Megbízható folyamatok szintézise P-gráf módszertanra alapozva

DOI:10.18136/PE.2023.836

DOKTORI (PhD) ÉRTEKEZÉS

Orosz Ákos

Témavezetők:
Friedler Ferenc, DSc, egyetemi tanár
Holczinger Tibor, PhD, egyetemi docens

Pannon Egyetem
Műszaki Informatikai Kar
Informatikai Tudományok Doktori Iskola

2022

Megbízható folyamatok szintézise P-gráf módszertanra alapozva

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

informatikai tudományok tudományágban

Írta: Orosz Ákos

Témavezető/i: Ferenc Friedler, DSc, Holczinger Tibor, PhD

Elfogadásra javaslom (igen / nem)

.....
(témavezető/k)

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

Veszprém, ...
(a Szigorlati Bizottság elnöke)

Az értekezést bírálóként elfogadásra javaslom:

Bíráló neve: .... igen /nem

.....
(bíráló)

Bíráló neve: .... igen /nem

.....
(bíráló)

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

Veszprém, ...
(a Bíráló Bizottság elnöke)

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

Veszprém, ...
(az EDHT elnöke)

Tartalmi kivonat

Folyamat hálózatok tervezésénél általában többféle szempontot kell érvényesíteni a legkedvezőbb hálózat kiválasztásához a gyakran nagyszámú lehetőség közül. A legtöbb ilyen szempont matematikai formában megfogalmazható egy optimalizálási feladat célfüggvényében. Így például a működési költség vagy a fenntarthatósági indikátor jól formalizálható. Van azonban olyan indikátor, amelynek a meghatározása a lehetséges esetek leszámlálásával adható meg, ezáltal közvetlen módon általában nem építhető be a célfüggvénybe. Ilyen indikátor a folyamat hálózat megbízhatósága.

Jelen dolgozat célja olyan új módszer elméleti alapjainak a kidolgozása, amely alkalmas összetett folyamat hálózatok megbízhatóságának a meghatározására önmagában, vagy folyamat hálózatok szintézise során. Ezen célok érdekében a dolgozat három fő témakörre bontható. Az első témakör egy olyan általános megbízhatósági formula meghatározása, amellyel tetszőleges rendszer vagy folyamat hálózat megbízhatósága meghatározható. Ezt egy, a P-gráf keretrendszeren alapuló, leszámlálás elvű algoritmussal teszi meg, és kitér az algoritmus számítási hatékonyságára, valamint a gyorsítási lehetőségekre is. A második témakör a rendszerbe építhető redundanciák fajtáinak és azok hatásainak elemzése. A harmadik témakör a megbízhatóságot is figyelembe vevő folyamat hálózat szintézis, ahol a kettő kidolgozott algoritmus különböző megközelítéssel képes generálni a lehetséges folyamat-megvalósítások Pareto-görbéjét a megbízhatóság és a költség skáláján.

Abstract

A fundamental problem of process design is to determine the network of the processing system. Several properties and indicators have to be considered during the design of processing systems for selecting the best system among the vast set of possibilities. Most of these indicators can be defined by a mathematical formulation that can be directly integrated into the objective function of an optimization problem. The operating cost and the sustainability index are two examples of indicators that can have proper formulation. There are other indicators, however, such as reliability, which can only be determined by a systematic enumeration of possible cases. Thus, a general reliability formulation cannot directly be integrated into the objective function.

The aim of the current work is to introduce the theoretical basis of a general method to determine the reliability of complex process network, and to integrate it into process network synthesis. The work consists of three main topics. The first topic defines the general reliability formula to determine the reliability of any process network. This is achieved via a P-graph-based enumeration method. The topic also elaborates on the computational complexity of the method, and the possible acceleration techniques. The second topic analyzes the possible redundancy options in a process network to improve its reliability. The third topic is about process network synthesis with integrated reliability consideration. In this topic two synthesis algorithms are introduced; both are capable of generating the Pareto-front of possible process design options based on reliability and cost.

Resumen

Un problema fundamental del diseño de procesos es la definición de la red del sistema de procesamiento. Durante el diseño de estos sistemas varias propiedades e indicadores deben ser considerados para seleccionar la mejor red entre el amplio conjunto de posibilidades disponibles. La mayoría de estos indicadores se pueden definir a través de una formulación matemática, la cual puede integrarse directamente en la función objetivo de un problema de optimización. El costo de operación y el índice de sostenibilidad son dos ejemplos de indicadores que pueden formularse adecuadamente. No obstante, existen otros indicadores que sólo pueden determinarse mediante la enumeración sistemática de los casos posibles, por ejemplo, la confiabilidad. Por lo tanto, una formulación general de la confiabilidad no puede integrarse directamente en la función objetivo.

El objetivo de este trabajo es presentar la base teórica de un método general para determinar la confiabilidad de una red compleja de procesos, e integrarla en la síntesis de redes de procesos. El trabajo está conformado por tres ejes temáticos. El primero es la definición de la fórmula general para determinar la confiabilidad de cualquier red de procesos. Esto se logra a través de un método de enumeración basado en gráficos de proceso, llamados *P-graphs*. Este eje también aborda la complejidad computacional del método y las posibles técnicas de aceleración. El segundo eje temático consiste en el análisis de las posibles opciones de redundancia en una red de procesos para mejorar su confiabilidad. En el tercer eje temático se plantea la síntesis de redes de procesos integrando el criterio de confiabilidad. En este eje también se presentan dos algoritmos de síntesis; ambos capaces de generar el frente de Pareto de las posibles opciones de diseño para el proceso basados en la confiabilidad y el costo.

Köszönetnyilvánítás

Nagyon köszönöm témavezetőimnek, különösen Friedler Ferenc professzor úrnak a sok éves közös munkát, iránymutatást és támogatást. Külön köszönöm Kovács Zoltánnak, akinek jegyzetei alapján a kutatási munka elindult. Köszönöm a Rendszer- és Számítástudományi Tanszéknek a megfelelő kutatási körülmények biztosítását. Köszönöm munkatársaimnak és kutatótársaimnak a sok együttműködést és segítséget. Köszönöm családomnak és barátaimnak a kitartó támogatást és motiválást a disszertációm elkészítéséhez.

Tartalomjegyzék

Ábrák jegyzéke

Táblázatok jegyzéke

Rövidítések jegyzéke

Rövidítések

ABB	Accelerated Branch-and-Bound
BDD	Dinary Decision Diagram
MILP	Mixed Integer Linear Programming
MINLP	Mixed Integer Non-Linear Programming
MSG	Maximal Structure Generator
PNS	Process Network Synthesis
SSG	Solution Structure Generator

Algoritmusok

RBO	Megbízhatóság kiszámítása a működőképes hálózatok alapján
SON	Strukturálisan működőképes hálózatok generálása
SRP	Rendszer strukturális megbízhatóságának meghatározása

1. fejezet

Bevezetés

A mindennapi életben ipari folyamatok által előállított termékek vesznek körül minket. Így például élelmiszerek, háztartási eszközök, informatikai eszközök, járművek, vagy sok minden más. Mindezek előállítása költséggel, biztonsági kockázattal és környezetszennyezéssel jár. Ezért fontos megtalálni a sok előállítási lehetőség közül azt, amely legjobban megfelel a mindenkori, de egyre szigorodó elvárásainknak.

A megfelelő gyártási folyamat kiválasztása során a leggyakrabban vizsgált szempont a folyamat beruházási- és működtetési költsége. Azonban jobb eredmény születhet, ha a tervezés során más szempontokat is figyelembe veszünk. Ilyen szempontok például a termeléssel járó környezetszennyezés, az előforduló veszélyfaktorok, valamint a folyamat megbízhatósága. A szempontok közül jelen munka a megbízhatóság kezelését segíti elő, amely egy fontos tulajdonság, hiszen ez határozza meg a meghibásodások, illetve vészhelyzetek bekövetkezésének esélyét. Mivel a folyamat működtetése során minimalizálni szeretnénk a katasztrofális meghibásodások mennyiségét, a tervezés során oda kell figyelni a megfelelő óvintézkedések és redundanciák beépítésére. A redundancia meghatározása során ráadásul több lehetőség közül kell választani, hiszen nem mindegy, hogy a tartalék egységek hol helyezkednek el a folyamaton belül, illetve, hogy mi a helyettesítés módja: egy-egy műveleti egység helyettesítése (lokális redundancia), vagy egy teljes részrendszeré (globális redundancia).

A folyamatba épített redundanciák hatékonysága a megbízhatóság segítségével mérhető. A megbízhatóság megfelelő meghatározásához speciális algoritmusok szükségesek, amelyek képesek a lehetőségek kimerítő leszámlálására. Ez a hagyományos, matematikai optimalizáláson alapuló módszerekkel nem valósítható meg, ezért van szükség kifejezetten erre a célra megalkotott megközelítésekre.

A megbízhatóság, mint fogalom, több különféle jelentéssel is bírhat a tervezendő rendszer elvárásai alapján. Bizonyos folyamatok esetén elegendő csak a folyamatot felépítő elemek (más szóval műveletek) strukturális kapcsolatait vizsgálni, más esetekben azonban fontos a műveletek átbocsátó képességeinek az egymáshoz igazítása is.

Egy-egy termék gyártására rengeteg különböző folyamatot lehet tervezni, egy mérnök általában nem képes saját erejéből az összes lehetőséget minden szempont szerint figyelembe venni. Ahhoz, hogy a megtervezett folyamat

ténylegesen optimális lehessen a lehetőségeket és korlátozásokat figyelembe véve, informatikai eszközökre van szükség, amelyek a számítógépek hatalmas számítási kapacitását kihasználva képesek a tervezést hatékonyabbá és megalapozottabbá tenni. Természetesen ezek az eszközök önmagukban nem képesek a folyamatok megtervezésére, hanem a tervező mérnökök a saját szaktudásukat felhasználva vezérlik az optimalizáló szoftvereket, majd értelmezik és realizálják az ezek által adott eredményeket.

Egy termék előállítását egy műveletekből felépülő folyamat hálózat végzi, amelynek megfelelő megtervezése során a tervezést végző mérnöknek számos lehetőség közül kell a megfelelőket kiválasztani úgy, hogy egy minél hatékonyabb rendszert kapjon, ami megfelel minden elvárásnak és feltételnek. A megvalósítandó rendszernek nem csak a terméket vagy termékeket kell előállítani, de közben az üzemeltető által támasztott elvárásoknak is meg kell felelni. Az elvárások közül az egyik legfontosabb természetesen a hatékonyság, hiszen ez adja az üzem működtetésének létjogosultságát. Ezen felül viszont oda kell figyelni többek között a folyamat környezeti hatásaira, szennyezőanyag kibocsájtására, az üzemeltető személyzet biztonságára, illetve a nem várt meghibásodások kezelésére. A nagy számú szempont miatt a megfelelő hálózat megtervezése egy bonyolult feladat, amely a lehetőségek sokasága miatt számítógépes segítséget igényel. Az ilyen segítő módszerek megalkotása egy szerteágazó kutatási terület, amelyben bár sok áttörő eredmény született, még mindig számos megoldatlan problémával találkozhatunk. Jelen dolgozat célja a rendelkezésre álló eszközök kibővítése olyan módszerekkel, amelyek fő fókusza a folyamatok megbízhatósága.

A folyamat tervezés legfőbb lépése a műveletek optimális hálózatának meghatározása, vagy más néven folyamat hálózat szintézis (process network synthesis, PNS). Ezen a területen az 1980-as évektől kezdve nagy számban jelennek meg publikációk. Az úttörő munkák közé tartoznak Papoulias és Grossmann [1], valamint Kocis és Grossmann [2] publikációi. Szintén jelentős új eredményeket és szemléletet hozott a P-gráf módszertan megjelenése is [3].

Termék vagy termékek előállítása különböző anyagok (nyersanyagok) felhasználásával, a tulajdonságaik többlépéses átalakításával történik. Az átalakítást a termelő folyamatban egy művelet reprezentálja, amelyet a PNS-ben műveleti egységgel reprezentálunk. Egy ilyen művelet vagy műveleti egység megfelelhet egy elemi átalakítási műveletnek (például egy elemi reakció lezajlása egy reaktorban), vagy egy magasabb szintű modell esetében jelölhet egy komplett részfolyamatot is (például a gyár egyik részlegét, ami egy folyamatnak megfelelően üzemel, de a tervezés során ezt nem kívánjuk módosítani, ezért egységként kezeljük). A PNS-ben minden műveleti egység rendelkezik a saját matematikai modelljével, ami leírja az általa végzett átalakítást, ami a kimeneti anyagait generálja a bemeneti anyagok felhasználásával. Ezen felül minden műveleti egységhez tartozik egy függvény, ami megadja a műveleti egységhez tartozó célfüggvény értéket (jellemzően a költségét) a műveleti egység és a hozzá kapcsolódó anyagáramok paramétereinek megfelelően. A különböző matematikai modellek sok különböző irányból és mélységben közelítik meg a feladatot. A legegyszerűbb esetben a modell csak az anyagok közötti strukturális kapcsolatokat tartalmazza. Részletesebb feladatleírásokban a műveleti egység

modellje lehet lineáris-, vagy nem lineáris függvény. Hasonlóan, a költségfüggvény is lehet konstans, lineáris, vagy nem lineáris.

Napjainkban a lineáris modell használata a gyakori, amely esetében a műveleti egységhez egy folytonos változó, a méret tartozik. A műveleti egység által felhasznált, illetve előállított anyagok mennyisége a műveleti egység méretével egyenesen arányos. Ezen felül a költségfüggvényt is lineáris függvényként adják meg. Habár a műveleti egység modellje lehet nem-lineáris is, vagyis a műveleti egység bemeneti és kimeneti anyagpontjai között lehetnek nem-lineáris összefüggések, ez az eddigi kutatások során ritkán fordult csak elő. Ennek elsődleges oka, hogy ilyen esetben a hálózat teljes matematikai modellje is nem-lineáris lesz, aminek a megoldása jóval nagyobb kihívásokat jelent a lineáris modellekhez képest.

Egy műveleti egység csak akkor tudja a feladatát elvégezni, ha minden szükséges bemeneti anyag rendelkezésére áll a megfelelő mennyiségben. Ehhez kettő feltételnek kell teljesülni. Egyrészt a műveleti egység minden bemeneti anyaga vagy nyersanyag, vagy valamely műveleti egység kimenete. Ezen felül, amennyiben a strukturális vizsgálaton felül az anyagegyensúly modellt is kezeli a feladat, akkor a bemeneti anyagok mennyisége is számít. Ekkor az anyagok mennyiségére teljesülnie kell a folyamatot leíró anyagegyensúly korlátozásoknak. Ha a műveleti egység működéséhez rendelkezésre állnak a bemeneti anyagok, akkor a műveleti egység minden kimeneti anyagot előállít a működését leíró modellnek megfelelően. Amennyiben adott műveleti egységeknek egy halmaza, azok meghatároznak egy hálózatot az anyagok közötti strukturális kapcsolatoknak megfelelően.

A megbízhatóság egy fontos indikátora a folyamatnak, viszont ennek figyelembe vétele a tervezés során több elméleti és gyakorlati nehézséget is felvet, amelyek közül a legjelentősebb a folyamatok működőképességének kombinatorikus jellege. A gyakorlatban a folyamat hálózatok kellően nagy megbízhatóságának elérése redundáns műveleti egységek beépítésével lehetséges. Egy ilyen folyamatban több különböző műveleti egység halmaz is létezik, amelyek a működőképességet biztosítják. A megbízhatóság meghatározása történhet a folyamat hálózat működését szimuláló vagy becslő sztochasztikus algoritmusok segítségével, illetve a működés lehetséges formáinak felsorolására alkalmas kombinatorikus algoritmusokkal. Az előbbi módszerek közé tartoznak például a Monte Carlo szimuláció [4], és a Markov láncok [5]. Ezen módszerek vagy csak becsülni képesek a megbízhatóságot, vagy a használhatóságot erősen korlátozza a megoldandó feladat mérete, valamint a szintézis feladatokat önmagukban nem képesek ellátni. A felsorolás alapú módszerek elsősorban kombinatorikus algoritmusokkal lehetségesek, hagyományos, tisztán matematikai optimalizálás alapon nem lehet megközelíteni. A P-gráf keretrendszer azonban megfelelő alapot nyújthat ennek a komplex feladatnak a megoldására [6]. A jelen munkában ismertetett módszer a folyamat hálózat szintézis irányából közelíti meg a megbízhatóság számítást is, ezáltal könnyen beépíthető szintézis algoritmusokba is. A módszert ezen felül nem korlátozza a folyamat hálózat strukturális komplexitása.

Egy műszaki rendszer megbízhatóságának kiszámítása speciális esetekben megtörténhet dekompozíciós eljárással is, amely a rendszert soros-, illetve

párhuzamos részrendszerek összességére bontja fel. A megbízhatóság aztán a részrendszerek megbízhatóságainak kiszámítása alapján meghatározható [7]. Azonban folyamat hálózat szintézishez ez az egyszerű módszer több okból sem megfelelő. Egy termelő folyamatban a műveleti egységek jellemzően komplex hálózatot alkotnak, akár több visszacsatolást, hurkot, vagy párhuzamos ágat is tartalmazva. Az ilyen rendszerek általában nem bonthatóak fel soros-, vagy párhuzamos részrendszerekre.

A termelőrendszerek tervezésére szükség van egy általános algoritmusra, ami bármilyen összetett struktúrájú rendszer esetén képes a megbízhatóságot meghatározni. A folyamat tervezésben és megbízhatósági analízisben sincs olyan algoritmus, amely ezt önmagában képes elvégezni, ezért egy új módszerre van szükség, amim a két területet együtt kezeli. A dolgozat alapja az az észrevétel, miszerint egy létező axióma alapú szintézis módszertan megfelelő alapot szolgáltat a probléma megoldásához. Ez a P-gráf módszertan [3], amelyet PNS feladatokban már sikeresen alkalmaztak ipari tervezési feladatok megoldására. A módszertan megfelelő kapcsolatot tud biztosítani a folyamat szintézis és a megbízhatósági analízis fogalomrendszer között. A közös terminológia alapja az lehetségesség és működőképesség kapcsolata, ahol az előbbi, PNS-ben használt fogalom sok hasonlóságot mutat az utóbbi, megbízhatósághoz kapcsolódó kifejezéssel. Ezen észrevétel alapján szigorú matematikai modell és algoritmus definiálható.

A P-gráf módszertan kombinatorikus matematikai összefüggésekkel írható fel, és az lehetséges hálózatok kombinatorikus tulajdonságaira épít. Mivel a folyamatok PNS-ben vizsgált lehetségessége és a megbízhatósági analízisben vizsgált működőképessége között szoros összefüggés van, a két tulajdonság gond nélkül vizsgálható párhuzamosan. Ehhez azonban egy új matematikai modellek szükségesek, különösen a struktúrák ábrázolását tekintve.

A dolgozat elsődleges célja egy olyan formalizmus bemutatása, amely áthidalja a folyamat tervezés és megbízhatóság együttes kezeléséből adódó strukturális nehézségeket. A megbízhatóság kiszámítására ismertetett állítások és algoritmusok függetlenek a folyamat műveleti egységeinek matematikai modelljétől.

A dolgozat másik célja pedig a megbízhatósági szempontokat is figyelembe vevő szintézis lehetőségeinek feltárása az előbbi formalizmus segítségével. Az ismertetett algoritmusok új irányból közelítik meg a feladat megoldását, ami új lehetőségekhez vezet a tervezett hálózatok felépítését tekintve.

1.1. Folyamat hálózat szintézis

A folyamat hálózat szintézis (PNS) célja összetett folyamatok tervezése a rendelkezésre álló lehetőségekből építkezve. A feladatok jellemzően komplex struktúrával rendelkeznek, ahol a folyamat összeállításához használható alkotóelemeket a feladat maga definiálja. A szintézis során ki kell választani a lehetőségek közül azokat az elemeket, amelyekből végül a folyamat felépül, definiálni kell, hogy ezek milyen módon kapcsolódnak egymáshoz, valamint meg kell határozni ezen egységek belső konfigurációját. A témakör az 1960-as évektől van jelen, mint kutatási téma, aminek első ismert munkái közé tartozik J.J. Siirola

PhD disszertációja [8], illetve publikációja [9]. A későbbiekben a téma hatalmasra fejlődött, és ahogy a számítástechnika erősödött, úgy a számítógépek által segített folyamat tervezés is. A témában a publikációs folyamat igazán az 1980-as években indult be. Ebből az időszakból kiemelkedő Douglas könyve [10], amely egy átfogó képet ad a folyamat szintézis alapelveiről és főbb módszereiről.

A folyamat szintézis leggyakoribb megközelítési módja a matematikai optimalizálás, ahol a korai kutatásokban a vegyes-egyész lineáris programozást (Mixed-Integer Linear Programming, MILP) alkalmazták [11], [12]. Ezen felül már ekkor is elindultak a kutatások a nemlineáris programozás (Mixed-Integer Non-Linear Programming, MINLP) alkalmazására [2].

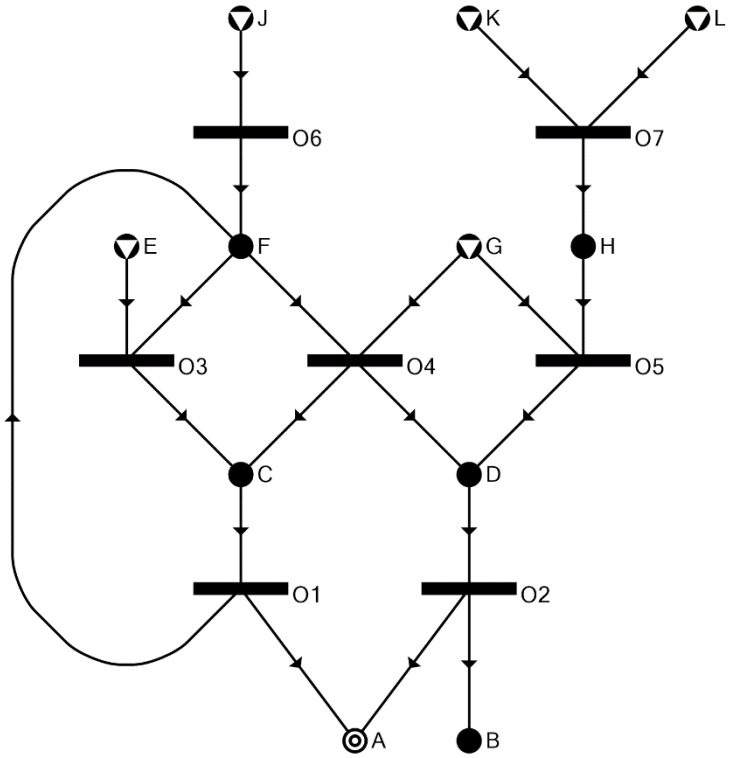
Mivel a szintézis során egy előre meghatározott halmazból válogatjuk ki a rendszer elemeit, így adja magát a szuperstruktúra elkészítése, amire aztán az optimalizálás ráépíthető. A szuperstruktúra lényege, hogy tartalmaz minden lehetséges alkotó elemet, ami szerepelhet a tervezendő folyamatban, valamint szintén tartalmaz minden lehetséges módot ezek összekapcsolására. A szuperstruktúra alapú módszerekben az optimális folyamatot a szuperstruktúra egy részeként határozzuk meg. A szuperstruktúra-alapú módszerek az 1990-es években terjedtek el igazán [13], [14], és azóta is folyamatosan jelen vannak. A P-gráf keretrendszer [15] is ezt az irányt követi, de fel lehet használni szétválasztási hálózatok optimalizálásához is [16]. Még a mai napig is jelentős mennyiségű új publikáció jelenik meg, ahol szuperstruktúra alapú módszereket alkalmaznak [17], [18].

A folyamat hálózat szintézis széles felhasználási körrel rendelkezik, amelyek között szerepelnek többek között az ütemezés [19], hőcserélő hálózatok szintézise [20], szétválasztási hálózatok megtervezése [21], vízhálózatok tervezése [22], vagy ellátási hálózatok szintézise [23]. Az eddig elért eredményekről, illetve a továbblépési lehetőségekről több értekezés is született az utóbbi években is [24]–[28].

