j$), valamint a feladat végrehajtása ($t_j\_by\_e_i$).

Egynél több előfeltétel esetén viszont már különbséget kell tenni a feltöltési műveletek között attól függően, hogy melyik feladat eredménye került áttöltésre (14-31. sor). Így a t_j feladat és e_i berendezés összerendelése után ($e_i\_to\_t_j$) minden előfeltételhez külön feltöltés előtti üres

Algoritmus 2: P-gráf modell generálása NIS tárolási stratégia esetén

input : $Problem(T, E, A, P, t_s, t_e, s, t, v, te, ts, cf, cp, ht)$: gyártás ütemezési probléma
output: $TCPNS(\mathcal{M}, \mathcal{P}, \mathcal{R}, \mathcal{O}, a, L, U, l, u, c, L_t, U_t, t_f, t_p)$: paraméteres TCPNS probléma

begin
  $\mathcal{R} := \emptyset$;
  foreach $e_i \in E$ **do**
    $\mathcal{R} := \mathcal{R} \cup \{r_k = e_i\}$;
    $U(r_k) = 1$;
    $L_t(r_k) = te(e_i)$;
    $U_t(r_k) = ts(e_i)$;
  end
  $\mathcal{P} := \emptyset$;
  foreach $t_j \in T$ ***where*** $\forall t_i \in T, t_j \cap P(t_i) = \emptyset$ **do**
    $\mathcal{P} := \mathcal{P} \cup \{p_k = (t_j\_done)\}$;
    $U(p_k) = 1$;
    $L(p_k) = ht(t_j)$;
    $U_t(p_k) = te(t_j)$;
    $\mathcal{M} := \mathcal{M} \cup \{(t_j\_transfered)\}$;
  end
  $\mathcal{M} := \emptyset$;
  foreach $t_j \in T$ ***where*** $\exists t_i \in T, t_j \cap P(t_i) \neq \emptyset$ **do**
    $\mathcal{M} := \mathcal{M} \cup \{m_k = (t_j\_done), (t_j\_transfered)\}$;
    $U(m_k) = 1$;
    $L(m_k) = ht(t_j)$;
    $U_t(m_k) = te(t_j)$;
  end
  $\mathcal{M} := \mathcal{M} \cup \mathcal{P} \cup \mathcal{R}$
  $\mathcal{O} := \emptyset$;
  generateTaskBranches$(Problem, TCPNS)$;
  foreach $t_j \in T$ **do**
    foreach $e_i \in A(t_j)$ **do**
      foreach $t_k \in T$ ***where*** $t_j \notin P(t_k), k \neq j$ **do**
        $\mathcal{O} := \mathcal{O} \cup \{(e_i\_t_j\_to\_t_k;$
        $\{e_i\_after\_t_j, t_j\_transfered\}; \{e_i\_before\_t_k e\})\}$;
      end
      foreach $t_k \in T$ ***where*** $t_j \in P(t_k), k \neq j$ **do**
        $\mathcal{O} := \mathcal{O} \cup \{(e_i\_t_j\_to\_t_k\_in;$
        $\{e_i\_after\_t_j, t_j\_done\}; \{e_i\_before\_t_k e\})\}$;
      end
    end
  end
end

$(e_i\_before\_t_j e\_t_k)$ állapot és egy feltöltés $(e_i\_load\_t_k\_to\_t_j)$ művelet kerül létrehozásra. Az utóbbi a feltöltés előtti és az előfeltétel végrehajtását jelző állapot (t_k_done) alapján előállítja az

áttöltés befejezését jelző állapotokat (t_k_done és $e_i\_before\_t_j$). A korábbi felépítéshez képest egy jelentős különbség, hogy a feltöltés utáni állapot csak $1/N$ arányban készül el egy feltöltés során, ahol $N = |P(t_j)|$, vagyis a t_j feladat előfeltételeinek száma. Ezzel a megoldással biztosítható, hogy egy rész feltöltés után a korábbi berendezés felszabadul, de t_j feladat végrehajtása csak akkor kezdődik meg, ha minden feltöltés megtörtént. A feladat végrehajtásának modellezése már megegyezik a korábban bemutatott működéssel.

Algoritmus 3: A feladat végrehajtást modellező gráf ágak generálása

input : $(T, E, A, P, t_s, t_e, s, t, v, t_{ee}, t_{es}, cf, cp, ht)$: gyártás ütemezési probléma
output: $(\mathcal{M}, \mathcal{P}, \mathcal{R}, \mathcal{O}, a, L, U, l, u, c, L_t, U_t, t_f, t_p)$: paraméteres TCPNS probléma

begin
  foreach $t_j \in T$, $|P(t_j)| <= 1$ **do**
    $t_k = P(t_j)$;
    foreach $e_i \in A(t_j)$ **do**
      $\mathcal{M} := \mathcal{M} \cup \{e_i\_before\_t_je, e_i\_before\_t_j, e_i\_after\_t_j\}$;
      $\mathcal{O} := \mathcal{O} \cup \{(e_i\_to\_t_j; \{e_i\}; \{e_i\_before\_t_je\}),$
       $(e_i\_load\_t_j; \{e_i\_before\_t_je, t_k\_done\}; \{e_i\_before\_t_j\}),$
       $o_k = (t_j\_by\_e_i; \{e_i\_before\_t_j\}; \{e_i\_after\_t_j, t_j\_done\})\}$;
      $u(o_k) = 1$;
      $cf(o_k) = cf(e_i)$;
      $L_t(o_k) = te(t_j)$;
      $U_t(o_k) = ts(t_j)$;
      $t_f(o_k) = t_f(t_j, e_i)$;
    end
  end
  foreach $t_j \in T$, $|P(t_j)| > 1$ **do**
    foreach $e_i \in A(t_j)$ **do**
      $\mathcal{M} := \mathcal{M} \cup \{e_i\_before\_t_j, e_i\_after\_t_j\}$;
      $\mathcal{O} := \mathcal{O} \cup \{o_{to} = (e_i\_to\_t_j; \{e_i\}; \{\})\}$;
      foreach $t_k \in P(t_j)$ **do**
        $\mathcal{M} := \mathcal{M} \cup \{e_i\_before\_t_je\_t_k\}$;
        $Outputs(o_{to}) = Outputs(o_{to}) \cup e_i\_before\_t_je\_t_k$;
        $\mathcal{O} := \mathcal{O} \cup \{(e_i\_load\_t_j\_to\_t_k; \{e_i\_before\_t_je\_t_k, t_k\_done\};$
         $\{t_k\_transfered, 1/|P(t_j)| * e_i\_before\_t_j\})\}$;
      end
      $\mathcal{O} := \mathcal{O} \cup \{o_k = (t_j\_by\_e_i; \{e_i\_before\_t_j\}; \{e_i\_after\_t_j, t_j\_done\})\}$;
      $u(o_k) = 1$;
      $cf(o_k) = cf(e_i)$;
      $L_t(o_k) = te(t_j)$;
      $U_t(o_k) = ts(t_j)$;
      $t_f(o_k) = t_f(t_j, e_i)$;
    end
  end
end

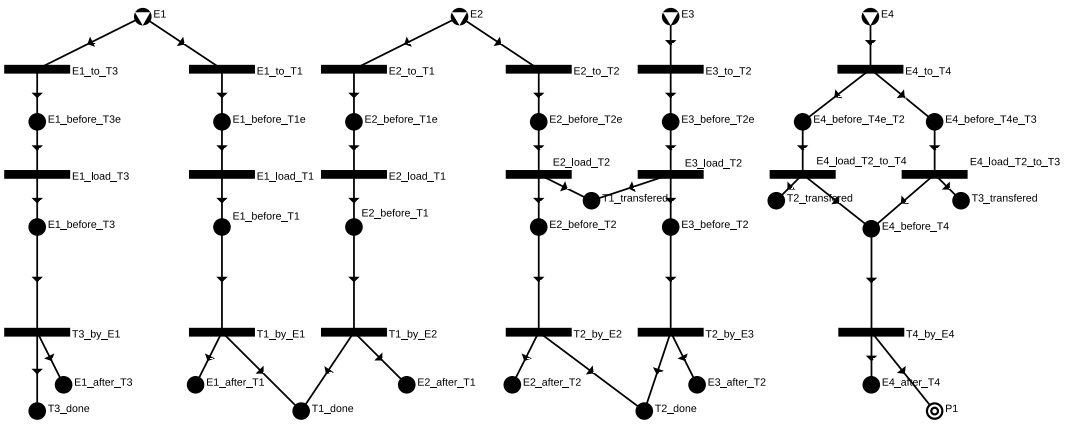
A 2. és 3. Algoritmus alkalmazásával automatikusan generálható egy NIS stratégiát megkö-

vetelő gyártás ütemezési problémát leíró P-gráf struktúra. A bevezetett feltöltési műveletek és a módosított átváltási műveletek révén a berendezések működése szinkronizálható a NIS stratégia feltételeinek megfelelően. A modell működését a bevezetőben definiált példa feladat megoldásával kerül szemléltetésre, amely részleteit a következő fejezet tartalmazza.

3.2.3. A példa feladat megoldása

Az előző fejezetben bevezetett modell generáló lépések alapján előállításra kerül a bevezető részben definiált gyártás ütemezési feladat P-gráf modellje NIS tárolási stratégia esetén.

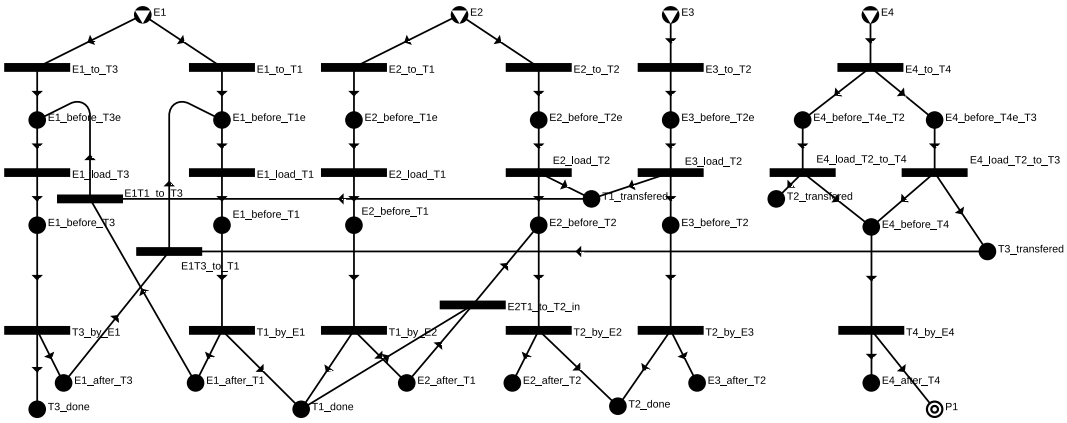
Az első lépésben felvételre kerülnek az erőforrások ($E1$ - $E4$), a feladatoknak megfelelő köztes ($T1\_done$ - $T4\_done$) és végtermékek ($P1$), valamint a 3.1. táblázatnak megfelelően a feladat végrehajtását leíró részgráfok. Egy ilyen részgráfban megtalálható a berendezés feladathoz való hozzárendelésére, a berendezés feltöltésére és a feladat megoldására vonatkozó műveletek. Ha egy feladatnak több előfeltétele is van a recept alapján, akkor minden előfeltételhez külön feltöltési művelet kerül létrehozásra. Erre példa a $T4$ feladat, amely csak akkor kezdhető meg, ha $T2$ és $T3$ feladat is befejeződött. A $T4$ feladatot csak $E4$ tudja elvégezni, így a hozzárendelés után ($E4\_to\_T4$) hozzá kell adni a $T2$ és $T3$ feladatoknak megfelelő üres állapotokat ($E4\_before\_T4e\_T2$, $E4\_before\_T4e\_T3$) és feltöltési műveleteket ($E4\_load\_T2\_to\_T4$, $E4\_load\_T3\_to\_T4$). A feltöltések során előállnak az áttöltés befejezését jelző állapotok ($T2\_transfered, T3\_transfered$), valamint a $T4$ feladat $E4$ berendezésbe történő töltése utáni állapot ($E4\_before\_T4$), amelyet 1/2-1/2 arányban hoz létre a két feltöltési művelet. Látható, hogy ez utóbbi csak akkor áll elő teljesen és így csak akkor kezdhető meg $T4$ feladat végrehajtása, ha mindkét feltöltés befejeződött. A recept alapján még szükséges a $T1\_transfered$ állapot kezelése is, amelyet minden olyan feltöltési művelet előállít, amely a $T2$ feladat végrehajtásához kapcsolódik, amelyek a mostani példában $E2\_load\_T2$ és $E3\_load\_T2$ műveletek lesznek. Az első lépés eredményeként felépített struktúra a 3.9. ábrán látható.



3.9. ábra. A modell generálás első lépésének eredménye

A generálás második lépésében a berendezések feladaton belüli átváltásait biztosító műveletek

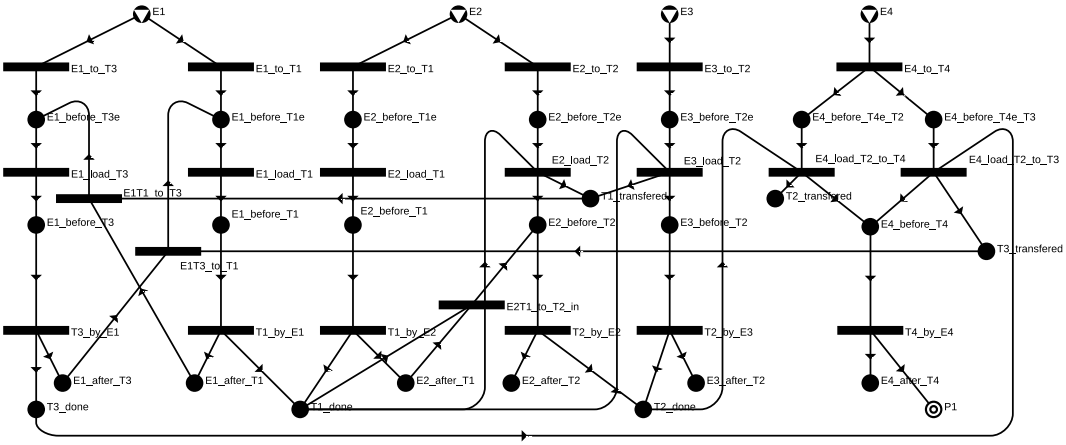
hozzáadása valósul meg. A recept alapján a $E1T1 \rightarrow E1T3$, $E1T3 \rightarrow E1T1$ és $E2T1 \rightarrow E2T2$ átállások értelmezhetők. NIS stratégia esetén minden átállási műveletnek a korábbitól eltérően két előfeltétele van, amely szerint a berendezésnek be kell fejeznie az előző feladatot ($E_i\_after\_T_k$) és az áttöltésnek is meg kell valósulnia ($T_k\_transfered$). A feltételek teljesülése esetén a berendezés átállhat a következő feladatnak megfelelő üres állapotba ($E_i\_before\_T_je$). Ennek megfelelően kerültek generálásra az $E1T1\_to\_T3$ és $E1T3\_to\_T1$ átváltási műveletek. Az $E2$ berendezésen belül megvalósuló váltást külön kell kezelni, mivel a $T1$ és $T2$ feladatok közvetlenül követik egymást a recept alapján, ezért nem történik tényleges áttöltést. Ezt úgy kezeljük, hogy egy olyan $E2T1\_to\_T2\_in$ váltási műveletet hozunk létre, amely követlenül használja fel a $T1\_done$ tokent és az eredménye az $E2$ berendezés már $T2$ feladat végrehajtására feltöltött állapota lesz. A három átváltási művelet hozzáadásával már a struktúra tartalmazza a receptnek megfelelő összes lehetséges végrehajtási sorrendet. A lépés eredményeként létrejövő P-gráf struktúra a 3.10. ábrán látható.



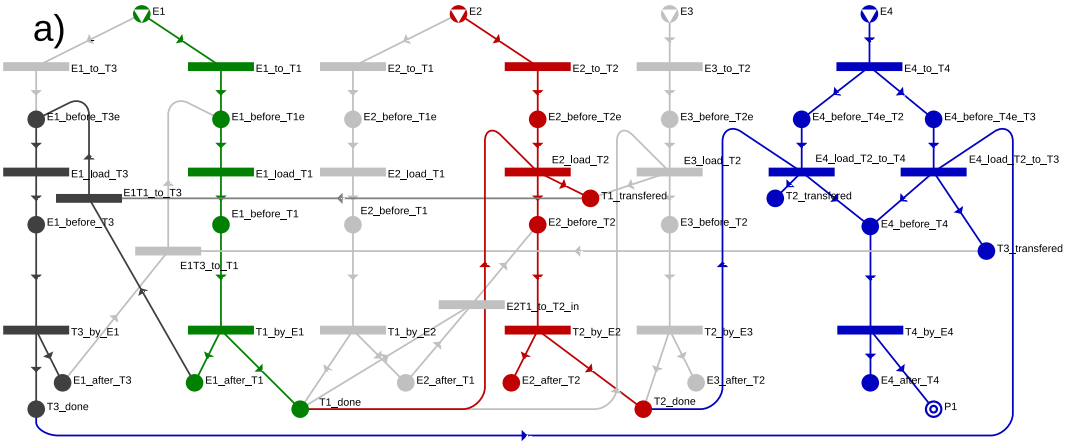
3.10. ábra. A modell generálás második lépésének eredménye

A harmadik lépés során a receptben meghatározott precedenciáknak megfelelő végrehajtási sorrend biztosítása érdekében további élekkel bővítjük a gráfot. A példa feladatban ez két esetben releváns: (1) $T2$ feladat feldolgozása csak akkor kezdődhet, amikor $T1$ befejeződött, ezért a $T1\_done$ új bemenetként kerül bekötésre a $T2$ feladathoz kapcsolódó feltöltést végző műveletekhez ($E2\_load\_T2$ és $E3\_load\_T2$), (2) $T4$ feladat végrehajtása csak $T2$ és $T3$ után kezdődhet, így $T2\_done$ a $E4\_load\_T2\_to\_T4$ művelethez, a $T3\_done$ pedig a $E4\_load\_T3\_to\_T4$ művelethez lesz új előfeltételeként hozzáadva. A probléma maximális struktúrája a 3.11. ábrán látható.

A jelenlegi példa feladathoz az UIS stratégia megoldása során már meghatározásra kerültek a receptnek megfelelő lehetséges végrehajtási sorrendek, amelyeket a 3.2. táblázatban kerültek felsorolásra. A 3.11. ábrán látható maximális struktúrának is tartalmaznia kell ugyanezeket a végrehajtási sorrendeket, amelyeket a 3.12., valamint a mellékletben szereplő B.6.13. - B.6.15. ábrák mutatnak be.



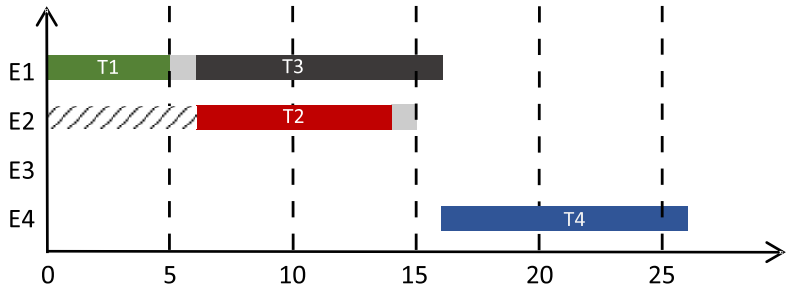
3.11. ábra. A modell generálás harmadik lépésének eredménye



3.12. ábra. Az optimális megoldás strukturális ábrázolása

A P-gráf alapján generált MILP modell megoldásával megkapjuk a feladat optimális ütemezését, amely a 3.12. ábrán látható. Az időváltozók értékei alapján felírható a 3.13. ábrán látható Gantt diagram is.

Az optimális megoldás azonos hozzárendeléseket tartalmaz, mint UIS stratégia esetén, de a műveletek kezdési időpontja a NIS stratégia miatt változik, így a termék előállítási ideje is megnőtt. A megoldásban az $E1$ berendezés egymás után hajtja végre a $T1$ és $T3$ feladatokat, de az áttöltés miatt, várakozni kell az $E2$ berendezésre, ami csak a 6. perctől lesz elérhető. Így az UIS stratégiánál látott megoldástól eltérően $T3$ feladat végrehajtása csak a 6. perctől kezdődhet meg, mivel addig az $E1$ berendezés a $T1$ feladat eredményét tárolja. A $T3$ későbbi kezdése miatt a $T4$ feladatot csak a 16. percben lehet megkezdeni, és így a $P1$ termék a korábbi 25 helyett csak a 26. percben áll elő.



3.13. ábra. A példa feladat optimális megoldásának Gantt diagramon való megjelenítése NIS stratégia esetén

A példa feladat megoldása során bizonyítást nyert, hogy az ismertetett modellezési eljárás képes megfelelően kezelni a NIS tárolási stratégiát.

3.3. Várakozás nélküli ütemezés

A ZW (Zero Wait) stratégia esetén megköveteljük, hogy a gyártási folyamat egy lépése során keletkező köztes termék feldolgozása azonnal, várakozás nélkül folytatódjon a következő lépéssel. A feltétel azt is jelenti, hogy a köztes anyag tárolása nem lehetséges sem külön tárolóban, sem pedig a berendezésekben. Ez korlátozás nagyon szigorú, ezért, ha egy teljes gyártási folyamatra alkalmazzuk, akkor csak nagyon speciális esetekben lesz megoldása a problémának. Ezért a gyakorlatban a ZW stratégiát inkább csak néhány köztes termékre szokták megkövetelni. Az ütemezésben ez úgy jelenik meg, hogy az azonnali feldolgozást igénylő köztes termékek előállítását és feldolgozását végző lépések közvetlenül egymás után hajtódnak végre várakozás nélkül. Természetesen van lehetőség külön berendezésen való végrehajtásra, de akkor is a következő feladat végrehajtását azonnal meg kell kezdeni.

Az eddig tárgyalt tárolási stratégiáktól eltérően a ZW stratégia kezeléséhez nem a P-gráf struktúra módosítására van szükség, hanem az 1.1.2. fejezetben tárgyalt MILP modell újabb feltételekkel való bővítésére, amelyek a következő fejezetben kerülnek bevezetésre.

3.3.1. A várakozás nélküli ütemezés modellezése

A ZW stratégia egy időbeni korlátozás, amely a gyártási folyamat egy lépése során előállított köztes anyag rendelkezésre állása és felhasználása között eltelt időt korlátozza, egész pontosan megköveteli a két időpont egyezését. Az eddig tárgyalt tárolási stratégiák kezelését a megfelelő P-gráf struktúra révén lehetett kezelni, mint például NIS stratégia esetén a feltöltési műveletek modellezése és az áttöltések előfeltételeinek bővítése. Viszont a ZW stratégia főként idő alapú megszorítás, így kezelése strukturálisan nem lehetséges. Ezért a ZW stratégia által támasztott időbeni megkötéseket a TCPNS problémához tartozó MILP modell szintjén kell kezelni. Ez egyben azt is jelenti, hogy a ZW feltételeket tartalmazó gyártás ütemezési probléma megoldásához az UIS vagy NIS stratégiának megfelelő struktúrát is használhatjuk, amelyeket bővítünk az azonnali

feldolgozást biztosító feltételekkel.

Ahogyan már a bevezetőben is említésre került, ha a várakozás nem megengedett, akkor az érintett köztes termék első rendelkezés állásához és a következő lépésben történő felhasználásához tartozó időpontoknak meg kell egyeznie. A MILP modellt ezzel a feltétellel kell bővíteni a ZW stratégia megfelelő kezeléséhez. Mivel feltételezzük, hogy nem a teljes folyamatra vonatkozik a korlátozás, ezért Z legyen azoknak a köztes termékeknek a halmaza, amelyekre érvényesíteni kell a ZW stratégiát, vagyis $z_j \in Z \subset M \setminus P \setminus R$. A MILP modellben a már korábban bevezetett 3.1. egyenlet megadja, hogy egy művelet nem kezdhető el, amíg minden előfeltétele nem teljesül. ZW stratégia esetén pedig egy új feltétel bevezetésével megköveteljük, hogy minden a Z halmazhoz tartozó köztes termék esetén a $t(z_j)$ anyag előállítási és az azt feldolgozó művelet $t(o_i)$ kezdési időpontja egyezzen meg; ezt írja le a 3.2. egyenlet.

$$o_i = (\alpha(o_i), \beta(o_i)) \in \mathsf{o}, \forall m_j \in \alpha(o_i) : t(o_i) \geq t(m_j) \tag{3.1}$$

$$o_i = (\alpha(o_i), \beta(o_i)) \in \mathsf{o}, \forall z_j \in \alpha(o_i) \cap Z : t(o_i) = t(z_j) \tag{3.2}$$

Az új feltétel ugyan megköveteli, hogy a z_i anyag előállítása után azonnal felhasználásra kerüljön, mégis ez a feltétel nem elégséges a helyes eredményhez. A probléma oka a műveletek során előálló anyagok rendelkezésre állási idejét meghatározó egyenletből származik, amely a relaxált modellben a 3.3. egyenlet alapján kerül meghatározásra.

$$o_i = (\alpha(o_i), \beta(o_i)) \in \mathsf{o}, \forall m_j \in \beta(o_i) :$$

$$t(m_j) \geq t(o_i) + x(o_i)tp(o_i) + y(o_i)(tb + tf(o_i)) - tb \tag{3.3}$$

A 3.3. egyenlet csak azt köti ki, hogy a feladat $t(m_j)$ rendelkezésre állási ideje legyen nagyobb, mint a feladat befejezésének ideje. Ez a feltétel általában helyes, de ZW stratégia esetén olyan megoldáshoz vezethet, ahol a megoldó algoritmus az anyag rendelkezésre állásának idejét a feladat befejezéséhez képest egy későbbi időpontra állítja annak érdekében, hogy a 3.2. feltétel teljesülni tudjon. Így matematikailag korrekt, de valóságban egy nem megvalósítható ütemezést kapunk. Ennek kiküszöbölésére egy újabb feltétel bevezetésére van szükség, amely a 3.3. egyenletben megadott alsó korlát mellé a z_j köztes anyagokra egy felső korlátot is megszab, ezzel kikényszerítve, hogy a $t(z_j)$ előállítási idejének a feladat tényleges befejezési ideje kerüljön beállításra. Ezt a felső korlátot mutatja be a 3.4. egyenlet.

$$o_i = (\alpha(o_i), \beta(o_i)) \in \mathsf{o}, \forall z_j \in \beta(o_i) \cap Z :$$

$$t(m_j) \leq t(o_i) + x(o_i)tp(o_i) + y(o_i)(tb - tf(o_i)) + tb \tag{3.4}$$

A Z halmazhoz tartozó köztes termékekre bevezetett új korlátozások révén kezelhető a ZW stratégia által előírt szigorú időbeni megkötés, de érdemes megvizsgálni, hogy Z halmaz miként definiálható. Általában a gyártási folyamat receptje alapján kerül meghatározásra, hogy mely köztes anyagok esetén követeljük meg a ZW stratégia alkalmazását. A receptben definiálva vannak a folyamat lépései és két egymást követő lépés egyértelműen meghatározza a közöttük lévő

köztes terméket. Így a recept szempontjából egymást követő t_i, t_j feladatok felsorolásával lehet definiálni a Z halmazt. Viszont az ütemezési feladatokhoz felírt P-gráf modellben két feladat végzési művelet között számos csomópont (állapot és művelet) szükséges a feladat váltás és a köztes termék kezelésének modellezésére. A TCPNS probléma MILP modellje viszont a P-gráf csomópontjaira határoz meg feltételeket, így felmerül a kérdés, hogy ha t_i és t_j feladatok közötti köztes termékre megköveteljük az azonnali feldolgozást, akkor a P-gráf mely csomópontjait kell a Z halmazhoz hozzáadni a megfelelő eredmény érdekében.