1.2. P-gráf keretrendszer

A folyamat hálózat szintézis feladatok egyik megközelítési lehetősége a P-gráf keretrendszer [15]. A módszertan axiómákra épülő kombinatorikus algoritmusok kidolgozására ad lehetőséget, melyek során a folyamatok strukturális tulajdonságai sokkal nagyobb szerepet kapnak, mint a tisztán matematikai optimalizáláson alapuló módszerek esetén. A feladatot irányított páros gráffal ábrázolja, ahol csúcsok két halmaza a műveleti egységek, illetve az anyagok. A csúcsok közötti összeköttetések jelzik a műveleti egységek által elvégzett átalakításokat. Az 1.1. ábrán látható egy példa P-gráf 7 műveleti egységgel, 5 nyersanyaggal, valamint 1 termékkel [29]. A szintézis feladatot a (P, R, O) hármassal adjuk meg, ahol a P a termékek halmaza, az R a nyersanyagok halmaza, az O pedig a műveleti egységek halmaza. Az 1.1 ábrán látható feladatra $P = \{A\}, R = \{E, G, J, K, L\}, O = \{O1, O2, \dots, O7\}$. Formális leírás esetén feltételezzük, hogy létezik egy M^* halmaz, amely tartalmazza az összes rendelkezésre álló anyagot, amely lehet műveleti egység bemenete vagy kimenete. A szintézis feladatot egy olyan (P, R, O) hármas írja le, ahol $P \subseteq M^*$, $R \subseteq M^*$, $O \subseteq$

$\wp'(M^*) \times \wp'(M^*)$, ahol $\wp'(M^*)$ az M^* halmaz összes nem üres részhalmazának halmaza.



1.1. ábra: Példa P-gráffal leírt folyamat hálózat szintézis feladatra [29]

Azáltal, hogy a folyamatot gráfként ábrázolja a módszertan, lehetőség adódik a kombinatorika és a gráfelmélet elemeinek kiaknázására a szintézis során. Ezeken az elveken a P-gráf módszertan 5 axiómát definiál a kombinatorikusan lehetséges hálózatokra, vagy másnéven a megoldás struktúrákra. Egy (m, o) P-gráf megoldás struktúra, ha az alábbi 5 axióma teljesül:

(S1) $P \subseteq m$

(S2) $\forall X \in m, X \in R \iff$ nincs az (m, o) P-gráfban (Y, X) él

(S3) $o \subseteq O$

(S4) $\forall Y_0 \in o$, létezik az (m, o) P-gráfban $[Y_0, Y_n]$ út, ahol $Y_n \in P$

(S5) $\forall X \in m, \exists(\alpha, \beta) \in o$, ahol $X \in \alpha \cup \beta$

Az axiómákat értelmezve a következő feltételeket mondják ki egy megoldás struktúrára:

(S1) A feladatban meghatározott minden termék szerepel benne,

(S2) Egy anyagpontot akkor és csak akkor nem gyárt semmi, ha az nyersanyag,

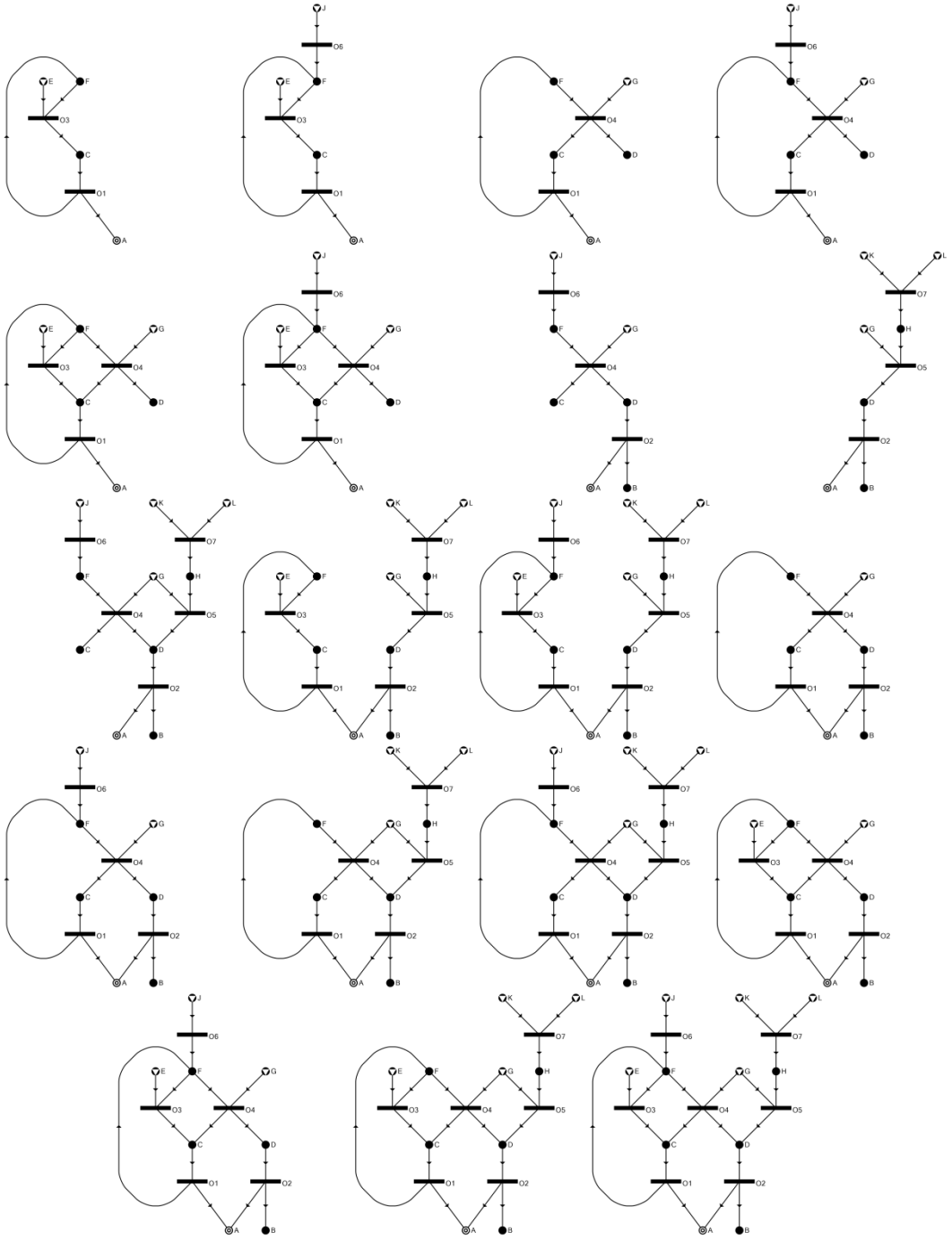
(S3) Csak olyan műveleti egységeket tartalmaz, amiket a feladat definiált,

(S4) Minden benne lévő műveleti egységből vezet út legalább egy termékhez, valamint

(S5) Csak olyan anyagot tartalmaz, ami legalább egy benne lévő műveleti egységnek bemenete vagy kimenete.

A P-gráf módszertan szerint csak olyan hálózat lehet jó megoldás egy szintézis feladatra, amely teljesíti ezt az öt axiómát; ezek a megoldás struktúrák, vagy kombinatorikusan lehetséges hálózatok. Folyamat hálózatok tekintetében a kombinatorikusan lehetséges hálózatok azok, amely strukturális felépítést

tekintve egy megvalósítható hálózatot reprezentálnak. Ezeknek egy részhalmazát adják a lehetséges hálózatok, amelyek a szintézis feladat tényleges, minden feltételt és elvárást teljesítő megoldásai. Az 1.1. ábra által definiált feladatnak 19 megoldás struktúrája van, amelyeket az 1.2. ábra sorol fel.



1.2. ábra: Az 1.1. ábrán látható feladat kombinatorikusan lehetséges részhálózatai [29]

A keretrendszer korai publikáció között három főbb algoritmus leírása jelent meg, amelyek a folyamat hálózat szintézis fontos lépéseit valósítják meg. Az első ilyen algoritmus az MSG [30], egy polinom-idejű algoritmus, amely képes a

feladat szigorú szuperstruktúráját (az úgynevezett maximális struktúrát) generálni a műveleti egységek adatai alapján. A maximális struktúra egyben a rendszer összes kombinatorikusan lehetséges struktúrájának uniója. Az SSG algoritmus [29] képes a maximális struktúra által leírt feladat összes kombinatorikusan lehetséges hálózatának generálására. Végül az ABB algoritmus [31] egy szétválasztás és korlátozás alapú algoritmus, amely képes meghatározni a feladat optimális megoldását, n-legjobb megoldását, vagy akár összes megoldását is.

A megjelenése óta a P-gráf módszertan a folyamat optimalizálás és tervezés több különböző területén is alkalmazásra került. A mai napig változatos témákban jelennek meg P-gráfot alkalmazó publikációk, ami igazolja a keretrendszer gyakorlati alkalmazhatóságát. Ilyen alkalmazási területek közé tartozik a multiperiódusos rendszerek szintézise [32]–[35], folyamatok ütemezése [36], [37], illetve szállító hálózatok tervezése és karbantartása [38]–[40].

Sok P-gráf-alapú munkában kapott fontos szerepet a fenntarthatóság, illetve annak különféle megközelítései [41]–[44]. Ebbe a témakörbe sorolhatóak a hőcserélő hálózatok [45], [46], illetve hőintegrált rendszerek [47] szintézise, az energiaellátás és energia szállítás tervezése [48]–[50], valamint a víztisztító hálózatok szintézise [51] is.

Több publikáció született, ami a létező rendszerek működése során fellépő nehézségeket vizsgálja, illetve ezeket figyelembe veszi a szintézis során. Ilyen tulajdonságok az elindíthatóság [52], a megbízhatóság [53], illetve a reziliencia [54].

A felsoroltakon kívül még nagy számmal vannak P-gráf alapú publikációk, és egyéb érintett témakörök is. Részletesebb ismertetést tudnak adni az áttekintő cikkek [26], [55], valamint a nemrégiben megjelent könyv [56] is.

A jelen dolgozatban ismertetett módszerek alkalmazzák az MSG és SSG algoritmusokat, a következőkben ezek rövid ismertetése található. A részletes formális leírások nem részei ennek a dolgozatnak, ezek megtalálhatóak az algoritmusokat közlő szakirodalomban [30], [29], valamint a korábban említett könyvben is [56].

Az MSG (Maximal Structure Generator) algoritmus célja a szintézis feladat maximális hálózatának meghatározása. A maximális hálózat az összes kombinatorikusan lehetséges részhálózat uniója, ami maga is egy kombinatorikusan lehetséges részhálózat. Az MSG algoritmus a működése során szűkítheti a szintézis feladatban definiált műveleti egységek halmazát, amennyiben ez szükséges.

Az algoritmus működése kettő fázisból áll. Az első fázis (redukció) eltávolítja azokat az műveleti egységeket a feladat leírásban megadott halmazból, amelyek a feladatban megadott nyersanyagok alapján nem képesek működni. A második fázis (konstrukció) felépíti a maximális hálózatot a megmaradt műveleti egységekből oly módon, hogy csak olyan műveleti egység kerüljön a maximális hálózatba, amelyik hasznos a termékek gyártása szempontjából.

Az SSG (Solution Structure Generator) algoritmus célja a kombinatorikusan lehetséges hálózatok generálása a maximális hálózatból kiindulva. Az algoritmus egy rekurzív kereső algoritmus, amely a keresési fát az alapján építi fel, hogy az

egyes anyagokat mely műveleti egységek gyártják. A keresési fa csúcsaiban mindig kiválaszt egy anyagot, amiről még nem hozott döntést, és az anyagot gyártó műveleti egységek alapján választja szét a feladatot részfeladatokra. Akkor áll le egy keresési irány, ha már nincs olyan anyag, amiről döntést kell hozni. Amennyiben a feladat nem definiál az 5 axiómán kívül egyéb feltételt a működőképességre, akkor a keresési fa minden levele egy kombinatorikusan lehetséges hálózat lesz.

1.3. Komplex rendszerek megbízhatósága

A megbízhatóság, mint fogalom több területen is értelmezhető gyakorlatilag ugyanazzal a jelentéssel. Azonban a tényleges definíció, és a meghatározás módja jelentősen változhat az alkalmazási terület függvényében. Például az építőiparban a megbízhatóság jellemzően az épület stabilitásával függ össze, egy darab építőelem (például) gerenda esetében a teherbírással hozható kapcsolatba, míg egy műszaki rendszernél a rendszer működőképességével a komponensek meghibásodásainak függvényében. A dolgozat műszaki rendszerekkel foglalkozik, ezért rendszer alatt minden esetben ezt fogjuk érteni.

Az a kérdés, hogy miként lehet egy rendszer megbízhatóságát meghatározni már szintén egy többévtizede kutatott téma. Ennek első lépése az volt, hogy miként lehet egy egység vagy eszköz megbízhatóságát meghatározni [57], majd elkezdődtek a kutatások a több elemi eszköz összességéből álló hálózatok, folyamatok megbízhatósági analízisére is [58], [59].

Manapság összetett rendszerek megbízhatóságának meghatározására két főbb irányvonal létezik: a determinisztikus és a sztochasztikus módszerek. Determinisztikus módszerről akkor beszélünk, amikor a számítás képes a rendszer tényleges megbízhatóságát meghatározni. Egyszerűbb felépítésű rendszerek esetén, mint a soros-, vagy párhuzamos struktúrával rendelkező rendszerek, gyakran még zárt képletek is alkalmazhatóak [7], [60]. Bonyolultabb felépítésű rendszerek esetében azonban a régóta ismert zárt képletek nem használhatóak. Ezzel szemben a sztochasztikus megközelítések jellegükből adódóan tetszőleges rendszerekre használhatóak, cserébe a megbízhatóságot nem pontosan, hanem csak valamilyen közelítéssel képesek megadni [61]–[63].

Léteznek determinisztikus algoritmusok, amelyek képesek bizonyos típus rendszerek megbízhatóságainak meghatározására. Például kommunikációs, vagy szállító hálózatok esetén alkalmazhatóak minimális elvágó halmazokon [64], [65], vagy minimális útvonalakon [66] alapuló módszerek. Az szakirodalomban fellelhető módszerek közül ezek hasonlítanak legjobban a jelen dolgozat által leírt algoritmushoz, azonban hátrányuk, hogy speciálisan csak hagyományos hálózatok esetén működőképesek. Hasonló elvet alkalmaznak a hibafák és a sikerfák [67], [68] is, amelyek elvileg bármilyen rendszerre alkalmazhatóak, azonban a hibafák, illetve sikerfák felírása emberi feladat. Ez a módszer így nem teljesen automatikus, hiszen a megbízhatóságot nem a rendszer modelljéből határozza meg, hanem egy attól különböző, emberi munkával elkészített hibafa modellből. További speciális algoritmusok alkalmazhatóak olyan esetben, amikor a rendszerben csak műveleti egység szintű redundancia található [60]. Ennek egy általánosabb esete a k-out-of-n rendszerek, amikor az n darab párhuzamos,

redundáns műveleti egységből legalább *k* darab működése szükséges [69], [70]. Azonban egyik determinisztikus algoritmus sem képes a megbízhatóságot automatikusan, a rendszer modellje alapján meghatározni.

Hasonló csoportosítás figyelhető meg akkor is, amikor a megbízhatóságot folyamat tervezésre során is figyelembe veszik. Míg egyszerűbb struktúrájú rendszerek esetén használhatóak zárt képletek [60], [71], addig bonyolultabb struktúrák esetén (vagy akár csak komplex redundancia és karbantartási szabályok mellett) már sztochasztikus módszereket alkalmaznak [72], [73].

Azon felül, hogy egy adott rendszer megbízhatósága hogyan határozható meg, az is fontos kérdés, hogy miként lehet a megbízhatóságot a folyamat tervezés során is figyelembe venni. Az egyik leggyakrabban vizsgált feladat a redundancia-optimalizálás [74], [75], ahol a rendszer megbízhatóságának növelését a rendszer elemeinek, műveleti egységeinek többszörözésével érik el. Ezt a fajta megközelítését a redundanciának már P-gráf alapokon is vizsgálták, ahol a hiba- és siker-fák, valamint az azokon keresett elvágó halmazok elvét követve számították ki a folyamat megbízhatóságát, és kiegészítették műveleti egység szintű redundanciával [53], [76]. A folyamat szintű redundanciákat az irodalomban csak nagyon ritkán és felületesen kezelik, egyszerűsített modellel és folyamattal [60], aminek fő oka ezek strukturális komplexitása.

A megbízhatóság és a szintézis együttes kezelése a legtöbb esetben két fázisban zajlik, ahol a megbízhatósági analízis ismételt meghívásával értékelik ki a potenciális megoldásokat. Erre a megbízhatósági analízisre gyakran használnak Markov-láncokat [77], illetve Monte-Carlo szimulációt [78]. Ezen felül gyakori még a heurisztika-alapú szintézis [79], [80].

2. fejezet

Folyamat hálózatok megbízhatósága

Ahogy a bevezetés is bemutatta, folyamat hálózatok megbízhatóságának meghatározására több módszer is ismert. Determinisztikus módszerek azonban csak speciális struktúrájú hálózatokra léteznek, általános esetekben sztochasztikus módszert alkalmaznak. A sztochasztikus módszerek jellemzően közelítő eredményt adnak, ami annál pontosabb, minél több időt tölt a módszer a számítással. Jelen dolgozat a determinisztikus módszerekkel foglalkozik. A dolgozat célja egy olyan determinisztikus módszer ismertetése, amely szisztematikusan képes tetszőleges struktúrájú hálózat megbízhatóságának pontos meghatározására.

Jelen munkában kihasználjuk a P-gráf módszertan kombinatorikus felépítését a komplex struktúrájú műszaki termelő rendszerek modellezésére. A P-gráf módszertannal leírt rendszerek megbízhatósága kombinatorikus algoritmusokkal meghatározható, függetlenül a struktúra bonyolultságától. Ezen felül a módszertan által biztosított algoritmusok segítségével a rendszer működőképességét nem csak strukturális szempontból, hanem anyagmérleg tekintetében is vizsgálhatjuk, valamint lehetőséget adunk komplex, tervezői feltételek figyelembe vételére.

Ez a fejezet a megbízhatóság meghatározásának részleteit tárgyalja. Az első alfejezet ismerteti az általános rendszer megbízhatósági formulát, ami tetszőleges rendszerek, folyamatok esetén használható. A második tartalmazza a számítási algoritmust. A harmadik alfejezet egy gázszállító hálózat esettanulmány több szempontból történő vizsgálatán keresztül szemlélteti a módszer használhatóságát. A negyedik alfejezet kiegészíti a módszert az anyagmérleg, illetve a folyamat matematikai modelljének figyelembe vételével. A negyedik alfejezet összefoglal lehetséges módszereket a számítási folyamatok gyorsítására.

2.1. Megbízhatósági formula

Folyamat hálózatok megbízhatóságának precíz meghatározása általános módszerrel csak akkor valósítható meg, ha az összes lehetséges működési mód megtalálását és felsorolását biztosítani lehet. Ehhez mindenképp kombinatorikus leszámláló algoritmusokra van szükség. Ezen algoritmusok esetén a leszámlálás alapját a strukturális tulajdonságok adják, így a jelenlegi munka strukturális vizsgálatokra építi a megbízhatóság meghatározását.

Az ismertetett módszer általános, tetszőleges folyamatra alkalmazható. A műveleti egységek matematikai modelljei nem korlátozzák az algoritmus működését, tetszőleges modell beépíthető. Ennek megfelelően a formula

bemutatása során a legegyszerűbb modellel dolgozunk. A módszer bármilyen, P-gráffal leírható rendszerre alkalmazható, nem korlátozza a rendszer struktúrájának bonyolultsága, lehetnek benne elágazások, csatlakozások, hurkok, komplex kapcsolatok is.

A rendszert vagy folyamatot egy P-gráffal ábrázoljuk, ahol a műveleti egységek jelölik a rendszer elemeit. Minden műveleti egységhez tartozik egy paraméter, ami az adott egység megbízhatóságát írja le. Ez az érték a műveleti egység megbízhatósága, és az i műveleti egységhez p_i-vel jelöljük ($i \in \{1,2, \dots, n\}, 0 \leq p_i \leq 1$). A számítási módszer feltétele, hogy a műveleti egységek meghibásodásai egymástól függetlenek. Összefüggő meghibásodások esetén szükség lenne az elemi okok meghatározására, ezt jelen munka nem tárgyalja. A műveleti egység megbízhatósága egy konstans érték, ami a rendszer működése során állandó. Az érték azt határozza meg, hogy mi annak a valószínűsége, hogy az adott műveleti egység képes az elvárt feladatát az adott időintervallum teljes hossza alatt megfelelő mértékben és minőségben elvégezni.

Egy adott pillanatban, a folyamat hálózat minden műveleti egysége vagy működőképes, vagy nem. Egy műveleti egység működőképességére vonatkozó pillanatnyi állapotát bináris változó határozza meg. Az i műveleti egység állapotát az x_i bináris változó jelzi, ahol

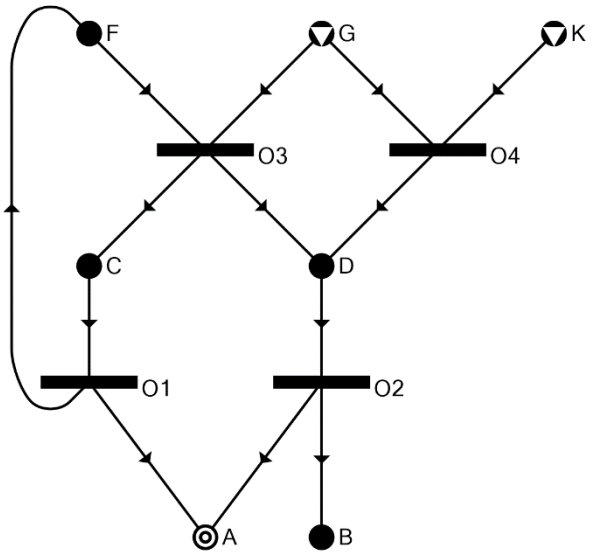
$$x_i = \begin{cases} 1, & \text{ha a műveleti egység működőképes} \\ 0, & \text{ha a műveleti egység nem működőképes} \end{cases} \quad (2.1)$$

Egy n műveleti egységet tartalmazó rendszer állapotát egy olyan n elemű, bináris vektor írja le, amelynek elemei a rendszer műveleti egységeinek állapotai: $X = (x_1, x_2, \dots x_n)$. A rendszer minden állapota egyértelműen megfeleltethető a működőképes műveleti egységek által meghatározott hálózatnak. Ez a hálózat mindig része az összes műveleti egység által meghatározott hálózatnak.

A rendszer egy adott állapota által leírt hálózat lehet működőképes, vagy nem működőképes. Itt fontos megkülönböztetni a működőképesség és a strukturális működőképesség fogalmát. A működőképesség figyelembe veszi a matematikai modell által definiált összes korlátozást és követelményt, közte az anyagmodellt is. Egy hálózat működőképes, ha képes minden terméket a *megfelelő mennyiségben és minőségben* gyártani. Ezzel szemben strukturális működőképesség esetében csak a rendszer szerkezete érdekes, más szóval az egységeinek strukturális kapcsolatai. Ekkor egy hálózat strukturálisan működőképes, amennyiben a felépítés szerkezete lehetővé teszi a működést. P-gráfos terminológiával ez azt jelenti, hogy van a hálózatnak egy olyan része, amely teljesíti az lehetséges hálózatok kombinatorikus axiómáit.