Ennek megválaszolására érdemes megvizsgálni, hogy a kérdéses időpontok melyik P-gráf csomópontoknál kerülnek megadásra. A $t(m_l)$, vagyis a köztes anyag első rendelkezésre állása a t_i feladat befejezésének időpontja, vagyis $t_i\_by\_o_m$ művelet eredményeként kerül meghatározásra. A $t(o_k)$ pedig a t_j feladat végrehajtásának, tehát $t_j\_by\_o_k$ művelet megkezdésének időpontját jelenti. Így könnyen belátható, hogy t_i és t_j feladat végrehajtását leíró műveletek közötti összes csomópontot fel kell venni a Z halmazba a megfelelő működéshez. Itt fontos megjegyezni, hogy mivel egy feladatot több berendezés is végrehajthat, ezért már egy azonnali feldolgozást igénylő köztes anyag esetén a Z halmaz egy komplex részgráfot fog tartalmazni.

A ZW stratégia a TCPNS probléma MILP modelljébe bevezetett új feltételekkel történő kezelését a következő fejezetben egy példán keresztül kerül szemléltetésre.

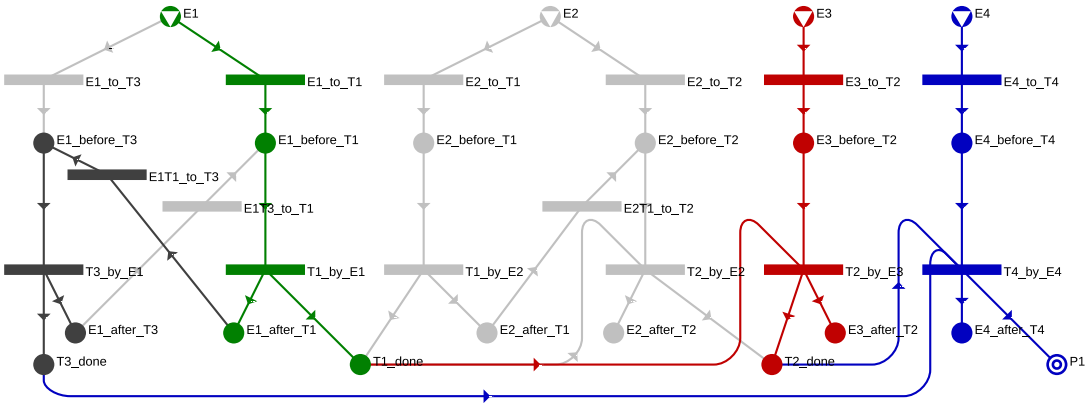
3.3.2. Példa feladat megoldása

Az előző fejezetben a MILP modell kibővítésre került azokkal a korlátozásokkal, amelyek lehetővé teszik a ZW stratégia által támasztott követelményeknek való megfelelést. A fejezet bevezetőjében bemutatott példa feladat először úgy kerül megoldásra, hogy az egész folyamatra megköveteljük a ZW stratégia alkalmazását, vagyis minden köztes termék bekerül a Z halmazba. Majd a második esetben csak a $T4$ feladat végrehajtása előtti köztes anyagok kerülnek a Z halmazba. A feladat megoldásához az UIS stratégiához generált maximális struktúrából indulunk ki (lásd: 3.3. ábra), így a P-gráf generálásának lépései itt nem kerülnek újra bemutatásra.

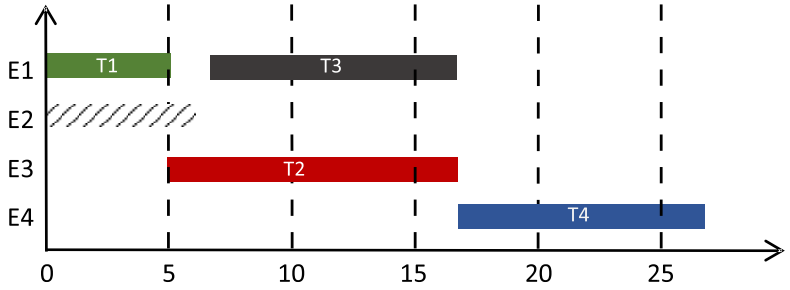
Példa#1 A teljes folyamaton értelmezett ZW stratégia: Az új feltétekkel bővített MILP modell megoldásával megkapjuk a feladat optimális megoldását, amelyet a 3.14. ábra a P-gráf strurktúrában, a 3.15. ábra pedig Gantt diagramon jeleníti meg.

A megoldásban az látható, hogy először $E1$ berendezés hajtja végre $T1$ feladatot, majd a ZW feltételnek megfelelően a $T2$ feldolgozása szünet nélkül folytatódik az $E3$ berendezésben. Mivel $T3$ feladatnak nincs előfeltétele, így az bármikor elkezdhető, ezért annak végrehajtása csak a 7. perctől indul, mivel így $T3$ és $T2$ egyszerre végez, ezért $T4$ feladat megkezdése esetén teljesül a ZW stratégia esetén előírt feltétel mindkét előfeltételre vonatkozóan. Így a termék a 27. percben fog előállni.

Érdemes végiggondolni, hogy a korábbi optimális megoldásban szereplő hozzárendelés időparaméterei megválaszthatók-e úgy, hogy az megfeleljen a ZW stratégiának. A $T1$ feladatnak az 1. percben kellene kezdődnie ahhoz, hogy a $T2$ való átváltás a ZW stratégiának megfelelően történjen. Ilyenkor $T2$ leghamarabb a 14. percben fejeződne be, ellenben $T3$ az $E1$ berendezésen csak a 16. percben. Mivel $T2$ későbbre mozgatása $T1$ hasonló mozgását eredményezné, amely



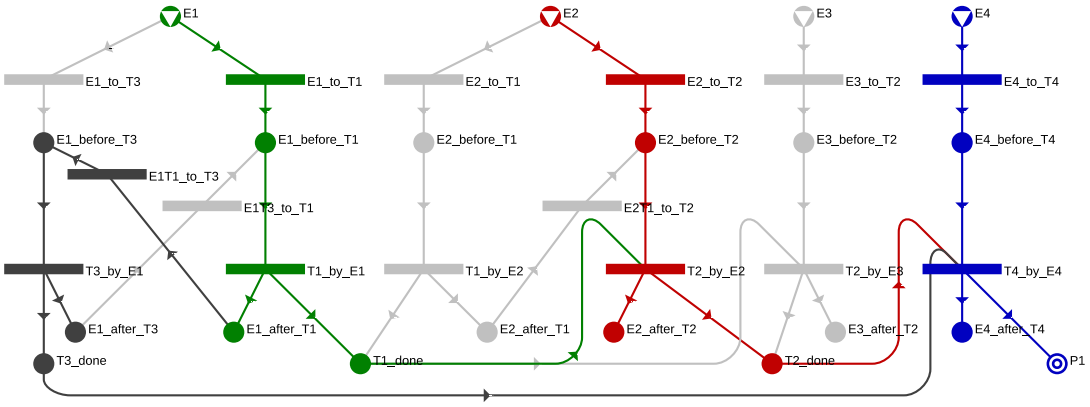
3.14. ábra. A példa optimális megoldása a struktúrában ábrázolva



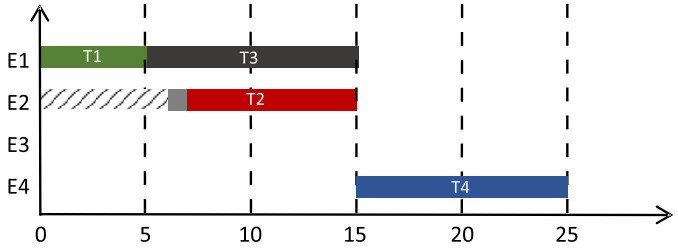
3.15. ábra. A példa optimális megoldása Gantt diagramon ábrázolva

hatására $T3$ is tovább tolódna, így belátható, hogy a $T4$ feladatra való átváltás ZW stratégia mellett nem megvalósítható ennél a hozzárendelésnél. Tehát a 3.15. ábrán látható megoldás a példa feladat optimális megoldása a teljes folyamatra feltételezett ZW stratégia esetén.

Példa#2 Köztes termékek szintjén meghatározott ZW stratégia: Az előző példával ellentétben most csak a $T4$ feladat végrehajtása előtti köztes anyagok esetén követeljük meg a ZW stratégiát. A recept alapján ez a $T2$ és $T3$ feladatok kimenetének azonnali feldolgozását jeleneti. Az UIS stratégiához generált P-gráf struktúrában ez a $T2\_done$ és $T3\_done$ állapotokat érinti, így ezt a két csomópontot fogjuk Z halmazba elhelyezni. Így a feladat újra futtatása után a 3.16. ábrán látható megoldás struktúrát és a 3.17. ábrán bemutatott Gantt diagramot kapjuk eredményül.



3.16. ábra. A 2. példa optimális megoldása a struktúrában ábrázolva



3.17. ábra. A 2. példa optimális megoldása Gantt diagramon ábrázolva

A megoldás struktúra ebben az esetben megegyezik az UIS és NIS stratégiánál kapott struktúrával, viszont a ZW stratégia miatt a feladatok ütemezése változik. A megoldás kulcsa továbbra is a $T3$ és $T2$ feladatok azonos időben való befejezése. Ennél a példánál az UIS stratégiának megfelelő alapmodell miatt a $T2$ feladat megvalósítása $T1$ és $T3$ feladatoktól függetlenül tud úgy megkezdődni, hogy $T3$ feladattal egyidőben fejeződjön be. Ez azért lehetséges, mert $T1$ feladat eredménye egy köztes tárolóban várakozhat, így $T3$ végrehajtása akár azonnal megkezdődhet, továbbá $T2$ feladatra pedig nincs korlátozás arra vonatkozóan, hogy $E2$ berendezés rendelkezésre állása után mikor kezdődjön meg a végrehajtás. Így a termék ebben a megoldásban a 25. percben lesz elérhető.

A két példa alapján látható, hogy a bevezetett feltételek lehetővé teszik, hogy a ZW stratégiát mind folyamat és mind anyag szinten is kezeljük a TCPNS optimalizáló keretrendszer alkalmazása során.

3.4. Vegyes tárolási stratégiák kezelése

A MIS (Mixed Intermediate Storage) tárolási stratégia esetén nem folyamat szinten határozzuk meg a köztes termékek tárolására vonatkozó stratégiát, hanem a gyártási folyamat lépései

során keletkező köztes termékekre szabjuk meg, hogy azokat UIS, NIS vagy ZW stratégiának megfelelően kell kezelni. Ez a megközelítés írja le legjobban a valós gyártórendszerek működését, mivel a korábban vizsgált szigorúbb korlátozásokat megfogalmazó stratégiák, mint a NIS és ZW jellemzően csak egy-egy köztes termék tárolására vonatkoznak, ezért a teljes folyamatra való alkalmazásuk egy kevésbé hatékony ütemezést eredményez. Így felmerül az igény a vegyes tárolási stratégiák használatára, amely egy olyan modellezési eljárást kíván meg, amely képes ezeket a köztes anyagok szintjén egy ütemezési modellben kezelni.

3.4.1. A vegyes tárolási stratégia modellezése

A 3.1. - 3.3. fejezetekben az egyes tárolási stratégiák TCPNS keretrendszerbe való modellezése és annak működése már részletesen bemutatásra került. A vegyes tárolási stratégia kezelése során a kérdés az, hogy lehetséges-e az egyes stratégiákat egy P-gráf modellben alkalmazni?

Ehhez meg kell vizsgálni az UIS, NIS és ZW modellek kapcsolatát. A ZW stratégia UIS vagy NIS stratégiával egy modellben való használata triviális, mivel ZW megkötést tartalmazó problémák megoldása során is az UIS vagy NIS alapmodellből kell kiindulni. Az is könnyen belátható, hogy ha az azonnali feldolgozást igénylő köztes anyagok Z halmaza üres, akkor a probléma megegyezik az alapmodell típusának megfelelő problémával. Így a fő kérdés az UIS és NIS stratégiát megvalósító modellek kapcsolata. A NIS stratégia modellezése során már említésre került, hogy a kiindulási modell megfelel az UIS stratégia esetén használt modellel, amely kibővítésre került az áttöltési műveletek megfelelő kezelése érdekében. Az is könnyen belátható, hogy ha csak az újonnan bevezetett feltöltési műveletekkel, illetve a precedencia élek eltérő kezelésével módosított gráfot vesszük figyelembe, akkor még mindig az UIS stratégia kezelésére alkalmas modellt kapjuk. Ez azt is jelenti, hogy ha a példa feladat megoldására ezt a modellt alkalmazzuk, akkor a megoldás is megegyezne az UIS esetben kapott megoldással. A NIS stratégiának megfelelő működés csak akkor érhető el, ha a váltási művelet előfeltételei között megjelenik a már elvégzett áttöltési művelet is. Tehát az UIS és NIS stratégia esetén alkalmazott modell között a fő eltérés az átváltási műveletek eltérő kezelése.

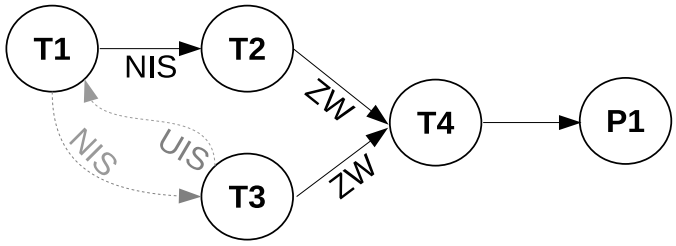
Ez alapján megállapítható, hogy vegyes tárolási stratégia esetén a probléma megoldásához generált P-gráf modell előállítása során az alapmodell felépítése a NIS stratégiának megfelelően kell történnie, annyi módosítással, hogy az átváltásoknál csak akkor vesszük fel előfeltételnek az áttöltés befejezését, ha a kiinduló feladat eredményére a recept NIS stratégiát határoz meg. Majd ez után külön megvizsgáljuk a ZW stratégiára vonatkozó megkötéseket, és a 3.3.1. fejezetben ismertetett módon feltöltjük a Z halmazt. Az így kapott TCPNS problémát a ZW stratégia kezelésére bevezetett feltételekkel együtt oldjuk meg. A bővített MILP modell akkor is helyes eredményt ad, ha Z halmaz üres, így nem szükséges a MILP modell módosítása attól függően, hogy a feladat ír-e elő ZW stratégiát vagy sem. Ez a megközelítés mindig alkalmazható, de természetesen tovább finomítható attól függően, hogy ha nincs előírva NIS stratégia a receptben, akkor elég egy kisebb UIS alapmodellből kiindulni.

Látható, hogy a korábban bemutatott modellek kombinálásával megoldható a vegyes tárolási stratégiát előíró gyártás ütemezési probléma a TCPNS keretrendszer használatával. A következő

fejezetben a korábban is használt példán keresztül kerül bemutatásra az eljárás működése.

3.4.2. Példa feladat megoldása

A vegyes tárolási stratégia modellezésének szemléltetéséhez módosítjuk a bevezetőben definiált példát úgy, hogy a receptet bővítjük a tárolási stratégiákkal. Ezt mutatja be a 3.18. ábra.

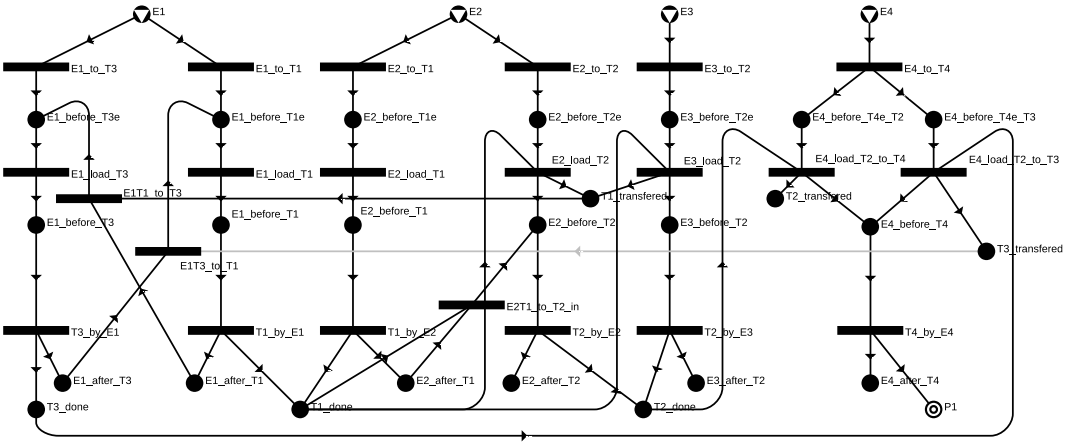


3.18. ábra. A példa feladat tárolási stratégiákkal bővített receptje

A recept alapján $T1$ feladatnál keletkező köztes anyag tárolása nem lehetséges (NIS stratégia), valamint a $T2$ és $T3$ feladatok eredményét azonnal fel kell dolgozni (ZW stratégia). Látható, hogy a receptbe felvételre került a $T1$ és $T2$ feladatok közé két újabb szürkével jelölt nyíl, amely nem a feladatok közötti precedenciát adja meg, hanem segít egyértelművé tenni, hogy ha az ütemezésben a két feladat egymás után következik, akkor a köztes termékre milyen tárolási stratégia lesz érvényes. Így $T1 \rightarrow T2$ sorrend esetén NIS, a $T3 \rightarrow T1$ sorrend esetén pedig UIS stratégia lesz érvényben.

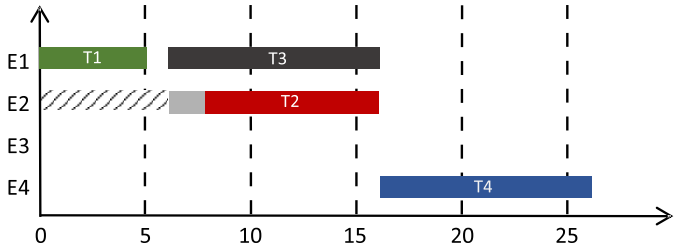
Mivel a gyártási folyamatnak van olyan lépése, amely eredményének tárolására nincs lehetőség, ezért az alapmodell is a NIS stratégiának megfelelő lépések során áll elő. A recept és a lehetséges összerendelések alapján a következő három váltási műveltet kell kezelni: $E1T1 \rightarrow E1T3$, $E2T1 \rightarrow E2T2$, $E1T3 \rightarrow E1T1$. Ezek közül az elsőnél a kiindulási feladat a $T1$, amely eredményére a NIS stratégia vonatkozik, így ennek megfelelően a feladat váltás csak az áttöltés után lehetséges. A második átmenet a berendezésen belül történik, amely kezelése UIS és NIS esetben azonos módon történik. Az utolsó váltásnál a $T3$ feladat a kiindulás, amelyre nincs megkötés, így az UIS stratégiának megfelelően történik a váltás. A problémához tartozó maximális struktúra a 3.19. ábrán látható, amely nagyrészt megegyezik a NIS stratégiánál bemutatott maximális struktúrával. Az eltérés szürkével látható, amely él elhagyásával a $E1T3 \rightarrow E1T1$ váltás már az UIS stratégiának megfelelően történik.

A $T2$ és $T3$ feladatok befejezése után keletkező köztes termékek feldolgozását azonnal meg kell kezdeni, így $T2$ és $T3$ feladat befejezésének és $T4$ feladat megkezdésének egy időpontra kell esnie. Ehhez a gráf megfelelő csomópontjait fel kell venni a Z halmazba, amely a ZW stratégiának megfelelő végrehajtást biztosítja. A 3.4.2. fejezetben az alapmodell az UIS stratégiának megfelelően került felépítésre, így ott elég volt a $T2\_done$ és $T3\_done$ felvétele a Z halmazba. Viszont ennél a példánál az eljárás a NIS stratégiának megfelelő modellből indult, ezért a $E4\_before\_T4$ állapotot is fel kell venni a Z halmazba a megfelelő működéshez.



3.19. ábra. A példa feladat maximális struktúrája

A példa megoldásával megkapjuk a probléma optimális ütemezését a 3.18. recept által definiált tárolási stratégiák esetén. A megoldás struktúra megegyezik az NIS esethez tartozó optimális megoldás struktúrájával, míg a megoldásban szereplő időparaméterek alapján felrajzolt Gantt diagram a 3.20. ábrán látható.



3.20. ábra. A példa optimális megoldása Gantt diagramon ábrázolva

A megoldásban $E1$ berendezés a $T1$ feladat befejezése után tárolja a köztes terméket, amíg $E2$ berendezés elérhetővé nem válik. A 6. percben megtörténik az áttöltés, így $E1$ megkezdheti $T3$ feladat végrehajtását is. Annak érdekében, hogy $T2$ és $T3$ feladatok végrehajtása egyszerre érjen véget, az $E2$ berendezés a feltöltés után még a 8. percig várakozik, és csak akkor kezdi meg a $T2$ feladat feldolgozását. Ez azért lehetséges, mert a NIS stratégia által előírt feltétel a köztes anyag áttöltésével már teljesül és nem szabályozza a következő feladat megkezdését. Így válik lehetségessé a $T4$ feladatot megelőző köztes termék esetén a ZW stratégia teljesítése. A feladat optimális ütemezése során a termék a 26. percben áll elő.

A példa feladat megoldása során bizonyítást nyert, hogy a vegyes tárolási stratégiát alkalmazó gyártási folyamatok ütemezése megoldható a TCPNS keretrendszer használatával.

3.5. A fejezet rövid összefoglalása

A második tézis során a 2. fejezetben bevezetett új ütemezési módszerhez kapcsolódó modellezési eljárás került kibővítésre az irodalomban már jól ismert tárolási stratégiák kezelésével. A gyártó folyamatok esetén a tárolási stratégia határozza meg a gyártás lépései során keletkező köztes termék tárolására vonatkozó szabályokat. A korlátozás vonatkozhat mennyiségre (UIS és NIS stratégia) és időre (ZW stratégia), de gyakorlati problémák esetén a vegyes megoldások (MIS stratégia) is jellemzőek.

Az előző fejezetben bevezetett alapmodell olyan műveletek és állapotokkal kerül bővítésre, amely révén ezek a tárolási stratégiák külön és egy modellben is kezelhetők. A bővített modell generálása szintén automatizálható, amelyet megvalósító algoritmusok is formálisan megadásra kerültek.

A tárolási stratégiákkal bővített ütemezési feladatok megoldását egy szemléltető példán keresztül mutattam be, amely alapján jól követhető a TCPNS modell változása a köztes anyag tárolására vonatkozó megkötések függvényében.

3.5.1. A fejezethez tartozó tézis

Új modellezési módszereket dolgoztam ki a tárolási stratégiával bővített gyártásütemezési feladatok időkorlátos folyamathálózat szintézis (TCPNS) problémaként való felírására. [24], [25]

(a) Meghatároztam az UIS, NIS, ZW és MIS tárolási stratégiával bővített gyártás ütemezési feladatok leírását időkorlátos folyamat-hálózat szintézis feladatként.

(b) Algoritmust dolgoztam ki, mely az összes lehetőséget tartalmazó szuperstruktúra generálása során a struktúrába építi azokat a logikai korlátokat, amelyek a tárolási stratégiából következnek.

(c) Az új ütemező módszer alkalmazhatóságát egy példa feladat megoldásával igazoltam.

3.5.2. A fejezet témaköréhez kapcsolódó publikáció

Nemzetközi folyóiratcikk

- Frits Márton és Bertók Botond: Process scheduling by synthesizing time constrained process-networks, kiadvány: Computer Aided Chemical Engineering, vol 33, oldal 1345-1350 (2014) (IF=0.54) [24]
- Frits Márton és Bertók Botond: Scheduling custom printed napkin manufacturing by P-graphs, kiadvány: Computers & Chemical Engineering, vol 141, oldal 107017 (2020) (IF=3.845) [25]

4. fejezet

Terepi munkavégzés ütemezése folyamatszintézis problémaként

A magas színvonalú szolgáltatások nyújtásához elengedhetetlen, hogy az infrastruktúra üzemeltetője rendszeresen karbantartsa a hálózatát, javítsa a felmerülő hibákat és elvégezzen minden olyan szakmai munkát, amely a szolgáltatás zavartalan működéséhez szükséges. Ehhez megfelelő tanúsítvánnyal, szakértelemmel és helyismerettel rendelkező szakemberek kellenek. A feladatok kiosztása során biztosítani kell, hogy azok olyan szerelőhöz legyenek hozzárendelve, akik rendelkeznek a megfelelő kompetenciákkal és tárgyi eszközökkel. Manapság az ügyfelek számára elfogadhatatlan, ha egy tevékenység várható megkezdését csak több órás intervallum pontossággal tudjuk előre meghatározni. A szakemberek pontos érkezési idején túl elvárás az azonnali értesítés minden esetleges változásról is. Az ilyen és hasonló ügyféligényeknek kiszolgálása érdekében olyan optimalizáló eljárásokra van szükséges, amelyek képesek nagyszámú feladat ütemezésére a szerelőkkel való összerendelésre vonatkozó komplex szabályrendszer kezelése mellett.

A terepi munkavégzés (Field Service Operation, FSO) ütemezését általában számítógépes rendszer (Field Service Management System, FSMS) is támogatja, amelybe rögzítik a bejövő feladatokat és kezelik a végrehajtás folyamatát. Az ilyen szoftverek jellemzően egy általános ütemező modult használnak, ami rendszerint számos paraméterrel konfigurálható, amely művelet nagy szakértelmet kíván. A konfigurációt a rendszer telepítésekor végzik, amely során próbálják a működést az adott vállalat igényeire szabni. Ez általában csak részben valósítható meg, ezért a szervezetnek is alkalmazkodnia kell a szoftverben megvalósított üzleti folyamatokhoz. Egy már konfigurált ütemező modul hatékonysága csak hosszabb távon értékelhető. A nem megfelelő hatékonyság, vagy a gyakorlatban nehezen kivitelezhető végrehajtási tervek szükségessé tehetik a paraméterek további módosítását, vagy szélsőséges esetben az automatikus rendszerfunkciók letiltását és a manuális tervezésre való visszaállást. Az ütemezés minősége a kézi, korábbi tapasztalatokon alapuló tervezés esetén nem ismert, de ha ez könnyebben megvalósítható, vagy jobban kezeli a szerelő csoportok egyedi igényeit, akkor a vállalat azt jobbra értékeli, mint egy elméletileg jobb (pl.: olcsóbb), de a gyakorlatban nehezebben megvalósítható végrehajtási tervet.

Ebben a fejezetben egy hibrid megoldás kerül bevezetésre a terepi munkavégzés ütemezési probléma megoldására, amely egy projekt együttműködés során meghatározott valós, ipari követelmények alapján került megtervezésre. A kidolgozott két optimalizálási modell iteratív megoldása révén lehetővé vált a nagyméretű problémák kezelése. Első lépésben egy lineáris programozási (LP) modell megoldásával a feladatokat diszkrét időintervallumokhoz és szerelőcsapatokhoz rendeljük a rendelkezésre álló kapacitások és a szükséges feltételek figyelembe vétele mellett. A lépés eredményeként a probléma kisebb részproblémákra bomlik, amelyeket már az időkorlátos folyamathálózat szintézissel (TCPNS) oldunk meg, amely révén előáll a feladatok pontos végrehajtási sorrendje. A TCPNS keretrendszer, valamint annak használata ütemezési feladatok megoldására a 2. fejezetben került ismertetésre. A módszer nagy előnye, hogy a gráf reprezentáció révén hatékonyan lehet modellezni a felmerülő egyedi igényeket, ezért érdemes megvizsgálni a TCPNS használati lehetőségeit a terepi munkavégzés ütemezéséhez kapcsolódóan.

4.1. A terepi munkavégzés ütemezése

Az egyetem és egy áramszolgáltató együttműködése során került megfogalmazásra a terepi munkavégzés (Field Service Operation, FSO) ütemezését leíró probléma, amely megoldására egy hibrid optimalizáló eljárás került kifejlesztésre, ami nagy mennyiségű feladat esetén is képes meghatározni a szerelő csapatok pontos munkatervét. Az FSO probléma megoldása során keressük azokat a szerelő csapatokat, amelyek megfelelő képességekkel és tárgyi feltételekkel rendelkeznek a feladatok elvégzéséhez.

A probléma megoldására egy kétlépcsős iteratív eljárás került kidolgozásra, ahol először egy diszkrét intervallumok alapján dolgozó relaxált modell segítségével dekomponáljuk a problémát, majd az egyes intervallumokat időkorlátos folyamathálózat szintézis (TCPNS) modell alkalmazásával ütemezzük [25]. A modellek ismertetése előtt bemutatásra kerülnek a feladatok és szerelő csapatok összerendelését meghatározó fogalmak és paraméterek.

A terepi munkák ütemezési probléma esetén minden t_i $(i = 1 \dots N)$ feladathoz meg kell határozni, hogy a rendelkezésre álló g_j $(j = 1 \dots M)$ szerelő csapatok közül melyik és pontosan mikor fogja azt elvégezni. Az FSO probléma definiálja a feladatok és szerelő csapatok esetén kezelt tulajdonságokat és a megfeleltetési eljárást, amely szükséges egy a gyakorlatban is kivitelezhető ütemezéshez.

Fontos paraméter a szakismeret (Skill), amely magába foglal minden olyan tudást és tapasztalatot, amely egy szerelőt jellemez. Ez lokáció független tulajdonság, de lehet lejárati dátuma, amelyet a hozzárendelés során figyelembe kell venni. Hasonló fogalom a képesség (Ability), amely azt határozza meg, hogy a csapat milyen feladatokat képes elvégezni. A munkaterületet zónákra osztjuk, ahol az egy zónán belül elvégezhető feladatokat a szerepkör (Role) határozza meg, amely egy zóna és egy képesség párral adható meg.

Egy feladat végrehajtási idejére vonatkozó korlátok alapján három csoportot különböztetünk meg:

– Fix időpontos feladatok: a feladat kezdésének időpontja már a rögzítéskor meghatározásra

kerül, amely az ütemezés során nem módosítható.

- Időablakos feladatok: a legkorábbi lehetséges kezdés és egy időablak kerül definiálásra a rögzítés során, amelyen belül a feladat kezdése szabadon ütemezhető, viszont az intervallumon belül a végrehajtásnak be is kell fejeződnie

- Határidős feladatok: jellemzően a kisebb prioritású karbantartási feladatok, amelyeket havonta, negyedévente vagy évente kell elvégezni és a rögzítéskor általában egy tágabb határidő kerül megadásra.

A feladatokhoz rögzítéskor egy feladat típus kerül hozzárendelésre, amely egyértelműen meghatározza a végrehajtáshoz szükséges személyi és tárgyi feltételeket, valamint a feladat kezdeti prioritását, amely az ütemezési folyamat során dinamikusan változik. A feladat típus által definiált feltételek a következők:

- A feladat megfelelő végrehajtásához szükséges szakértők száma, amely definiálja a lehetséges minimális, maximális és optimális számot is. Az optimálistól való eltérést hosszabb végrehajtási idővel vagy extra költséggel büntetjük.

- Terület specifikus szerepkörök, amely szükségesek az adott feladat végrehajtásához (olvasó, szerelő, mérőóra cserélő stb.)

- A feladat elvégzéséhez szükséges szakismeret. Az elvárt készségek kielégítése érdekében a különböző szakismerettel rendelkező végrehajtókat csapatokba szervezzük.

- A feladat elvégzéséhez szükséges speciális eszközök, mint például mérőműszer, autó, daruskocsi stb.

A feladat sablon, a helyszín és az időparaméterek megadásával a feladat már egyértelműen definiálva van. Fontos megjegyezni, hogy a szerelő csapatok összeállítása az ütemező modultól függetlenül történik, bemenetként már a kész csapatok kerülnek átadásra. Az ütemezés során csapat szintű tulajdonságokkal dolgozunk, amit a csapattagok egyéni tulajdonságai alapján határozunk meg, amelyek a következők:

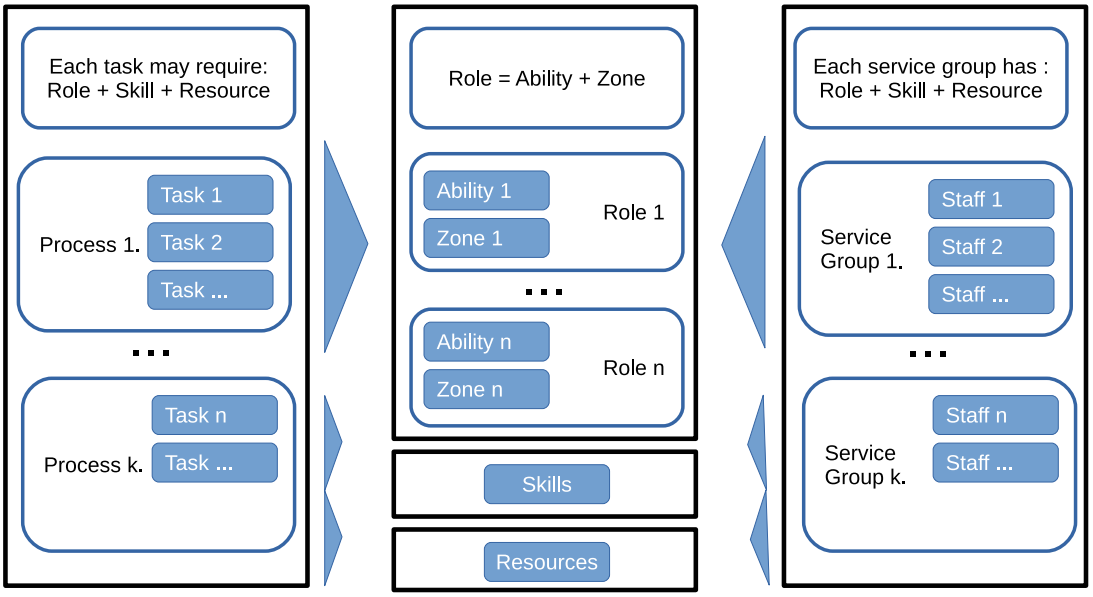
- képességek, amik szükségesek a feladat elvégzéséhez (olvasó, szerelő, mérőóra cserélő) és csak bizonyos zónákban érvényesíthetők;

- szakismeret;

- eszközökre és anyagokra vonatkozó feltételek (milyen erőforrásokat rendeltek hozzá és milyen erőforrásokat kezelhet), beleértve a mérőeszközt, a személyi állományt, az autót stb.

A 4.1. ábra a feladatok által támasztott követelmények és a szerelő csapatok képességei közötti összefüggést mutatja be.

A szerelő csapatok szervizkocsival közlekednek, amellyel általában a raktárból vagy valamelyik csapattag házától indulnak a nap elején. A szervizkocsi egyben egy mobil raktár is, amely



4.1. ábra. A feladatok által támasztott követelmények és a szerelő csapatok képességei közötti összefüggés

készletszintje ismert az ütemezés elején. Ugyancsak ismert minden csapat esetén a következő raktár látogatás időpontja. Eddig az időpontig csak olyan feladat rendelhető a szerelő csapathoz, amely eszköz és anyagigénye kielégíthető a szervizkocsi készletszintje alapján. Feltételezzük, hogy a raktár látogatás során képes felvenni minden olyan eszközt és anyagot, amely a már beütemezett feladatok végrehajtásához szükséges. Ezért, a látogatás utáni időszakban, csak a képességekre vonatkozó feltételeket ellenőrizzük, a tárgyi követelményeket már nem.

A terepi munkavégzés optimalizálásának nehézsége egyrészt a probléma méretéből, valamint a lehetséges összerendelésekre vonatkozó számos feltételből ered. A feladat menedzsment rendszerben egyidejűleg akár 2.000 különböző típusú és prioritású feladat is lehet rögzítve. A feladatok nagy száma mellett a terepen dolgozó szakemberek száma is aránylag magas, a kapcsolódó projekt leírásban például 170 leolvasó, 190 szerelő és 240 hálózati karbantartó került feltüntetésre. A tervezéskor megközelítőleg 50 csapat napi feladatvégzését kell ütemezni, amely során a tervezési horizont a rövidebb 3-5 naptól, a hosszabb, akár 30 napos periódusig dinamikusan változhat.

Az optimalizálás során olyan célok kerültek megfogalmazásra, mint a hatékonyság és az ügyfelek elégedettségének növelése, valamint a költségek csökkentése. Viszont ezen célok általában konkurensek, ezért biztosítani kell a célfüggvényben ezek relatív súlyának finomhangolási lehetőségét. Minden feladat számára meghatározásra kerül egy üzleti érték, annak prioritása és határideje alapján. A fejlesztés során alkalmazott célfüggvényben maximalizáljuk az elvégzett feladatok üzleti értékének és a végrehajtás során felmerült költségek különbségét. A feladat érték meghatározásának menetét, valamint a célfüggvény formális leírását a 4.2. fejezet részletezi.

4.2. Az ütemezés eredményének értékelése

A fejezetben bemutatásra kerülő ütemezési algoritmusok célja a szerelő csapatok feladatainak ütemezése az időkorlátok, a csapatok képességeinek, a raktár készletének, valamint a megvalósítás költségeinek figyelembe vételével. Az ütemezés eredményének értékeléséhez először meg kell határozni a célfüggvényt. A kapcsolódó számításokat a következő fejezet mutatja be részletesen, amely során alkalmazott jelöléseket a 4.1. táblázat foglalja össze.

4.1. táblázat. A feladat üzleti értékének meghatározása során használt jelölések

Jelölés	Leírás
$v(t_i)$	A t_i feladat értéke
$p(t_i)$	A t_i feladat kezdeti prioritása
$pd(t_i)$	A t_i feladat dinamikus prioritása
$td(t_i)$	A t_i feladat határideje
$tf(t_i)$	A t_i feladat várható végrehajtási ideje
$tcf(t_i, g_j)$	A t_i feladat g_j szerelő csapat által történő végrehajtásának fix költsége
PCV	Prioritás költség érték
MP	Maximálisan megengedett prioritás
TPM	Idő prioritás maximum
PDT	A prioritás duplázás gyakorisága
$TimeLeft(t_i)$	A t_i feladat legkésőbbi megkezdéséig fennmaradó idő
$TimePrio(time)$	Az időprioritás szorzót meghatározó függvény
$LatestStart(t_i)$	A t_i feladat legkésőbbi megkezdésének ideje

4.2.1. A célfüggvény

Az ütemezés során minden feladat kiértékelésre kerül, amelyre a továbbiakban, mint a $v(t_i)$ feladat értékére fogunk hivatkozni. Ez az érték függ a $p(t_i)$ kezdeti prioritástól, a legkésőbbi megkezdési időpontig fennmaradó időtől, valamint a prioritás költség értéktől (PCV). Az utóbbi megadja a maximális üzleti értéket a maximális prioritású (MP) feladat elvégzése esetén. Ha az így meghatározott üzleti érték kisebb, mint a végrehajtás várható költsége, akkor az eljárás során a feladat nem kerül ütemezésre, mivel nem éri meg végrehajtani az aktuális prioritási szinten. A feladat prioritása egy véges skálán mozog, a PCV értéke minél magasabb, annál nagyobb fontosságot jelent. Ha a feladat prioritása 0, akkor nem kerül végrehajtásra.

A feladatok értéke a 4.1. egyenlet alapján határozható meg:

$$v(t_i) = \frac{p(t_i)}{MP} TimePrio(TimeLeft(t_i))PCV \tag{4.1}$$

ahol $p(t_i)$ a t_i feladat kezdeti prioritása és $TimePrio(time)$ pedig egy függvény, amely visszaadja az idő prioritás szorzót, amely a határidőhöz közelítve növekszik. Ha a paraméterek alapján a határidők túlságosan hangsúlyosak, akkor felülírhatják a fogyasztói prioritásokat. Ha kevésbé hangsúlyosak, akkor az alacsony prioritású feladatok határidejét szinte elhanyagolják. A $TimeLeft(t_i)$

függvény meghatározza a feladat lehetséges legkésőbbi megkezdéséig hátralévő időt:

$$TimeLeft(t_i) = LatestStart(t_i) - CurrentTime \tag{4.2}$$

ahol

$$LatestStart(t_i) = td(t_i) - tf(t_i) \tag{4.3}$$

a legkésőbbi kezdés időpontja a $td(t_i)$ határidőtől és a feladat várható $tf(t_i)$ végrehajtási idejétől függ.

Minden g_j szerelő csapat és t_i feladat párra a $tcf(t_i, g_j)$ függvény meghatározza a végrehajtás várható költségét, ha t_i feladatot g_j csapat hajtja végre.

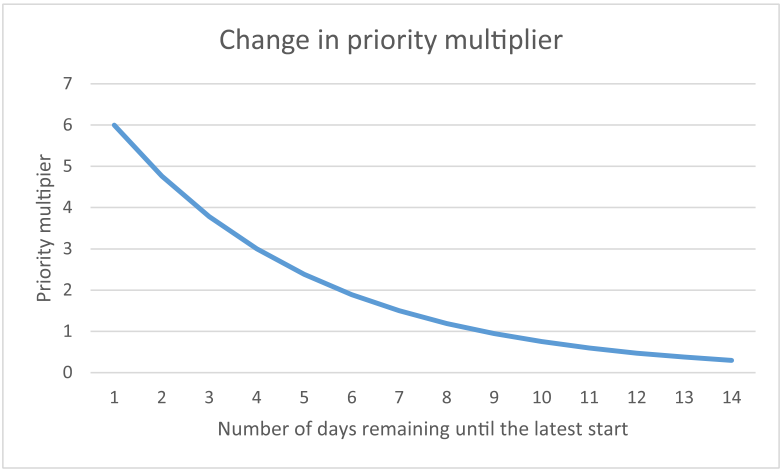
Az optimalizálás során a cél az üzleti érték maximalizálása az adott ütemezési időszakban, vagyis maximalizáljuk a $v(t_i) - tcf(t_i, g_j)$ különbségek összegét minden sikeresen ütemezett feladatra. A fejezetben bemutatott optimalizálás modell célfüggvényét a 4.4. egyenlet írja le.

$$\sum_{\forall t_i \in T} v(t_i) - tcf(t_i, g_j) \rightarrow max \tag{4.4}$$

4.2.2. Dinamikus priorizálás

A feladat típusa alapján meghatározásra kerül egy kezdeti prioritási szint, amely például a karbantartási munkáknál kisebb, mint a hibajavítások esetén. Ezért, ha biztosítani akarjuk a kisebb prioritású feladatok határidő előtti elvégzését, akkor a prioritást dinamikusan növelnünk kell. A határidő közeledtét úgy vesszük figyelembe, hogy az ütemezésben szereplő feladatok prioritását változtatjuk a feladat legkésőbbi megkezdéséig fennmaradó idő függvényében. A dinamikus prioritás meghatározása a 4.5. formula alapján történik:

$$pd(t_i)) = TPM/2^{\frac{TimeLeft(t_i)}{PDT}} \tag{4.5}$$



4.2. ábra. A prioritás szorzó változása a legkésőbbi megkezdésig fennmaradó idő függvényében

A TPM konstans megadja a maximális számot ameddig a prioritás a határidő miatt növelhető, míg a PDT meghatározza, hogy a prioritás milyen gyakran duplázódik. Például, ha a TPM 6 nap és a duplázási idő 3 nap, akkor a prioritás szorzó 6,00, 4,76, 3,78, 3,00, 2,38, 1,89, 1,50, 1,19 és 0,94, amikor 0, 1, ..., 8 nap van hátra a legkésőbbi kezdési időpontig. Tehát amikor már nincs idő (0 nap), akkor 6; három nappal korábban ennek fele, azaz 3. Ha több idő áll rendelkezésre, például 8 nap van hátra, akkor a prioritás szorzó 1-re csökken; lásd a 4.2. ábrát.

Megjegyzés: az ütemezés során a fennmaradó idő percben kerül meghatározásra, így egy napon belül is a korábban elvégzendő feladatnak nagyobb prioritása van.

4.3. Terepi munkavégzés ütemezése TCPNS problémaként

A terepi munkavégzés ütemezési probléma megoldása során a különböző lokációhoz kötődő feladatok kerülnek kiosztásra a szerelő csapatok számára, miközben fenn kell tartani a szakértők kapacitása, a javítási költségek és az utazási idő közötti egyensúlyt. A műszak elején minden szerelő csapat számára elérhetőnek kell lennie az aznapi feladatok végrehajtására vonatkozó tervnek. Az ütemezés során biztosítania kell, hogy minden feladat csak olyan szerelő csapathoz kerüljön hozzárendelésre, amely megfelel a feladat által támasztott feltételeknek úgy, mint a szükséges szakismeret, az érvényes tanúsítványok, valamint a tárgyi és anyagi szükségletek, továbbá a várható végrehajtási és utazási időt is figyelembe kell venni.

A 2. fejezetben már bemutatásra került, hogy miként modellezhető egy ütemezési probléma TCPNS segítségével, amely során a cél egy olyan P-gráf modell építése, amely tartalmazza az összes lehetséges ütemezést, ezzel biztosítva, hogy az optimális megoldás is része a struktúrának. Az így kialakított P-gráfot maximális struktúrának nevezzük, amely tartalmazza az összes lehetséges feladat-csapat összerendelést, valamint az összes lehetséges feladat végrehajtási sorrendet is.

A modellezési eljárás alapgondolata az, hogy minden ütemezendő berendezést erőforrásként kell felvenni, majd egy olyan P-gráf struktúrát kell felépíteni, amely biztosítja, hogy a berendezés, mint erőforrás, bármely potenciális feladat számára rendelkezésre álljon, a problémában definiált feltételeknek megfelelően. A struktúrának biztosítani kell, hogy a berendezést leíró erőforrás mozgása a gráfban csak úgy történhet, ha minden időpillanatban pontosan egy csomópontnál van jelen és pontosan egységnyi mennyiséggel. Így biztosítható, hogy egy berendezés egyidőben csak egy feladathoz kerüljön hozzárendelésre. Általánosságban elmondható, hogy a P-gráf modellben lévő aktivitások az egyes feladatok alternatív erőforrások általi teljesítését írják le, míg az elvégzett feladatokat a folyamattól függően köztes vagy végső célként modellezük.

4.3.1. A terepi munkavégzés ütemezésének P-gráf modellje

A modellezés első lépéseként, a P-gráf csomópont típusait és a probléma elemeit kell megfeleltetni, amelyet a 4.2. táblázat mutat be.

Minden szerelő csapatot és a feladat végrehajtásához szükséges tárgyi eszközt külön erőforrás csomóponttal írunk le. Az ütemezés célja a gráfban végcélként reprezentált feladatok végrehajtá-

4.2. táblázat. A P-gráf csomópontok és az ütemezési probléma elemeinek megfeleltetése

P-gráf csomópont	Az ütemezési probléma eleme
Erőforrás	Szerelő csapatok, eszközök és anyagok
Cél	Elvégzett feladatok, érkezési pont a műszak végén
Aktivitások	Feladat végrehajtás és utazás

sa. További követelményként jelent meg, hogy a szerelő csapat a műszak végére egy meghatározott pontra érkezzen meg, amelyet kötelező feladatként szintén egy végcél típusú csomóponttal modellezünk. A szerelő csapatok fő tevékenységei a feladatok végrehajtása és a helyszínek közötti utazások.

A modell generálás lépései egy kisméretű példán kerülnek szemléltetésre, ahol két szerelő csapatot és négy feladatot kell ütemezni. Először az összerendelést meghatározó tulajdonságok kerülnek definiálásra. A példában megkülönböztetünk három szakismeretet (Repair Electrical Systems, Maintaining Equipment és Electrical Installations), egy képességet (Mechanic), és két zónát (North, South). A képesség és a zónák alapján két szerepkör definiálható: Mechanic-In-Nort, Mechanic-In-South. A példában annyi egyszerűsítéssel élünk, hogy minden csapat csak egy szerelőből áll, így a személyes és csoport szintű tulajdonságok megegyeznek, amelyeket a 4.3. táblázat mutat be.

4.3. táblázat. A csapatok tulajdonságai

Csoport	Szakmai tapasztalat	Szerepkör	Utazási költség
G1	Repair Electrical Systems, Maintaining Equipment, Electrical Installations	Mechanic-In-North, Mechanic-In-South	2000
G2	Maintaining Equipment, Electrical Installations	Mechanic-In-North, Mechanic-In-South	2000

Egy feladat rögzítése során meghatározásra kerül annak típusa, amely egyértelműen definiálja, hogy milyen feltételeknek kell megfelelnie egy szerelő csapatnak ahhoz, hogy elvégezhesse az adott feladatot. A példa során használt feladat típusokat a 4.4. táblázat mutatja be.

4.4. táblázat. Feladat típus definiciók

Feladat típus	Prioritás	Végrehajtási idő	Képesség	Szakismeret
Repair	30	60	Mechanic	Repair Electrical Systems
Maintenance	10	90	Mechanic	Maintaining Equipment
Installation	20	120	Mechanic	Electrical Installations

Ahogyan a 4.4. táblázatban is látható, a feladat típus meghatározza a szükséges képességet, de mivel itt nincs információnk a helyszínre vonatkozóan, ezért a szerepkör a feladat sablonok

esetén még nem értelmezett. A 4.5. táblázat már az ütemezendő feladatokat és azok paramétereit mutatja be.

4.5. táblázat. A feladatok és paramétereik

Név	Feladat sablon	Zóna	Legkorábbi kezdés	Legkésőbbi befejezés
A	Repair	North	08:00	10:00
B	Maintenance	North	-	16:00
C	Installation	North	11:30	-
D	Repair	South	10:00	12:00

Az időparaméterek alapján látható, hogy az A és D feladat esetén időablak, a B feladathoz csak határidő és C feladathoz pedig fix kezdési időpont került meghatározásra. Az egyszerűbb megjelenítés miatt most csak a feladathoz tartozó zóna került megadásra, de a gyakorlatban pontos cím kerül rögzítésre, amely alapján pontos utazási idő számolható. További egyszerűsítés, hogy az indulási és érkezési pontok megegyeznek minden szerelő csapat esetén. A 4.6. táblázat megadja a feladatok közötti utazási időket.

4.6. táblázat. A feladatok közötti utazási idők percben kifejezve

	G1_Home	G2_Home	A	B	C	D
G1_Home	0	-	40	30	10	25
G2_Home	-	0	40	10	15	30
A	40	40	0	35	25	30
B	30	10	35	0	40	20
C	10	15	25	40	0	15
D	25	30	30	20	15	0

A 4.3.-4.5. táblázatok alapján meghatározhatóak a lehetséges összerendelések. Tekintsünk egy példát: az A feladat egy *Repair* típusú munka a $North$ zónában, azért a $Mechanic-In-North$ képesség és a $RepairElectricalSystem$ szakismeret megléte szükséges a végrehajtáshoz, amely a példában csak a $G1$ csapatnál érhető el. Hasonló módon megállapítható, hogy A és D feladatokat csak $G1$, valamint B és C feladatokat mindkét szerelő csapat végre tudja hajtani.