Például a 2.1. ábrán látható hálózat teljesíti a strukturális feltételeket, amennyiben $X = (1,0,1,0)$, ellenben nem, ha $X = (1,1,0,0)$, feltéve, hogy a műveleti egységek sorrendje a vektorban O1, O2, O3, O4. A rendszer működőképessége a benne lévő egységek állapotától függ, ez az n paraméterrel rendelkező Ψ függvénnyel írható le:

$$\Psi(x_1, x_2, \dots, x_n) = \begin{cases} 1, & \text{ha a rendszer működőképes} \\ 0, & \text{ha a rendszer nem működőképes} \end{cases} \quad (2.2)$$



2.1. ábra: Példa hálózat a megbízhatósági fogalmak szemléltetésére

Jelölje Ω a Ψ függvény értelmezési tartományát, vagyis az összes n hosszú bináris vektor halmazát.

$$\Omega = \{0,1\}^n = \{(0,0,\dots,0),(0,0,\dots,1),\dots,(1,1,\dots,1)\} \tag{2.3}$$

Az Ω halmaz minden elemének megfeleltethető a rendszer egységeinek egy részhálózata. Az Ω halmaz felbontható két diszjunkt részre, a működőképes, illetve a nem működőképes részhálózatok halmazaira. A valószínűség számítással való konzekvencia érdekében a rendszer egy állapota nevezhető a rendszer egy eseményének is, vagy röviden rendszer-eseménynek. Jelölje U az Ω halmaz működőképes elemeinek halmazát:

$$U = \{(x_1, x_2, \dots, x_n) \in \Omega : \Psi(x_1, x_2, \dots, x_n) = 1\} \tag{2.4}$$

A rendszer $\hat{r}$ megbízhatósága annak a valószínűségét jelöli, hogy a rendszer működőképes. Ez az U halmaz valószínűsége, vagyis $\hat{r} = P(U)$. Az i műveleti egység megbízhatóságát p_i jelöli, vagyis p_i a valószínűsége annak, hogy $x_i = 1$.

$$\begin{aligned} p_i &= P(x_i = 1) \\ 1 - p_i &= P(x_i = 0) \end{aligned} \tag{2.5}$$

Az egységek meghibásodásainak függetlenségéből adódóan egy $(b_1, b_2, \dots, b_n)$ bináris vektorral jelölt rendszer esemény ($b \in \{0,1\}, i = 1,2,\dots,n$) valószínűsége:

$$P\big((x_1, x_2, \dots, x_n) = (b_1, b_2, \dots, b_n)\big) = \prod_{i=1}^{n} {p_i}^{b_i}(1 - p_i)^{(1-b_i)} \tag{2.6}$$

A rendszer eseményei kölcsönösen kizáróak, ezért az U halmaz valószínűsége, vagyis a rendszer megbízhatósága, a benne lévő események valószínűségeinek összegével egyenlő:

$$\hat{r} = P(U) = \sum_{(x_1,x_2,\dots,x_n)\in U} \left(\prod_{i=1}^{n} {p_i}^{x_i}(1 - p_i)^{(1-x_i)} \right) \tag{2.7}$$

A rendszer $\hat{r}$ megbízhatósága így egyértelműen meghatározható tetszőleges $p_1, p_2, \dots, p_n$ értékekre $(0 \le p_i \le 1, i = 1,2,\dots,n)$, amennyiben a működőképes részhálózatok halmaza adott. Ezért az (2.7) formula a *rendszer megbízhatósági formula*.

A következőkben tekintsünk néhány példát a módszer alkalmazására speciális felépítésű rendszerek megbízhatóságának meghatározására. A kapott eredményeken látható, hogy ugyanazt adják, mint az ezen hálózatokra létező zárt képletek [7], [81]. Az első példa egy olyan rendszer, ahol a műveleti egységek párhuzamosan vannak egymással. Egy ilyen rendszer egészen addig működőképes, amíg bármelyik műveleti egysége működőképes, vagyis:

$$\begin{gathered} U = \left\{ (x_1, x_2, \dots, x_n) \in \Omega : \sum_{i=1}^{n} x_i \geq 1 \right\} \\ P(U) = 1 - P(\Omega \setminus U) \\ \Omega \setminus U = \{(0,0,\dots,0)\} \\ P(\Omega \setminus U) = \prod_{i=1}^{n} (1 - p_i) \\ \hat{r} = P(U) = 1 - \prod_{i=1}^{n} (1 - p_i) \end{gathered} \tag{2.8}$$

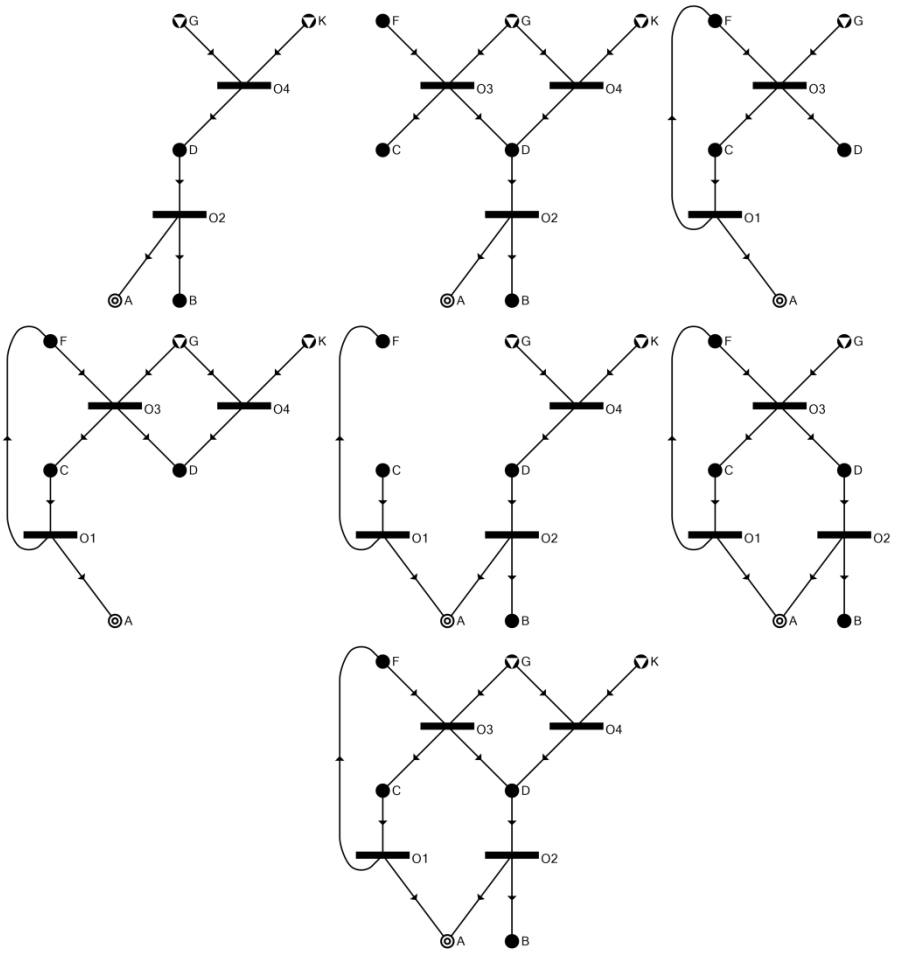
Egy másik egyszerű eset, amikor nincs a rendszerben redundancia, vagyis minden műveleti egységre szükség van a rendszer működéséhez. Ilyenek a soros rendszerek, ahol:

$$\begin{gathered} \Psi(x_1, x_2, \dots, x_n) = x_1 \cdot x_2 \cdot \dots \cdot x_n \\ U = \{(1,1,\dots,1)\} \\ \hat{r} = P(U) = \prod_{i=1}^{n} p_i \end{gathered} \tag{2.9}$$

Egyéb speciális struktúrájú rendszerek megbízhatóságának meghatározására is léteznek módszerek, azonban nincs olyan általános módszer, amely algoritmikus módon képes tetszőleges struktúrájú rendszer megbízhatóságát megadni. Az (2.7) formula megadja a számítás módját, amennyiben az U halmaz ismert. A nehezebb feladat a működőképes struktúrák algoritmikus felsorolása, amely megfelelő módszertan hiányában nem működik. Ennek részletezése előtt egy példán szemléltetjük a formula működését.

2.1. szemléltető példa

A példához tartozó hálózat a 2.1. ábrán látható. A hálózat akkor működőképes, ha az A-val jelölt terméket gyártja. A G és K csúcsok jelölik az elérhető nyersanyagokat. A termék gyártásához nincs szükség az összes műveleti egységre. Például, ha csak az O2 és O4 műveleti egységek működőképesek (vagyis $X = (0,1,0,1)$), a fennmaradó hálózat továbbra is képes az A terméket gyártani. Azonban, ha mind az O1 és az O2 műveleti egységek meghibásodnak, a rendszer nem lesz működőképes. A 2.2. ábra felsorolja az összes működőképes részhálózatot.



2.2. ábra: A 2.1. ábrán látható hálózat összes működőképes részhálózata

A működőképes részhálózatok halmazát fel lehet írni az U halmazzal is, mint $U = \{(0,1,0,1), (0,1,1,1), (1,0,1,0), (1,0,1,1), (1,1,0,1), (1,1,1,0), (1,1,1,1)\}$. A (2.7) formulát alkalmazva a rendszer megbízhatósága meghatározható: $\hat{r} = (1 - p_1) * p_2 * (1 - p_3) * p_4 + (1 - p_1) * p_2 * p_3 * p_4 + p_1 * (1 - p_2) * p_3 * (1 - p_4) + p_1 * (1 - p_2) * p_3 * p_4 + p_1 * p_2 * (1 - p_3) * p_4 + p_1 * p_2 * p_3 * (1 - p_4) + p_1 * p_2 * p_3 * p_4$. Ha például $p_1 = p_2 = 0{,}90$ és $p_3 = p_4 = 0{,}95$, akkor a rendszer megbízhatósága 0,9789. Ha $p_1 = 0{,}95$, $p_2 = 0{,}98$, $p_3 = 0{,}96$, és $p_4 = 0{,}99$, akkor a rendszer megbízhatósága 0,9974.

Amennyiben a rendszer működőképessége nem csak a strukturális kapcsolatoktól függ, hanem egyéb követelmények is vannak, akkor a működőképes hálózatok halmaza tovább szűkíthető. Ilyenre lehet példa a 2.1. szemléltető példa esetében, ha a termékre van egy mennyiségi elvárás, például naponta legalább 100 kg gyártása. Az egyes műveleti egységeknek lehet egy kapacitása, így előfordulhat, hogy az O1 és O2 műveleti egységek külön csak legfeljebb 70-70 kg kibocsájtással rendelkeznek. Ez esetben csak azok a hálózatok lehetnek működőképesek, amelyekben az O1 és O2 műveleti egység egyszerre működőképes (és a bemeneteik is rendelkezésre állnak), így az U halmaz kisebb lesz: $U = \{(1,1,1,0), (1,1,1,1)\}$. Ennek megfelelően a megbízhatóság is csökken: $\hat{r} = p_1 * p_2 * p_3 * (1 - p_4) + p_1 * p_2 * p_3 * p_4 = 0{,}8938$.

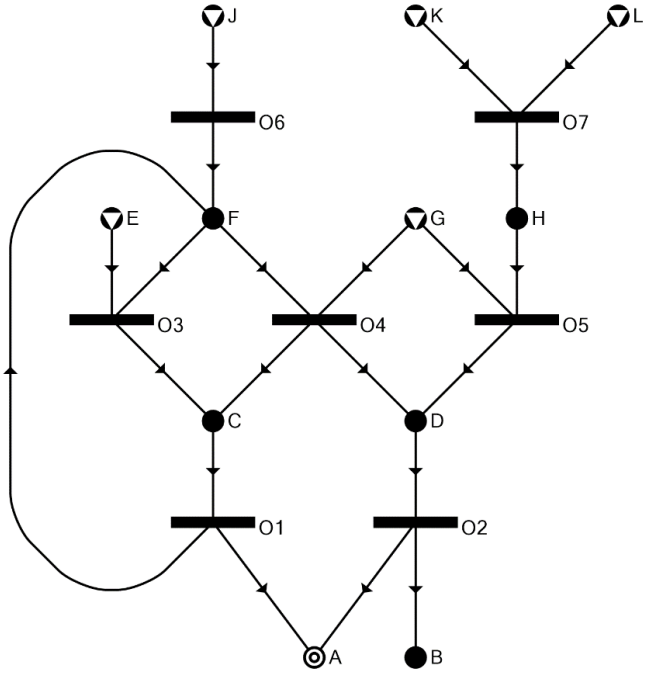
Olyan kisméretű, egyszerű rendszerekre, mint ami a 2.1. szemléltető példában szerepelt, a működőképes részhálózatok halmaza egyszerűen meghatározható. Nagyobb méretű, komplex kapcsolatokat tartalmazó hálózatokhoz viszont általános, szisztematikus módszerre van szükség, ezt mutatjuk be a következőkben.

2.2. Folyamat hálózatok megbízhatósága: strukturális eset

A 2.1. szemléltető példa megmutatta, hogy amennyiben a működőképes részhálózatok halmaza ismert, a rendszer megbízhatósága könnyen meghatározható az (1.7) formula segítségével. A következő kérdés, hogy miként lehet ezt a halmazt algoritmikus módon meghatározni. Ehhez az Ω halmazból kell a működőképes hálózatokat kiválogatni, amihez a P-gráf módszertan biztosít megfelelő eszközöket. Ez az alfejezet a strukturális megbízhatóság meghatározását ismerteti, a működési megbízhatóság kezeléséhez kellő kiegészítéseket egy későbbi fejezet tartalmazza.

2.2. szemléltető példa

A módszer szemléltetése egy irodalomban található folyamat hálózaton [29] keresztül történik, ami a 2.3. ábrán látható. A folyamat hét műveleti egységből áll, és egy terméket (A csúcs) gyárt 5 nyersanyag felhasználásával (E, G, J, K és L csúcsok). A hálózatnak 128 részhálózata van, amelyekből 19 elégíti ki a P-gráf keretrendszer által definiált 5 axiómát, ezek a kombinatorikusan lehetséges hálózatok, amik az 1.2. ábrán láthatóak. A P-gráf keretrendszerben található SSG [29] algoritmus alkalmas a struktúrák generálására.



2.3. ábra: A 2.2. szemléltető példa folyamat hálózata

Minden lehetséges hálózat egyben kombinatorikusan lehetséges is. Fordítva ez nem igaz (hiszen lehetnek nem strukturális jellegű megkötések), ezért jelen fejezet csak a strukturális korlátozásokat vesz figyelembe az lehetségességhez és működőképességhez. Ekkor minden kombinatorikusan lehetséges hálózat működőképes. Továbbá, egy működőképes hálózat kiegészítése új művelet egységekkel nem befolyásolja a termékek gyártásához szükséges strukturális követelményeket, ezért az így kapott hálózat is működőképes lesz. Ezek alapján megfogalmazható a működőképesség feltétele: egy hálózat működőképes, amennyiben van kombinatorikusan lehetséges részhálózata.

A működőképes részhálózatok U halmaza meghatározható az Ω halmaz minden elemének tesztelésével. Amennyiben a teszt pozitív eredményt ad, az adott hálózat az U halmazba kerül, egyébként nem. A tesztre a P-gráf keretrendszer biztosít megfelelő eszközöket.

A maximális hálózat az összes kombinatorikusan lehetséges részhálózat uniója. Ezen felül egy hálózat működőképes, amennyiben van kombinatorikusan lehetséges részhálózata. Ezekből következik, hogy egy, az Ω halmazban lévő részhálózat teszteléséhez azt kell megvizsgálni, hogy a hozzá tartozó maximális hálózat létezik-e, vagy üres. Amennyiben létezik, a részhálózat működőképes lesz, és az U halmazba kerül. Ha egy részhálózathoz tartozó maximális hálózat üres, az a részhálózat nem működőképes. A P-gráf keretrendszerben lévő MSG algoritmus [3], [30] képes meghatározni a maximális hálózatot, és ezzel együtt annak létezését is tetszőleges hálózatokra, így az Ω halmaz elemeinek tesztelése ezzel az algoritmussal történik. Érdemes megemlíteni, hogy az MSG algoritmus működése polinomiális komplexitású a műveleti egységek számához mérve.

Az Ω halmaz minden elemének tesztelése az MSG algoritmus segítségével meghatározza a működőképes részhálózatok U halmazát. Az MSG algoritmus hatékony, ennek ellenére a folyamat sok számítást igényelhet, amennyiben az Ω halmaznak sok eleme van (a rendszerben sok a műveleti egység). Jelen fejezet célja megmutatni, hogy a probléma megoldható algoritmikus módon. A hatékonyság növeléséről később lesz szó.

A bevezetett elvek alapján felírható a megbízhatóság meghatározására alkalmas általános eljárás. Az eljárás a P-gráf keretrendszer algoritmusaiból, valamint új algoritmusokból áll össze. Az első algoritmus, SRP, felügyeli a teljes számítási folyamatot. Az SRP algoritmus meghívja az MSG, SON és RBO algoritmusokat. Ezek formális leírásai találhatóak lentebb.

1. Algoritmus: A folyamat strukturális megbízhatóságának meghatározására

SRP algoritmus

Bemeneti adatok:

Folyamat hálózat a következő formában:

P: A termékek halmaza: a folyamat által elérendő célok

R: A nyersanyagok halmaza: a folyamat kiinduló pontjai

O: A műveleti egységek halmaza

A P, R, és O halmazok globális adatok az összes algoritmus számára

$\boldsymbol{p} = (p_1, p_2, \dots, p_n)$: p_i az i műveleti egység megbízhatósága

Kimenet:

A folyamat $\hat{r}$ megbízhatósága

Egyéb adatok:

U: A strukturálisan működőképes részhálózatok halmaza, az SON algoritmus kimenete

begin

MSG(O)

SON(O, U)

if $U \neq \emptyset$ **then**

megáll (nincs strukturálisan működőképes részhálózat)

end if

RBO($O, U, \boldsymbol{p}, \hat{r}$)

$\hat{r}$ kiírása a kimenetre

end

2. Algoritmus: A maximális hálózat generálására [3]

MSG algoritmus (meghívás: MSG(O))

Bemeneti adatok:

O: A műveleti egységek halmaza

Kimenet:

O: A maximális hálózatot leíró halmaz (az algoritmus a bemeneti halmazt szűkíti le a maximális hálózatra)

begin

Az eljárás részletes leírása a [3] hivatkozásban található

end

3. Algoritmus: A strukturálisan működőképes részhálózatok generálására

SON algoritmus (meghívás: SON(O, U))
Bemeneti adatok:
O: A műveleti egységek halmaza
Kimenet:
U: A strukturálisan működőképes részhálózatok halmaza
Egyéb adatok:
n: A műveleti egységek száma ($n = |O|$)
O_X: műveleti egységek egy halmaza ($O_X \subseteq O$)
$\boldsymbol{x} = (x_1, x_2, \dots, x_n)$: bináris vektor

begin
$U = \emptyset$
for all $\boldsymbol{x} \in \{0,1\}^n$:
$O_X = \{o_i : o_i \in O \text{ and } x_i = 1 \ \forall i = 1, 2, \dots, n\}$
MSG(O_X)
if $O_X \neq \emptyset$ **then**
$U = U \cup \{\boldsymbol{x}\}$
end if
end for
end

4. Algoritmus: A megbízhatóság ($\hat{r}$) meghatározására

RBO algoritmus (meghívás: RBO($O, U, \boldsymbol{p}, \hat{r}$))
Bemeneti adatok:
A műveleti egységek O halmaza
A strukturálisan működőképes részhálózatok U halmaza
$\boldsymbol{p} = (p_1, p_2, \dots, p_n)$: p_i az i műveleti egység megbízhatósága
Kimenet:
$\hat{r}$: A megbízhatóság
Egyéb adatok:
n: A műveleti egységek száma ($n = |O|$)
$\boldsymbol{x} = (x_1, x_2, \dots, x_n)$: bináris vektor

begin
$\hat{r} = 0$
for all $\boldsymbol{x} \in U$:
$\hat{r} = \hat{r} + \prod_{i=1}^{n} p_i^{x_i} (1 - p_i)^{(1-x_i)}$
end for
end

Az algoritmusok szemléltetésre kerülnek a korábbi két példán keresztül.

2.1. szemléltető példa (további vizsgálat)

A példához tartozóan az eljárás P, R, és O halmazai a 2.1. ábrának megfelelően a következők: $P = \{\mathrm{A}\}$, $R = \{\mathrm{G}, \mathrm{K}\}$, $O = \{01, 02, 03, 04\}$. A műveleti egységek megbízhatóságait a $\boldsymbol{p} = (0{,}9;\ 0{,}9;\ 0{,}95;\ 0{,}95)$ vektor adja meg. Az eljárás először meghívja az MSG algoritmust az O halmazra, ami visszaadja az O halmazt változatlanul, mivel az általa jelölt hálózat kielégíti az 5 axiómát.

Ezután az SON algoritmus kezdi el a hálózatok generálását. Kezdetben az U halmaz üres. A 4 műveleti egység miatt az algoritmus 16 iterációt végez, ebből kettőnek a részletes leírása következik.

Az $\boldsymbol{x} = (0,0,1,1)$ esetben: $O_X = \{03, 04\}$, amelyik halmazra az MSG algoritmus az $O = \emptyset$ eredményt adja. Ez a részhálózat ezért nem működőképes.

Az $\boldsymbol{x} = (0,1,0,1)$ esetben: $O_X = \{02, 04\}$, amelyik halmazra az MSG algoritmus az $O = \{02, 04\}$ eredményt adja. Ez a hálózat ezért strukturálisan működőképes, és az algoritmus az U halmazba helyezi azt: $U = \{(0,1,0,1)\}$.

Az algoritmus hasonlóan végig nézi az Ω halmaz mind a 16 elemét. Ennek végeztével meghatározta az U halmazt, mint: $U = \{(0,1,0,1),\ (0,1,1,1),\ (1,0,1,0),\ (1,0,1,1),\ (1,1,0,1),\ (1,1,1,0),\ (1,1,1,1)\}$. A 2.2. ábra mutatja az U halmazban lévő 7 hálózat P-gráf reprezentációját.

Végül az RBO algoritmus meghatározza $\hat{r}$ értékét a (2.7) formula segítségével; ez lesz a rendszer megbízhatósága. A példa esetében ez $\hat{r} = 0{,}9789$.

2.2. szemléltető példa (további vizsgálat)

Első lépésben az MSG algoritmus meghatározza, hogy a 2.3. ábrán látható hálózatot kell-e szűkíteni. Ebben a példában nincs kiszedetett műveleti egység. Ezután az SON algoritmus generálja a 64 elemű U halmazt, aminek minden eleme egy strukturálisan működőképes részhálózatot jelöl. A teljes halmaz alább látható.