A példa feladathoz tartozó P-gráf struktúra építésének első lépésében a szerelő csapatoknak megfelelő erőforrások ($G1$ és $G2$) kerülnek felvételre, majd az ütemezendő feladatok (A, B, C, D) és a csapatok érkezési pontjai ($G1\_End$, $G2\_End$) végcélként jelennek meg a gráfban. A csapatokat reprezentáló erőforrások felső korlátját 1-re állítjuk, amely azt eredményezi, hogy a csapat nem kerül több feladathoz is hozzárendelésre ugyanazon időben. Ennél a problémánál feltételezzük, hogy az ütemezendő feladatok száma nagyobb, mint amennyit ténylegesen ütemezni tudunk, ezért általánosan a termékek alsó korlátját, vagyis a minimálisan előállítandó mennyiséget 0-ra állítjuk. A felső korlát 1-re állítása pedig biztosítja, hogy minden feladatot maximum egyszer végezzünk el.

A feladatok végrehajtásának üzleti értékét a 4.1. formula alapján számolhatjuk. A példában

szereplő feladatok értékei a 4.7. táblázatban láthatóak. A számolás során a konstansok értékei a következők voltak: Prioritás költség érték (PCV) = 100000, Prioritás max (PM)= 100, Idő prioritás max (TPM) = 10, Prioritás duplázási idő (PDT)= 2.

4.7. táblázat. A feladatok üzleti értéke

Feladat	Prioritás	A legkésőbbi kezdésig hátralévő idő [h]	Idő prioritás	Üzleti érték
A	30	1	7,071067812	212132
B	10	6.5	1,051120519	10511
C	20	3.5	2,973017788	59460
D	30	5	3,535533906	106066

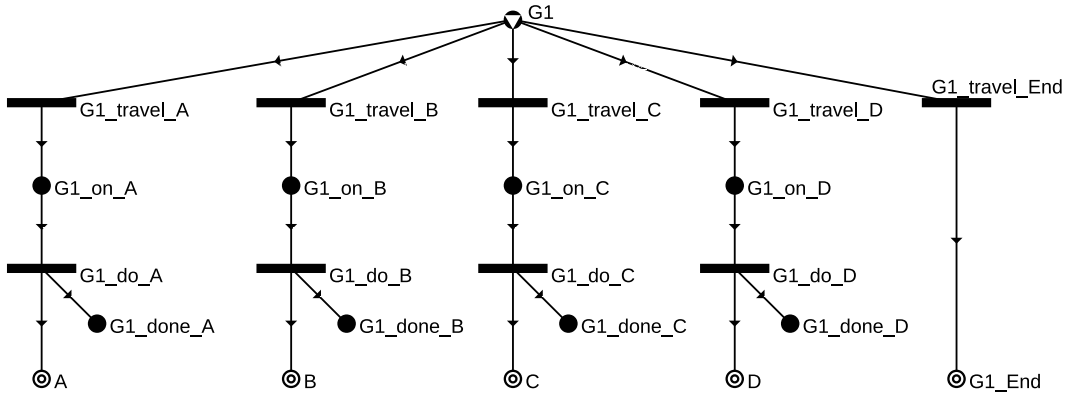
A P-gráf modell segítségével kezelhetőek a kötelezően elvégzendő vagy már előre beütemezett feladatok is. Ebben az esetben a feladatokat leíró végtermékek alsó korlátját is 1-re kell állítani. Erre példa a nap végén az adott helyszínre való érkezés (g_j_End).

A struktúra generálás következő lépésében a végrehajtás folyamatát leíró gráf ágak kerülnek előállításra minden olyan feladathoz, amelyet az aktuálisan vizsgált szerelő csoport el tud végezni. Például a $G1$ csapat el tudja végezni a A feladatot, de ehhez először a helyszínre kell utazni ($G1\_travel\_A$), ha megérkeztek ($G1\_on\_A$), akkor elkezdhetik a végrehajtást ($G1\_do\_A$), amely eredményeként előáll az elvégzett feladat (A) és a csapat készen áll egy újabb feladat elvégzésére ($G1\_done\_A$). Az utazási műveletek esetén ($g_i\_travel\_t_j$) a végrehajtási idő megegyezik a pontos utazási idővel, amelyet a 4.6. táblázat alapján tudunk meghatározni. A feladat végrehajtási műveletek esetén ($g_i\_do\_tj$) a felső és alsó korlátot is 1-re állítjuk, valamint a várható végrehajtási idő alapján kerül meghatározásra az aktivitás fix ideje. Továbbá a feladat típusának megfelelően beállított legkorábbi kezdési és legkésőbbi befejezési időkkel kontrollálható, hogy mikor történjen a végrehajtás. Ezek a paraméterek jellemzően a műszak kezdetétől számolva kerülnek meghatározásra, ami a példában reggel 8 óra. Például a D feladat esetén a legkorábbi kezdés 10 órakor lehetséges, így 120 perc kerül beállításra, valamint 12 óráig be kell fejezni, így határidőnek 240-t kell megadni. A 4.3. ábra bemutatja a $G1$-es csapathoz generált ágakat.

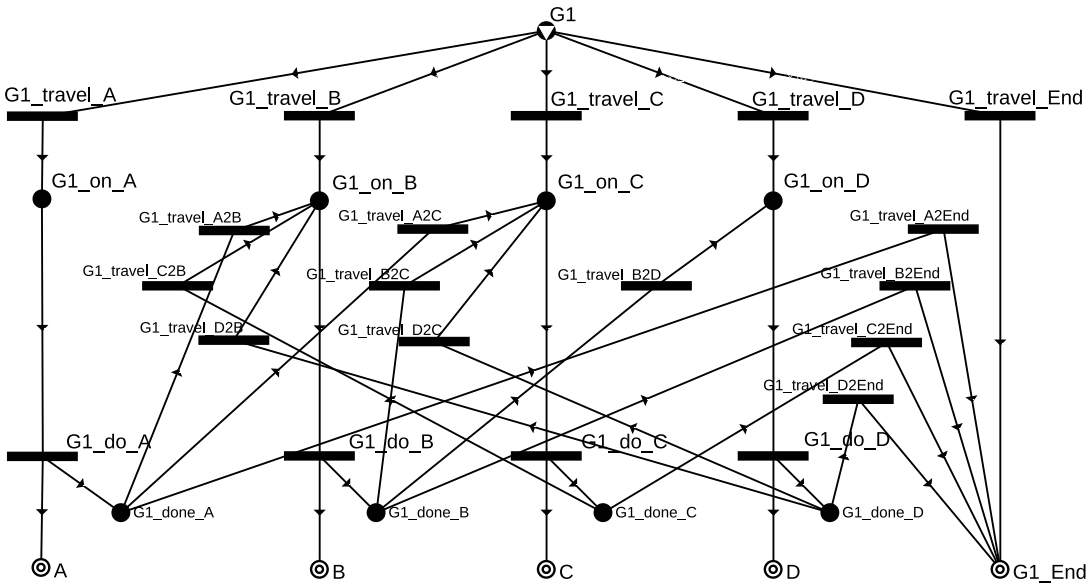
A modell jelenlegi állapotában a $G1$ csapatnak lehetősége van eljutni az összes lehetséges feladathoz (A,B,C vagy D), képes végrehajtani, majd a műveletek után várakozik. Ahhoz, hogy el tudjon jutni további feladatokhoz a gráfot tovább kell bővíteni, hogy a csapatoknak lehetőségük legyen eljutni egy másik feladat helyszínére. Ez új aktivitások ($g_j\_travel\_t_i2t_k$) bevezetésével valósítható meg, amelyek reprezentálják a feladat befejezése utáni várakozó állapotból ($g_j\_done\_t_i$) való eljutást egy másik t_k feladat helyszínére. Az utazáshoz szükséges idő szintén a távolság mátrix alapján meghatározható (lásd. 4.6. táblázat). Az így felépített gráfot mutatja be a 4.4. ábra.

Az ütemezési feladatok megoldásához generált P-gráf modell egy maximális struktúra, amely tartalmazza az összes lehetséges ütemezést. A 4.4. ábrán látható gráf maximális struktúra a $G1$ csapatra vonatkozóan, tehát tartalmazza az összes lehetséges feladat végrehajtási sorrendet. Ezek közül grafikusan ábrázolva mutatnak be néhányat a mellékletben szereplő C.6.16. és a C.6.17.

ábrák.

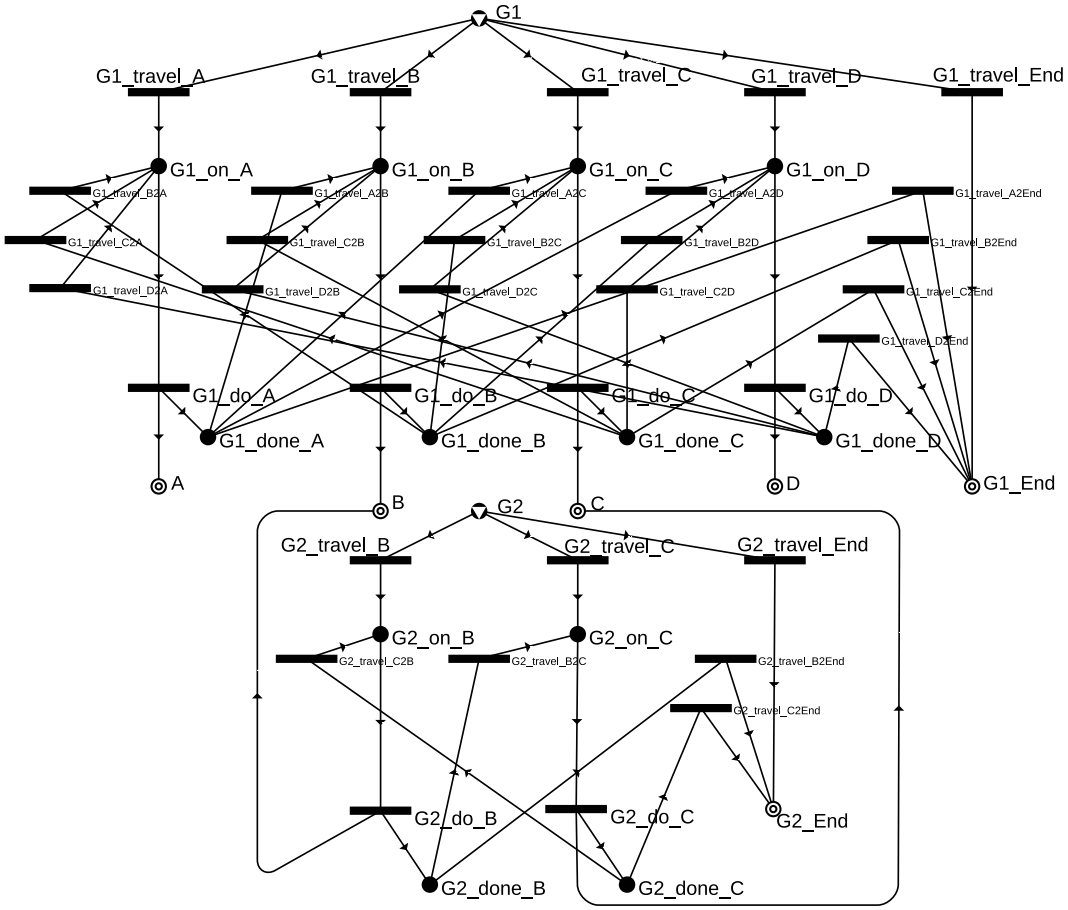


4.3. ábra. A $G1$ feladathoz generált végrehajtást leíró ágak.



4.4. ábra. A feladatok közötti utazásokkal bővített P-gráf modell

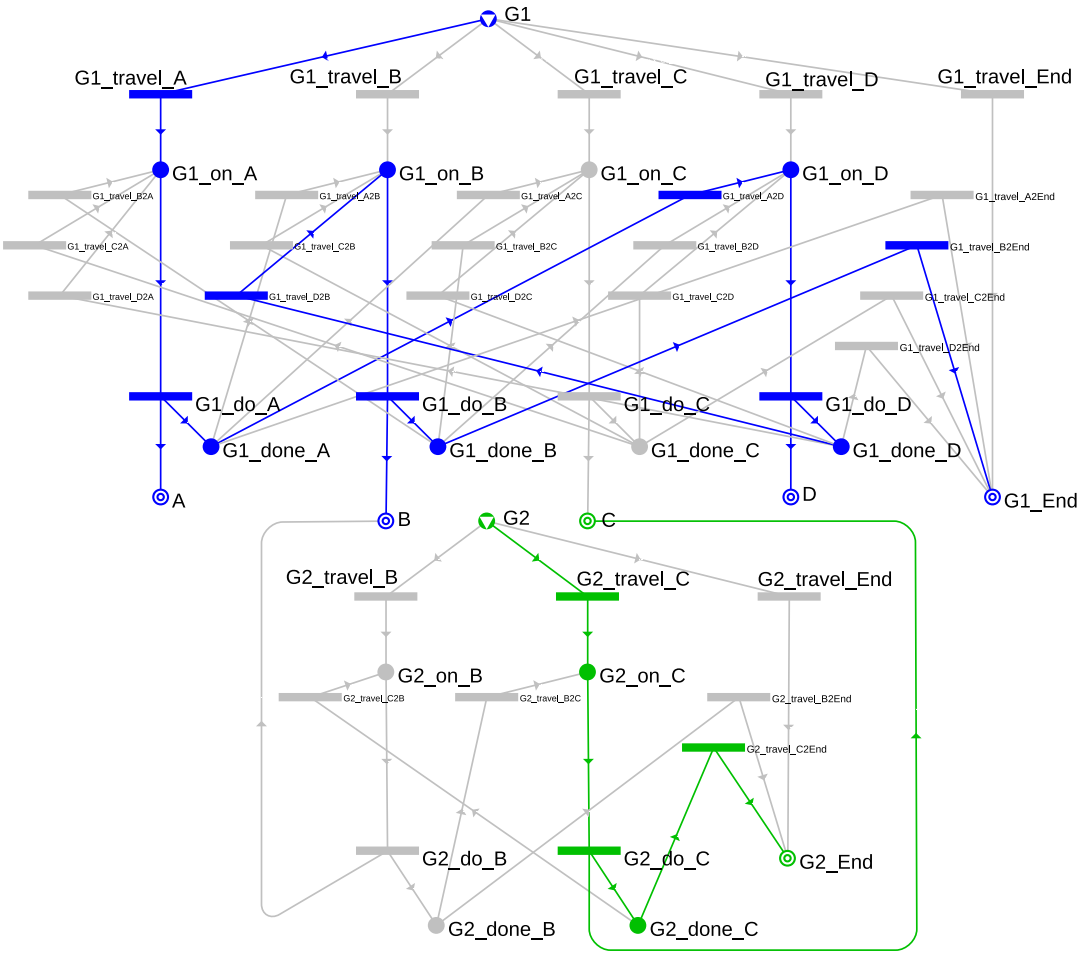
Hasonló lépésekkel generálható a maximális struktúra a $G2$-es csapat esetén is, ahol a B és C feladatok végrehajtását kell modellezni. A teljes feladatra vonatkozó maximális struktúra esetén ügyelni kell arra, hogy ha egy feladatot több csapat is végre tud hajtani, akkor is csak egy a feladatnak megfelelő végcél szerepeljen a gráfban. Így biztosított, hogy minden feladat csak egyszer kerül végrehajtásra. A 4.5. ábrán látható a példa feladat maximális struktúrája, amely már mindkét csapatra vonatkozóan tartalmazza az összes lehetséges ütemezést.



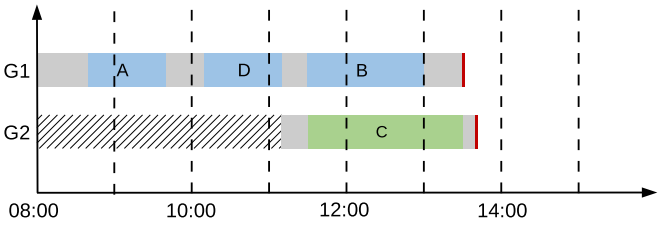
4.5. ábra. A példa probléma maximális struktúrája

A P-gráf alapján automatikusan generálható a vele ekvivalens MILP modell, amely megoldásával megkapjuk a megoldás struktúrát, amely a 4.6. ábrán látható. A megoldásban $G1$ csapat a A, D és B feladatokat hajtja végre, amelyek a gráfban kék színnel láthatóak, míg a $G2$ csapathoz a zölddel jelölt C feladat került hozzárendelésre. Mindkét csapat esetén a színezés mindig az erőforrásból indul majd az utazás és a feladat végzés aktivitásokon keresztül jut el a nap végén az érkezési pozícióba.

Ha a megoldásban szereplő aktivitásokat egy időhorizonton ábrázoljuk, akkor egy Gantt diagramot kapunk, amely egyértelműen meghatározza az egyes szerelő csapatok által elvégzendő feladatokat és azok pontos ütemezését. A 4.7. ábrán látható a példa probléma optimális megoldása alapján felrajzolt Gantt diagramm, amelyen a $G1$ csapathoz tartozó feladatok kékkel, a $G2$ csapathoz tartozó feladatok pedig zölddel kerültek jelölésre, valamint szürkével láthatóak a feladatok között utazások is.



4.6. ábra. A példa probléma optimális megoldása



4.7. ábra. A példa probléma optimális megoldása Gantt diagrammon ábrázolva

4.3.2. Az eszköz és anyagigények modellezése

A bemutatott lépések alapján előállított P-gráf modell képes szolgáltatni a terepi munkavégzés ütemezési probléma optimális megoldását a feladatban definiált komplex hozzárendelési szabályok és a pontos utazási idők kezelése mellett. Viszont további követelményként jelent meg, a

feladatok elvégzéséhez szükséges eszközök és anyagok, valamint a szervizkocsihoz tartozó raktárkészlet kezelése. Minden szerelő csapat esetén ismert, hogy mikor várható a következő raktárlátogatás. A probléma specifikációja alapján a raktárlátogatásig csak olyan feladatokat lehet a csapathoz rendelni, amelyek az aktuális raktárkészlet alapján megvalósíthatók. Ha az aktuális ütemezési periódus a raktárlátogatás után van, akkor az eddig bemutatott modellt használhatjuk, viszont előtte szükséges az eszközök és anyagok modell szintű kezelése. A raktárkészlet kezeléséhez minden szerelő csapathoz új erőforrásként kerülnek felvételre a releváns eszközök és anyagok, amelyek felső korlátját a rendelkezésre álló mennyiségnek megfelelően kell beállítani. Tegyük fel, hogy a $Installation$ típusú feladat elvégzéséhez egy "elektromos mérőóra" szükséges, így ez csak olyan csapathoz rendelhető, amelynek legalább egy elektromos mérő elérhető a szervizkocsiban (vagy az ütemezési időszak a raktárlátogatás utánra esik). Tegyük fel továbbá, hogy a $G2$ csoportnak van egy mérőórája, ezért egy új $G2\_meter$ erőforrást veszünk fel a gráfba, amelynek a felső korlátját 1-re állítjuk. A $G2\_meter$ előfeltétele minden olyan tevékenységnek, amely egy olyan feladat teljesítését ($g_j\_do\_t_i$) írja le, amelyet g_j el tud végezni és egy elektromos mérőóra szükséges hozzá. Az aktuális példában csak a C feladat $Installation$ típusú, így a $G2\_do\_C$ művelethez, mint bemenetet kell bekötni a $G2\_meter$ erőforrást. A 4.8. ábrán látható a $G2\_meter$ kezelésével bővített maximális struktúra.

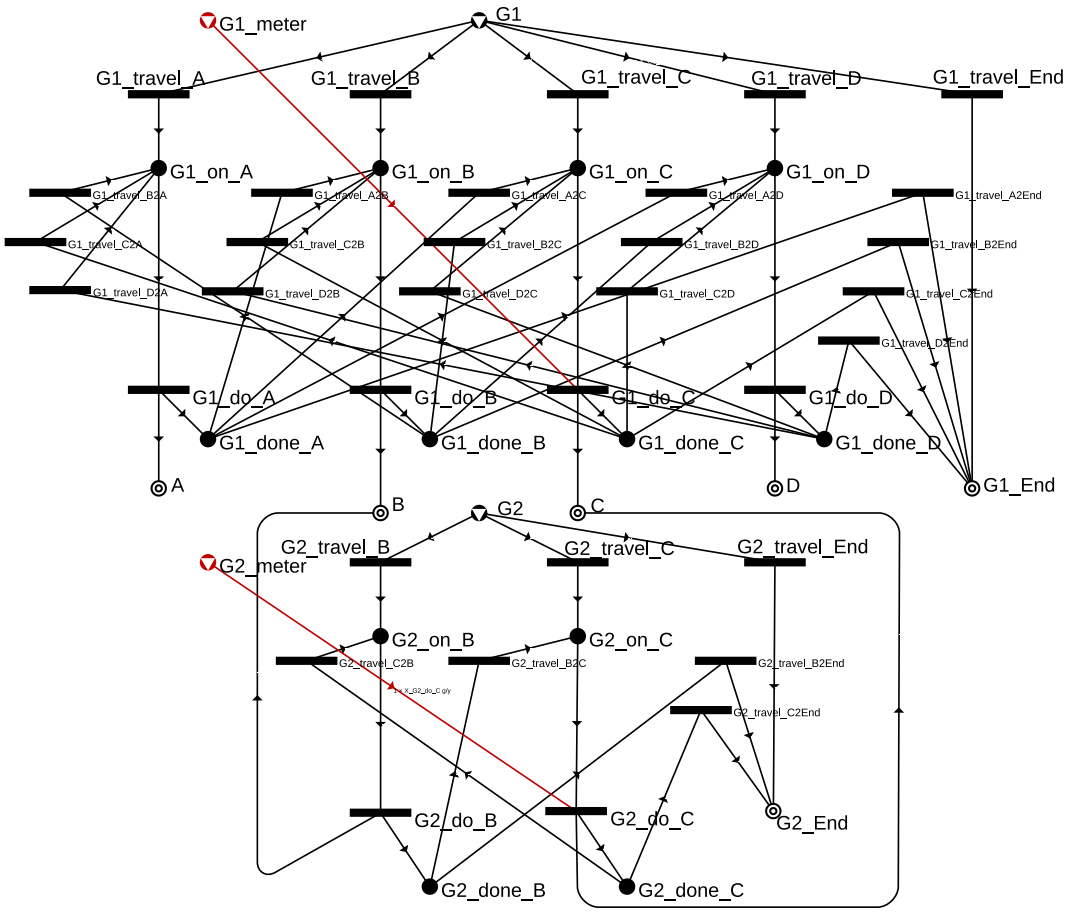
A példa problémában tehát feltételezzük, hogy csak $G2$ csapatnak van merőórája, ezért a korábbi optimális megoldás már nem megvalósítható. A módosított P-gráf modell megoldását a 4.9. ábra grafikusan, a 4.10. ábra pedig Gantt diagrammon ábrázolva mutatja be. Az új eredmény alapján a B és C feladatok ütemezése változott, mivel C feladat végrehajtási feltételeit $G1$ már nem teljesítette. Az ábrán látható, hogy a C feladat megkezdése előtt a $G2$ csapatnak várakoznia kell, mert csak így volt teljesíthető az előírt időkorlát.

A raktárkészlet kezelésére folytonos változókat használunk, amelyek korlátozhatják a feladatok végrehajtását. Ez a megközelítés akkor is működik, ha például egy kábel felhasználását modellezük, ahol az egyes feladatok elvégzése során felhasznált szokásos mennyiség általában nem egész érték.

A mobil raktárkészlet kezelése azonban jó példa arra, hogy a TCPNS keretrendszerrel milyen hatékonyan lehet az új követelményeket kezelni. A szoftver megvalósítás szempontjából a gráf építő algoritmus csak kis mértékű módosításával az ütemezési probléma definíciójában jelentkező új komplex feltétel is kezelhetővé válik.

4.3.3. Az ebédidő modellezése

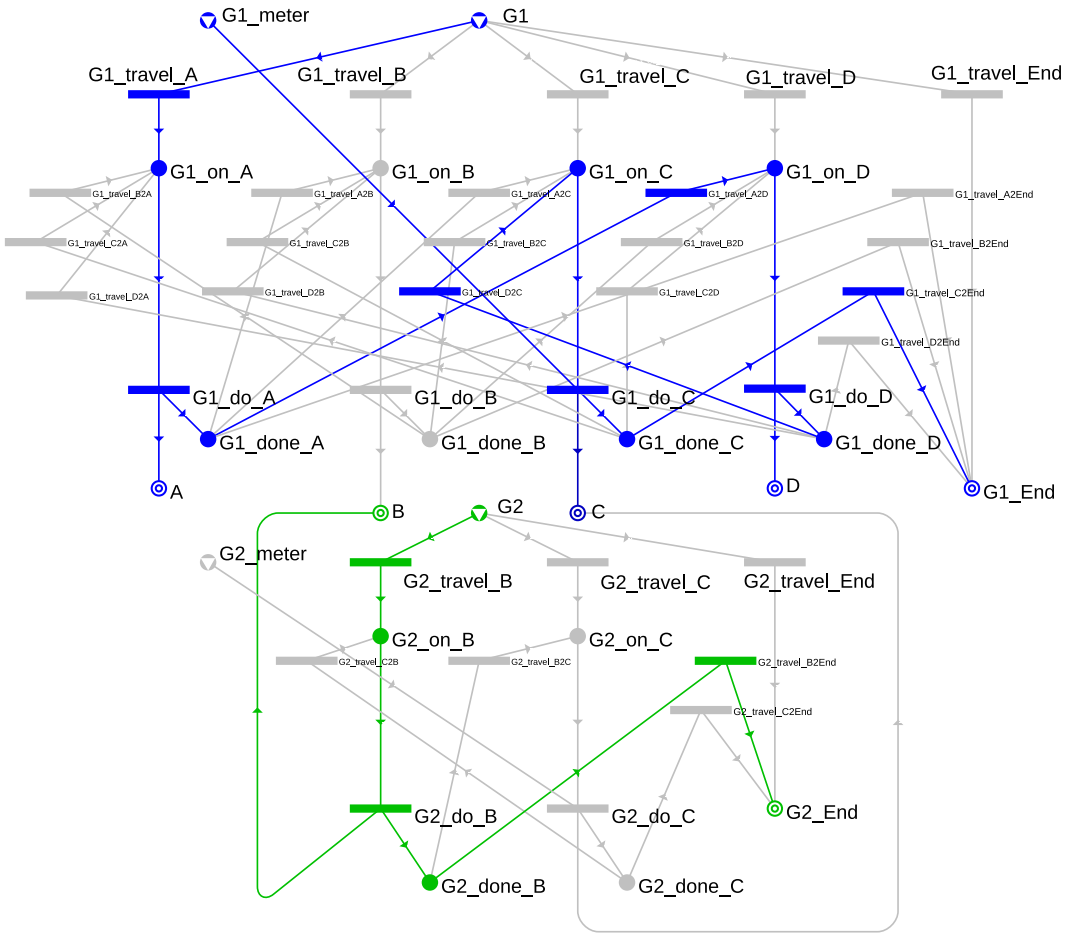
Egyedi követelményként jelent meg, hogy minden szerelő csapat ütemezése során 20 perc ebédidőt kell biztosítani egy előre definiált időintervallumban (pl. 11: 00-13: 00). Az ebédidő kezelését a korábbi 4.5. ábrán bemutatott modell kibővítésén keresztül mutatom be. Tehát a modellnek biztosítania kell az ebédhez szükséges extra időt két feladat elvégzése között a megadott időintervallumban. Ehhez a modellben egy új célként jelenik meg az adott napra vonatkozó ebédidő (g_j_lunch), amelynek mivel nincs üzleti értéke, ezért csak akkor kerül ütemezésre, ha kötelezővé tesszük az alsó korlát 1-re állításával. A következő kérdés, hogy ezt a célt milyen művelet fogja



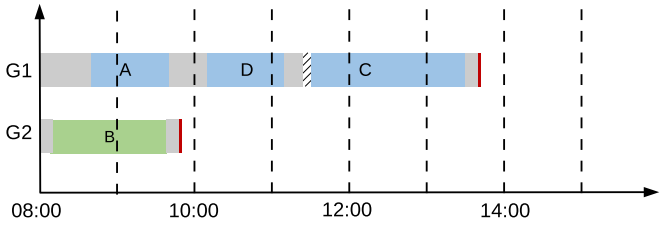
4.8. ábra. Az anyagok kezelésével bővített maximális struktúra

előállítani? Kézenfekvő megoldás lenne, ha ugyanúgy modelleznénk, mint a többi feladatot, így új ágként jelenne meg minden csapat esetén. Mivel az nem ismert, hogy melyik két feladat végrehajtása közé kerülne beszúrásra, ezért minden feladatból elérhetőnek kellene lennie, valamint bármely feladattal folytatódhat a végrehajtás. Ennél a megoldásnál a probléma az utazási idők meghatározásánál jelentkezne. Nézzük egy példát, amikor t_i és t_j feladatok között, közvetlenül t_i utánra terveznénk az ebédszünetet. Ilyenkor az ebéd megkezdéséhez nem kell utazási idővel számolni, mivel t_i helyszínén megoldható. Viszont a tovább lépéshez már szükség lenne a g_k_lunch és t_j közötti utazási időre, ami bár megegyezik t_i és t_j közötti idővel, de g_k_lunch már nem hordoz helyszín információt. Így ez a megoldás csak akkor lehetne járható, ha az ebéd egy előre meghatározott helyen, például egy vállalati menzán történik, mert így előre ismertek lennének a helyszínek és az utazási idők is. Ebben az esetben a távolságmátrixot ki kellene bővíteni a menza és a feladatok közötti távolsággal, így pontosan tudnánk, milyen utazási időre van szükségünk ahhoz, hogy eljussunk a menzába és az ebéd után következő feladatra.

Ehelyett egy dinamikusabb megoldást került alkalmazásra, amely során az utazási műveletek



4.9. ábra. Az anyagok kezelésével bővített modell optimális megoldásának grafikus ábrázolása

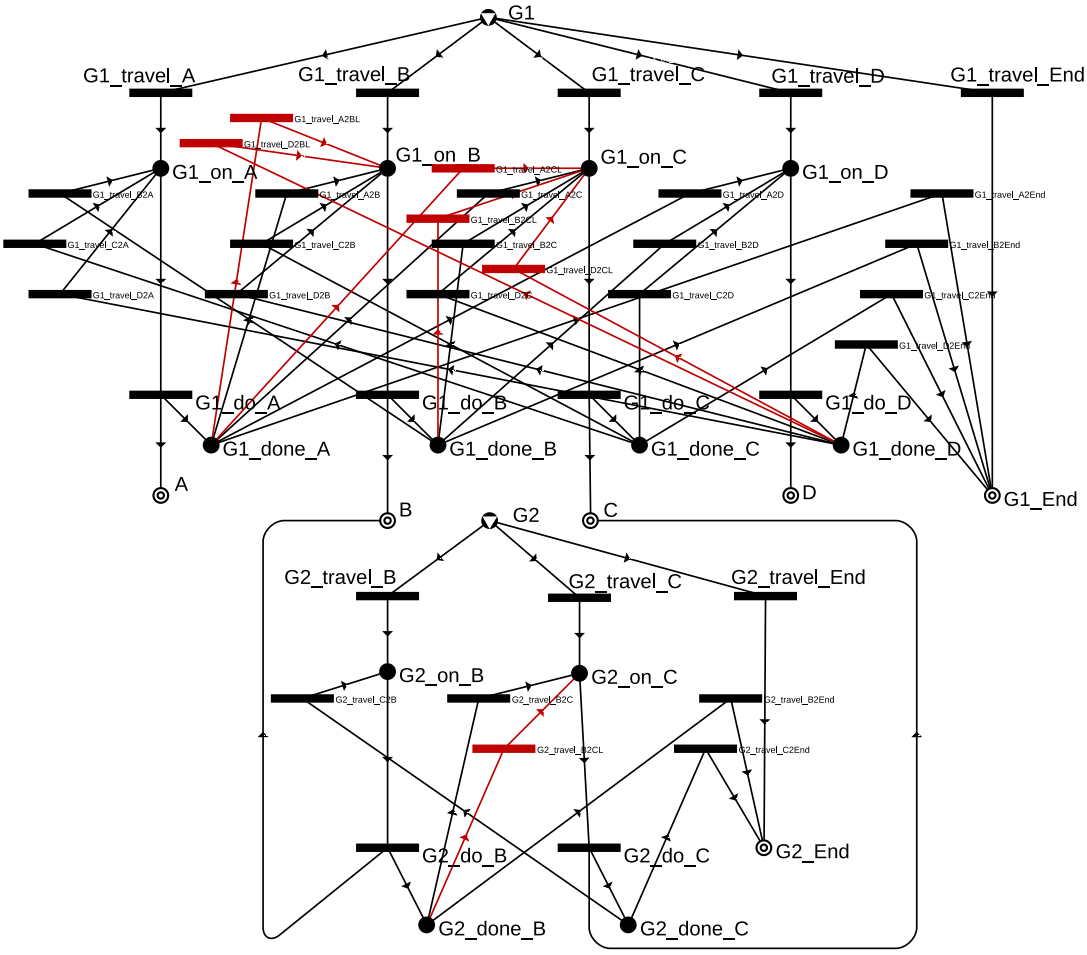


4.10. ábra. Az anyagok kezelésével bővített modell optimális megoldásának Gantt diagrammon való ábrázolása

esetén van lehetőség az extra 20 perces ebédidő beiktatására. Ezért a modellben kétféle utazást leíró csomópontot veszünk fel: az egyik megegyezik a már korábban használttal ($g_j\_travel\_t_i2t_k$), míg a másik esetén ($g_j\_travel\_t_i2t_k\_lunch$) a bemenet szintén az előző feladat elvégzése utáni állapot lesz ($g_j\_done\_t_i$), de ez már két kimenettel rendelkezik, ahol az egyik a következő feladat

($g_j\_on\_t_k$), a másik pedig az ebédidőt reprezentáló végcél (g_j_lunch). Ez utóbbi esetben az utazási idő a távolságmátrixban megadott érték plusz 20 perc lesz. Ez a megközelítés biztosítja, hogy pontosan egy utazás során 20 perccel hosszabb menetidővel kalkuláljunk.

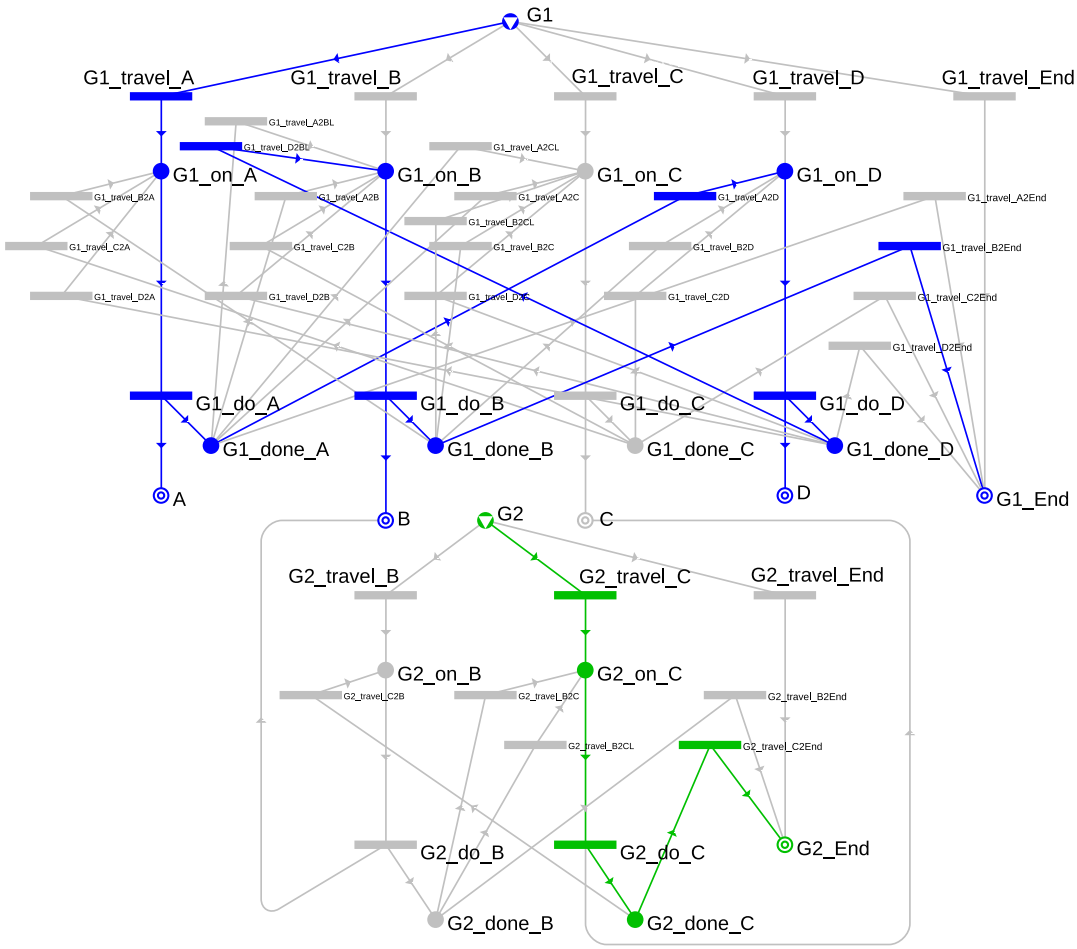
A példa feladatot úgy módosítjuk, hogy 11:00 és 13:00 óra között biztosítani kell minden csapat számára egy 20 perces ebédszünetet. Először érdemes megvizsgálni minden feladat pár esetén, hogy az időparaméterek alapján az utazás beleesik-e a meghatározott intervallumba. Például a D feladat legkésőbbi kezdési ideje 11:00, ezért előtte nem lehetséges beiktatni az ebédidőt, így minden olyan utazás esetén, amely célja a D feladat, nem szükséges generálni az ebédidővel meghosszabbított utazási műveletet. A feladatok időparamétereinek vizsgálata során arra jutunk, hogy a következő utazások esetén releváns a hosszabb utazási idő: A-B, A-C, D-B, B-C, D-C. Ezek alapján került kibővítésre az eredeti probléma maximális struktúrája (lásd: 4.5.), amely eredményét a 4.11. ábra mutatja be.



4.11. ábra. Az ebédidő kezeléssel bővített maximális struktúra

A módosított probléma optimális ütemezését grafikusan a 4.12. ábra mutatja be, amely na-

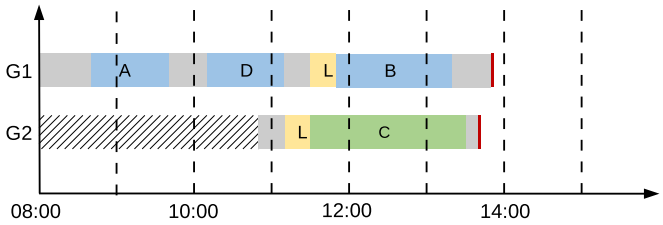
gyon hasonlít az eredeti probléma megoldására, a különbség csak az utazási műveletekben van. A megoldás Gantt diagrammja a 4.13. ábrán látható, amelyen megfigyelhető, hogy a feladatok kiosztása és sorrendje nem változott, de a sárgával jelölt ebédidők már megjelennek az ütemezésben.



4.12. ábra. Az ebédidő kezelésével bővített modell optimális megoldásának grafikus ábrázolása

A fejezetben ismertetett TCPNS modell képes előállítani a 4.1. fejezetben specifikált terepi munkavégzés ütemezési probléma megoldását. A P-gráf modell algoritmikusan generáltható az ismertetett lépések mentén. Az így felépített modell biztosítja a megfelelő feladat és szerelő csapat összerendelést, a pontos utazási időkkel való számolást, a mobil raktárkészlet kezelést, valamint az olyan egyedi követelményeknek való megfelelést, mint például az ebédidő ütemezése. A P-gráf modell alapján közvetlenül előállítható a vele ekvivalens MILP modell (lásd: 1.1.2.), amelynek megoldásával megkapjuk a probléma optimális ütemezését.

A vegyes-egész programozási modellek egyik jellemzője, hogy nagyméretű problémák esetén a megoldás jelentősen lassulhat. Sok más területen (gyártási ütemezés, üzleti folyamatok üte-



4.13. ábra. Az ebédidő kezelésével bővített modell optimális megoldásának ábrázolása Gantt diagrammon

mezése, autonóm járművezérlés stb.) a TCPNS önmagában elegendő lehet, hogy a valós méretű problémákat elfogadható időn belül megoldja. Viszont a terepi munkavégzés ütemezési probléma esetén a nagyszámú feladat és szerelő csoport kezelésére már nem volt alkalmazható, ezért további megoldások kifejlesztésére volt szükség. A probléma méretének csökkentése érdekében kidolgoztunk egy diszkrét intervallumokkal dolgozó relaxált modellt, amely a problémát kisebb részproblémákra bontja, amelyek már megoldhatók a TCPNS segítségével. A kapacitás tervezés és a TCPNS-alapú ütemezés iteratív megoldásával már lehetővé vált az ipari méretű problémák megoldása is.

4.3.4. A terepi munkavégzés ütemezéséhez generált P-gráf modell komplexitása

A terepi munkavégzés ütemezése során generált P-gráf szuperstruktúra méretét és komplexitását számos paraméter befolyásolja. A probléma egy újabb feladattal való bővítése esetén a gráf annyi a feladat végrehajtási folyamatot leíró ággal bővül, ahány szerelő csapat képes annak végrehajtására. Hasonlóan, egy újabb szerelő csapat esetén minden az általa elvégezhető feladatnak megfelelően egy újabb végrehajtást leíró ággal bővül a gráf. A lehetséges hozzárendelések számát tehát nagyban befolyásolja a csoportok paraméterei (szakismeret, képességek, szerepkörök és a mobil raktárkészlet) és a feladatok által előírt követelmények. További meghatározó tényező, a feladatok közötti lehetséges váltások (utazások) száma, amely szintén jelentősen megnövelheti a modell méretét. Ha minden feladat megoldható a teljes tervezési periódus alatt, akkor a feladatok közötti összes lehetséges utazással bővíteni kell a gráfot. A speciális követelmények, mint például a kötelező ebédidő kezelése további extra műveletek hozzáadását és újabb feltételek kezelését teheti szükségessé.

Egy adott problémához generált P-gráfban lévő aktivitások számának növekedésével arányosan növekszik a MILP modell egész változóinak száma is, amely jelentősen befolyásolja a modell megoldásához szükséges időt. Ezért érdemes megvizsgálni, hogy a bemutatott P-gráf alapú megközelítés mekkora méretű modellek megoldását teszi lehetővé. A 4.8. táblázat az átlagos megoldási időket mutatja be a feladatok és szerelő csapatok számának függvényében. A tesztelés során minden paraméter beállítással legalább tíz futtatás történt, valamint a szoftver futási ideje 20 percben került maximalizálásra.

A tesztelési eredmények elemzése alapján arra lehet következtetni, hogy a megoldási időt

4.8. táblázat. Megoldási idők a feladatok és a szerelő csapatok számának függvényében (sec)

Feladatok	2	4	6	8	10	12	14	16	18
1 Csapat	0.64	0.61	1.15	1.47	3.11	6.44	16.98	318	794
2 Csapat	0.42	0.63	1.56	2.95	9.01	57.98	1190	x	x
4 Csapat	0.46	1.34	2.89	10.02	91.35	449.84	x	x	x
6 Csapat	0.50	2.03	11.09	74.02	1045	x	x	x	x
8 Csapat	0.55	3.06	45.03	352.03	x	x	x	x	x
10 Csapat	0.58	4.13	172.25	x	x	x	x	x	x
12 Csapat	0.60	3.02	341.15	x	x	x	x	x	x

* Intel Core i5-3210M 2.5 GHz, 16 GB RAM

negatívan befolyásolja a feladatok időparamétereinek és prioritásának kicsi szórása, valamint az átállási idők szimmetriája (távolságmátrix). Viszont látható, hogy egyetlen szerelő csapat esetén is csak 18 feladat ütemezhető a meghatározott 20 perces időkorláton belül. Ezzel szemben a 4.1 fejezetben ismertetett ipari méretű problémák lényegesen nagyobbak, több száz vagy akár néhány ezer ütemezendő feladatot is tartalmazhatnak. Ezért szükséges volt egy a 4.4 fejezetben bevezetésre kerülő gyors előfeldolgozást biztosító eljárásra, amely a feladatok szerelő csapatokhoz és idő intervallumokhoz való rendelése révén csökkenti a pontosabb, perc alapú ütemezést megvalósító TCPNS alapú modell komplexitását.

4.4. Relaxált modell diszkrét intervallumokkal

A 4.3. fejezetben bemutatott TCPNS modell egyik jellemzője, hogy a gráf alapú modell alkalmazása nagyban segíti az adott probléma struktúrájának megértését, valamint támogatja az egyedi követelményekhez igazított ütemezési modell kidolgozását. Általában a modellezés kezdeti szakaszában ezeket a gráfokat manuálisan készítjük el, majd meghatározásra kerülnek azok a modell generáló lépések, amelyek segítségével lehetővé válik a nagy méretű problémákhoz tartozó modellek előállítása. Ez az eljárás a legtöbb gyakorlati probléma esetén működik, de ha az adott probléma nagyszámú feladatot és/vagy berendezést (szerelő csapatot) tartalmaz, akkor a modell megoldása során jelentős lassulás tapasztalható. A terepi munkavégzés ütemezési feladat esetén nagyságrendileg 50+ szerelő csapat és 1-2 ezer feladat kezelésére kell számítani. A korábbi tapasztalatok és a kezdeti modelleken végzett tesztek azt mutatják, hogy ekkora méretű probléma TCPNS-sel való megoldása nem lehetséges elfogadható időn belül.

Ezért a jelenlegi probléma megoldása során szükség volt egy olyan lépésre, amely a problémát kisebb részproblémákra bontja, amelyek már megoldhatók a bemutatott modellezési eljárással. Ehhez egy olyan lineáris programozási modell kifejlesztésére volt szükség, amely képes nagyszámú feladat esetén is elvégezni a feladatok szerelő csapatokhoz, valamint az előre meghatározott, de nem rögzített hosszúságú időintervallumhoz rendelést. A lépés eredményeként létrejött összerendelések időintervallumonként és csapatonként külön-külön kerülnek ütemezésre a TCPNS modell segítségével.

A kapacitásalapú tervezés során több egyszerűsítésre is szükség volt a nagyméretű feladatok kezelésének biztosításához, így előfordulhat, hogy egy adott intervallumhoz és szerelő csapathoz rendelt feladatot a TCPNS modell alapján nem lehet ütemezni, például az átlagostól jelentősen eltérő utazási idők miatt. Ennek kiküszöbölésére a TCPNS modellek megoldása után a kapacitástervezési lépés újra lefut, ahol az ütemezés során kihagyott feladatok újra elérhetőek lesznek. Természetesen a korábban sikeresen ütemezett feladatokat és az ebből fakadó kapacitás csökkenést is kezelni kell. Az újabb iterációk során a még nem ütemezett feladatok hozzárendelhetők egy másik csoporthoz és/vagy időszakhoz. Ezt a két lépést addig kell iteratívan végrehajtani, amíg az új iteráció során van újabb összerendelés.

A kapacitásalapú tervezéshez a munkaterületet zónákra, a tervezési periódust időintervallumokra osztjuk. A diszkrét intervallumokat használó relaxált modell felírásához meg kell adni az időintervallumok sorozatát ($i_1, i_2, \ldots, i_k, \ldots, i_T$), a szerelő csapatok halmazát ($g_1, g_2, \ldots, g_i, \ldots, g_N$), az ütemezendő feladatok halmazát ($t_1, t_2, \ldots, t_j, \ldots, t_N$), valamint a csoportok számára már korábban kiosztott feladatokat.

A lineáris programozási modell felírásához a következő változók bevezetésére volt szükség:

- Minden i_k időintervallumra, és t_j feladatra felveszünk egy $x_{i,j,k}$ változót, ha a g_i csapat el tudja végezni a t_j feladatot a korábban hozzárendelet feladatok mellett.

$$0 <= x_{i,j,k} <= 1$$

- Jelölje $z_{i,k,l}$, hogy g_i csapat az i_k intervallumban hány feladatot végez a z_l zónában:

$$z_{i,k,l} = \sum_{Zone(t_j)=z_j} x_{i,j,k}$$

- Továbbá $u_{i,k,l}$ megadja, hogy a g_i csoport végez-e feladatot a z_l zónában az i_k időintervallumon belül:

$$x_{i,j,k} <= u_{i,k,l}, \forall t_j, \text{ ahol } Zone(t_j) = z_l$$

- Végül legyen r_i, amely megadja, hogy hány zóna váltás tett a g_i csapat a i_k időintervallumban.

$$r_i >= (\sum_{\forall z_l} u_{i,k,l}) - 1$$

A célfüggvény az értékteremtés mínusz a költségek összege, ahol az utazási időket zónákon belüli és a zónák közötti átlagos utazási idővel becsüljük az alábbiak szerint:

$$\sum_{\forall g_i} \sum_{\forall t_j} \sum_{\forall i_k} x_{i,j,k}(V(t_j) - TimeCost(g_i, TaskTime(t_j)) - \sum_{\forall g_i} \sum_{\forall i_k} \sum_{\forall z_l} z_{i,k,l} TravelCost(q_i, AvgTTiZ(z_l)) - \sum_{\forall g_i} \sum_{\forall i_k} r_{i,k} TravelCost(g_i, AvgTTbZ()) \rightarrow max$$

ahol $TaskTime(t_j)$ a t_j feladat végrehajtásához szükséges idő, a $TravelCost(q_i, time)$ az idő alapú utazási költség a g_i csapatra vonatkozóan, végül pedig $AvgTTiZ(z_l)$ és $AvgTTbZ$ az átlagos utazási idő a zónán belül illetve a zónák között.

Természetesen egyik csoport sem végezhet több munkát egyik időintervallumban sem, mint amennyi a szabad kapacitása a korábbi ütemezése alapján, ezért minden g_i csoportra:

$$\begin{gathered}\sum_{\forall t_j}\sum_{\forall i_k} x_{i,j,k} TaskTime(t_j) \\ + \sum_{\forall i_k}\sum_{\forall z_l} z_{i,k,l} AvgTTiZ(z_l) + \sum_{\forall g_i}\sum_{\forall i_k} r_{i,k} AvgTTbZ()) \\ <= FreeCapacity(g_i, i_k, Sch(g_i))\end{gathered}$$

A következő raktárlátogatás előtt elvégezhető feladatok anyagigénye nem lehet több mint amennyi a g_i csoportnál rendelkezésre áll:

$$\sum_{\forall t_j}\sum_{\forall i_k, End(i_k)<SVT(g_i)} x_{i,j,k} TMQ(t_j, m_q) <= GS(g_i, m_q)$$

ahol $SVT(g_i)$ a g_i csapat következő raktárlátogatásának időpontja, $TMQ(t_j, m_q)$ az m_q anyagból szükséges mennyiség a t_j feladat elvégzéséhez, és $GS(q_i, m_q)$ az m_q anyag készlet szintje, amely elérhető g_i csapat mobil raktárában.