$U = \{(0, 1, 0, 0, 1, 0, 1), (0, 1, 0, 0, 1, 1, 1), (0, 1, 0, 1, 0, 1, 0), (0, 1, 0, 1, 0, 1, 1),$
$(0, 1, 0, 1, 1, 0, 1), (0, 1, 0, 1, 1, 1, 0), (0, 1, 0, 1, 1, 1, 1), (0, 1, 1, 0, 1, 0, 1),$
$(0, 1, 1, 0, 1, 1, 1), (0, 1, 1, 1, 0, 1, 0), (0, 1, 1, 1, 0, 1, 1), (0, 1, 1, 1, 1, 0, 1),$
$(0, 1, 1, 1, 1, 1, 0), (0, 1, 1, 1, 1, 1, 1), (1, 0, 0, 1, 0, 0, 0), (1, 0, 0, 1, 0, 0, 1),$
$(1, 0, 0, 1, 0, 1, 0), (1, 0, 0, 1, 0, 1, 1), (1, 0, 0, 1, 1, 0, 0), (1, 0, 0, 1, 1, 0, 1),$
$(1, 0, 0, 1, 1, 1, 0), (1, 0, 0, 1, 1, 1, 1), (1, 0, 1, 0, 0, 0, 0), (1, 0, 1, 0, 0, 0, 1),$
$(1, 0, 1, 0, 0, 1, 0), (1, 0, 1, 0, 0, 1, 1), (1, 0, 1, 0, 1, 0, 0), (1, 0, 1, 0, 1, 0, 1),$
$(1, 0, 1, 0, 1, 1, 0), (1, 0, 1, 0, 1, 1, 1), (1, 0, 1, 1, 0, 0, 0), (1, 0, 1, 1, 0, 0, 1),$
$(1, 0, 1, 1, 0, 1, 0), (1, 0, 1, 1, 0, 1, 1), (1, 0, 1, 1, 1, 0, 0), (1, 0, 1, 1, 1, 0, 1),$
$(1, 0, 1, 1, 1, 1, 0), (1, 0, 1, 1, 1, 1, 1), (1, 1, 0, 0, 1, 0, 1), (1, 1, 0, 0, 1, 1, 1),$
$(1, 1, 0, 1, 0, 0, 0), (1, 1, 0, 1, 0, 0, 1), (1, 1, 0, 1, 0, 1, 0), (1, 1, 0, 1, 0, 1, 1),$
$(1, 1, 0, 1, 1, 0, 0), (1, 1, 0, 1, 1, 0, 1), (1, 1, 0, 1, 1, 1, 0), (1, 1, 0, 1, 1, 1, 1),$
$(1, 1, 1, 0, 0, 0, 0), (1, 1, 1, 0, 0, 0, 1), (1, 1, 1, 0, 0, 1, 0), (1, 1, 1, 0, 0, 1, 1),$
$(1, 1, 1, 0, 1, 0, 0), (1, 1, 1, 0, 1, 0, 1), (1, 1, 1, 0, 1, 1, 0), (1, 1, 1, 0, 1, 1, 1),$
$(1, 1, 1, 1, 0, 0, 0), (1, 1, 1, 1, 0, 0, 1), (1, 1, 1, 1, 0, 1, 0), (1, 1, 1, 1, 0, 1, 1),$
$(1, 1, 1, 1, 1, 0, 0), (1, 1, 1, 1, 1, 0, 1), (1, 1, 1, 1, 1, 1, 0), (1, 1, 1, 1, 1, 1, 1)\}$

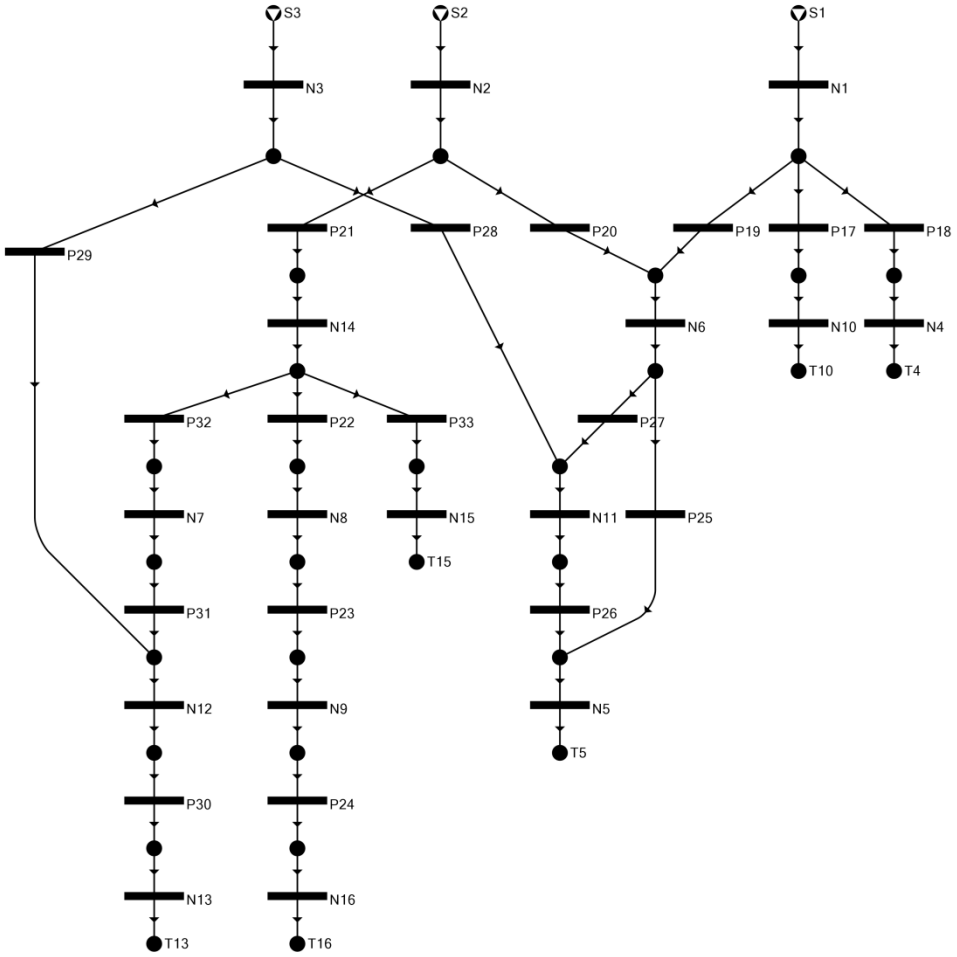
Az U halmaz alapján a rendszer megbízhatósági formula felírható. Amennyiben a műveleti egységek megbízhatóságát megadó vektor $\boldsymbol{p} = (0{,}95; 0{,}999; 0{,}999; 0{,}95; 0{,}99; 0{,}99; 0{,}999)$, a rendszer megbízhatósága $\hat{r} = 0{,}9999$, ahogy az RBO algoritmus meghatározza.

Esettanulmány

A fejezet a Shelby County-ban található MLGW (Memphis Light, Gas, and Water) hálózat gázszállító részét [82] elemzi. A hálózatba a gáz három betápláló állomáson keresztül jut, és hat végső állomásba lehet azt eljuttatni, ahonnét már nincs kimenő szállítás. A szállításra hét köztes állomás van. A gáz minden csővezetékben csak az egyik irányba mehet. A rendszer megbízhatóságának meghatározásához szükség van a működési kritériumra, vagyis definiálni kell, hogy mi a minimálisan elvárt szolgáltatás. Erre a hálózatra a minimális elvárás, hogy a kiválasztott cél állomásokba eljusson a gáz. Az eredeti publikáció [82] nem definiálta ezeket az állomásokat. A fejezet három esetet vizsgál különböző kritériumokkal.

A hálózat P-gráf reprezentációja a 2.4. ábrán található. A rendszerben kétféle műveleti egység van: állomások és csővezetékek. A 16 állomást az N1-N16 műveleti egységek jelölik, a 17 csővezetékeket a P17-P33 műveleti egységek. A három betápláló állomás, N1, N2 és N3 bemenetei a rendszer nyersanyagai. A termékek nincsenek az ábrán jelölve, hiszen ezek majd a működési kritériumban meghatározott állomások kimenetei lesznek. Azon részhálózatok működőképesek, amelyek képesek a gázt az összes „termékhez” elszállítani.

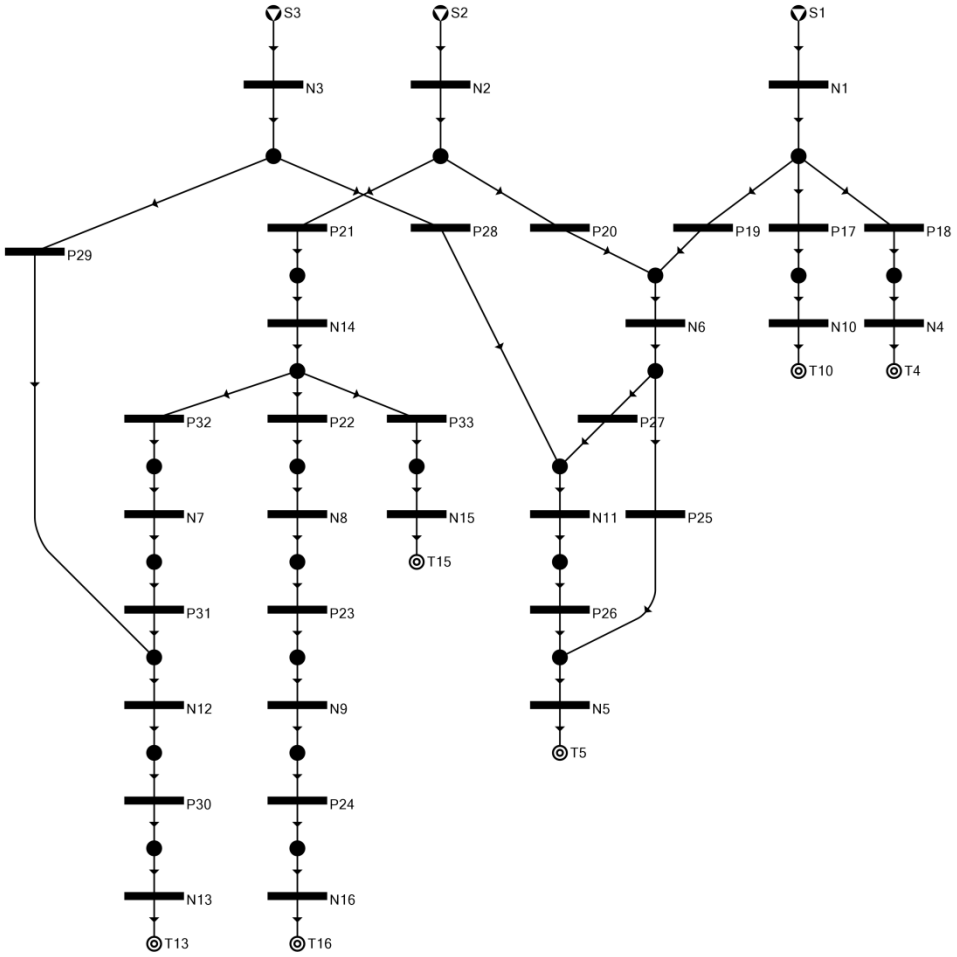
A szállító hálózat vizsgálata három különböző esetre történik. Először a teljes feladat megoldása következik az eredeti publikáció alapján. Ezután két, szintén korábban publikált, egyszerűsített példa megoldása következik, mint a teljes feladat speciális esetei.



2.4. ábra: A gázszállító hálózat [82] P-gráf reprezentációja

1. eset: minden cél állomást ki kell szolgálni, mind az állomások, mind elosztópontok közötti csővezetékek meghibásodhatnak

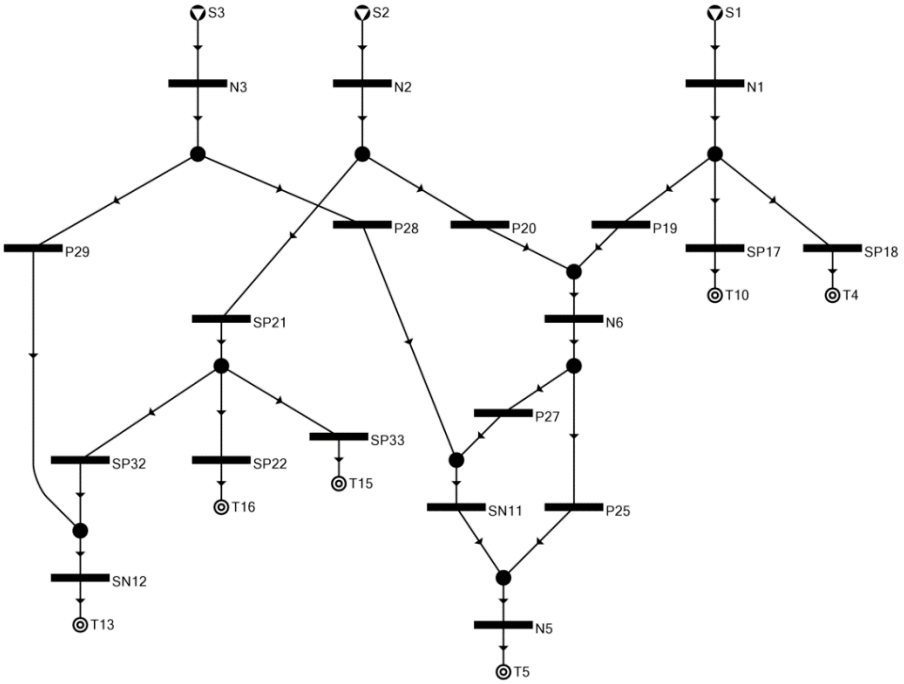
Feltesszük, hogy a rendszer bármely műveleti egysége meghibásodhat. A vizsgálat során a termékek a T4, T5, T10, T13, T15 és T16 csúcsok. A megfelelő P-gráf ábrázolás a 2.5. ábrán látható. Az állomások, illetve csővezetékek megbízhatóságai több paramétertől is függhetnek. Például egy csővezeték esetén annak hossza vagy térbeli elhelyezkedése kihatással lehet a megbízhatóságára. Az eredeti publikációban [82] nem szerepeltek megbízhatósági értékek sem az állomásokhoz, sem a csővezetékekhez, ezért jelen vizsgálat során minden műveleti egység kapott egy megbízhatóságot. A P17, P18, …, P33 csúcsokkal jelölt csővezetékekhez rendre a 0,9993, 0,9994, 0,9985, 0,9997, 0,9996, 0,9996, 0,9997, 0,9996, 0,9991, 0,9989, 0,9991, 0,9995, 0,9998, 0,9998, 0,9994, 0,9997 és 0,9994, értékeket rendeltük. Az állomások megbízhatóságait egységesen 0,9990 értéknek vesszük. Fontos megjegyezni, hogy ezek a megbízhatósági értékek csak a végeredményt befolyásolják, nincsenek hatással sem az eljárás menetére, sem annak számítási igényére.



2.5. ábra: Az 1. esethez tartozó P-gráf reprezentáció

A 2.5. ábrán jelölt hálózatban több helyen is szerepel műveleti egységek sorba kötött lánca. Egy ilyen láncnak vagy minden eleme szerepel egy működőképes hálózatban, vagy egyik sem, ezért megbízhatósági szempontból helyettesíthetőek egy eredő műveleti egységgel. Például a P22, N8, P23, N9, P24 és N16 egységek alkotnak egy ilyen láncot, amit lehet egy SP22 műveleti egységgel helyettesíteni. Az új műveleti egység megbízhatósága a helyettesített műveleti egységek megbízhatóságainak szorzata. Az összes lehetséges helyettesítést elvégezve az eredményt a 2.6. ábra mutatja, ahol az új, SN11, SN12, SP17, SP18, SP21, SP22, SP32 és SP33 jelölésű műveleti egységek megbízhatóságai rendre 0.9979, 0.9978, 0.9983, 0.9984, 0.9986, 0.9959, 0.9981 és 0.9984. A megbízhatóságot meghatározó eljárásnak az új rendszerben csak 19 műveleti egységgel kell dolgoznia az eredeti 33 helyett, amivel a hatékonyság jelentősen megnő.

A feladat ezután a rendszer megbízhatóságának meghatározása az összes termék egyidejű kiszolgálására vonatkozóan. Az eljárás első lépéseként az MSG algoritmus eltávolítja a műveleti egységeket, amelyek nem tudnak a működésben részt venni (ha van ilyen). Jelen esetben minden műveleti egység hozzá tud tenni a folyamat működéséhez, nincs kizárás.



2.6. ábra: A 2.5. ábra hálózata a sorba kötött egységek összevonása után

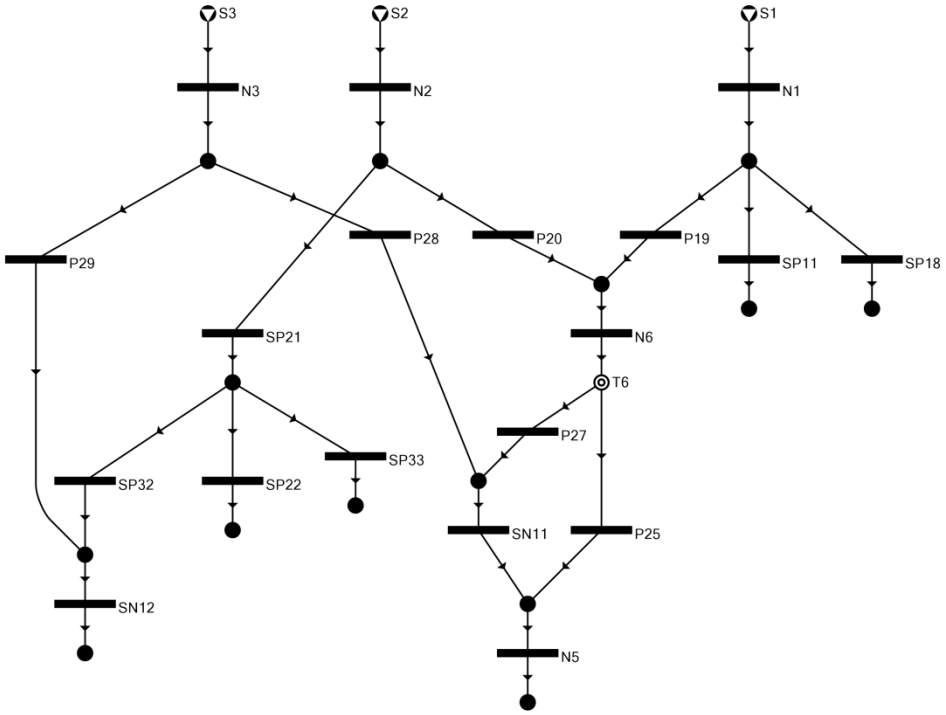
Az SON algoritmus meghatározza a strukturálisan működőképes részhálózatok *U* halmazát, ami 219 elemű. Ezt az *U* halmazt és a műveleti egységek megbízhatóságait felhasználva a rendszer megbízhatóságát az RBO algoritmus meghatározza, az eredmény 0,9845. Ha a csővezetékek meghibásodásait figyelmen kívül hagyjuk, vagyis a megbízhatóságukat 1-nek vesszük, akkor a rendszer megbízhatósága 0,9880 lesz. A megbízhatóság kiszámítása nagyjából 20 másodpercet vett igénybe egy átlagos laptopon (Linux Mint operáció rendszer, Intel Core i5 2,5 GHz processzor). Az eljárást a 2.5. ábrán lévő eredeti hálózatra futtatva az *U* halmaz 801 elemű lesz, a rendszer megbízhatósága viszont ugyanúgy 0,9845, ahogy az várható is. Azonban a 2.5. ábrán lévő hálózat esetén a 33 műveleti egység által definiált keresési tér ilyen módon történő végig járása ugyanazon a laptopon több, mint 3 napot (nagyjából 90 órát) vesz igénybe. Ebből látható, hogy a hálózat méretének csökkentése milyen nagy hatással lehet a számítási kapacitásra. A számítási idő csökkentése ezért egy fontos része a megbízhatóság meghatározásának, amire a 2.4. fejezet ismertet további lehetőségeket.

2. eset: Chang és Song [83] *által vizsgált paraméterek*

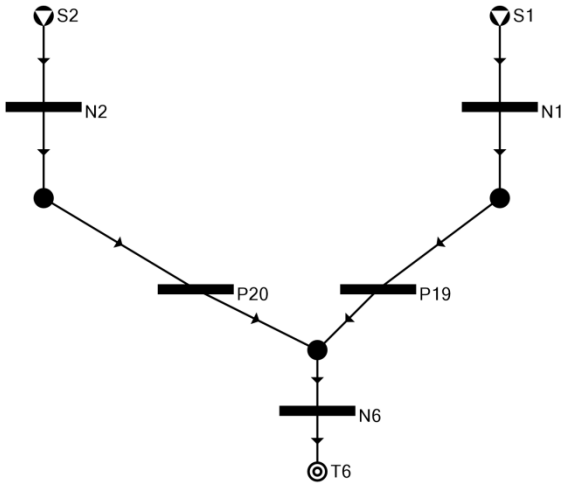
A műveleti egységek megegyeznek az 1. esetben vizsgáltakkal. A működési kritérium jelen esetben az, hogy a T6 jelölésű csúcsba eljusson a gáz, a megfelelő hálózatot a 2.7. ábra szemlélteti a sorba kötött egységek összevonása után. Az 1. esettel ellentétben itt a T4, T5, T10, T13, T15 és T16 csúcsok nem termékek.

Ahogy az eljárás leírja, az első lépés az MSG algoritmus futtatása, ami kiszűri a cél elérése szempontjából használhatatlan műveleti egységeket, ezzel csökkentve a feladat méretét. Jelen esetben ez jelentősen csökkenti a hálózatot, összesen 5 műveleti egység marad meg. A fennmaradó hálózat a 2.8. ábrán látható.

Az SON algoritmus generálja a 7 elemű *U* halmazt, a kapott megbízhatóság pedig 0,9990, amennyiben a műveleti egységek megbízhatóságai az 1. esetben megadottakkal megegyeznek. A számítási idő elhanyagolható.



2.7. ábra: A 2. eset P-gráf reprezentációja, a működési feltétel a gáz eljuttatása a T6 csúcsba



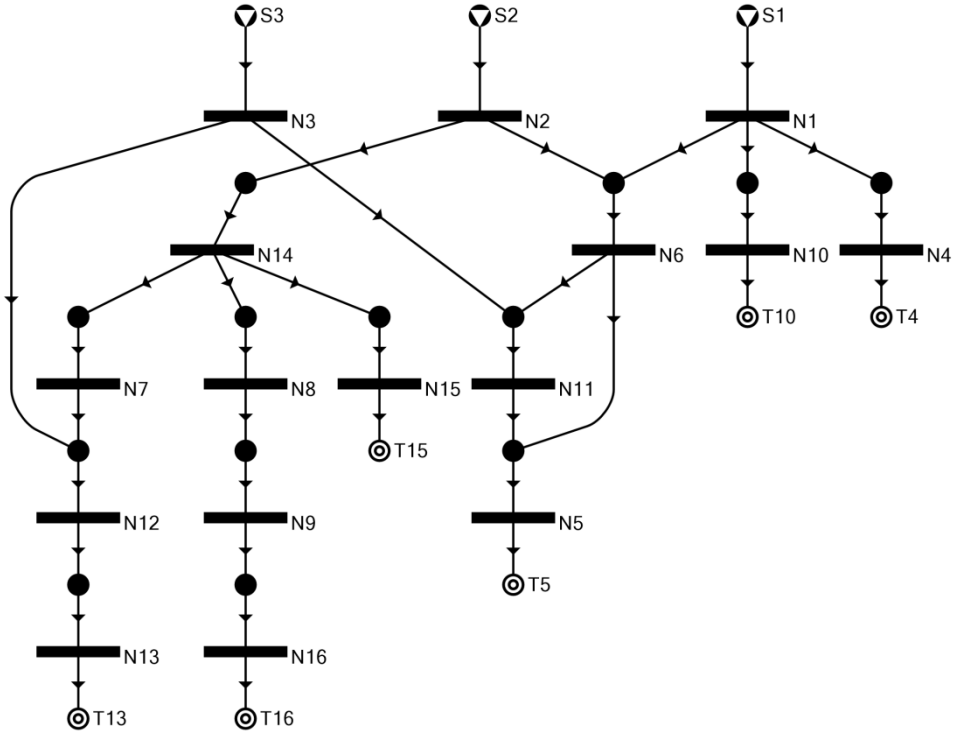
2.8. ábra: A 2. eset hálózata, miután az MSG algoritmus redukálta a 2.7. ábra hálózatát

3. eset: A Kim és Kang [84] *által vizsgált feladat*

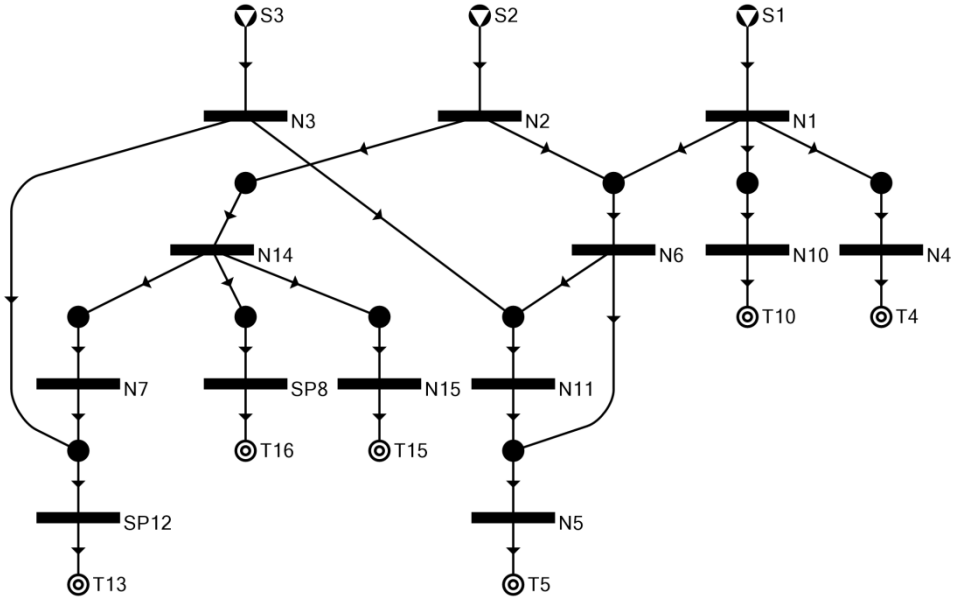
Kim és Kang [84] csak az állomások megbízhatóságait vizsgálták, feltételezve hogy a csővezetékek nem hibásodnak meg. A működési kritérium, hogy a gáz eljusson a 6 végső állomásba, vagyis a T4, T5, T10, T13, T15 és T16 csúcsok a termékek. Az ide tartozó P-gráfot a 2.9. ábra mutatja, a csővezetékekhez tartozó

műveleti egységek ebben nem szerepelnek.

Ahogy az 1. esetben, itt is összevonhatóak a sorba kötött műveleti egységek. A 2.9. ábra hálózata kettő ilyen láncot tartalmaz, az egyik az N12 és N13 műveleti egységekből áll, a másik az N8, N9 és N16 műveleti egységekből. Ezek rendre az SP12 és SP8 műveleti egységekkel helyettesíthetőek, az eredményt a 2.10. ábra mutatja.



2.9. ábra: A 3. esethez tartozó gázszállító hálózat P-gráf modellje



2.10. ábra: A 3. esethez tartozó 2.9. ábra hálózata a sorba kötött műveleti egységek összevonása után

Az SON algoritmus meghatározza az U halmazt 8 elemmel:
$U = \{(1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1), (1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1),$
$(1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1), (1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1),$
$(1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1), (1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1),$
$(1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1), (1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)\}$

Az RBO algoritmus három paraméterezésre is meghatározta a rendszer megbízhatóságát, ezek a 2.1. táblázatban láthatóak. Az SP8 és SP12 összevont egységek megbízhatóságai a korábbiaknak megfelelően kerültek meghatározásra, ezek szintén láthatóak a 2.1. táblázatban. A második paraméterezés az 1. esettel egyezik meg, amennyiben ott a csővezetékek megbízhatóságát 1-nek vesszük. A számítási idő elhanyagolható.

2.1. táblázat: A 3. eset gázhálózatának megbízhatósága 3 paraméterezésre

N1-N16 műveleti egységek megbízhatósága	0,9950	0,9990	0,9999
SP8 műveleti egység megbízhatósága	0,9851	0,9970	0,9997
SP12 műveleti egység megbízhatósága	0,9900	0,9980	0,9998
Rendszer megbízhatósága	0,9416	0,9881	0,9988

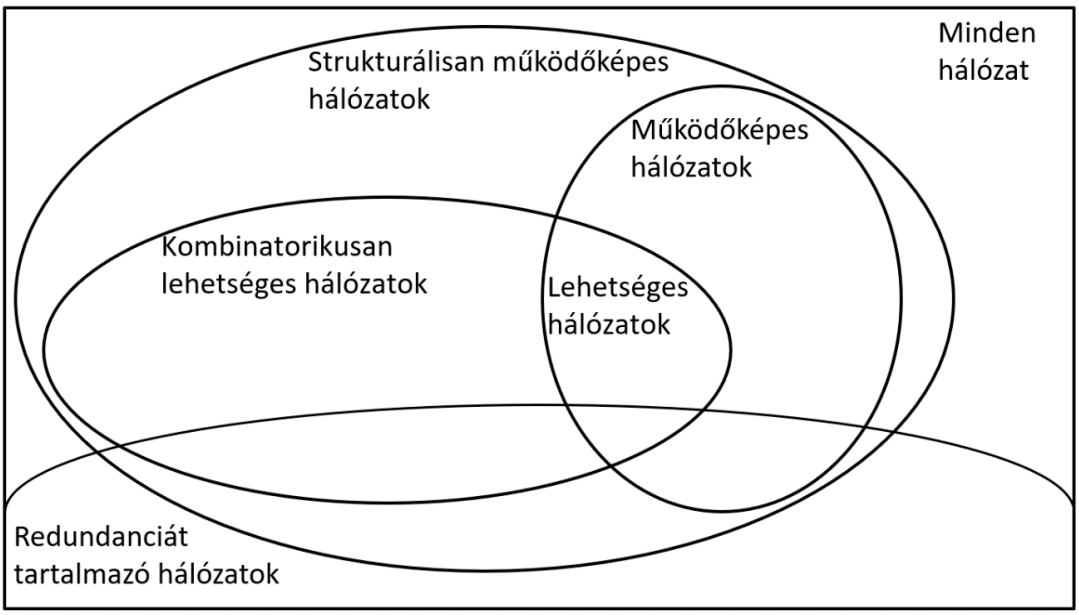
2.3. Folyamat hálózatok megbízhatósága: általános eset

Az eddigiek során strukturális megbízhatóságról volt szó, ahol az U halmazba a strukturálisan működőképes részhálózatok kerültek. Ezek azok a hálózatok, amelyek képesek mindegyik terméket előállítani. Ahogy korábban volt róla szó, minden kombinatorikusan lehetséges hálózat (olyan hálózat, amely teljesíti az 5 axiómát) ebbe a csoportba tartozik. Ugyanakkor vannak olyan strukturálisan működőképes hálózatok, amelyek nem teljesítik mindegyik axiómát.

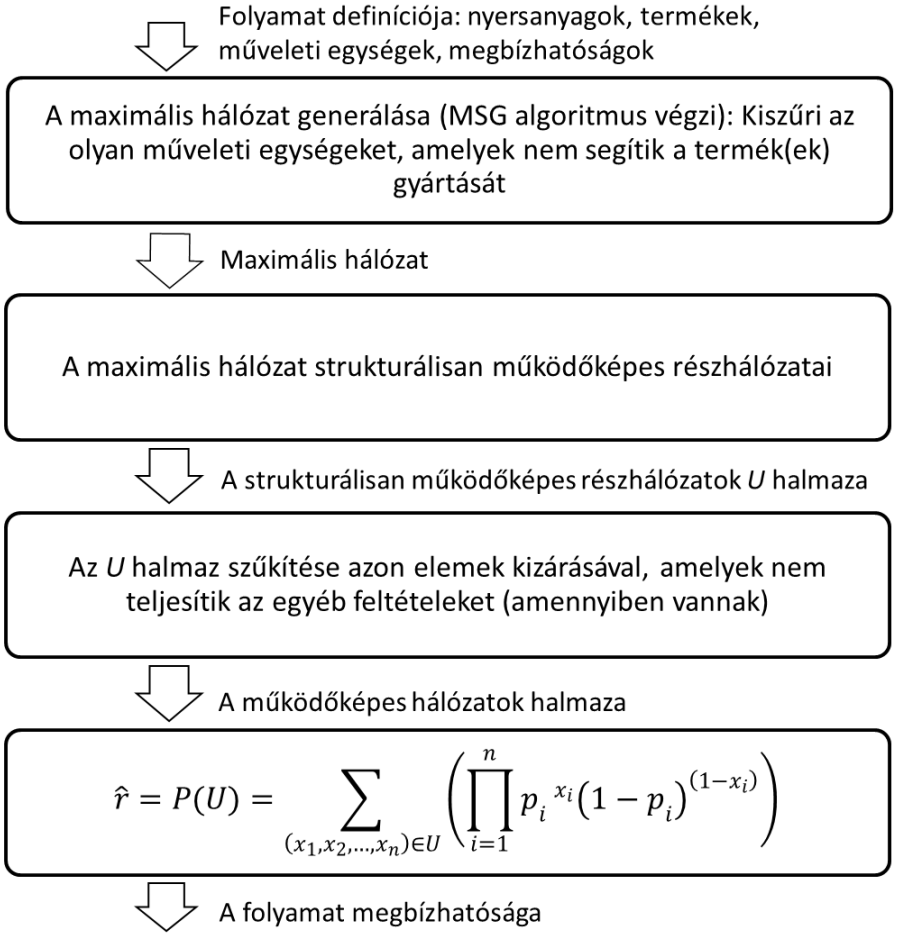
Vannak olyan műszaki rendszerek, ahol a műveleti egységek strukturális megléte nem elég a rendszer működéséhez, hanem egyéb feltételeknek is teljesülni kell. Például vegyipari folyamatok esetén fontos a megfelelő anyagösszetétel a műveleti egységek bemenetén, valamint a teljes rendszerre vonatkozó anyagmérleg teljesítése is. Az olyan hálózatokat, amelyek kombinatorikusan lehetségesek, és ezen felül megfelelnek az egyéb feltételeknek is, lehetséges hálózatoknak nevezzük. Hasonlóan, működőképes hálózatként hivatkozunk azokra a strukturálisan működőképes hálózatokra, amelyek minden feltételt figyelembe véve is képesek a megfelelő működésre. A 2.11. ábrán láthatóak a dolgozat során kezelt hálózat típusuk, valamint azok kapcsolatai.

Amikor a strukturális működőképességen felül egyéb feltételeknek is teljesülni kell, akkor a működőképes struktúrákat kell generálni és összegyűjteni az U halmazba. Ennek a módja, hogy minden hálózatot kettő (vagy több) vizsgálat ellenőriz. Az első vizsgálat a strukturális működőképesség meghatározása. Amennyiben egy hálózat strukturálisan működőképes, akkor kerül sor a következő vizsgálatra, amely a tényleges működőképességet határozza meg. Ilyen vizsgálatból több is lehet, de célszerű összevonni őket. A 2.12. ábra mutatja a teljes algoritmust a rendszer megbízhatóságának meghatározására, amely a

strukturális szempontokon kívül egyéb feltételeket is képes figyelembe venni a működőképességre.



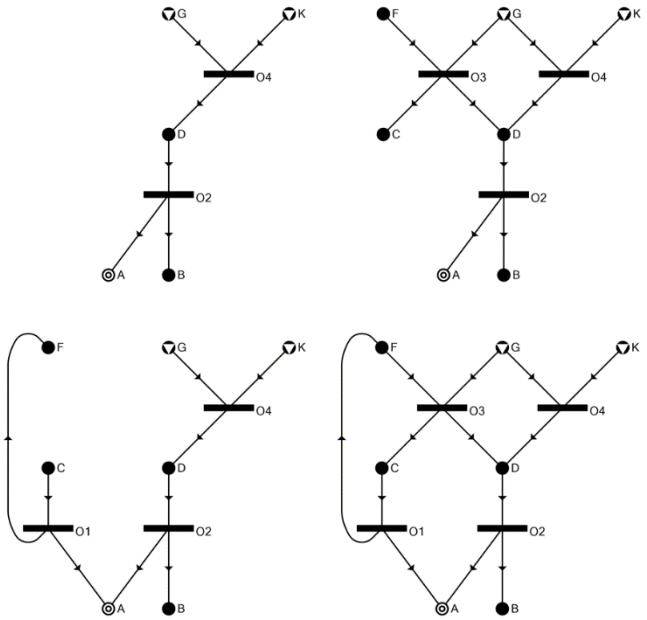
2.11. ábra: A különböző hálózat típusok és egymással való kapcsolatuk



2.12. ábra: Rendszer megbízhatóságának meghatározása

2.1. szemléltető példa (további vizsgálat)

A 2.1. ábrán látható 2.1. szemléltető példa strukturális megbízhatóságát egy korábbi fejezet már meghatározta. Az U halmazban 7 strukturálisan működőképes hálózat volt. Ezek közül azonban lehet, hogy nem mind működőképes az anyagmérleg tekintetében. Például a folyamat felépítése olyan, hogy az O4 műveleti egység nélkül vagy nem képes a terméket megfelelő mennyiségben legyártani, vagy képes, de csak az üzemeltető számára elfogadhatatlan mértékű veszteséggel. Ezt a strukturális vizsgálat nem képes eldönteni, ezért szükség van további szűrésre. Jelen feladat esetében a P-gráf keretrendszer által biztosított anyagmérleg-modell segítségével ellenőrizhető a részhálózatok működőképessége. A szűrés után az U halmaz 4 elemet tartalmaz: $U = \{(0,1,0,1),(0,1,1,1),(1,1,0,1),(1,1,1,1)\}$, ezek P-gráf reprezentációi a 2.13. ábrán láthatók. A rendszer megbízhatósága: $\hat{r} = (1-p_1) * p_2 * (1-p_3) * p_4 + (1-p_1) * p_2 * p_3 * p_4 + p_1 * p_2 * (1-p_3) * p_4 + p_1 * p_2 * p_3 * p_4$. Ha $p_1 = p_2 = 0{,}90$ és $p_3 = p_4 = 0{,}95$, akkor a rendszer megbízhatósága 0,8550.



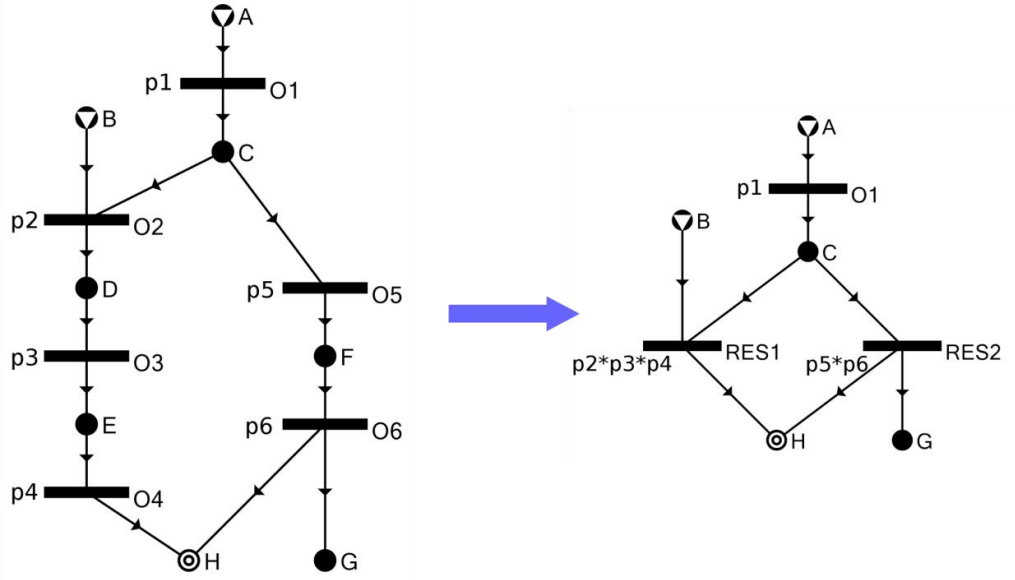
2.13. ábra: A 2.1. szemléltető példa működőképes részhálózatai

2.4. Gyorsítási eljárások a megbízhatóság meghatározására

Az algoritmusok gyakorlati használhatóságában nagy szerepet játszik a számításigényük. Az ismertetett algoritmus futási ideje elsősorban a hálózatban lévő műveleti egységek számától függ, hiszen az határozza meg a részhálózatok számát. Nagyméretű hálózatok esetén fontos a számítási hatékonyság növelése. Jelen fejezet erre ismertet néhány lehetőséget, amelyek között vannak mind strukturális, mind algoritmizálási megoldások.

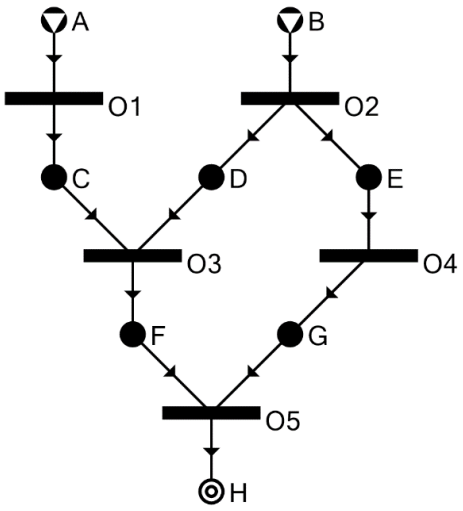
A hálózat egyszerűsítése

Az első hatékonyság-növelő módszer már szerepelt a gázszállító hálózat esettanulmánya során. Ez az egyszerű részrendszerek összevonása. A 2.14. ábra mutatja, hogy olyan sorban lévő műveleti egységeket, amelyek a sor elejét és végét kivéve más műveleti egységekkel nincsenek összeköttetésben, a megbízhatóság szempontjából össze lehet vonni egy műveleti egységgé. Az így kapott eredő műveleti egység megbízhatósága a sorban lévő műveleti egységek megbízhatóságainak szorzata, a (2.9) formula alapján, ahogy az ábrán is látható.



2.14. ábra: Példa sorba kapcsolt egységek összevonására

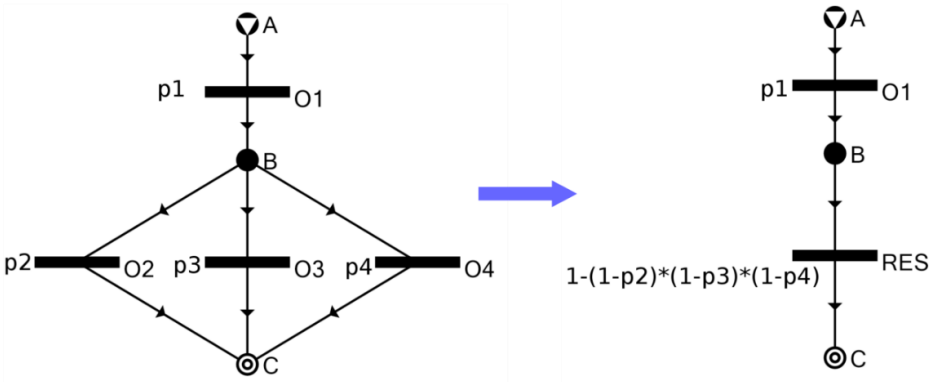
Az összevonás kiterjeszthető olyan részrendszerekre is, amelyek nincsenek feltétlenül sorba kötve, de minden bennük lévő műveleti egységre egyszerre van szükség egy működőképes hálózatban. Ilyen esetben ez a részrendszer ugyanolyan szabályok szerint kezelhető, mint egy soros részrendszer. Erre mutat példát a 2.15. ábra. Itt az O5-ös műveleti egység értelemszerűen kötelező a H anyag gyártásához. Így, amennyiben a H anyag része a hálózatnak, akkor az O5-ös műveleti egység is. Az O5-ös műveleti egységnek 2 bemente van, F és G, így, ha az O5 műveleti egység része a hálózatnak, akkor az F és G anyagokat gyártani kell, ami csak az O3 és O4 műveleti egységek segítségével érhető el. Ugyanezzel az indoklással élve az O1 és O2 műveleti egységekre is szükség van. A levezetésből következik, hogy a 2.15. ábrán látható részrendszer műveleti egységei vagy mind egyszerre szerepelnek a hálózatban, vagy egyik sem, és ezért a részhálózat helyettesíthető egyetlen műveleti egységgel.



2.15. ábra: Olyan részrendszer, ahol minden műveleti egységre egyszerre van szükség a műveleti egységek több bemeneti anyagai miatt

Ezt az elvet még tovább lehet fejleszteni az összevonásos redukció elvét használva [85]. Az összevonásos redukció képes meghatározni az összes olyan műveleti egység halmazt egy P-gráffal ábrázolt folyamatban, amelyekre igaz, hogy egy tetszőleges kombinatorikusan lehetséges hálózatban vagy mindegyik szerepel, vagy egyik sem. A megbízhatósági számítások tekintetében minden ilyen halmaz helyettesíthető egy eredő műveleti egységgel.

Hasonlóan össze lehet vonni párhuzamosan kötött műveleti egységeket is, ahogy a 2.16. ábrán látható. Ilyen részrendszer tipikusan akkor fordul elő, amikor a műveleti egységek mellé redundáns egységek kerülnek a megbízhatóság javítása céljából. A műveleti egységek ugyanazt a feladatot látják el, de esetleg eltérhetnek hatékonyságban, költségben, megbízhatóságban. Az ilyen részrendszer is összevonható egy eredő műveleti egységgé, aminek a megbízhatósága a párhuzamos rendszerek megbízhatóságának képletéből jön ki, amit a (2.8) formula ír le.



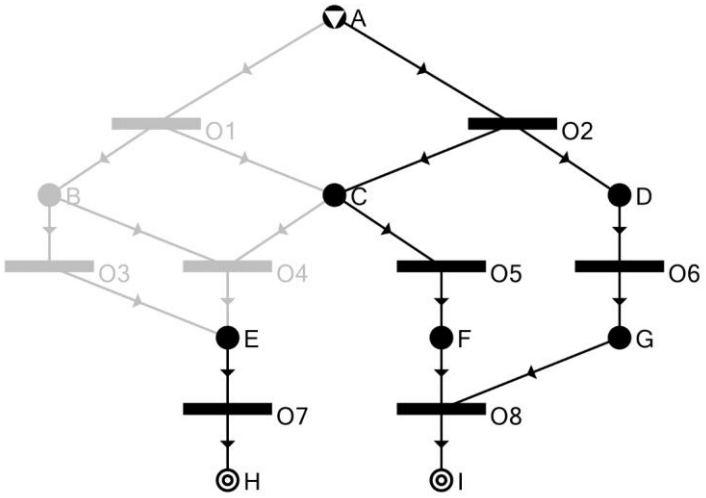
2.16. ábra: Példa párhuzamos részrendszer összevonására.

A részrendszerek összevonását megvalósító strukturális módszerek a hálózat struktúrájától függő mértékben képesek a megbízhatóság számítását gyorsítani. Minél kevesebb műveleti egység marad a számítandó modellben, annál kisebb lesz a megvizsgálandó részhálózatok halmaza.

Kritikus műveleti egységek

Egy másik módszer a számítási hatékonyság növelésére a kritikus műveleti egységek megkeresése. Kritikusnak azt a műveleti egységet nevezzük, amelyik mindegyik lehetséges hálózatnak része. Amennyiben nem szeretnénk az összes lehetséges hálózatot felsorolni ennek meghatározására, úgy a strukturálisan kritikus műveleti egységek megkeresésére léteznek megoldások. Az első egy egyszerű megoldás, a neutrális kiterjesztésen alapul [31], amely egy nem lehetséges hálózatra képes meghatározni olyan műveleti egységek egy halmazát, amelyek mindenképp szükségesek a hálózat lehetséges hálózattá kiegészítésében. Az algoritmus elve, hogy megkeresi azokat a hálózatban lévő, de még nem gyártott anyagokat, amik nem nyersanyagok, és csak egy műveleti egység lehetséges a legyártásukra. Az ilyen műveleti egységeket kötelező a hálózat kiegészítésébe bevenni. Az algoritmus ezt a lépést rekurzívan ismétli.

A neutrális kiterjesztést alkalmazva az üres hálózatra, aminek csak a termékek az elemei, az megad minden olyan műveleti egységet, amelyek kritikusak a termékek legyártásához. Erre mutat példát a 2.17. ábra.



2.17. ábra: A sötéttel jelölt műveleti egységek kritikusak a termék legyártásához, ezeket a neutrális kiterjesztés határozta meg.

Ezen felül azonban lehetnek olyan kritikus műveleti egységek is, amelyek elrejtve szerepelnek a hálózat belsejében, és a neutrális kiterjesztés rekurzív algoritmusa nem találja meg őket. A 2.17. ábrán látható hálózatban ilyen az O1 műveleti egység. Átgondolható, hogy az O1 műveleti egység nélkül a H termék nem gyártható, azonban a neutrális kiterjesztés ezt nem tudja észre venni. Amennyiben ezek megkeresésére is igényt tartunk, akkor ismételten az összevonásos redukció [85] algoritmusa az, amellyel ezeket is meg lehet keresni.

A kritikus műveleti egységek felismerése azért gyorsítja a megbízhatóság kiszámítását, mert felhasználható, hogy ezek minden lehetséges hálózatban szerepelnek. Ezért a lehetséges hálózatok keresése során nem kell sem generálni, sem kiértékelni azokat a hálózatokat, amelyek nem tartalmazzák az összes kritikus műveleti egységet. A 2.17. ábrán látható hálózat esetében, amennyiben az összes műveleti egységet figyelembe véve kell generálni az összes részhálózatot, akkor $2^8 = 256$ hálózatot kell egyesével tesztelni. Ha azonban kihasználjuk, hogy

a műveleti egységek közül 6 kritikus, akkor csak a maradék 2 az, amit figyelembe kell venni a generáláskor, ez összesen $2^2 = 4$ hálózatot jelent.

Bináris döntési diagram

A következő gyorsító eljárás a hálózat generáló algoritmust módosítja. A módszer során a generáló algoritmusban a hálózatok generálásának sorrendje módosul a bináris döntési diagramnak (binary decision diagram, BDD) megfelelően [86]. A BDD elve, hogy a nem ismert bináris változókból egy döntési fát épít fel. A cél, hogy ne kelljen az n változó által definiált 2^n méretű keresési teret egyesével bejárni, hanem egy vizsgálattal akár több eset hovatartozását is meghatározhassuk.

Rendszerek működőképessége esetén a rendszer koherens tulajdonságát aknázza ki a BDD. Egy rendszer koherens, ha a Ψ működőképességi függvény monoton, és a rendszer egyik műveleti egysége sem irreleváns a működőképesség szempontjából. Ez azt jelenti, hogy ha egyik műveleti egység sem működik, akkor a rendszer sem, valamit egy működőképes rendszerhez új egységgel kiegészítve, az nem okoz meghibásodást. Formálisan a feltételeket a (2.10) és (2.11) formulák írják le.