Egy feladatot legfeljebb egyszer lehet elvégezni, bármennyi üzleti értéke is van, ezért minden t_j feladatra:

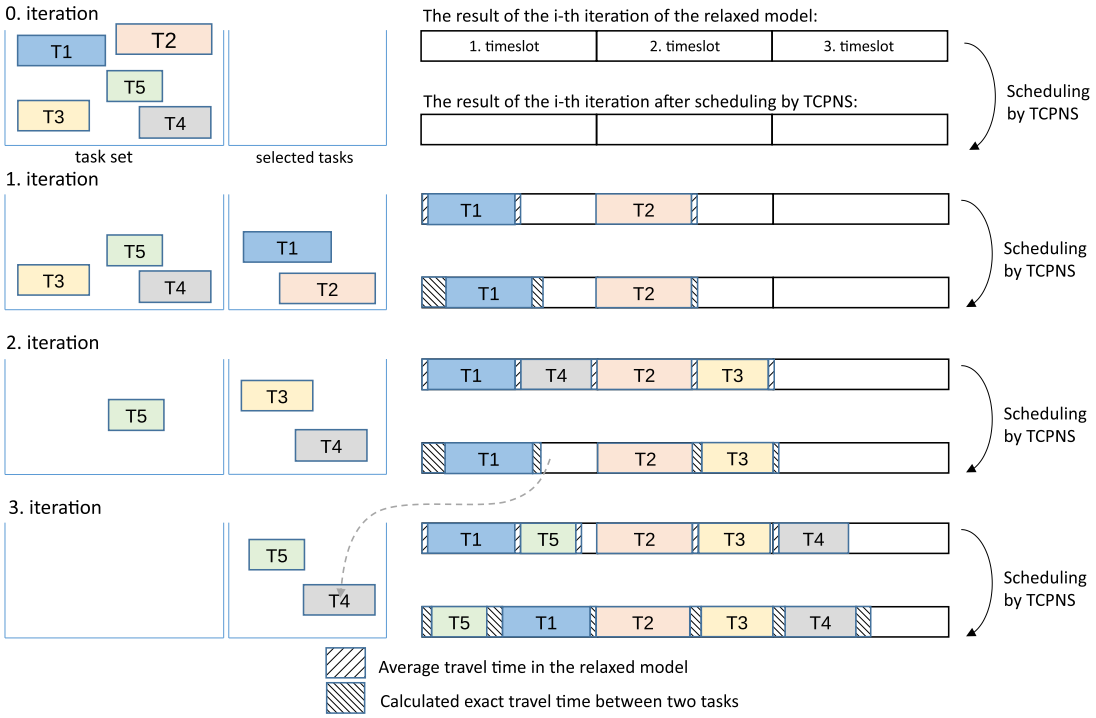
$$\sum_{\forall g_i}\sum_{\forall i_k} x_{i,j,k} <= 1$$

A fenti szabad kapacitás alapú modell egy lineáris programozás (LP) feladat, mely az optimális ütemezés egy jó közelítését adja, egyszerre véve figyelembe az összes csapat képességét, költségét és szabad kapacitását. A fenti modell megoldásának eredménye az $x_{i,j,k}$ értéke, mely 0 és 1 közötti értékkel jelzi, hogy melyik t_j feladatot, melyik g_i csoportnak, melyik i_k időintervallumban lenne érdemes megvalósítania. Ha minden $x_{i,j,k}$ folytonos változó, akkor előfordulhat, hogy a válasz nem egyértelmű: ugyanazon feladat különböző időintervallumokban, vagy különböző csoportoknál is megjelenik egynél kisebb súllyal. Ha minden $x_{i,j,k}$-t egész (bináris) változónak veszünk, akkor a válasz egyértelműen 0 vagy 1 lesz, ugyanakkor a megoldó módszer bonyolultsága komplexitást ugrik, ekkor már nem LP hanem vegyes egész lineáris programozási feladatról, azaz MILP-ről beszélünk, amely visszavezet az eredeti problémához.

Ezért szükségesek a folytonos változók használata, azaz az LP modell megtartása és egy küszöb érték bevezetése, amely felett a $x_{i,j,k}$ értéket úgy tekintjük, hogy hozzárendelés átadható a TCPNS modell számára, amely a pontosabb ütemezés révén véglegesíti vagy felbontja azt. Minél magasabb a küszöbérték, annál kevesebb, az LP megoldásban javasolt hozzárendelés kerül átadásra TCPNS modellek számára. Az általunk fejlesztett szoftverben ennek a küszöbnek az alapértelmezett értéke 0,5 volt.

Az LP modell generálása előtt a potenciális, még nem ütemezett feladatok üzleti értékei kerülnek maghatározásra a 4.2. fejezetben bemutatott módon, majd az előre meghatározott számú, legmagasabb értékkel rendelkező, legígéretesebb feladatokat választjuk ki ütemezésre. Az LP modell eredménye alapján minden szerelő csapathoz és időintervallumhoz külön TCPNS modell

kerül létrehozásra, amely már pontos utazási idők mellett próbálja az ütemezést elvégezni. Ha az ütemezés sikeres, akkor az LP megoldása által javasolt feladatok véglegesen hozzárendelésre kerülnek a csapathoz, így azok már a következő iterációban nem módosíthatók. Természetesen a csapat szabad kapacitása a kiosztott feladatoknak megfelelően csökken. Ha a TCPNS modell a pontos utazási idők miatt nem tudja ütemezni az LP által javasolt feladatot, akkor az a következő iteráció során újra elérhető lesz más csapatok és/vagy időintervallumok számára. Az LP-modell minden új iterációjához a TCPNS által visszaadott feladatok mellett további feladatokat tölt be egy paraméterként megadott maximális elemszám alapján.



4.14. ábra. Az LP és TCPNS modell iteratív megoldásának szemléltetése

A 4.14. ábra egy 5 feladatot tartalmazó példán keresztül szemlélteti az iteratív működést. Minden iteráció során két feladat kerül kiválasztásra, ami a relaxált modell megoldásával egy időintervallumhoz kerül hozzárendelésre, amely során átlagos utazási idővel számolunk. A következő lépésben minden időintervallum külön kerül ütemezésre a TCPNS segítségével, ahol két egymást követő feladat lokációja alapján már pontos utazási időkkel kalkulálhatunk, ezért néhány esetben az LP által javasolt összerendelés felbontásra kerül. Az első iterációban $T1$ és $T2$ feladat került kiválasztásra, majd hozzárendelésre rendre az első és második intervallumhoz. A második iterációban a kiválasztott $T3$ és $T4$ feladatokat az LP rendre a második és első intervallumhoz rendeli hozzá. Viszont a TCPNS megoldása során $T4$ ütemezése sikertelen, így visszakerül a még ütemezésre váró feladatok közé. A harmadik iterációban $T5$ mellett $T4$ újra kiválasztásra kerül, majd a relaxált modell a rendre az első és harmadik intervallumhoz rendeli őket hozzá.

4.9. táblázat. Megoldási idők a feladatok és a szerelő csapatok számának függvényében (sec)

Feladatok száma:	50	100	200	500	1000
10 Csapat	25.7	49.43	86.82	345.7	677.1
20 Csapat	24.66	52.9	129.57	480.81	960.75
30 Csapat	41.92	69.11	150.48	523.96	999.57
40 Csapat	42.22	82.14	160.1	564.73	1052.17
50 Csapat	48.13	89.62	190.25	727.88	*1378.95*

* Intel Core i5-3210M 2.5 GHz, 16 GB RAM

A TCPNS ütemezés eredményeként $T5$ marad az első intervallumba, de a végrehajtási sorrend változik, először $T5$ kerül végrehajtásra, majd ezt követően $T1$; a $T4$-es feladat viszont már sikeresen ütemezésre kerül a harmadik intervallumban. Így a harmadik iteráció eredményeként mind az 5 feladat ütemezésre került.

A két modell iteratív végrehajtása lehetővé teszi a nagyméretű ütemezési problémák megoldását. A relaxált modell hosszabb tervezési időszakra, akár több napra is képes gyorsan eredményt szolgáltatni, még akkor is, ha az egy iterációban kiválasztott feladatok száma százas nagyságrendű. Ezután a TCPNS néhány órás intervallumban képes optimális eredményt szolgáltatni néhány másodperces megoldási idővel.

A bemutatott kétszintű modell tesztelése során megvizsgálásra került, hogy alkalmas-e a 4.1 fejezetben bemutatott ipari méretű problémák megoldására, 20 perces megoldási időn belül egy gyengébb, kétmagos, 2,5 GHz-es processzorral rendelkező notebookon. A 4.9 táblázat összefoglalja a javasolt megoldási módszerrel elért futási időket a feladatok és szerelő csapatok számának függvényében, egy hosszabb, 30 napos ütemezés során. Korlátozott számítási kapacitás mellett az ütemezhető feladatok száma 40 szerelő csapat esetén megközelíti az 1000-et, 50 csapatnál pedig a 800-at. Így látható, hogy a tisztán PNS-alapú megoldásokhoz képest a megoldható probléma nagysága jelentősen megnőtt, például 6 helyett 1000-nél több feladatot lehet ütemezni 10 szolgáltatócsoportra. Ennek eredményeként a megoldható probléma mérete megfelel az ipari partnerek követelményeinek.

4.5. A fejezet rövid összefoglalása

A fejezetben egy a fuvarszervezési (VRP) feladatok osztályához tartozó valós gyakorlati probléma, a terepi munkavégzés ütemezése megoldására kidolgozott módszer került bevezetésre. A probléma során a különböző helyszínekhez köthető feladatokat kell komplex összerendelési szabályrendszer alapján kiosztani a szerelő csapatok számára. Az ütemezés során figyelembe kell venni az utazási időket, a feladatra vonatkozó időkorlátokat, a mobil raktárkészletet, valamint olyan speciális igényeket, mint az ebédidő biztosítása a feladatok végrehajtása során.

A terepi munkavégzés ütemezési probléma jellemzője, hogy nagyszámú feladatot és szerelő csapatot kell egyszerre kezelni, ezért annak megoldása tisztán a TCPNS keretrendszer által használt MILP modellel már nem lehetséges a bináris változók nagy száma miatt. Ezért egy két-szintű

iteratív megoldás került kidolgozásra, amely első lépésben egy diszkrét intervallumokkal dolgozó relaxált modell segítségével dekomponálja a feladatot, amely eredményeként létrejövő részfeladatok már megoldhatók TCPNS segítségével. A két modell megoldása több ciklusban addig kerül végrehajtásra, amíg lehetőség van új összerendelések megadására.

A matematikai programozással összehasonlítva az időkorlátos folyamathálózat szintézis és a P-gráf alapú megközelítés előnye, hogy az optimalizálási modell automatikusan generálható, így az optimalizálási folyamat szoftveres támogatása könnyen megvalósítható és a grafikus reprezentáció révén egyszerűen elmagyarázható a végfelhasználó számára. Továbbá a dolgozatban bevezetett kétlépcsős iteratív megvalósítás lehetővé teszi a terepi munkavégzés ütemezését, több mint ezer ütemezett feladattal és az optimális útvonalakkal. Például 50 szerelő csapat esetén is, belátható időn belül szolgáltat eredményt egy hagyományos számítógéppel is, ami jelentős előrelépés a tucatnyi feladathoz és szerelő csapathoz képest, amelyeket gyakorlatban egy precedencia alapú MILP modellel kezelni lehet.

4.5.1. A fejezethez tartozó tézis

Kidolgoztam egy két-lépcsős iteratív módszert a terepi munkavégzés ütemezési probléma időkorlátos folyamathálózat szintézissel (TCPNS) való megoldására. [102], [103], [104],

(a) Meghatároztam a terepi munkavégzés ütemezési feladatok leírását időkorlátos folyamathálózat szintézis feladatként.

(b) Kidolgoztam a terepi munkavégzés ütemezési probléma dekomponálására egy LP alapú diszkrét intervallumokat alkalmazó relaxált modellt a szerelő csapatok kapacitásainak tervezésére.

(c) Kidolgoztam egy iteratív eljárást, amely a kapacitástervező és a TCPNS modell iteratív végrehajtásával lehetővé teszi ipari méretű feladatok megoldását.

4.5.2. A fejezet témaköréhez kapcsolódó publikáció

Nemzetközi folyóiratcikk

– Frits Márton és Bertók Botond: Routing and scheduling field service operation by P-graph, Computers & Operations Research (IF=4,008) (2021), 105472

5. fejezet

Összefoglalás

A dolgozatban olyan új, tudományos eredmények kerülnek bemutatásra, amelyek alkalmazásával az időkorlátos folyamathálózat szintézis (TCPNS) feladatként írhatóak fel és oldhatóak meg komplex ipari ütemezési, erőforrás-hozzárendelési és fuvarszervezési problémák. A TCPNS a folyamathálózat-szintézis időparaméterek kezelésével bővített kiterjesztése. A 2.1. fejezetben részletesen bemutatásra kerül, hogy miként építhető egy ütemezési feladathoz olyan maximális P-gráf struktúra, amely tartalmazza a feltételeknek megfelelő összes lehetséges összerendelést, így az optimális ütemezést is. A maximális struktúra alapján felírható egy vele ekvivalens MILP modell, amely megoldásának megfelelő értelmezése révén megadható az eredeti probléma optimális ütemezése. A modell bővítésével olyan esetek kerültek kezelésre, mint az egy feladat végrehajtásának több berendezésen való arányos megosztása, valamint a korlátos köztes tárolók alkalmazása. Az új módszer működését egy valós gyártás ütemezési feladat, az egyedi lenyomatos szalvéta gyártási probléma megoldásával igazoltuk.

Szakaszos gyártási folyamatok ütemezése során az egyik meghatározó paraméter a köztes termékekre vonatkozó tárolási stratégia. Az irodalomban számos, a már jól ismert stratégiák kezelésére vonatkozó megoldás található, de ezek főként az egyszerűbb változatokat vizsgálják. Az 3. fejezetben olyan modellezési technikák kerültek bevezetésre, amelyek modell szinten képesek kezelni a végtelen számú köztes tárolót (UIS), a köztes tároló nélküli (NIS) gyártási folyamatokat, illetve a köztes termék azonnali feldolgozását megkövetelő (ZW) feltételeket. Definiálásra kerültek az adott tárolási stratégiának megfelelő működést biztosító struktúra felépítésének lépései, amely működését szemléltető példákon keresztül igazolom. A fejezetben összehasonlításra kerülnek az egyes modellek, amely alapján megállapításra került, hogy a különböző stratégiák egy modellben is kezelhetőek, így nem csak a teljes folyamatra, hanem a folyamat lépéseinek szintjén is meg lehet határozni az elvárt tárolási stratégiát.

Az utolsó, 4. fejezetben egy a fuvarszevezési (VRP) feladatok osztályába tartozó összetett ütemezési probléma, a terepi munkavégzés optimalizálása feladat került modellezésre, implementálásra és megoldásra. A feladat fő nehézsége a komplex összerendelési szabályrendszer mellett az utazási idők kezelése az ütemezés során. Olyan modell megalkotására volt szükség, amely képes kezelni a feladatokra vonatkozó időkorlátokat, a pontos utazási időket és a mobil raktárkészletet.

A feladat modellezése során a korábban bevezetett TCPNS alapú módszert bővítettem az egyedi igények kezelésével. A szerelő csapatok és az elvégzendő feladatok várható nagy száma miatt a feladat megoldása egy tisztán MILP alapú megoldással nem lehetséges, ezért szükség volt egy relaxált modell kidolgozására, amely LP alapú modell révén nagy méretű feladatok megoldására is alkalmazható. A relaxált modell megoldásával a problémát kisebb részproblémákra osztjuk, amelyek már TCPNS segítségével megoldhatók. A két modell iteratív végrehajtásával lehetővé vált ipari méretű feladatok megoldása is.

Összességében kijelenthető, hogy a TCPNS keretrendszerhez kapcsolódó eredmények újak, és önálló kutatás részeként kerültek megvalósításra. A kutatás során kidolgozott eljárások implementálásra kerültek és alkalmasak valós ipari feladatok megoldására. Elmondható tehát, hogy nem csak elméleti, de gyakorlatban is hasznos eredmények születtek.

6. fejezet

Új tudományos eredmények

6.1. Tézisek

1. **Kidolgoztam egy új időkorlátos folyamathálózat szintézisen (TCPNS) alapuló módszert ütemezési és folyamathálózat szintézis feladatok együttes megoldására. [24], [25], [98], [99], [100], [101]**

   (a) Meghatároztam az ütemezési feladatok leírását időkorlátos folyamathálózat szintézis feladatként

   (b) Megmutattam, hogy az általam javasolt TCPNS formalizmus lehetővé teszi az ütemezés és folyamat szintézis együttes megvalósítását, beleértve a folytonos értékű erőforrás korlátok kezelését, feladatok tetszőleges arányban való megosztását több berendezés között, a tevékenységek volumenétől folytonos függvény szerint változó műveletvégzési idők figyelembe vételét az ütemezés során.

   (c) Algoritmust dolgoztam ki olyan szuperstruktura generálására, mely tartalmazza egy ütemezési feladat összes lehetséges megoldását, így garantáltan tartalmazza az optimális megoldást is.

   (d) Az új ütemező módszer alkalmazhatóságát egy valós ipari probléma, az egyedi lenyomtatos szalvéta gyártás ütemezési feladat megoldásával igazoltam.

2. **Új modellezési módszereket dolgoztam ki a tárolási stratégiával bővített gyártásütemezési feladatok időkorlátos folyamathálózat szintézis (TCPNS) problémaként való felírására. [24], [25]**

   (a) Meghatároztam az UIS, NIS, ZW és MIS tárolási stratégiával bővített gyártás ütemezési feladatok leírását időkorlátos folyamat-hálózat szintézis feladatként.

   (b) Algoritmust dolgoztam ki, mely az összes lehetőséget tartalmazó szuperstruktúra generálása során a strukturába építi azokat a logikai korlátokat, amelyek a tárolási stratégiából következnek.

(c) Az új ütemező módszer alkalmazhatóságát egy példa feladat megoldásával igazoltam.

3. **Kidolgoztam egy két-lépcsős iteratív módszert a terepi munkavégzés ütemezési probléma időkorlátos folyamathálózat szintézissel (TCPNS) való megoldására. [102], [103], [104], [26],**

   (a) Meghatároztam a terepi munkavégzés ütemezési feladatok leírását időkorlátos folyamathálózat szintézis feladatként.

   (b) Kidolgoztam a terepi munkavégzés ütemezési probléma dekomponálására egy LP alapú diszkrét intervallumokat alkalmazó relaxált modellt a szerelő csapatok kapacitásainak tervezésére.

   (c) Kidolgoztam egy iteratív eljárást, amely a kapacitástervező és a TCPNS modell iteratív végrehajtásával lehetővé teszi ipari méretű feladatok megoldását.

6.2. Az értekezés témaköréből készült publikációk

Az értekezés témaköreiből az alábbi publikációk születtek:

Nemzetközi folyóiratcikkek

– Frits Márton és Bertók Botond: Process scheduling by synthesizing time constrained process-networks, kiadvány: Computer Aided Chemical Engineering, vol 33, oldal 1345-1350 (2014) (IF=0.54) [24]

– Frits Márton és Bertók Botond: Scheduling custom printed napkin manufacturing by P-graphs, kiadvány: Computers & Chemical Engineering, vol 141, oldal 107017 (2020) (IF=4.33)[25]

– Frits Márton és Bertók Botond: Routing and scheduling field service operation by P-graph, Computers & Operations Research, oldal 105472 (2021), (IF=4,008) [26]

Nemzetközi konferencia-kiadványokban megjelent közlemények

– Frits Márton és Bertók Botond: Time Constrained Process-Network Synthesis: Solving Production Scheduling Problems, kiadvány: Future Internet Services ASCONIKK 2014: Extended Abstracts. II. 5-10. oldal (2014) [105]

Nemzetközi konferencia előadások

– Frits Márton és Bertók Botond: Process scheduling by synthesizing time constrained process-networks, konferencia: European Symposium on Computer Aided Process Engineering, Budapest, Magyarország (2014) [24]

– Frits Márton és Bertók Botond: Maximize Aircraft Utilization by P-graphs, konferencia: International Congress on Sustainability Science Engineering, Balatonfüred, Magyarország (2015) [98]

– Frits Márton és Bertók Botond: Planning and Scheduling Aircraft Missions to Eliminate Empty Legs, konferencia: European Working Group on Location Analysis Meeting 2015., Budapest, Magyarország (2015) [99]

– Frits Márton és Bertók Botond: Time Constrained Process-Network Synthesis: Solving Production Scheduling Problems, konferencia: ASCONIKK 2014., Veszprém, Magyarország (2014)[105]

– Frits Márton és Bertók Botond: Mobile Workforce Assignment and Scheduling Optimization, konferencia: VOCAL Optimization Conference: Advanced Algorithms 2016., Esztergom, Magyarország (2016) [102]

– Pekárdy Milán, Frits Márton és Bertók Botond: Software framework for minimizing resource needs of field operations, konferencia: SPIL 2017 (Energy, water, emission, & waste in industry an cities), Brno, Csehország (2017) [103]

– Frits Márton: Development of Smart city solutions by the University of Pannonia in cooperation with IBM Hungary, konferencia: HEinnovate Hungary Conference, Budapest, Magyarország (2017) [104]

– Frits Márton és Bertók Botond: Scheduling of custom printed napkin manufacturing by P-graphs, konferencia: VOCAL Optimization Conference: Advanced Algorithms 2018., Esztergom, Magyarország (2018) [100]

– Bertók Botond, Frits Márton: Software framework for minimizing resource needs of field operations, konferencia: SPIL 2020 (Energy, water, emission, & waste in industry an cities), Brno, Csehország (2020) [101]

Irodalomjegyzék

[1] Arwa Mohamed és tsai. "Software-defined networks for resource allocation in cloud computing: A survey". *Computer Networks* 195 (2021), 108151. old.

[2] Fatemeh Ebadifard és Seyed Morteza Babamir. "Federated Geo-Distributed Clouds: Optimizing Resource Allocation Based on Request Type Using Autonomous and Multi-objective Resource Sharing Model". *Big Data Research* 24 (2021), 100188. old.

[3] Essam H Houssein és tsai. "Task scheduling in cloud computing based on meta-heuristics: Review, taxonomy, open challenges, and future trends". *Swarm and Evolutionary Computation* (2021), 100841. old.

[4] N Narayanan Prasanth és tsai. "Crossbar switch scheduling algorithms for high performance computing: A comprehensive review". *Materials Today: Proceedings* (2020).

[5] Nimisha Patel és Hiren Patel. "Energy efficient strategy for placement of virtual machines selected from underloaded servers in compute Cloud". *Journal of King Saud University-Computer and Information Sciences* 32.6 (2020), 700–708. old.

[6] Yaser Jararweh és tsai. "Energy efficient dynamic resource management in cloud computing based on logistic regression model and median absolute deviation". *Sustainable Computing: Informatics and Systems* 19 (2018), 262–274. old.

[7] Nikos Hatziargyriou. *Microgrids: architectures and control*. John Wiley & Sons, 2014.

[8] Thomas Strasser és tsai. "A review of architectures and concepts for intelligence in future electric energy systems". *IEEE Transactions on Industrial Electronics* 62.4 (2014), 2424–2438. old.

[9] Roland Bálint, Attila Fodor és Attila Magyar. "Model-based Power Generation Estimation of Solar Panels using Weather Forecast for Microgrid Application". *Acta Polytechnica Hungarica* 16.7 (2019), 149–165. old.

[10] Anna I Pózna, Katalin M Hangos és Attila Magyar. "Temperature dependent parameter estimation of electrical vehicle batteries". *Energies* 12.19 (2019), 3755. old.

[11] Botond Bertok és Aniko Bartos. "Renewable energy storage and distribution scheduling for microgrids by exploiting recent developments in process network synthesis". *Journal of Cleaner Production* (2019).

[12] Márton Greber, Attila Fodor és Attila Magyar. “Generalized persistent fault detection in distribution systems using network flow theory”. *IFAC-PapersOnLine* 53.2 (2020), 13568–13574. old.

[13] Carlos A Méndez és tsai. “State-of-the-art review of optimization methods for short-term scheduling of batch processes”. *Computers & chemical engineering* 30.6-7 (2006), 913–946. old.

[14] Ali Allahverdi és tsai. *Scheduling in manufacturing systems: new trends and perspectives*. 2018.

[15] Ali Allahverdi. “A survey of scheduling problems with no-wait in process”. *European Journal of Operational Research* 255.3 (2016), 665–686. old.

[16] Braulio Brunaud és tsai. “Batch scheduling with quality-based changeovers”. *Computers & Chemical Engineering* 132 (2020), 106617. old.

[17] Mohammad Mahdi Ahmadian és tsai. “Four decades of research on the open-shop scheduling problem to minimize the makespan”. *European Journal of Operational Research* (2021).

[18] Jacques F Gouws és Thokozani Majozi. “Usage of inherent storage for minimisation of wastewater in multipurpose batch plants”. *Chemical Engineering Science* 64.16 (2009), 3545–3554. old.

[19] Robert Adonyi és tsai. “Incorporating heat integration in batch process scheduling”. *Applied Thermal Engineering* 23.14 (2003), 1743–1762. old.

[20] Christodoulos A Floudas és Xiaoxia Lin. “Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review”. *Computers & Chemical Engineering* 28.11 (2004), 2109–2129. old.

[21] Humyun Fuad Rahman, Ripon K Chakrabortty és Michael J Ryan. “Memetic algorithm for solving resource constrained project scheduling problems”. *Automation in Construction* 111 (2020), 103052. old.

[22] E Sanmarti és tsai. “Combinatorial framework for effective scheduling of multipurpose batch plants”. *AIChE Journal* 48.11 (2002), 2557–2570. old.

[23] Sebastian Panek és tsai. “Scheduling of multi-product batch plants based upon timed automata models”. *Computers & Chemical Engineering* 32.1-2 (2008), 275–291. old.

[24] M Frits és B Bertok. “Process scheduling by synthesizing time constrained process-networks”. *Computer Aided Chemical Engineering*. 33. köt. Elsevier, 2014, 1345–1350. old.

[25] Marton Frits és Botond Bertok. “Scheduling custom printed napkin manufacturing by P-graphs”. *Computers & Chemical Engineering* 141 (2020), 107017. old.

[26] Marton Frits és Botond Bertok. “Routing and scheduling field service operation by P-graph”. *Computers & Operations Research* (2021), 105472. old.

[27] E Kondili, CC Pantelides és RWH Sargent. “A general algorithm for short-term scheduling of batch operations—I. MILP formulation”. *Computers & Chemical Engineering* 17.2 (1993), 211–227. old.

[28] Jose M Pinto és Ignacio E Grossmann. “A continuous time mixed integer linear programming model for short term scheduling of multistage batch plants”. *Industrial & Engineering Chemistry Research* 34.9 (1995), 3037–3051. old.

[29] X Zhang és RWH Sargent. “The optimal operation of mixed production facilities—a general formulation and some approaches for the solution”. *Computers & Chemical Engineering* 20.6-7 (1996), 897–904. old.

[30] CA Méndez, GP Henning és J Cerdá. “An MILP continuous-time approach to short-term scheduling of resource-constrained multistage flowshop batch facilities”. *Computers & Chemical Engineering* 25.4-6 (2001), 701–711. old.

[31] CA Méndez, GP Henning és J Cerdá. “Optimal scheduling of batch plants satisfying multiple product orders with different due-dates”. *Computers & Chemical Engineering* 24.9-10 (2000), 2223–2245. old.

[32] Jaime Cerdá, Gabriela P Henning és Ignacio 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 (1997), 1695–1707. old.

[33] M Hegyháti és tsai. “Practical infeasibility of cross-transfer in batch plants with complex recipes: S-graph vs MILP methods”. *Chemical Engineering Science* 64.3 (2009), 605–610. old.

[34] Thokozani Majozi és Ferenc Friedler. “Maximization of throughput in a multipurpose batch plant under a fixed time horizon: S-graph approach”. *Industrial & engineering chemistry research* 45.20 (2006), 6713–6720. old.

[35] Mate Hegyhati. “Batch process scheduling with eS-graph: A case study”. *Chemical Engineering Transactions* 70 (2018), 115–120. old.

[36] Rajeev Alur, Salvatore La Torre és George J Pappas. “Optimal paths in weighted timed automata”. *Theoretical Computer Science* 318.3 (2004), 297–322. old.

[37] Ansgar Fehnker. “Scheduling a steel plant with timed automata”. *Proceedings Sixth International Conference on Real-Time Computing Systems and Applications. RTCSA’99 (Cat. No. PR00306)*. IEEE. 1999, 280–286. old.

[38] Balázs Dávid, Máté Hegyháti és Miklós Krész. “Linearly priced timed automata for the bus schedule assignment problem”. *2018 4th International Conference on Logistics Operations Management (GOL)*. IEEE. 2018, 1–7. old.

[39] Su Nguyen és tsai. “Genetic programming for job shop scheduling”. *Evolutionary and Swarm Intelligence Algorithms*. Springer, 2019, 143–167. old.

[40] John Park és tsai. "An investigation of ensemble combination schemes for genetic programming based hyper-heuristic approaches to dynamic job shop scheduling". *Applied Soft Computing* 63 (2018), 72–86. old.

[41] Florian Grenouilleau és tsai. "A set partitioning heuristic for the home health care routing and scheduling problem". *European Journal of Operational Research* 275.1 (2019), 295–303. old.

[42] Sicheng Zhang és TN Wong. "Integrated process planning and scheduling: an enhanced ant colony optimization heuristic with parameter tuning". *Journal of Intelligent Manufacturing* 29.3 (2018), 585–601. old.

[43] B Bertok és tsai. "Superstructure approach to batch process scheduling by S-graph representation". *Computer Aided Chemical Engineering*. 29. köt. Elsevier, 2011, 1105–1109. old.

[44] F Friedler és tsai. "Graph-theoretic approach to process synthesis: axioms and theorems". *Chemical Engineering Science* 47.8 (1992), 1973–1988. old.

[45] JB Varga, F Friedler és LT Fan. "Parallelization of the accelerated branch-and-bound algorithm of process synthesis: application in total flowsheet synthesis". *Acta Chimica Slovenica* 42 (1995), 15–15. old.

[46] FERENC Friedler és tsai. "Combinatorial algorithms for process synthesis". *Computers & chemical engineering* 16 (1992), S313–S320.

[47] Mate Barany és tsai. "Solving vehicle assignment problems by process-network synthesis to minimize cost and environmental impact of transportation". *Clean Technologies and Environmental Policy* 13.4 (2011), 637–642. old.

[48] Rozália Lakner, Ferenc Friedler és Botond Bertók. "Synthesis and analysis of process networks by joint application of P-graphs and Petri nets". *International Conference on Application and Theory of Petri Nets and Concurrency*. Springer. 2017, 309–329. old.

[49] J.C. Garcia-Ojeda, B. Bertok és F. Friedler. "Planning evacuation routes with the P-graph framework". *Chemical Engineering Transactions* 29 (2012), 1531–1536. old. DOI: 10.3303/CET1229256.

[50] Károly Kalauz és tsai. "Extending process-network synthesis algorithms with time bounds for supply network design". *Chemical Engineering* 29 (2012).

[51] P Castro, APFD Barbosa-Póvoa és H Matos. "An improved RTN continuous-time formulation for the short-term scheduling of multipurpose batch plants". *Industrial & engineering chemistry research* 40.9 (2001), 2059–2068. old.

[52] Georgios M Kopanos, Luis Puigjaner és Michael C Georgiadis. "Optimal production scheduling and lot-sizing in dairy plants: the yogurt production line". *Industrial & Engineering Chemistry Research* 49.2 (2010), 701–718. old.

[53] Maria Di Mascolo, Clea Martinez és Marie-Laure Espinouse. "Routing and scheduling in Home Health Care: A Literature Survey and Bibliometric Analysis". *Computers & Industrial Engineering* (2021), 107255. old.

[54] Mustafa Misir és tsai. “Security personnel routing and rostering: a hyper-heuristic approach”. *Proceedings of the 3rd International Conference on Applied Operational Research*. 3. köt. Tadbir; Canada. 2011, 193–205. old.

[55] Jean-François Cordeau és tsai. “Scheduling technicians and tasks in a telecommunications company”. *Journal of Scheduling* 13.4 (2010), 393–409. old.

[56] J Arturo Castillo-Salazar, Dario Landa-Silva és Rong Qu. “Workforce scheduling and routing problems: literature survey and computational study”. *Annals of Operations Research* 239.1 (2016), 39–67. old.

[57] Mogens Fosgerau és Leonid Engelson. “The value of travel time variance”. *Transportation Research Part B: Methodological* 45.1 (2011), 1–8. old.

[58] Sergio R Jara-Diaz. “Allocation and valuation of travel-time savings”. *Handbook of transport modelling*. Emerald Group Publishing Limited, 2007.

[59] George B Dantzig és John H Ramser. “The truck dispatching problem”. *Management science* 6.1 (1959), 80–91. old.

[60] Martin Desrochers, Jan Karel Lenstra és Martin WP Savelsbergh. “A classification scheme for vehicle routing and scheduling problems”. *European Journal of Operational Research* 46.3 (1990), 322–332. old.

[61] Gilbert Laporte és Ibrahim H Osman. “Routing problems: A bibliography”. *Annals of operations research* 61.1 (1995), 227–262. old.

[62] Raafat Elshaer és Hadeer Awad. “A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants”. *Computers & Industrial Engineering* 140 (2020), 106242. old.

[63] Gilbert Laporte. “Fifty years of vehicle routing”. *Transportation science* 43.4 (2009), 408–416. old.

[64] Burak Eksioglu, Arif Volkan Vural és Arnold Reisman. “The vehicle routing problem: A taxonomic review”. *Computers & Industrial Engineering* 57.4 (2009), 1472–1483. old.

[65] Gilbert Laporte és Yves Nobert. “A branch and bound algorithm for the capacitated vehicle routing problem”. *Operations-Research-Spektrum* 5.2 (1983), 77–85. old.

[66] Michel L Balinski és Richard E Quandt. “On an integer program for a delivery problem”. *Operations research* 12.2 (1964), 300–304. old.

[67] Roberto Baldacci, Eleni Hadjiconstantinou és Aristide Mingozzi. “An exact algorithm for the capacitated vehicle routing problem based on a two-commodity network flow formulation”. *Operations research* 52.5 (2004), 723–738. old.

[68] Wai Yuen Szeto, Yongzhong Wu és Sin C Ho. “An artificial bee colony algorithm for the capacitated vehicle routing problem”. *European Journal of Operational Research* 215.1 (2011), 126–135. old.

[69] Sener Akpinar. “Hybrid large neighbourhood search algorithm for capacitated vehicle routing problem”. *Expert Systems with Applications* 61 (2016), 28–38. old.

[70] László T Kóczy, Péter Földesi és Boldizsár Tüű-Szabó. "An effective discrete bacterial memetic evolutionary algorithm for the traveling salesman problem". *International Journal of Intelligent Systems* 32.8 (2017), 862–876. old.

[71] Paolo Toth és Daniele Vigo. "Branch-and-bound algorithms for the capacitated VRP". *The vehicle routing problem*. SIAM, 2002, 29–51. old.

[72] Paolo Toth és Daniele Vigo. "The granular tabu search and its application to the vehicle-routing problem". *Informs Journal on computing* 15.4 (2003), 333–346. old.

[73] Guenther Fuellerer és tsai. "Ant colony optimization for the two-dimensional loading vehicle routing problem". *Computers & Operations Research* 36.3 (2009), 655–673. old.

[74] Michel Gendreau és tsai. "A Tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints". *Networks: An International Journal* 51.1 (2008), 4–18. old.

[75] Olli Bräysy és Michel Gendreau. "Vehicle routing problem with time windows, Part I: Route construction and local search algorithms". *Transportation science* 39.1 (2005), 104–118. old.

[76] Jonathan F Bard, George Kontoravdis és Gang Yu. "A branch-and-cut procedure for the vehicle routing problem with time windows". *Transportation Science* 36.2 (2002), 250–269. old.

[77] Niklas Kohl és Oli BG Madsen. "An optimization algorithm for the vehicle routing problem with time windows based on Lagrangian relaxation". *Operations research* 45.3 (1997), 395–406. old.

[78] Jean-François Cordeau, Gilbert Laporte és Anne Mercier. "A unified tabu search heuristic for vehicle routing problems with time windows". *Journal of the Operational research society* 52.8 (2001), 928–936. old.

[79] Jean-François Cordeau, Gilbert Laporte és Anne Mercier. "Improved tabu search algorithm for the handling of route duration constraints in vehicle routing problems with time windows". *Journal of the Operational Research Society* 55.5 (2004), 542–546. old.

[80] Luca Maria Gambardella, Éric Taillard és Giovanni Agazzi. "Macs-vrptw: A multiple colony system for vehicle routing problems with time windows". *New ideas in optimization*. Citeseer. 1999.

[81] Russell Bent és Pascal Van Hentenryck. "A two-stage hybrid local search for the vehicle routing problem with time windows". *Transportation Science* 38.4 (2004), 515–530. old.

[82] Toshihide Ibaraki és tsai. "Effective local search algorithms for routing and scheduling problems with general time-window constraints". *Transportation science* 39.2 (2005), 206–232. old.

[83] Defu Zhang és tsai. "A hybrid algorithm for a vehicle routing problem with realistic constraints". *Information Sciences* 394 (2017), 167–182. old.

[84] Geoff Clarke és John W Wright. “Scheduling of vehicles from a central depot to a number of delivery points”. *Operations research* 12.4 (1964), 568–581. old.

[85] Manfred Gronalt, Richard F Hartl és Marc Reimann. “New savings based algorithms for time constrained pickup and delivery of full truckloads”. *European Journal of Operational Research* 151.3 (2003), 520–535. old.

[86] Amalia I Nikolopoulou és tsai. “Moving products between location pairs: Cross-docking versus direct-shipping”. *European Journal of Operational Research* 256.3 (2017), 803–819. old.

[87] Yannis Ancele és tsai. “Toward a more flexible VRP with pickup and delivery allowing consolidations”. *Transportation Research Part C: Emerging Technologies* 128 (2021), 103077. old.

[88] Yousef Maknoon és Gilbert Laporte. “Vehicle routing with cross-dock selection”. *Computers & Operations Research* 77 (2017), 254–266. old.

[89] Jose Caceres-Cruz és tsai. “Rich vehicle routing problem: Survey”. *ACM Computing Surveys (CSUR)* 47.2 (2014), 1–28. old.

[90] Kenneth Sörensen, Marc Sevaux és Patrick Schittekat. ““Multiple Neighbourhood” Search in Commercial VRP Packages: Evolving Towards Self-Adaptive Methods”. *Adaptive and multilevel metaheuristics*. Springer, 2008, 239–253. old.

[91] Julia Rieck és Jürgen Zimmermann. “A new mixed integer linear model for a rich vehicle routing problem with docking constraints”. *Annals of Operations Research* 181.1 (2010), 337–358. old.

[92] Marius M Solomon. “Algorithms for the vehicle routing and scheduling problems with time window constraints”. *Operations research* 35.2 (1987), 254–265. old.

[93] Baris Ozkan, Eren Ozceylan és Suleyman Mete. “Planning of Vehicle Routes for the Exam Booklet Distribution: A GIS-based Solution Approach”. *IFAC-PapersOnLine* 53.2 (2020), 11225–11230. old.

[94] University of Pannonia. *P-graph Studio*. 2013. URL: `http://p-graph.org`.

[95] Microsoft Corp. *Microsoft Excel*. 2016. verzió. 2016. URL: `https://office.microsoft.com/excel`.

[96] Thorsten Koch. “Rapid Mathematical Prototyping”. Dissz. Technische Universität Berlin, 2004.

[97] Robin Lougee-Heimer. “The Common Optimization INterface for Operations Research: Promoting open-source software in the operations research community”. *IBM Journal of Research and Development* 47.1 (2003), 57–66. old.

[98] M Frits és B Bertok. *Maximize Aircraft Utilization by P-graphs*. 2015.

[99] M Frits és B Bertok. *Planning and Scheduling Aircraft Missions to Eliminate Empty Legs*. 2015.

[100] M Frits és B Bertok. *Scheduling of custom printed napkin manufacturing by P-graphs.* 2018.

[101] B Bertok és M Frits. *Process Scheduling and Load Distribution: Time Interval and Precedence Based Formulation by P-graphs.* 2020.

[102] M Frits és B Bertok. *Mobile Workforce Assignment and Scheduling Optimization.* 2016.

[103] M Pekardy, M Frits és B Bertok. *Software framework for minimizing resource needs of field operations.* 2017.

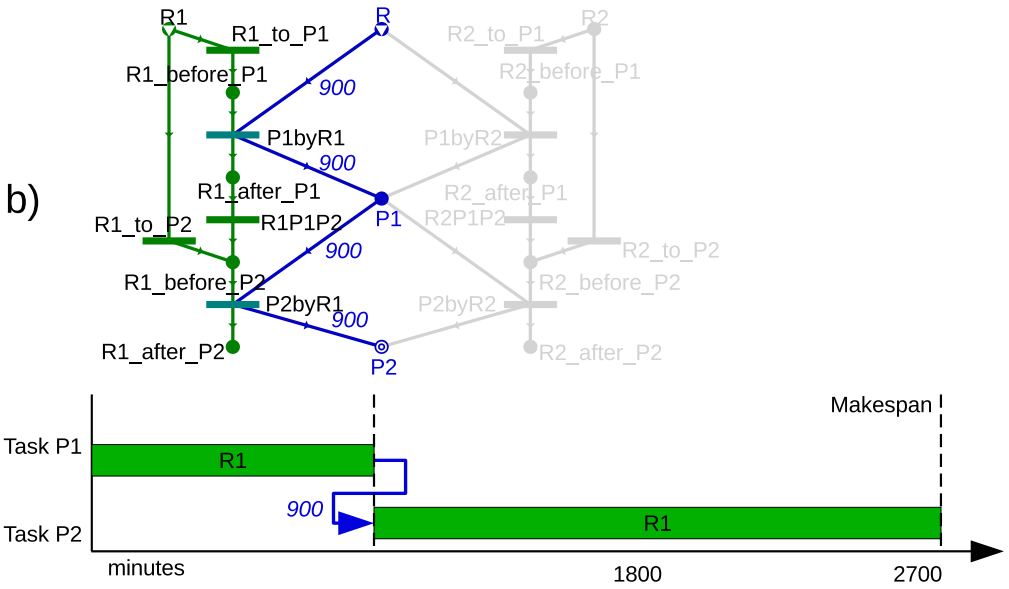
[104] M Frits. *Development of Smart city solutions by the University of Pannonia in cooperation with IBM Hungary.* 2017.

[105] M Frits és B Bertok. "Time Constrained Process-Network Synthesis: Solving Production Scheduling Problems". *ASCONIKK 2014* (2014), 13. old.

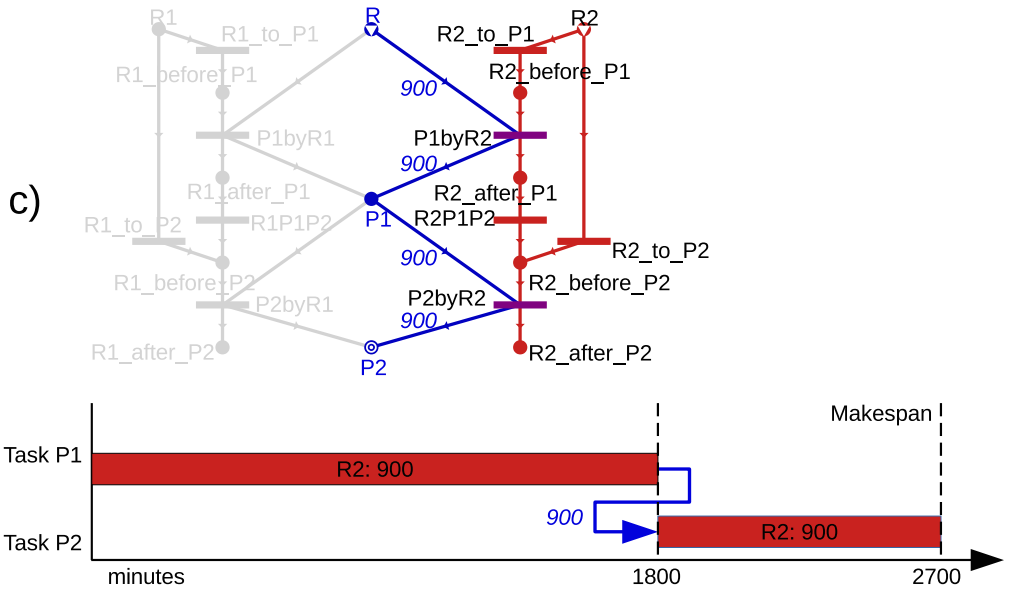
Melléklet

A) Ütemezés időkorlátos folyamathálózat szintézissel

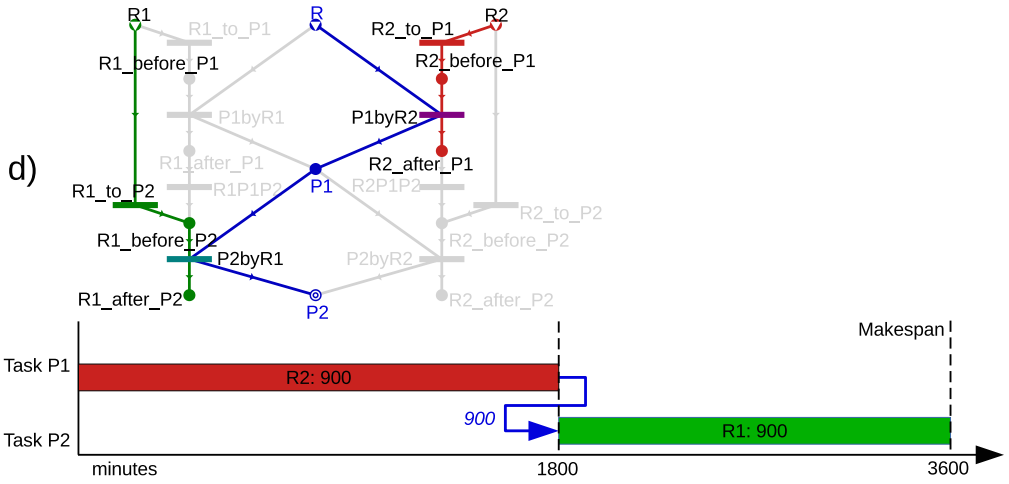
Anyagmennyiségek kezelése



6.1. ábra. A példa feladat lehetséges további ütemezésének megjelenítése anyagáramokkal

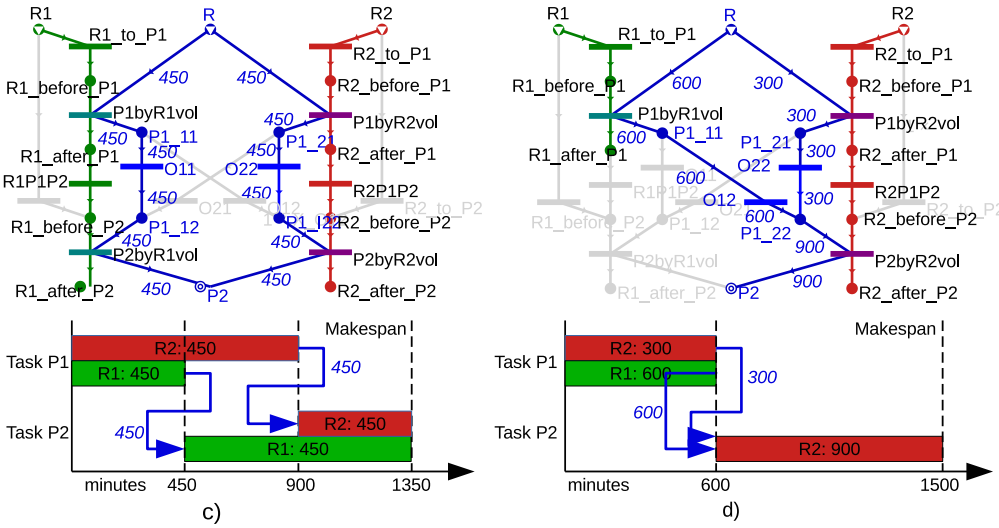


6.2. ábra. A példa feladat lehetséges további ütemezésének megjelenítése anyagáramokkal

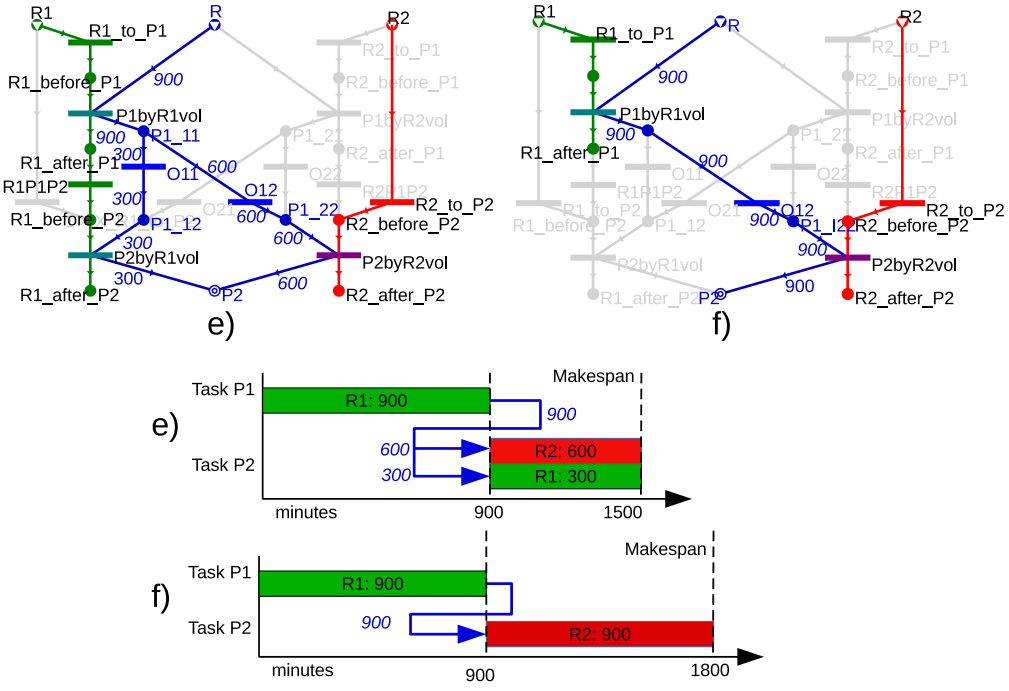


6.3. ábra. A példa feladat lehetséges további ütemezésének megjelenítése anyagáramokkal

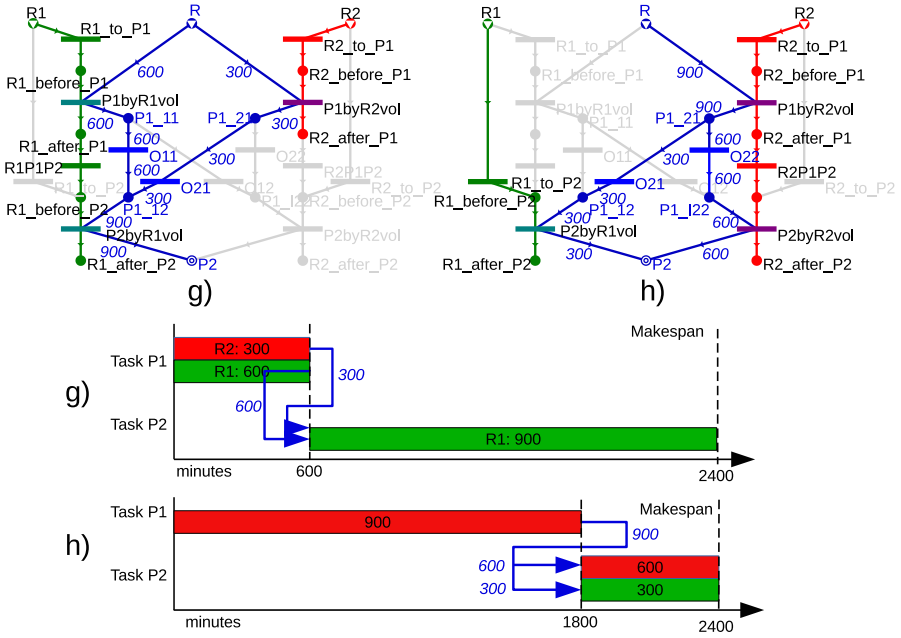
Részterhelés kezelése



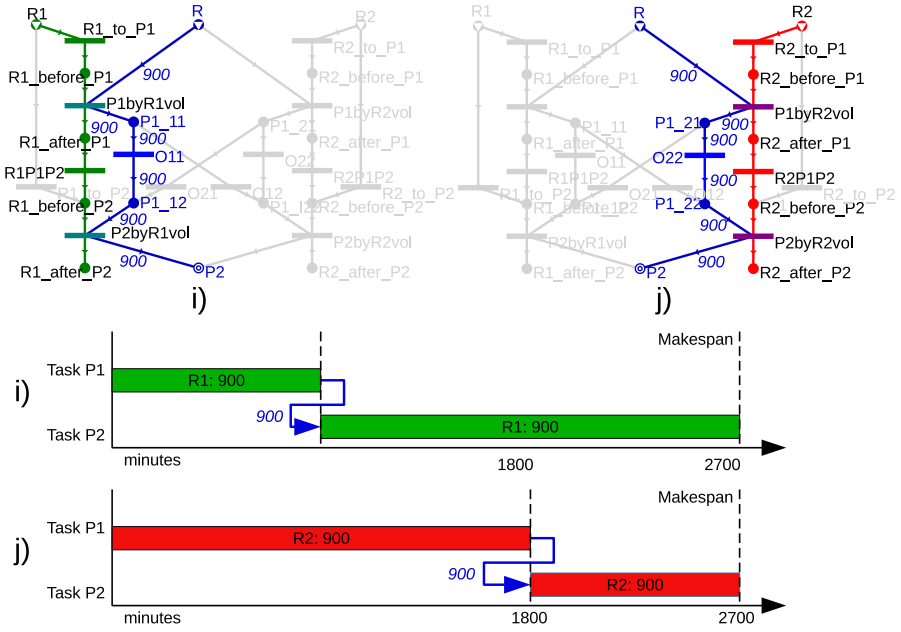
6.4. ábra. Példa: Lehetséges ütemezések a feladatok aszinkron párhuzamos végrehajtásával