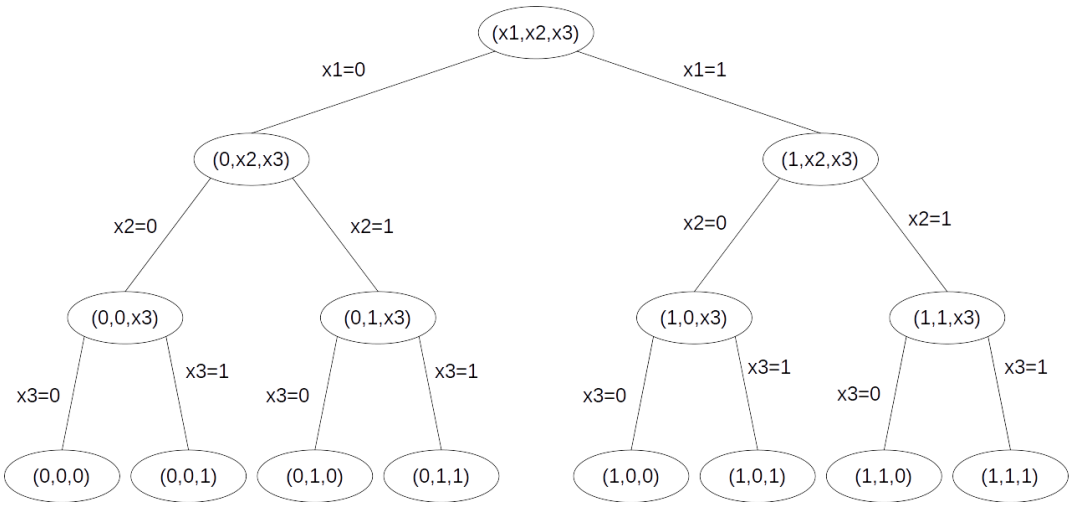
$$\Psi(0,0,\dots,0) = 0 \tag{2.10}$$

$$x_i \geq y_i \;\; \forall\, i = 1,2,\dots,n \;\; \Longrightarrow \;\; \Psi(x_1, x_2, \dots, x_n) \geq \Psi(y_1, y_2, \dots, y_n) \tag{2.11}$$

A koherens tulajdonságot kihasználva a működőképességi függvény BDD alapú generálása során a bináris fa gyökere egy olyan csúcs, ahol még egyik műveleti egységről sem született döntés. Itt az algoritmus ellenőrzi, hogy a rendszer működőképes-e akkor, ha minden műveleti egység működőképes ($\Psi(1,1,\dots,1) = 1$ teljesül-e). A koherens tulajdonság alapján feltehető, hogy ha egyik műveleti egység sem működőképes, akkor a rendszer sem ($\Psi(0,0,\dots,0) = 0$).

A fa minden csúcsához döntések sorozata vezet el, amelyek mindegyike során egy műveleti egységet vagy beveszi a módszer a hálózatba, vagy kizár abból. A gyökér csúcsban a döntés alapja a sorrendben első műveleti egység, a következő szinten a második műveleti egység, és így tovább. Minden csúcs gyakorlatilag hálózatok egy halmazát jelöli, ahol bizonyos műveleti egységekről már megszületett a döntés, a többiről viszont még nem, és esetükben a bináris értékek tetszőleges kombinációban szerepelhetnek. A 2.18. ábra mutat egy példát a teljes BDD-re 3 műveleti egység esetén. Minden csúcsban a 0 érték azt jelenti, hogy az a műveleti egység nem része a hálózatnak, az 1 azt jelenti, hogy része, a változó értékek (x_1, x_2, x_3) pedig azt jelentik, hogy arról csak a későbbi lépésekben születik döntés.

Minden csúcsban az algoritmus kettő rendszer állapotot határoz meg, amelyek a legjobb-, és a legrosszabb esetekhez tartoznak. A legjobb eset egy adott csúcsból kiindulva, ha minden még változó értéket 1-esre állítunk, a legrosszabb eset pedig az, ha nullára. Ezek az adott csúcs alatti részfához tartozó levelek közül a legkevesebb, illetve legtöbb hibát tartalmazó levelek. Például 5 műveleti egység esetén az $(1,0,x_3,x_4,x_5)$ csúcsból kiindulva a legjobb levél az (1,0,1,1,1), míg a legrosszabb levél az (1,0,0,0,0). Ez a két levél azért érdekes, mert a rendszer koherens tulajdonsága alapján ezek korlátozzák az összes többi levél esetében a rendszer állapotát (2.12).



2.18. ábra: Példa bináris döntési diagramra 3 műveleti egység esetén

$$\Psi(1,0,0,0,0) \leq \Psi(1,0,x_3,x_4,x_5) \leq \Psi(1,0,1,1,1) \tag{2.12}$$

Általánosan, ha $b_1, b_2, \dots, b_i$ jelölik a már megadott bináris értékeket egy csúcsban, k darab döntés után, és $x_{k+1}, x_{k+2}, \dots, x_n$ a további, még nem meghatározott értékeket, akkor a szabályt a (2.13) formula írja le.

$$\begin{aligned}\Psi(b_1, b_2, \dots, b_k, 0,0, \dots,0) &\leq \Psi(b_1, b_2, \dots, b_k, x_{k+1}, x_{k+2}, \dots, x_n) \\ &\leq \Psi(b_1, b_2, \dots, b_k, 1,1, \dots,1)\end{aligned} \tag{2.13}$$

Jelölje a legrosszabb esetet, vagyis az alsó korlátot α, a legjobbat, vagyis a felső korlátot pedig β, vagyis általánosan:

$$\begin{aligned}\alpha(b_1, b_2, \dots, b_k, x_{k+1}, x_{k+2}, \dots, x_n) &= \Psi(b_1, b_2, \dots, b_k, 0,0, \dots,0) \\ \beta(b_1, b_2, \dots, b_k, x_{k+1}, x_{k+2}, \dots, x_n) &= \Psi(b_1, b_2, \dots, b_k, 1,1, \dots,1)\end{aligned} \tag{2.14}$$

Ebből adódik a következő kettő logikai következtetés:

- Ha egy csúcsban a legjobb lehetséges esetben sem működőképes a rendszer, akkor egyikben sem lesz az.
- Ha egy csúcsban a legrosszabb lehetséges esetben is működőképes a rendszer, akkor mindegyikben az lesz.

A kettő közül bármelyik feltétel teljesül, az azt jelenti, hogy a csúcs gyerekeinek megvizsgálására már nincsen szükség, hiszen már most is mindre tudjuk, hogy mit kapunk eredménynek. Így tehát a vizsgálatok egy jelentős részét ki lehet hagyni.

A fa minden csúcsában 2 vizsgálat eredményét kell meghatározni. Habár ez csúcsonként kettő működőképességi tesztet jelentene, ezek között sok a redundáns vizsgálat. Minden köztes csúcsnak kettő gyereke van, egy, ahol a következő bináris változó értéke 0, és egy másik, ahol 1. Vegyük észre, hogy a csúcs nullás irány alsó korlátja megegyezik a csúcséval, és hasonlóan az 1-es csúcs felső korlátja is megegyezik a csúcséval (2.15).

$$\begin{aligned}\alpha(b_1, b_2, \dots, b_k, 0, x_{k+2}, \dots, x_n) &= \alpha(b_1, b_2, \dots, b_k, x_{k+1}, x_{k+2}, \dots, x_n) \\ \beta(b_1, b_2, \dots, b_k, 1, x_{k+2}, \dots, x_n) &= \beta(b_1, b_2, \dots, b_k, x_{k+1}, x_{k+2}, \dots, x_n)\end{aligned} \tag{2.15}$$

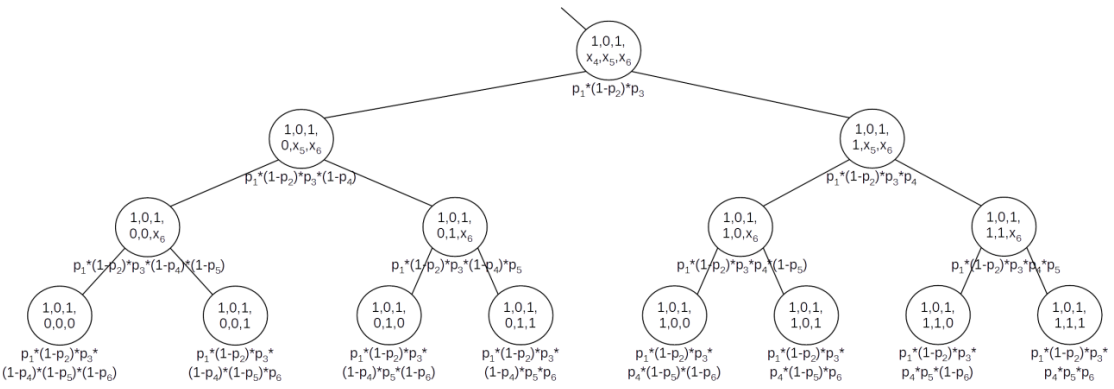
Így tehát minden csúcsra igaz (a gyökeret kivéve), hogy a két meghatározandó korlát közül az egyik megegyezik a szülőnek ugyanezzel a korlátjával. Ez azt jelenti, hogy a gyökéren kívül minden csúcsra csak egy működőképességi tesztet kell elvégezni.

Ahhoz, hogy a rendszer megbízhatósága BDD segítségével meghatározható legyen, a számítást igazítani kell fában történő vágások esetére. A hagyományos számítás a fa működőképes hálózatokat adó leveleihez határozza meg a hozzájuk tartozó valószínűségeket, és ezeket adja össze. A vágásoknak azon fajtája, amikor már a legjobb eset sem működőképes, nem befolyásolja a számítást, hiszen azon az ágon egyik levél valószínűségét sem kell a megbízhatóságba beszámítani.

Azoknál a vágásoknál, ahol a legrosszabb eset is működőképes, ott a csúcsból induló részfában minden levél működőképes. Amennyiben a csúcsban az első k bináris érték adott ($(b_1, b_2, \dots, b_k, x_{k+1}, x_{k+2}, \dots, x_n)$ csúcs), akkor még szabadon van $n-k$ bináris érték. A csúcs alatti részfa 2^{n-k} levelet fog tartalmazni, ahol ezek a változó értékek az összes lehetséges érték-kombinációt felveszik. A levelek valószínűségeiben az első k változóhoz tartozó részek közösek, ezek kiemelhetőek. A kiemelés után megmaradt részek egy $n-k$ változóból álló teljes eseményhalmazt írnak le, ahol így a valószínűségek összege 1-et ad eredményül. Így a csúcshoz tartozó teljes részfa eredő valószínűsége a csúcshoz tartozó $b_1, b_2, \dots, b_k$ értékek alapján határozható meg (2.16).

$$P\big((b_1, b_2, \dots, b_k, x_{k+1}, x_{k+2}, \dots, x_n)\big) = \prod_{i=1}^{k} {p_i}^{b_i}(1-p_i)^{(1-b_i)} \tag{2.16}$$

A 2.19. ábra mutat egy példát, a csúcs megbízhatóságának meghatározására 6 műveleti egység esetén, ahol az adott csúcsban az első háromról hoztunk döntést. Az ábrán az $(1,0,1,x_4,x_5,x_6)$ csúcsról kiderül, hogy a részfájában minden levél működőképes hálózatot jelöl (mivel az alsó korlátja, vagyis az (1,0,1,0,0,0) csúcs is működőképes), így a számításban itt vágást lehet végezni. Az algoritmus nem vizsgálja tovább a részfát, hanem meghatározza az eredő valószínűséget ($p_1 * (1-p_2) * p_3$), és ezt számolja bele a rendszer megbízhatóságába. A (2.17) számítás kifejti az eredő valószínűség számításának helyességét erre a példára.



2.19. ábra: Az $(1,0,1,x_4,x_5,x_6)$ csúcs részfájában minden levél működőképes, így azok megvizsgálása nélkül a csúcs eredő valószínűsége számítható bele a megbízhatóságba

$$
\begin{aligned}
&P\big((1,0,1,x_4,x_5,x_6)\big) = \\
&P\big((1,0,1,0,0,0)\big) + P\big((1,0,1,0,0,1)\big) + P\big((1,0,1,0,1,0)\big) + P\big((1,0,1,0,1,1)\big) \\
&\quad + P\big((1,0,1,1,0,0)\big) + P\big((1,0,1,1,0,1)\big) + P\big((1,0,1,1,1,0)\big) \\
&\quad + P\big((1,0,1,1,1,1)\big) = \\
&p_1 * (1-p_2) * p_3 * (1-p_4) * (1-p_5) * (1-p_6) + p_1 * (1-p_2) * p_3 \\
&\quad * (1-p_4) * (1-p_5) * p_6 + p_1 * (1-p_2) * p_3 * (1-p_4) * p_5 \\
&\quad * (1-p_6) + p_1 * (1-p_2) * p_3 * (1-p_4) * p_5 * p_6 + p_1 \\
&\quad * (1-p_2) * p_3 * p_4 * (1-p_5) * (1-p_6) + p_1 * (1-p_2) * p_3 \\
&\quad * p_4 * (1-p_5) * p_6 + p_1 * (1-p_2) * p_3 * p_4 * p_5 * (1-p_6) \\
&\quad + p_1 * (1-p_2) * p_3 * p_4 * p_5 * p_6 = \\
&p_1 * (1-p_2) * p_3 \\
&\quad * \big((1-p_4) * (1-p_5) * (1-p_6) + (1-p_4) * (1-p_5) * p_6 \\
&\quad + (1-p_4) * p_5 * (1-p_6) + (1-p_4) * p_5 * p_6 + p_4 * (1-p_5) \\
&\quad * (1-p_6) + p_4 * (1-p_5) * p_6 + p_4 * p_5 * (1-p_6) + p_4 * p_5 \\
&\quad * p_6\big) = \\
&\quad p_1 * (1-p_2) * p_3 * 1 = p_1 * (1-p_2) * p_3
\end{aligned}
\tag{2.17}
$$

Maximális számítási mélység

Az ismertetett megbízhatóság számító algoritmus számára a legnagyobb kihívás a sok műveleti egységeket tartalmazó folyamatok kezelése. Bár, ahogy a korábbiakból látható, több lehetőség is van a hatékonyság növelésére, de a nagyméretű és komplex struktúrájú feladatok kezelését ezek sem tudják elég hatékonnyá tenni. Ez pusztán abból adódik, hogy a determinisztikus algoritmus által kombinatorikusan bejárt lehetőségek száma túl nagy lesz. Amennyiben a hálózat méretének összevonásokkal való csökkentése után is túl sok a műveleti egység, a pontos érték helyett csak egy közelítő érték határozható meg rövid időn belül.

A közelítő érték meghatározására egy determinisztikus lehetőség egy maximális számítási mélység meghatározása, amely jelen algoritmus esetében azt jelenti, hogy nem vizsgáljuk meg az n műveleti egységből adódó mind a 2^n részhálózatot. Helyette meghatározunk egy F_{max} értéket, amely azt jelöli, hogy legfeljebb mennyi hibás műveleti egységet engedünk meg a vizsgált hálózatokban. Például, ha $F_{max} = 3$, akkor csak azokat a részhálózatokat járja végig az algoritmus, amelyeken legfeljebb 3 meghibásodott műveleti egység található. A módszer azon az elven alapul, miszerint a műveleti egységek megbízhatóságai jellemzően magas értékek, vagyis a meghibásodások valószínűsége kicsi. Így egy sok meghibásodást tartalmazó részhálózat esetén a (2.6) formulában a szorzatnak több eleme is kicsi érték lesz, vagyis a szorzás eredménye közel lesz a nullához. Még ha egy ilyen részhálózat működőképes is, a valószínűsége annyira kicsi, hogy a beleszámolása a képletbe nem eredményez jelentős változást. Természetesen óvatosan kell kezelni ezt a megfigyelést, hiszen, ha egy hálózatban sok a redundancia, akkor az ilyen kis valószínűségű részhálózatok közül sok működőképes lesz, ami már megmutatkozhat a megbízhatóságban.

A gyorsítási módszer során felmerülő legfőbb kérdés ezért természetesen az, hogy az F_{max} értékét hogyan célszerű megválasztani. Ennek megválaszolására hibabecslő számításokat kell végezni. Egy folyamat hálózatban a műveleti

egységek megbízhatósági eltérőek lehetnek. A következő számításokhoz tegyük fel, hogy minden műveleti egység megbízhatósága a p_{min} és p_{max} értékek között van: $p_{min} \leq p_i \leq p_{max}, \quad i = 1,2, \dots, n$

Ekkor egy olyan részhálózat esetén, amely pontosan k meghibásodást tartalmaz, a hozzá tartozó valószínűségi érték is becsülhető. Mivel a műveleti egységek megbízhatóságai helyett a p_{min} és p_{max} értékeket használjuk a becslésben, ezért mindegy, hogy a pontosan k meghibásodást tartalmazó részhálózatok közül melyiket vizsgáljuk, a becslés ugyanúgy igaz lesz. Így az általánosság elvesztése nélkül vizsgálhatjuk azt a hálózatot, ahol az $1,2, \dots, k$ műveleti egységek hibásodnak meg, és a $k+1, k+2, \dots, n$ műveleti egységek nem. A részhálózat valószínűségének meghatározásakor ez esetben $b_1 = b_2 = \cdots = b_k = 0$, és $b_{k+1} = b_{k+2} = \cdots = b_n = 1$. Ekkor a részhálózat valószínűségét a (2.18) formula írja le, majd erre a becslés a (2.19) formulában látható.

$$P\big((0,0, \dots, 0,1,1 \dots, 1)\big) = \prod_{i=1}^{k}(1 - p_i) * \prod_{i=k+1}^{n} p_i \tag{2.18}$$

$$(1 - p_{max})^k * {p_{min}}^{n-k} \leq \prod_{i=1}^{k}(1 - p_i) * \prod_{i=k+1}^{n} p_i \leq (1 - p_{min})^k * {p_{max}}^{n-k} \tag{2.19}$$

Értelemszerűen, minél közelebb vannak egymáshoz a p_{min} és p_{max} értékek, annál élesebb ez a becslés. Következőnek vegyük azt, hogy az n műveleti álló hálózatban, ha csak a legfeljebb K meghibásodást tartalmazó részhálózatokat vesszük figyelembe a számítás során, és ezt a p_{min} és p_{max} érték segítségével becsüljön, akkor milyen összefüggés határozható meg. A teljes rendszer megbízhatóságát alulról tudjuk becsülni, ha minden műveleti egység megbízhatóságát p_{min}-re állítjuk, valamint felülről tudjuk becsülni, ha p_{max}-ra. Ha az így kapott alsó, illetve felső becslést $\hat{r}_{min}$ és $\hat{r}_{max}$ jelöli, akkor $\hat{r}_{min} \leq \hat{r} \leq \hat{r}_{max}$. Ez akkor is igaz, ha csak a legfeljebb K meghibásodást tartalmazó részhálózatokat vesszük figyelembe, ahol jelölje a tényleges értéket, valamint az alsó és felső becslést $\hat{r}^K$, $\hat{r}^K_{min}$, illetve $\hat{r}^K_{max}$. Ekkor $\hat{r}^K_{min} \leq \hat{r}^K \leq \hat{r}^K_{max}$. Ez utóbbi állítás a rendszer koherenciája miatt teljesül, hiszen a maximum K meghibásodást megengedő számításra úgy is lehet tekinteni, mintha a rendszer működési feltételeibe határoznánk meg, hogy K-nál több meghibásodással nem tud működni (és ezt le is lehet írni a Ψ^K függvénnyel). Habár a sok meghibásodást tartalmazó részhálózatok esetén már gyakori, hogy a műveleti egységek megbízhatóságainak csökkentése növeli a részhálózat valószínűségét, mivel $\hat{r}^K$, $\hat{r}^K_{min}$, és $\hat{r}^K_{max}$ értékek meghatározásakor a módosított U halmaz koherens rendszert ír le, így bármely műveleti egység megbízhatóságának csökkentése biztosan csökkenti a rendszer megbízhatóságát is. Valamint az is teljesül, hogy $\hat{r}^K_{min} \leq \hat{r}_{min}$, $\hat{r}^K \leq \hat{r}$, valamint $\hat{r}^K_{max} \leq \hat{r}_{max}$.

Mivel $\hat{r}^K \leq \hat{r}$, a fő kérdés az F_{max} érték meghatározásánál, hogy mennyire szeretnénk közel kerülni a tényleges megbízhatósághoz. Más szóval, mi az az ϵ határérték, ami alá kell vinni a $\hat{r} - \hat{r}^K$ értéket. Ez az $\hat{r} - \hat{r}^K$ érték nem más, mint az K-nál több meghibásodást tartalmazó, de még működőképes részhálózatok valószínűségeinek összege, vagyis azon $(x_1, x_2, \dots, x_n)$ részhálózatoké, ahol $x_1 +$

$x_2 + \cdots + x_n < n - K$. Értelemszerűen ezt felülről lehet becsülni, ha összeadjuk az összes, K-nál több meghibásodást tartalmazó részhálózat valószínűségét, a működőképességtől függetlenül. Erre lehet még egy felső becslést adni, ha alkalmazzuk a (2.19) formulát minden ilyen részhálózatra. A (2.20) számítás részletezi a levezetést. Egy n műveleti egységet tartalmazó rendszerben pontosan k meghibásodás tartalmazó részhálózatból $\binom{n}{k}$ darab van. Mivel ezek valószínűsége mind ugyanazzal az értékkel becsülhető felül, ezért a formulában összevonhatóak, és a becslésre a (2.21) formula írható fel.

$$\begin{aligned}
\hat{r} - \hat{r}^K &\leq \sum_{\substack{(x_1,x_2,\ldots,x_n)\in U \\ x_1+x_2+\cdots+x_n<n-K}} \left(\prod_{i=1}^{n} {p_i}^{x_i} (1-p_i)^{(1-x_i)} \right) \leq \\
&\leq \sum_{\substack{(x_1,x_2,\ldots,x_n)\in \Omega \\ x_1+x_2+\cdots+x_n<n-K}} \left(\prod_{i=1}^{n} {p_i}^{x_i} (1-p_i)^{(1-x_i)} \right) \leq \\
&\leq \sum_{\substack{(x_1,x_2,\ldots,x_n)\in \Omega \\ x_1+x_2+\cdots+x_n<n-K}} \left(\prod_{\substack{i=1 \\ x_i=0}}^{n} (1-p_{min}) * \prod_{\substack{i=1 \\ x_i=1}}^{n} p_{max} \right) = \\
&= \sum_{\substack{(x_1,x_2,\ldots,x_n)\in \Omega \\ x_1+x_2+\cdots+x_n<n-K}} \left((1-p_{min})^{n-(x_1+x_2+\cdots+x_n)} * {p_{max}}^{x_1+x_2+\cdots+x_n} \right)
\end{aligned} \tag{2.20}$$

$$\hat{r} - \hat{r}^K \leq \sum_{k=K+1}^{n} \left(\binom{n}{k} * (1-p_{min})^k * {p_{max}}^{n-k} \right) \tag{2.21}$$

Az így kapott (2.21) formula bármilyen n, K, p_{min} és p_{max} értékekre gyorsan meghatározható. Mivel az n, p_{min} és p_{max} értékek a feladat paraméterei, így a számítás során csak a K állítható. A formula jobb oldalán lévő érték a K függvényében monoton csökkenő. Így tehát az F_{max} értéket arra a legkisebb K-ra kell megválasztani, amelyre a (2.21) formula jobb oldali értéke kisebb, mint ϵ.

A becslés alkalmazására vegyünk egy példát $n = 30$ műveleti egységgel. A hagyományos ekkor $2^n = 1\,073\,741\,824$ részhálózatot vizsgálna végig, ami már egy elég nagy érték ahhoz, hogy az algoritmus hosszú ideig fusson. A 2.2. táblázat mutatja a kiszámolt hiba becsléseket különböző K, p_{min}, és p_{max} értékek esetén. Minden p_{min}, és p_{max} páros esetén az első oszlopban a K értéke látható, a második oszlopban a $\hat{r} - \hat{r}^K$-ra adott felső becslés, másszóval az, hogy legfeljebb mennyivel térünk el a valós megbízhatóságtól, a harmadik oszlopban pedig a megvizsgált részhálózatok darabszáma. Minden p_{min}, és p_{max} párokhoz csak néhány K értékre látható az eredmény. Az ezeknél kisebb K értékekre túl nagy a várható hiba, a nagyobb K értékekre pedig annyira közel lesz a tényleges megbízhatósághoz, hogy nem éri meg ennél több részhálózatot megvizsgálni. Az értékeket megvizsgálva látható az, amire a (2.21) formulát elemezve is lehet következtetni: minél nagyobb a p_{min}, illetve minél kisebb a p_{max}, annál kisebb a hiba, vagyis annál kisebb K értékre adhat a közelítés megfelelően pontos eredményt.