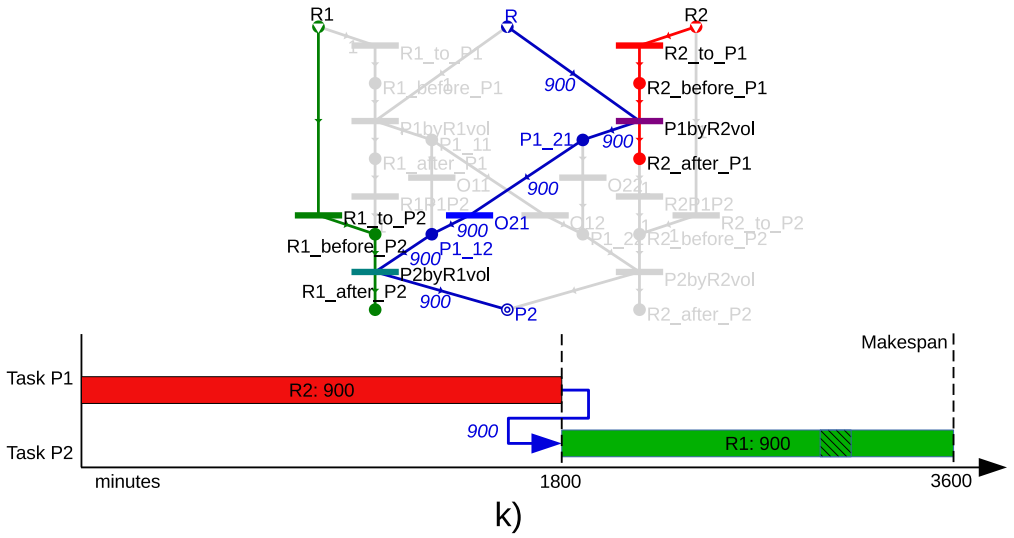
6.5. ábra. Példa: Lehetséges ütemezések a feladatok aszinkron párhuzamos végrehajtásával



6.6. ábra. Példa: Lehetséges ütemezések a feladatok aszinkron párhuzamos végrehajtásával

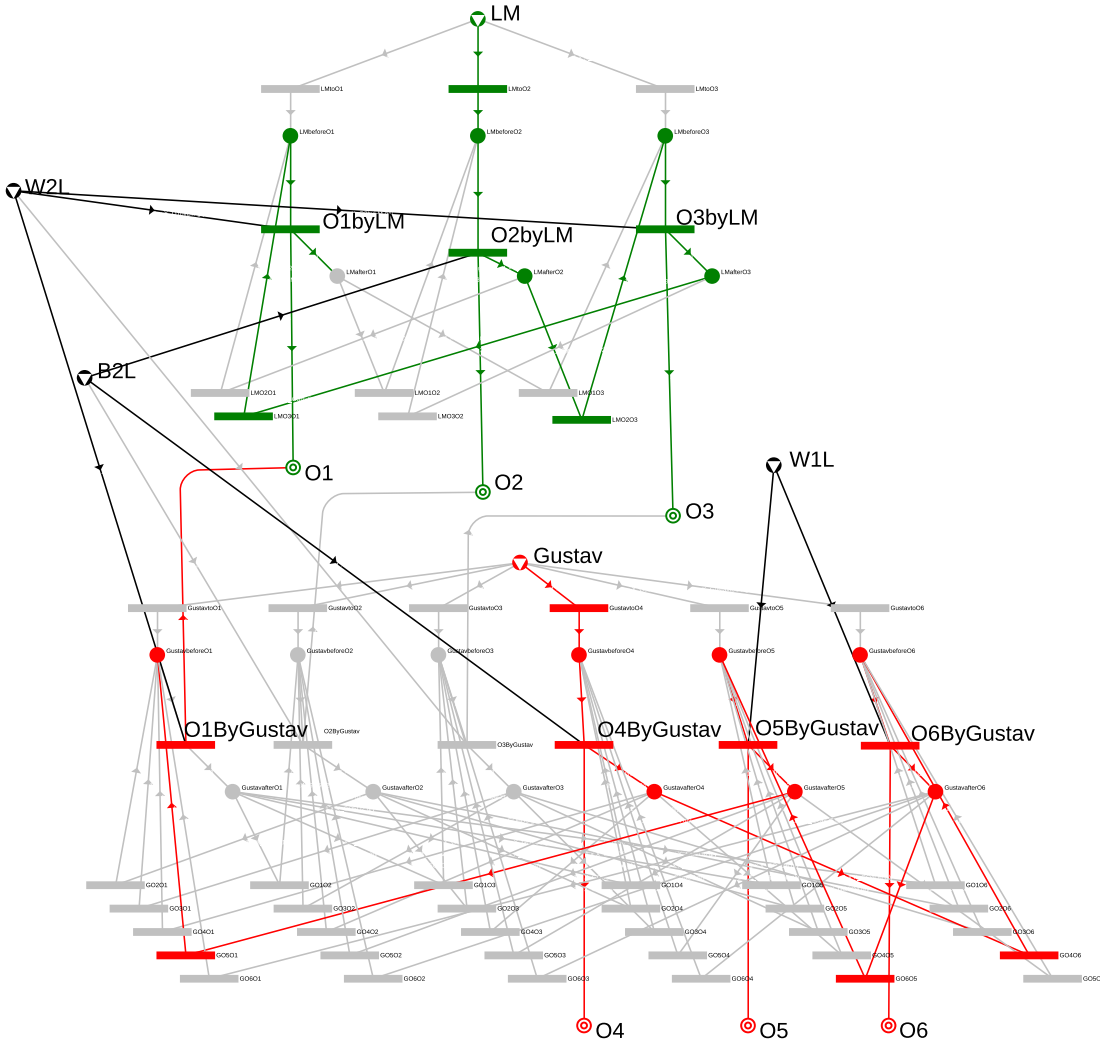


6.7. ábra. Példa: Lehetséges ütemezések a feladatok aszinkron párhuzamos végrehajtásával



6.8. ábra. Példa: Lehetséges ütemezések a feladatok aszinkron párhuzamos végrehajtásával

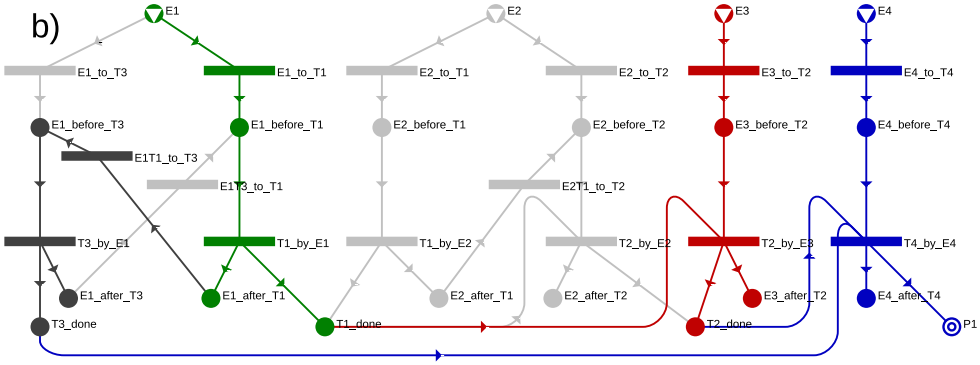
Esettanulmány: Egyedi lenyomtatos szalvéták gyártása



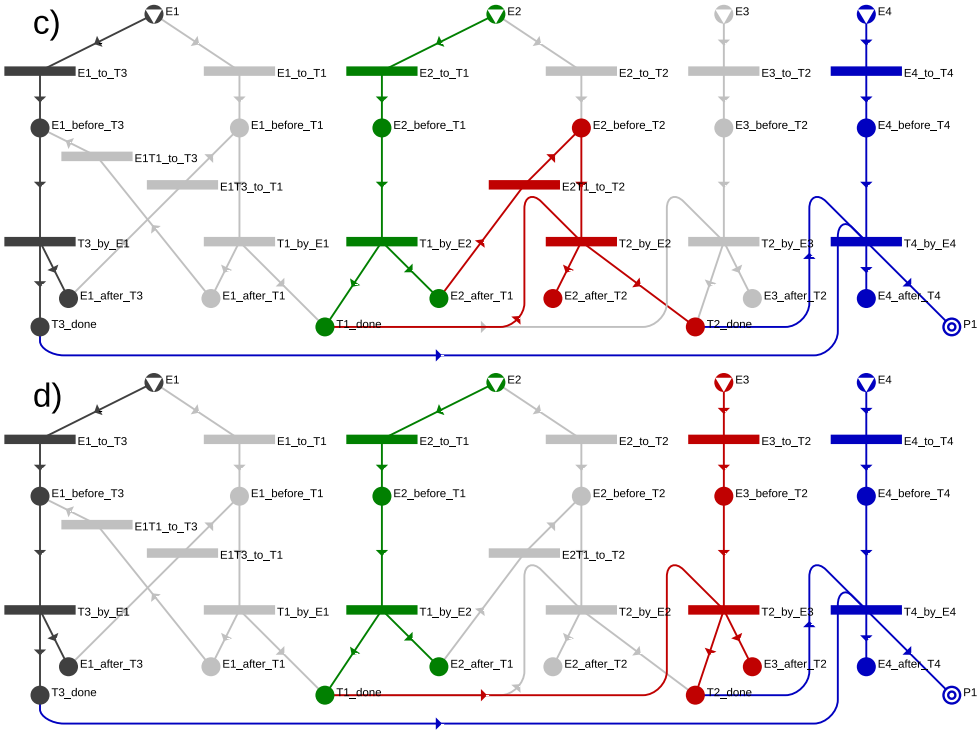
6.9. ábra. A c) eset megoldásának ábrázolása a maximális struktúrában

B) Tárolási stratégiák modellezése

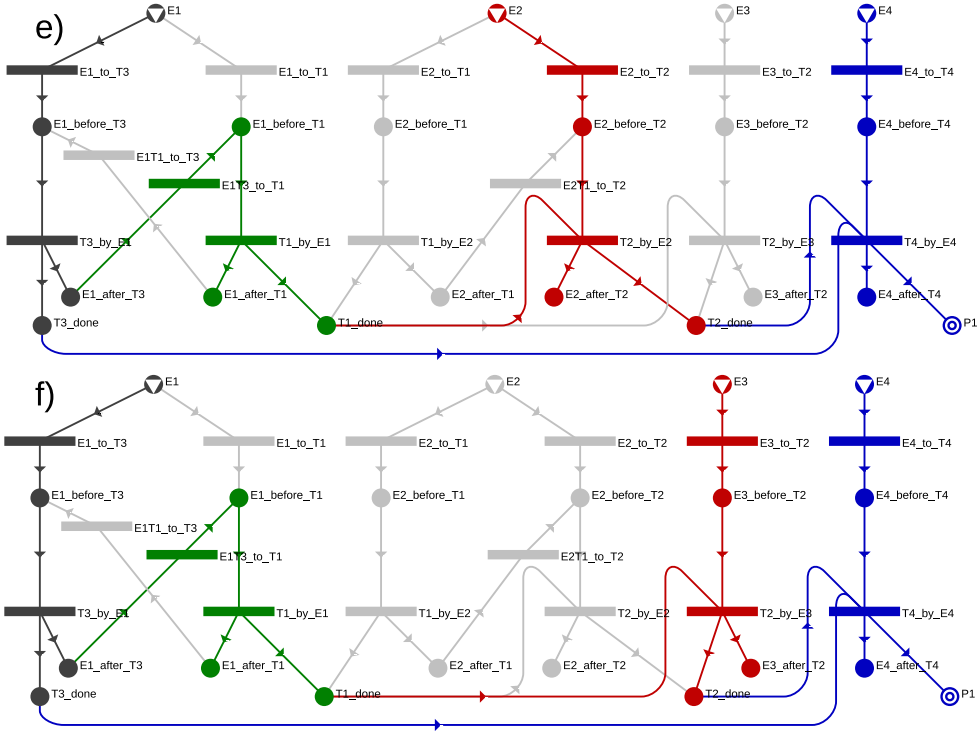
Végtelen köztes tárolók ütemezése



6.10. ábra. A b) megoldás strukturális ábrázolása

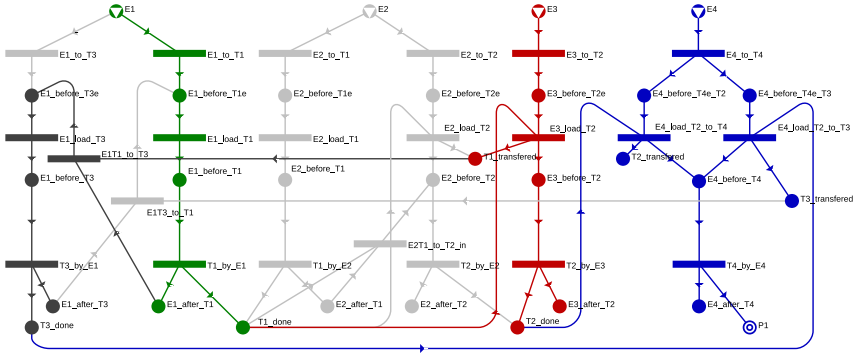


6.11. ábra. A c) és d) megoldások strukturális ábrázolása

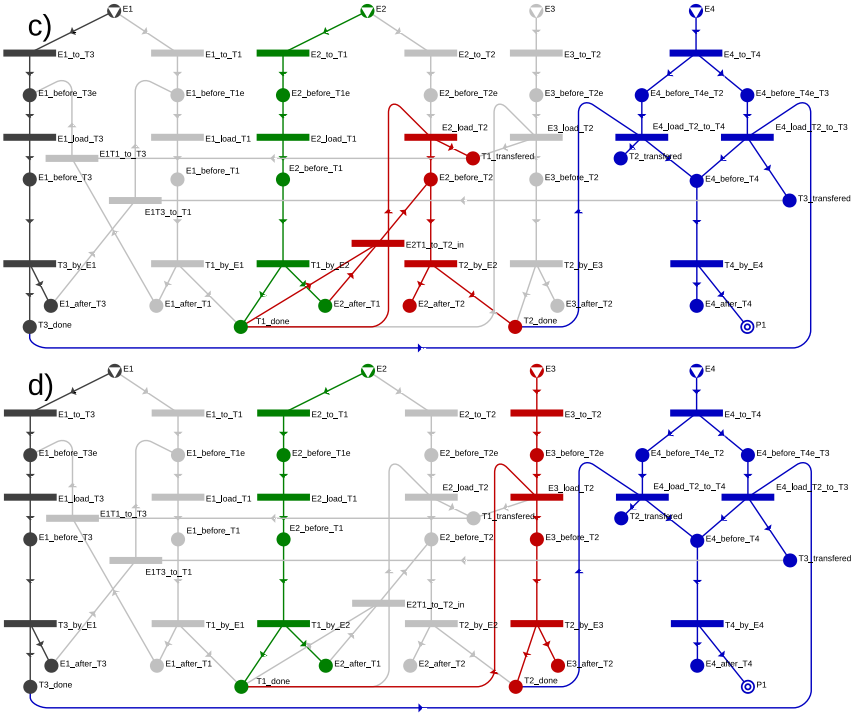


6.12. ábra. Az e) és f) megoldások strukturális ábrázolása

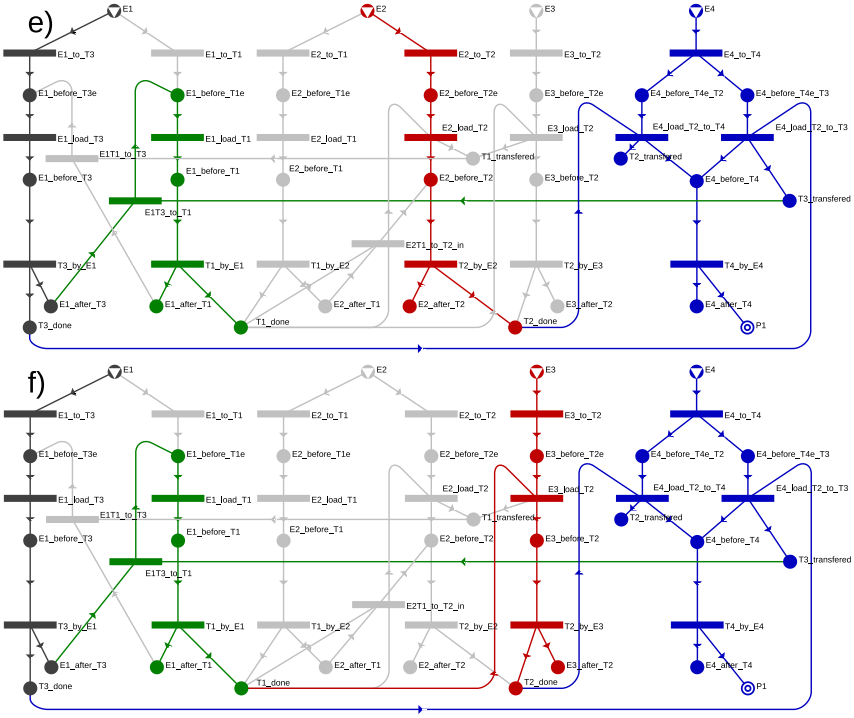
Köztes tárolók nélküli ütemezés



6.13. ábra. A b) megoldás strukturális ábrázolása

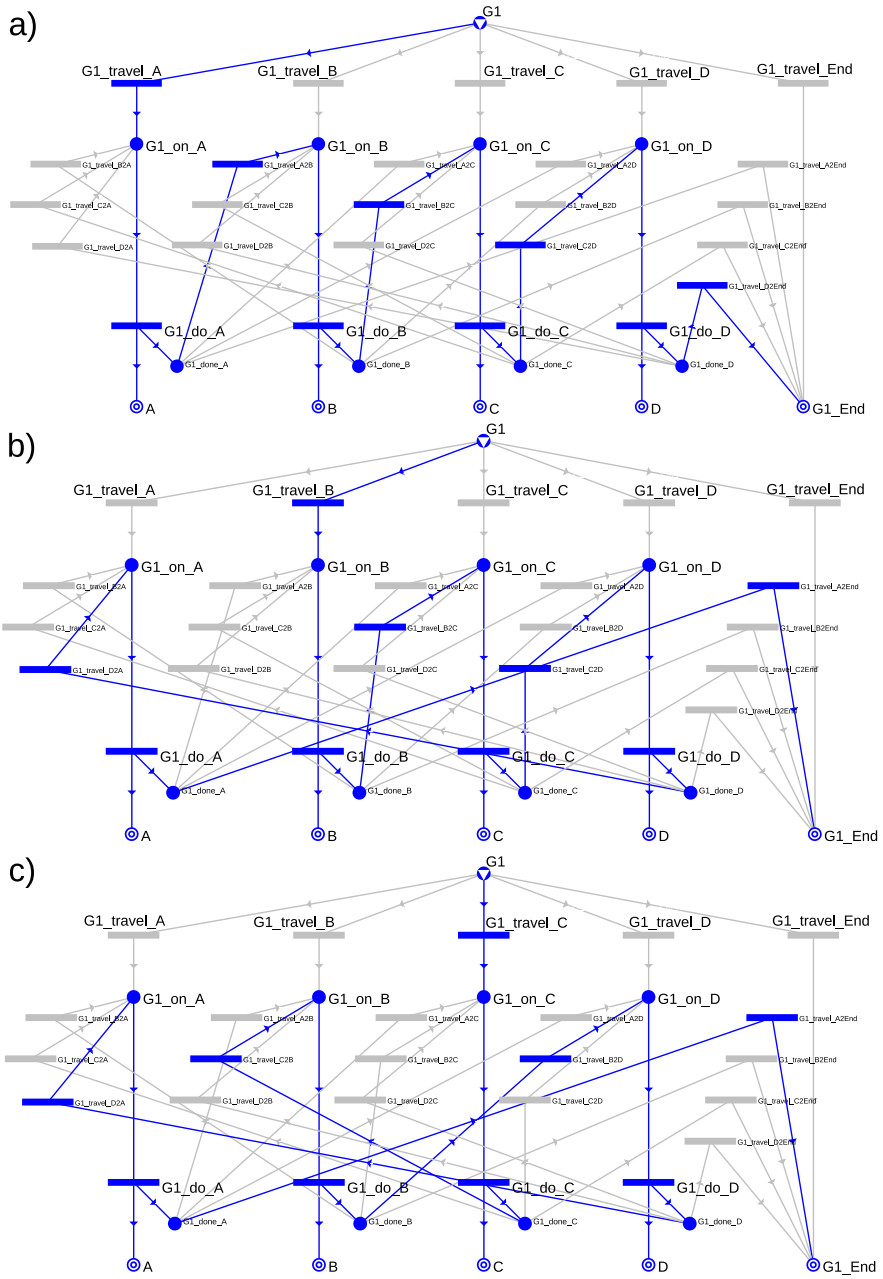


6.14. ábra. Az c) és d) megoldások strukturális ábrázolása

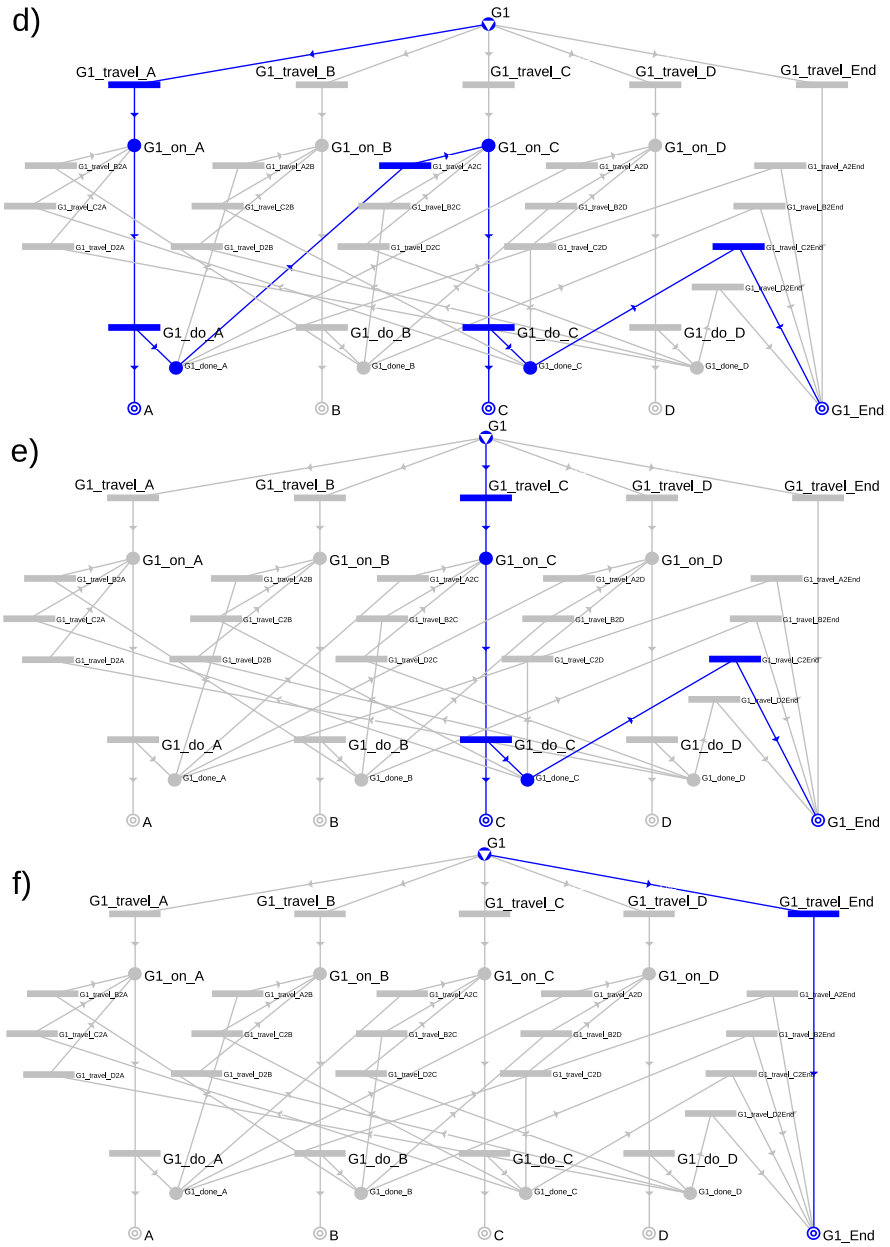


6.15. ábra. Az e) és f) megoldások strukturális ábrázolása

C) Terepi munkavégzés ütemezése



6.16. ábra. Példák megoldás struktúrákra, ahol a végrehajtási sorrend a következő: a) *A-B-C-D-End*, b) *B-C-D-A-End*, c) *C-B-D-A-End*



6.17. ábra. Példák megoldás struktúrákra, ahol a végrehajtási sorrend a következő: d) *A-C-End*, e) *C-End*, f) *End*