2.2. táblázat: A (2.21) formula által adott hibabecslés eredménye $n = 30$ műveleti egységre, különböző K, p_{min}, és p_{max} paraméterekkel

$p_{min} = 0{,}99;\ p_{max} = 0{,}99$			$p_{min} = 0{,}99;\ p_{max} = 0{,}999$		
K	$\hat{r} - \hat{r}^K \leq$	részhálózatok	K	$\hat{r} - \hat{r}^K \leq$	részhálózatok
3	$2{,}22 * 10^{-4}$	4 526	3	$2{,}82 * 10^{-4}$	4 526
4	$1{,}16 * 10^{-5}$	31 931	4	$1{,}45 * 10^{-5}$	31 931
5	$4{,}83 * 10^{-7}$	174 437	5	$6{,}00 * 10^{-7}$	174 437
6	$1{,}66 * 10^{-8}$	768 212	6	$2{,}05 * 10^{-8}$	768 212
7	$4{,}81 * 10^{-10}$	2 804 012	7	$5{,}87 * 10^{-10}$	2 804 012
8	$1{,}18 * 10^{-11}$	8 656 937	8	$1{,}43 * 10^{-11}$	8 656 937
$p_{min} = 0{,}95;\ p_{max} = 0{,}99$			$p_{min} = 0{,}95;\ p_{max} = 0{,}999$		
K	$\hat{r} - \hat{r}^K \leq$	részhálózatok	K	$\hat{r} - \hat{r}^K \leq$	részhálózatok
7	$2{,}09 * 10^{-4}$	2 804 012	7	$2{,}54 * 10^{-4}$	2 804 012
8	$2{,}53 * 10^{-5}$	8 656 937	8	$3{,}05 * 10^{-5}$	8 656 937
9	$2{,}64 * 10^{-6}$	22 964 087	9	$3{,}16 * 10^{-6}$	22 964 087
10	$2{,}39 * 10^{-7}$	53 009 102	10	$2{,}84 * 10^{-7}$	53 009 102
11	$1{,}89 * 10^{-8}$	107 636 402	11	$2{,}23 * 10^{-8}$	107 636 402
12	$1{,}31 * 10^{-9}$	194 129 627	12	$1{,}53 * 10^{-9}$	194 129 627

2.5. A fejezethez kapcsolódó tézis

1. Tézis: Általános formulát dolgoztam ki folyamat hálózatok megbízhatóságának meghatározására, feltéve, hogy a folyamatot alkotó műveleti egységek megbízhatósága adott.

a) A formula a működőképes hálózatok P-gráf algoritmusokon alapuló kombinatorikus leszámlálására épül.
b) A formulával meghatározható a strukturális megbízhatóság, ha a leszámlálás a kombinatorikusan lehetséges struktúrák alapján történik, valamint az általános megbízhatóság, ha a leszámlálás a lehetséges struktúrák alapján történik
c) Gyorsító eljárásokat javasoltam a formula meghatározásához szükséges számítási igény csökkentésére.
d) Bemutattam a formula alkalmazását több esettanulmánnyal.

2.6. A fejezethez kapcsolódó publikációk

Á. Orosz, F. Friedler, P. S. Varbanov, and J. J. Klemes, “Systems reliability, footprints and sustainability,” *Chemical Engineering Transactions*, vol. 63, pp. 121–126, 2018, doi: 10.3303/CET1863021.

Z. Kovacs, A. Orosz, and F. Friedler, “Synthesis algorithms for the reliability analysis of processing systems,” *Central European Journal of Operations Research*, vol. 27, no. 2, pp. 573–595, 2019, doi: 10.1007/s10100-018-0577-0.

3. fejezet

Strukturális redundancia: lokális és globális

Egy műszaki rendszert redundancia-mentesnek nevezünk, ha bármely műveleti egységének meghibásodása a rendszer meghibásodását jelenti. Műszaki rendszerek esetén fontos a magas megbízhatóság, amit redundancia-mentes felépítéssel nem lehet megvalósítani, vagyis redundáns rendszereket kell tervezni. A megfelelő redundancia meghatározása a szintézis vagy tervezés folyamat része.

Egy redundáns rendszernek több különböző működési módja van, amelyek az aktív műveleti egységekben térnek el. A működési módok közül a legkedvezőbb az alap működési mód, amely a rendszer alapértelmezett működésének felel meg. Amennyiben az alap működési mód szerint nem tud üzemelni a rendszer (egy vagy több műveleti egység meghibásodásának következtében), az üzemeltetés valamelyik tartalék működési módra vált, amennyiben van olyan, ami az eddigi meghibásodások után még működőképes. A rendszer működési módjainak száma, illetve azok paraméterei nagymértékben függenek a rendszerben lévő redundanciáktól.

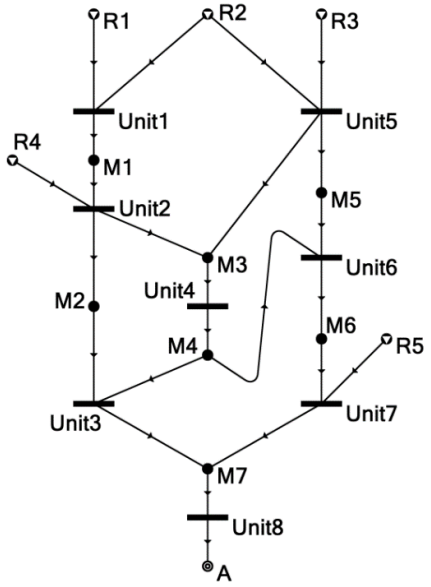
A gyakorlatban leggyakrabban használt redundancia az műveleti egység szintű (vagy lokális) redundancia, ahol egy műveleti egység mellé párhuzamosan helyezünk további, ugyanazt a műveletet megvalósító műveleti egységeket. A műveleti egységek közül az egyik aktív az idő nagy részében, a többi tartalék a meghibásodások esetére. A tartalék műveleti egységek lehetnek az aktív műveleti egységgel megegyezők, de lehetnek eltérőek is. A lokális redundanciának egy alternatív változata a k-out-of-n redundancia, ahol n ekvivalens műveleti egység van párhuzamosan, és ezek közül legalább k működése szükséges a rendszer működőképességéhez.

Az műveleti egység szintű redundancia megvalósítása egyszerű, de nem mindig adja meg az optimális rendszert. A rendszerben a tartalék működési módok jellemzően csak a teljes idő kis részében aktívak, az idő nagy részét a rendszer az alap működési módban tölti. A költségre optimalizált rendszerben a tartalék működési mód hálózata jelentősen eltérhet az alap működési módétól, különösen, ha a redundanciát biztosító tartalék műveleti egységek beruházási költsége alacsony. Ez a folyamat szintű (vagy globális) redundancia, ahol a műveleti egységek meghibásodásai esetén a rendszer valamely funkcióit egy tartalék részrendszer végzi el, amelynek hálózata eltérhet az eredetitől. A következő példa szemlélteti ezt a lehetőséget egy szintézis feladaton keresztül.

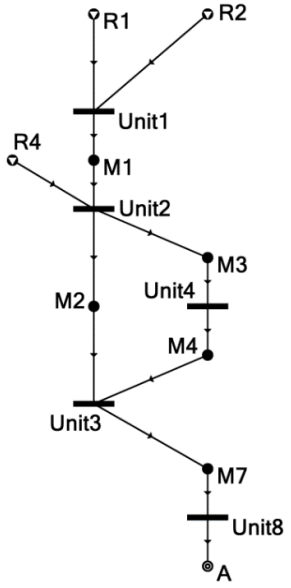
3.1. szemléltető példa

A 3.1. ábrán látható a szintézis feladat maximális hálózata. A folyamat funkciója az A csúccsal jelölt termék gyártása úgy, hogy a folyamat

megbízhatósága legalább 0,999 legyen. Amennyiben a szintézis csak a költséget veszi figyelembe, az optimális megoldás a 3.2. ábrán látható hálózat 9 312 USD/y költséggel. Ez egy redundancia-mentes hálózat, aminek a megbízhatósága (0,989) nem teljesíti az elvárásokat. Ahhoz, hogy a szintézis a megbízhatóságot is figyelembe vegye, nem elég a legkisebb költségű megoldás megtalálása. Ehelyett a P-gráf keretrendszer által biztosított SSG algoritmus segítségével felsorolható az összes lehetséges hálózat. Ezek mindegyikére a költség mellett a megbízhatóság is meghatározható a 2. fejezetben ismertetett algoritmussal. Így megkapjuk az összes lehetséges megoldást költséggel és megbízhatósággal, amik közül a megfelelő kiválasztható.



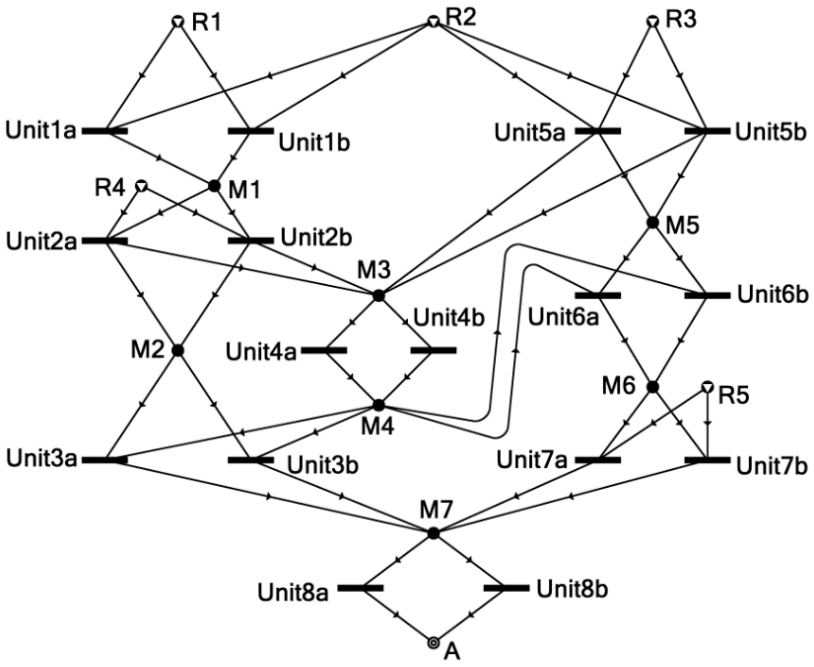
3.1. ábra: A 3.1. szemléltető példa maximális hálózata



3.2. ábra: A 3.1. ábra szintézis feladatának optimális megoldása, a megbízhatósági feltétel nem teljesül

Ennek a szintézis feladatnak egyik redundancia mentes megoldása sem képes az elvárt megbízhatóság elérésére, mindenképp szükség van redundanciára. Ezért a redundanciát már a szintézis során figyelembe kell venni. Jelenleg ennek megvalósításához az szükséges, hogy a maximális struktúra tartalmazza az összes lehetőséget a redundancia alkalmazására. Ezáltal az SSG algoritmus képes generálni az összes hálózatot, amelyekben a megengedett mennyiségű redundancia összes lehetséges alkalmazása szerepel. Az így generált hálózatok mindegyikének megbízhatósága külön meghatározható, ezzel megkapva a lehetséges megoldásokat.

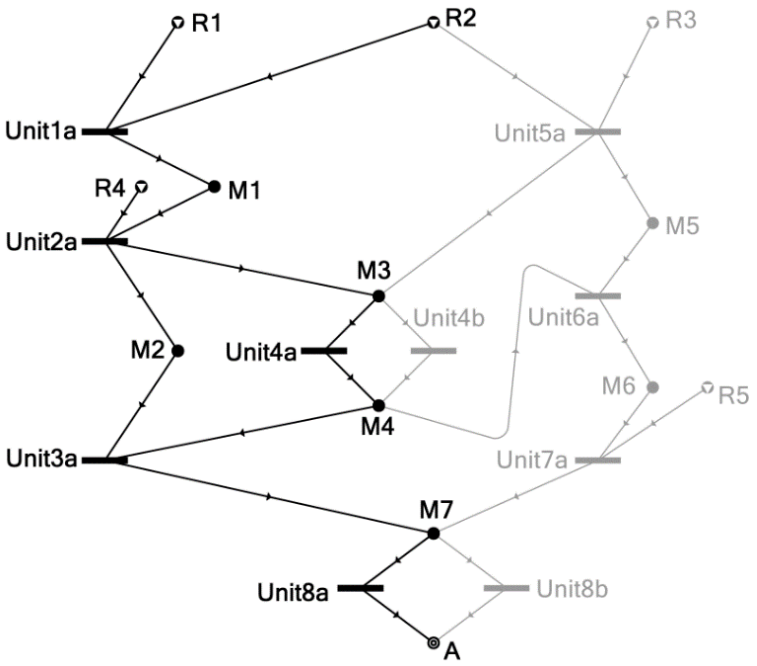
A 3.3. ábrán látható a maximális hálózat, ezúttal úgy, hogy minden műveleti egység mellé egy redundáns tartalékot helyezünk, amiket az ábra külön jelöl. Ez engedi meg a szintézis során a műveleti egység szintű redundancia lehetőségét. A valóságban egy műveleti egységhez több redundáns tartalék egység is tartozhat, így a gyakorlatban szükség lehet arra, hogy a maximális hálózaton 3, 4, vagy akár több redundáns műveleti egység legyen egy feladatra. A megoldás során azonban javasolt a redundáns lehetőségek számát minimalizálni, hiszen a lehetőségek növelésével jelentősen nő a megoldások száma is, ami növeli a futási időt. Jelen feladat esetében a műveleti egységek megbízhatóságai alapján meghatározható, hogy a kívánt megbízhatóság eléréséhez 2 párhuzamos műveleti egység is elegendő.



3.3. ábra: A 3.1. ábrán lévő maximális hálózat kiegészítése redundáns műveleti egységekkel

A szintézis által generált hálózatok megbízhatóságát is figyelembe véve a 3.4. ábra mutatja azt a hálózatot, amely a megbízhatósági korlátot teljesíti, és ezek közül a legkisebb költséggel rendelkezik. A vastag vonalak jelölik az alap működési módhoz tartozó hálózatot, a vékony szürke vonalak pedig a tartalék műveleti egységeket. Látható, hogy ez az optimális megoldás tartalmaz folyamat

szintű és műveleti egység szintű redundanciát is. A hagyományos módszer, a 3.2. ábra hálózatának kiegészítése műveleti egység szintű redundanciával, nagyobb költséget eredményez. Ez mutatja, hogy fontos a folyamat szintű redundanciát figyelembe venni, ha a folyamat szintézis tartalmaz megbízhatósági feltételt.



3.4. ábra: A 3.3. ábra szintézis feladatának optimális megoldása, ahol a megbízhatósági feltétel is teljesül (a vastag vonalak jelölik az alap működési módot)

Az lokális és globális redundancia kapcsán egy gyakran felmerülő kérdés, hogy mikor melyiket éri meg alkalmazni. A lokális redundancia mellett több érv is szól, amelyek közül a legyakoribb az, hogy egyszerű. A folyamat műveleti egységeinek ismeretében a redundancia beépítéséhez annyit kell tenni, hogy be kell szerezni több ugyanolyan műveleti egységet, és azokat párhuzamos módon kell beépíteni a rendszerbe. Egyszerűen látható, hogy mi a reakció egy meghibásodás esetén, valamint a megbízhatóság meghatározása is könnyebb.

Vannak azonban olyan szempontok, amelyek a globális redundanciát támogatják. Az egyik szempont a költség. Globális redundanciával az alap működési mód nagyobb részei teljesen kiválthatóak hiba esetén. A tartalék működési mód jellemzően a rendszer működési idejének csak egy töredékét teszi ki. Ilyen esetben úgy költség szempontjából hatékonyabb lehet, ha a redundáns működést biztosító műveleti egységek egy olyan részhálózatot alkotnak, amelynek a működtetési költsége arra a rövid időre nagyobb, mint az alap működésé, cserébe a beruházási költsége jelentősen kisebb. Ezen felül a globális redundancia hibatűrőbb is lehet, amennyiben figyelembe vesszük, hogy a műveleti egységek meghibásodása kihatással lehet más műveleti egységekre is. Egy veszélyes anyagokkal dolgozó műveleti egység meghibásodása megkárosíthatja a vele párhuzamosan beszerelt, redundáns egységet is, ezáltal ellehetetlenítve a tartalék működést. A globális redundanciát biztosító műveleti

egységek viszont lehetnek az üzem egy távolabbi részében, így kisebb az esélye az alap és a tartalék egységek egyöntetű meghibásodásának.

3.1. A fejezethez kapcsolódó tézis

2. Tézis: Globális strukturális redundanciát megengedő folyamat hálózatok esetére igazoltam, hogy a gyakorlatban általában alkalmazott műveleti egység szintű (lokális) redundancia kizárhatja a globálisan optimális hálózatot, így az optimum garantált megatalálásához a folyamat szintű (globális) redundanciát is figyelembe kell venni.

3.2. A fejezethez kapcsolódó publikációk

A. Orosz, Z. Kovacs, and F. Friedler, “Processing Systems Synthesis with Embedded Reliability Consideration,” *Computer Aided Chemical Engineering*, vol. 43, pp. 869–874, 2018, doi: 10.1016/B978-0-444-64235-6.50152-2.

Á. Orosz and F. Friedler, “Synthesis technology for failure analysis and corrective actions in process systems engineering,” *Computer Aided Chemical Engineering*, vol. 46, pp. 1405–1410, 2019, doi: 10.1016/B978-0-12-818634-3.50235-6.

4. fejezet

Folyamat hálózat szintézis a megbízhatóság figyelembe vételével

Folyamat hálózatok tervezésére a tradicionális megoldás a feladat dekomponálása volt kisebb feladatok sorozatára. Ennek egyik legismertebb példája a Douglas [10] által publikált hierarchikus eljárás. Ezen módszerek sorozatban megoldandó lépései nem függetlenek egymástól, így viszont nem garantálható a globális optimum megtalálása. Ezek a szekvenciális módszerek bizonyos feladatok esetén képesek lehetnek az optimális megoldás megtalálására, de a legtöbb esetben inkább csak közelítő megoldást adnak. Következésképp, a lépések együttes elvégzése lehetőséget ad a kapott eredmények javítására, így fontos a hatékony algoritmusok készítése szempontjából. Ez igaz akkor is, amikor több indikátort is figyelembe kell venni az optimalizálás során, mint a megbízhatóság kiszámításának integrálása a költség alapú folyamat hálózat szintézisbe. Az így kapott módszer képes a feladat adott feltételek melletti legjobb megoldásának megtalálására. A megbízhatóság bonyolult modellezése miatt nem létezik hatékony integrált módszer a feladatra.

Csak kevés publikáció tárgyalja a megbízhatóság figyelembe vételét szintézis során. Pistikopoulos et al. [87] a műveleti egység szintű redundancia optimalizálását vizsgálták folyamatokban. Goel et al. [60] egy meglévő HDA folyamatot egészítettek ki redundáns egységekkel. Terrazas-Moreno et al. [88] a megbízhatóságot és a rugalmasságot is figyelembe vették a gyártó kapacitások és köztes tárolók optimalizálása során. Ezek a módszerek azonban nem képesek a szisztematikus, minden megoldást kimerítően generáló szintézisre.

Ahhoz, hogy a megbízhatóság kiszámítása beépíthető legyen szintézis algoritmusokba, egy olyan eszközre van szükség, ami egyszerre képes megvalósítani a szintézist és a megbízhatóság meghatározást. Jelen munkában ezt a szerepet a PNS feladatok megoldásához alkotott P-gráf keretrendszer [15] tölti be. A keretrendszer megfelelő kiegészítésével a megbízhatósági szempont integrálható a szintézisbe. A kapcsolat kulcsa a folyamatok működőképessége és a kombinatorikus lehetségesség közötti összefüggés. Mindkét tulajdonság erősen struktúra alapú, A P-gráf keretrendszer megfelelően tudja ábrázolni mindkettőt az integrált algoritmusokban.

Benjamin et al. [89] sikeresen alkalmazták a P-gráf keretrendszert egy bioenergia létesítmény kockázat elemzésére csökkentett kapacitás esetére. Benjamin [90] elvégezte a kockázat elemzést bizonytalan kereslet esetére is. Tan et al. [91] minimalizálta a termelő rendszer meghibásodásokból adódó működési zavarait.

A P-gráf keretrendszer a működőképes hálózatok strukturális tulajdonságara

épít. Ezek a tulajdonságok általánosak, függetlenek a műveleti egységek működését leíró matematikai modellektől, ezért a keretrendszer maga is általános. A dolgozatban szereplő példákban több modell is előfordul. A műveleti egységek meghibásodásai jelen esetben egymástól független, idő-invariáns események. Ezen felül a műveleti egységek megbízhatóságai ismertek, a feladatok bemeneti adatai között szerepelnek.

4.1. Szintézis algoritmus kiterjesztése megbízhatóság figyelembe vételére

A korábban ismertetett módszer alkalmas egy hálózat megbízhatóságának meghatározására. Jelen fejezet megmutatja, hogy ezt a számítást be lehet építeni a P-gráf módszertan szintézis algoritmusaiba.

A szintézis mindig az MSG algoritmus [30] futtatásával kezdődik, amely potenciálisan csökkenti a feladat méretét. Ezután az optimális megoldás meghatározása során az első lépés az összes kombinatorikusan lehetséges hálózat generálása az SSG algoritmus [3] segítségével, majd ezek egyesével kiértékelése következik. A kiértékelés után a kapott hálózatok közül a legkedvezőbb kiválasztható.

Ahhoz, hogy a költség optimális hálózat meghatározható legyen, definiálni kell a költség kiszámításának módját. Hagyományos szintézis esetén a rendszer költsége két részből tevődik össze, a beruházási költségből és az üzemeltetési költségből. A beruházási költséget egyszer kell kifizetni, a rendszer kezdeti felépítése során. Ez tartalmazza a műveleti egységek árát, valamint beleszámolható a beüzemelés, összeszerelés költsége is. Az üzemeltetési költség folyamatos, amíg a rendszer működik, tipikusan az üzemeltetéshez szükséges erőforrások (áram, víz) árait, valamint a nyersanyagok beszerzését és a melléktermékek kezelését tartalmazza. Az üzemeltetési költséget valamilyen időszakra vetítve szokás megadni, például éves szinten. A beruházási költség egyszeri, de a szintézis során definiálni szokás egy megtérülési időintervallumot, amire leosztva a költség beleépíthető az üzemeltetési költségbe.

A feladat definíciója meghatározhat egyéb költség elemeket. A megbízhatósági számítások során például a tartalék műveleti egységek költségeit is figyelembe kell venni. Értelemszerűen a tartalék műveleti egységekhez is tartozik beruházási költség, ami egyszerűen hozzáadható a többi beruházási költséghez. A rendszer üzemeltetési költségét elsősorban az alap működési mód szerinti üzemeltetésre kell meghatározni, hiszen ez a tervezett működés. Alacsony megbízhatóságú rendszerek esetében a tartalék működési módok üzemeltetési költségeit is fontos a költségfüggvénybe építeni. Erre a leggyakoribb megoldás az átlagos üzemeltetési költség meghatározása, amely az egyes működési módok várható üzemideje és azok üzemeltetési költsége alapján meghatározza a rendszer átlagos költségét (a várható értéket) az adott időintervallumra. Nagy megbízhatóságú rendszerek az idő nagy részét az alap működési mód szerint üzemelve töltik, a tartalék működési módokra csak nagyon ritkán van szükség. Ilyen rendszerek esetében a tartalék működési módok üzemeltetési költségétől akár el is lehet tekinteni a szintézis folyamatban, hiszen a magas megbízhatóság miatt az átlagos

üzemeltetési költség és az alap működési mód üzemeltetési költsége között elhanyagolható az eltérés. Viszont nagy megbízhatóságú rendszerek esetében is lehet a tartalék működési módok üzemeltetési költségére egy felső határt szabni.

A dolgozat elsősorban nagy megbízhatóságú rendszerek tervezésével foglalkozik, így az üzemeltetési költség meghatározása során az alap működési mód üzemeltetési költsége lesz a fő szempont, míg a beruházási költség természetesen minden műveleti egység árát tartalmazza. A feladat figyelembe veheti a műveleti egységek karbantartási költségét, ami szintén az üzemeltetési költség része lesz. Magasabb megbízhatóság magasabb karbantartási költséget von maga után. A tartalék működési módok kiválasztását a megbízhatóságon kívül egyéb korlátok befolyásolhatják.

Fontos meghatározni a célfüggvényt is, amit a szintézis során optimalizálni kell. Hagyományos szintézis feladatoknál a célfüggvény egyszerűen a hálózat költsége. Amikor azonban több mérőszám is tartozik egy hálózathoz, mint jelen esetben a költség és a megbízhatóság, akkor többcélú optimalizálásról van szó, és a célfüggvény nem ennyire egyértelmű.

Jelen munkában a többcélú optimalizálás három irányát tekintjük, amelyek megbízható rendszerek tervezése esetén előfordulnak. A szemléltető példa mindig definiálja az adott feladat célfüggvényét.

A leggyakoribb megközelítés a szakirodalomban az, hogy a megbízhatóságot beépítik a költségbe, így a szintézis során egy célfüggvény lesz. Ez történhet az átlagos üzemeltetési költség meghatározásával, ahol az üzemeltetési költség várható értékét határozzák meg a megbízhatóság és az egyes működési módok valószínűségei alapján. Másik lehetőség egy alsó korlát meghatározása a megbízhatóságra, ahol a túl alacsony megbízhatóság büntetést (plusz költséget) von maga után. Lehetséges felső korlátot is meghatározni, és jutalmazni, ha ennél magasabb a megbízhatóság.

Többcélú optimalizálásnál gyakori a Pareto front meghatározása, vagyis a Pareto optimális megoldások megkeresése. Ez a megközelítés több megoldást is ad a feladatra, amelyek közül egyik sem dominálja a másikat, vagyis a lista nem tartalmaz olyan rendszert, aminél lehet magasabb megbízhatóságút tervezni olcsóbban. A generált hálózatok közül a legkedvezőbb kiválasztása a döntéshozó feladata.

A harmadik megközelítés, amikor az indikátorok közül az egyiket kinevezzük belső indikátornak, a többit pedig külső indikátornak. A belső indikátor adja meg az optimalizálás célfüggvényét, a külső indikátorok pedig korlátozásokat határoznak meg. Megbízható rendszerek szintézise során a belső indikátor tipikusan a költség, a külső indikátor pedig a megbízhatóság. Például a cél egy olyan rendszer, aminek a költsége minimális, miközben a megbízhatósága elér vagy meghalad egy előre meghatározott értéket.

A feladat P-gráf formában modellezése után az MSG algoritmus meghatározza a maximális hálózatot. Ezután az SSG algoritmus generálja az összes kombinatorikusan lehetséges hálózatot. A következő lépésben az algoritmus az összes generált hálózatot megvizsgálja működőképesség, költség, illetve megbízhatóság szempontjából. A működőképesség vizsgálata történhet matematikai modell, folyamat analízis által vagy szimulációs programmal. Ennek része a működőképes hálózatok paraméterezése, ami alapján a költségük

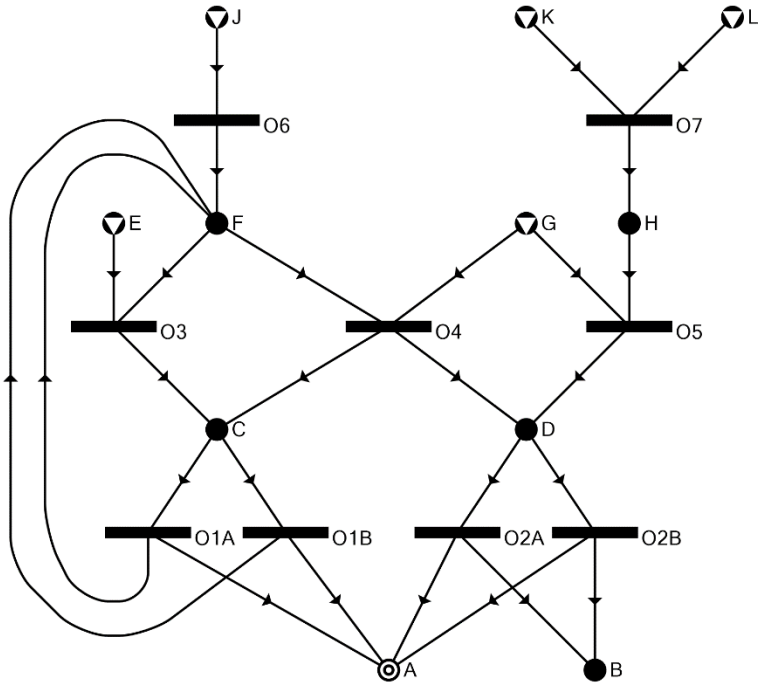
egyértelműen meghatározható. Az algoritmus ezzel meghatározza a lehetséges hálózatok halmazát. Végül minden így generált hálózat megbízhatósága meghatározható a 2. fejezetben leírt algoritmus szerint.

A módszer eredménye azon hálózatok halmaza, amelyek mind kombinatorikusan, mind az egyéb feltételeknek megfelelően lehetségesek, valamint ezen hálózatok költsége és megbízhatósága. Az eredmények alapján a legkedvezőbb megoldás a tervező igényeinek megfelelően meghatározható.

Ahogy a folyamat hálózat szintézis feladatoknál jellemző, a módszer futási ideje természetesen exponenciális, azonban a tényleges futási idő jelentősen eltérő lehet különböző feladatokra. Jelen módszer két fázisból áll. Az első az SSG algoritmus végrehajtása, amelynek futási ideje elsősorban a kombinatorikusan lehetséges hálózatok számától függ. Ehhez még hozzá adódik a működőképességi teszt futtatása minden hálózatra, ami a mögöttes matematikai modell függvénye. A második a hálózatok megbízhatóságának meghatározása. Mivel minden hálózat megbízhatóságát meg kell határozni, így a futási idő függ működőképes hálózatok számától, valamint azok méretétől. A legtöbb működőképes hálózat jellemzően jóval kevesebb műveleti egységből áll, mint a maximális struktúra, így a megbízhatóság meghatározása a legtöbb hálózat esetében gyors művelet, csak a nagyobb méretű hálózatok esetén tarthat több ideig.

4.1. szemléltető példa

A szemléltető példa MSG algoritmus által generált maximális hálózata a 4.1. ábrán látható. A műveleti egységekre egy egyszerű modell érvényes, vagyis minden műveleti egységnek csak egy fix beruházási költsége van, a folyamat költsége pedig a műveleti egységek költségeinek összege. Az anyagáramok, anyagköltségek, illetve a működési költségek most nem részei a feladatnak. Minden műveleti egység megbízhatósága állandó, és független az időtől. A megbízhatóság-, és költség adatok a 4.1. táblázatban láthatóak. Az O1A és O1B műveleti egységek alternatív választási lehetőségek. Az O1B egység nagyobb megbízhatósággal rendelkezik, mint az O1A egység, de drágább is. Az O2A és O2B műveleti egységekkel ugyanez a helyzet.

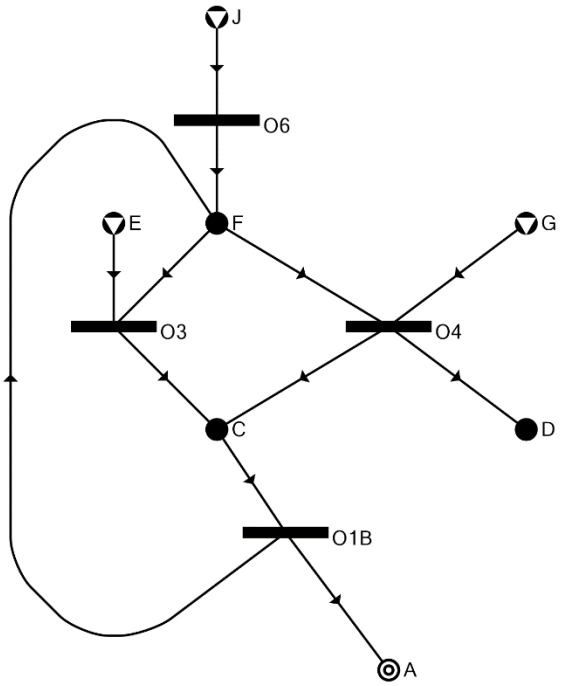


4.1. ábra: A 4.1. szemléltető példa maximális hálózata

A szemléltető példában 58 kombinatorikusan lehetséges hálózat van. Ezek mindegyikéhez meghatározható a költség és a megbízhatóság. A hálózat költsége a benne lévő műveleti egységek költségeinek összege. A megbízhatóságot a korábban ismertetett SRP algoritmus határozza meg. Példa gyanánt a 4.2. ábrán látható, #12-es hálózat költsége 30, a megbízhatósága pedig 0,99994999.

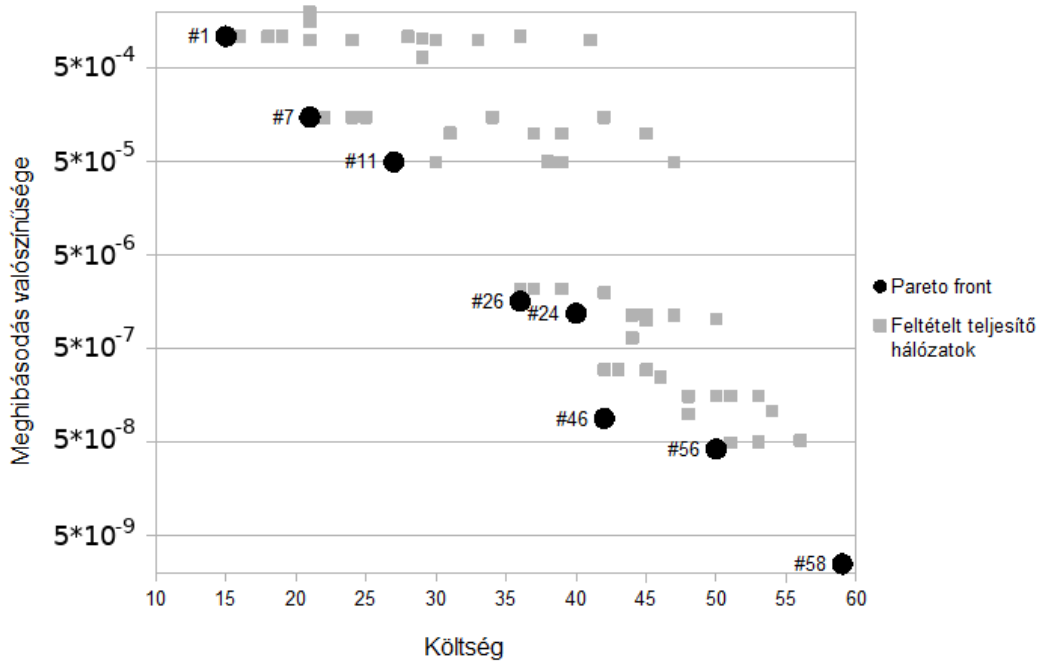
4.1. táblázat: A 4.1. szemléltető példa műveleti egységeinek megbízhatóságai és költségei

Műveleti egység	Megbízhatóság	Költség
O1A	0,99900	10
O1B	0,99995	16
O2A	0,99900	12
O2B	0,99995	20
O3	0,99990	5
O4	0,99990	6
O5	0,99950	4
O6	0,99950	3
O7	0,99950	5



4.2. ábra: A #12-es hálózat, a 4.1. szemléltető példa 58 lehetséges hálózatának egyike

Illusztráció gyanánt az 58 kombinatorikusan lehetséges hálózatra a költség és meghibásodási valószínűség összehasonlítását a 4.3. ábra mutatja, ahol a meghibásodás valószínűsége a hálózat megbízhatóságát 1-ből kivonva kapható meg, és logaritmikus skálán ábrázoltuk. 8 Pareto megoldás van a hálózatok között, ezek a 4.3. ábrán kiemelten láthatóak, a részletes adataik pedig a 4.2. táblázatban szerepelnek. Mindegyik hálózathoz látható a költség, megbízhatóság, meghibásodási valószínűség és a strukturálisan működőképes részhálózatok száma.



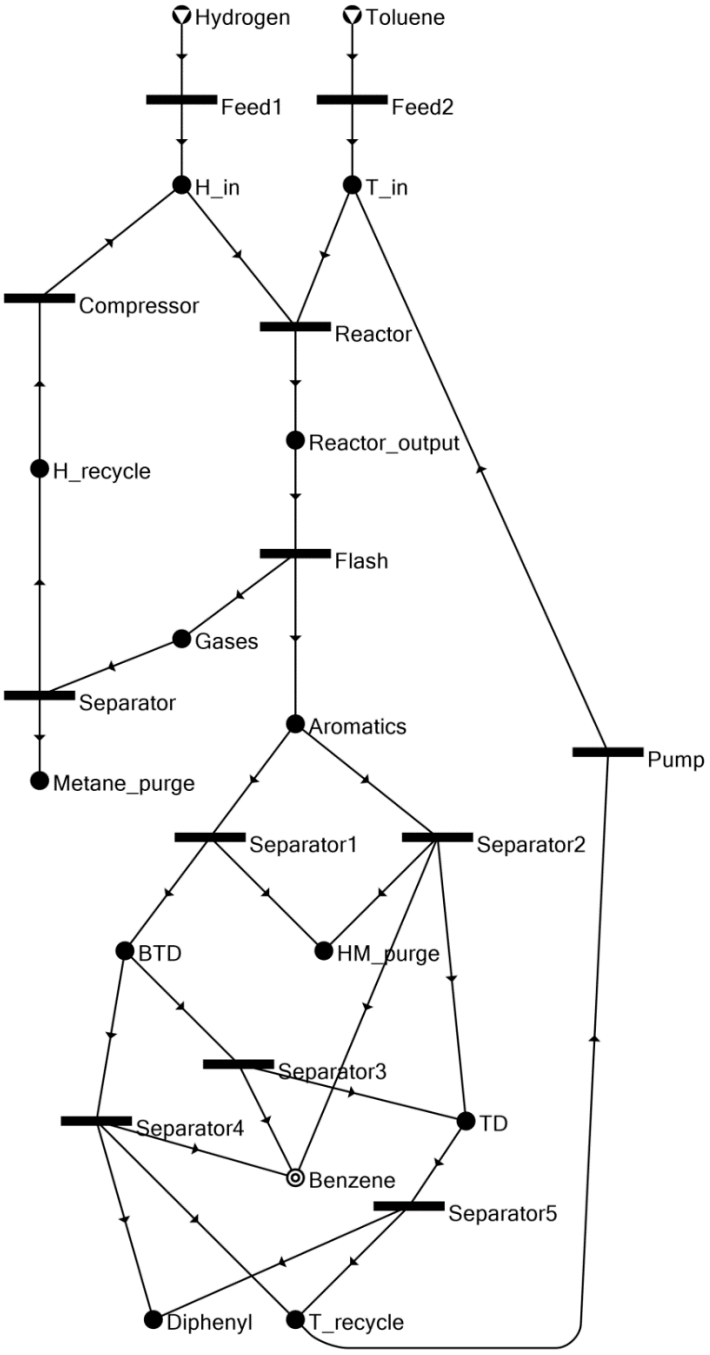
4.3. ábra: A 4.1. szemléltető példa 58 lehetséges hálózatának meghibásodási valószínűsége és költsége

4.2. táblázat: A 4.1. szemléltető példa Pareto megoldásainak adatai

Hálózat	Műveleti egységek	Megbízhatóság	Költség	Meghibásodási valószínűség	Strukturálisan működőképes részhálózatok száma
#1	O1A, O3	0,9989001000	15	0,0010999000	1
#7	O1B, O3	0,9998500050	21	0,0001499950	1
#11	O1B, O3, O4	0,9999499900	27	0,0000500100	3
#24	O1A, O2A, O4, O5, O6, O7	0,9999987997	40	0,0000012003	23
#26	O1A, O2A, O3, O4, O6	0,9999983907	36	0,0000016093	14
#46	O1B, O2A, O3, O4, O6	0,9999999100	42	0,0000000900	14
#56	O1B, O2B, O3, O4, O6	0,9999999575	50	0,0000000425	14
#58	O1B, O2B, O3, O4, O5, O6, O7	0,9999999975	59	0,0000000025	64

4.2. szemléltető példa

A jól ismert HDA folyamat [10] szintézise a szemléltető példa, amelyhez a maximális hálózat a 4.4. ábrán látható. Annak érdekében, hogy a megbízhatósági feltételek megvalósíthatóak legyenek, a maximális hálózatot több redundáns műveleti egységgel is ki kell egészíteni, különösen a reaktor, a szivattyú és a kompresszor esetében. A 4.3. táblázat tartalmazza a műveleti egységek költségeit és megbízhatóságait. A redundáns műveleti egységek darabszámára és kiválasztására nincs általános korlát, azonban jelen példa esetében, praktikai okokból, minden hálózat pontosan egy reaktort, maximum két pumpát, és maximum két kompresszort tartalmazhat.

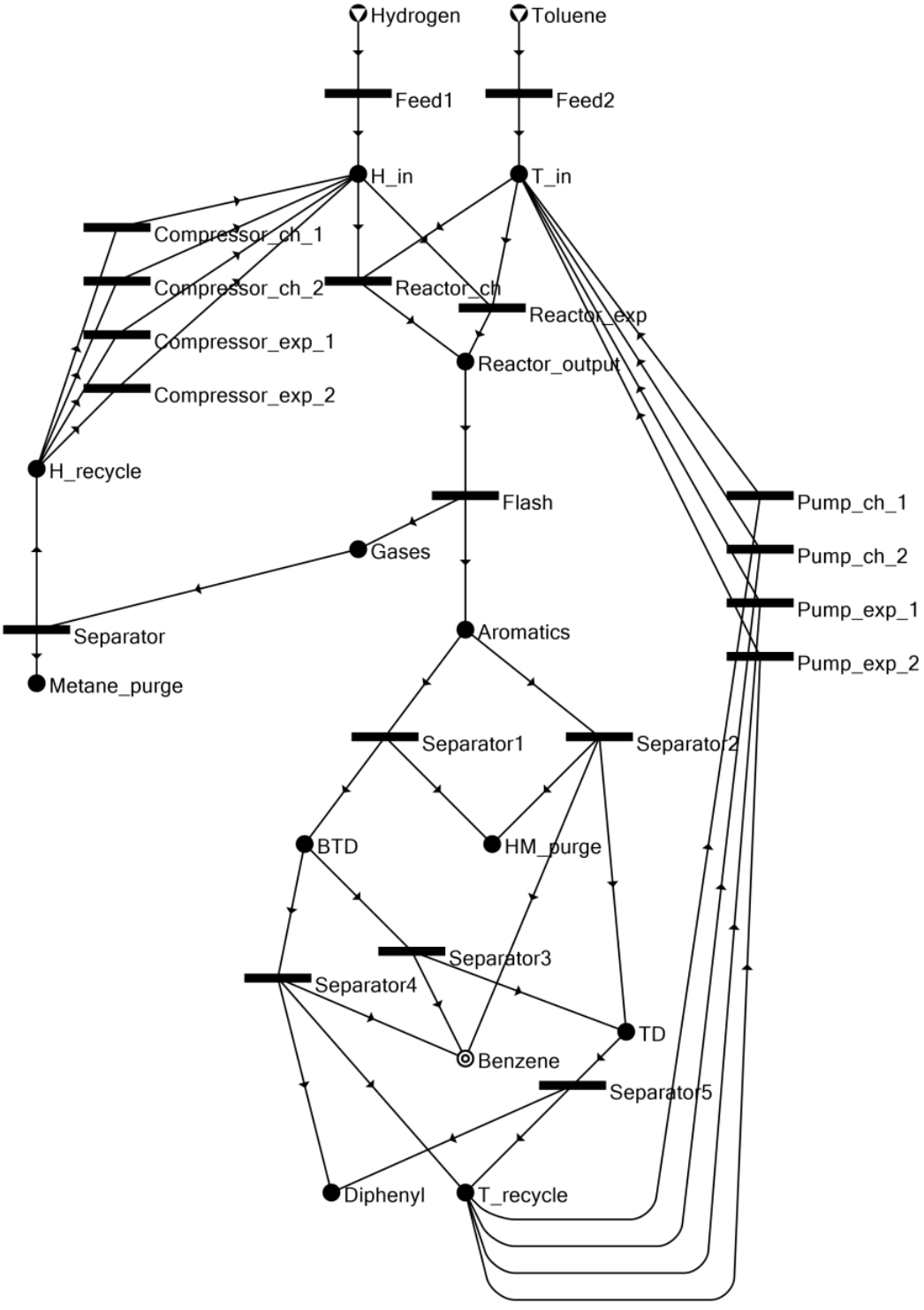


4.4. ábra: A 4.2. szemléltető példa maximális hálózata a redundáns opciók nélkül

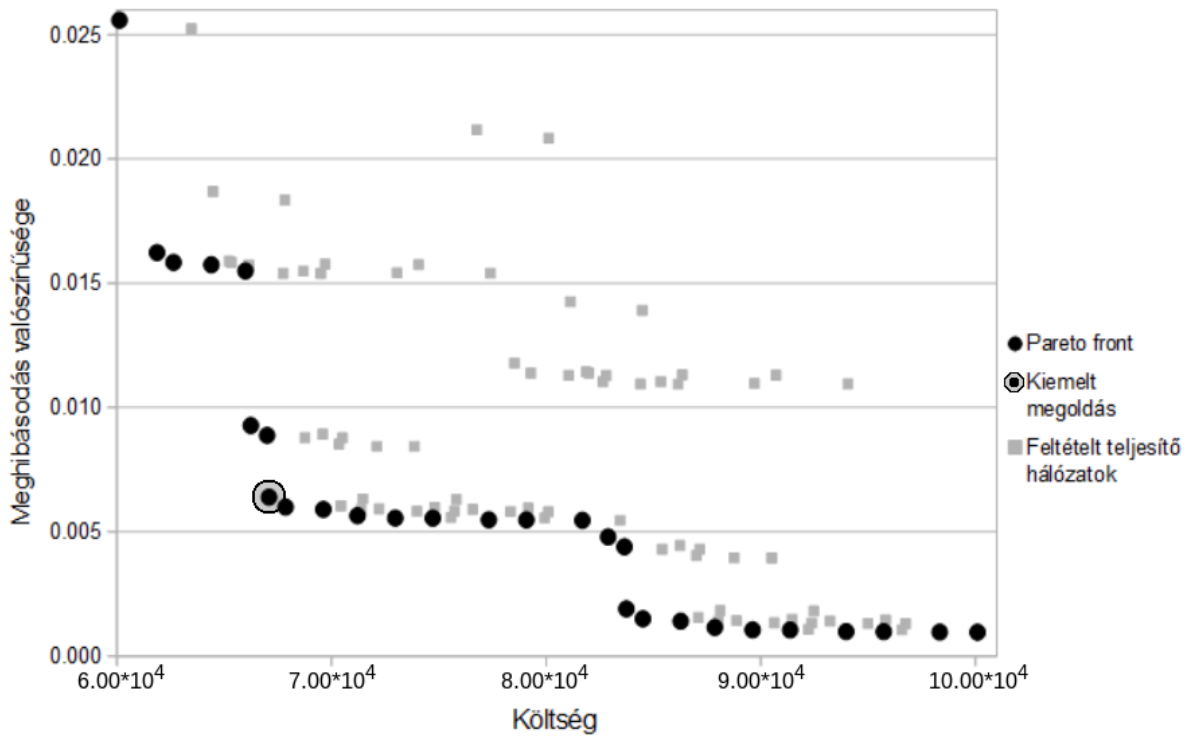
4.3. táblázat: A 4.2. szemléltető példa műveleti egységeinek adatai

Műveleti egység	Fix költség (CU)	Proporcionális költség (CU)	Megbízhatóság	Meghibásodás valószínűsége	Maximális kapacitás
Flash	1500	0,3	0,99995	0,00005	
Feed1	0	0,0	0,99995	0,00005	
Feed2	0	0,0	0,99995	0,00005	
Compressor_ch	1325	0,9	0,99000	0,01000	41550
Compressor_exp	2150	1,6	0,99950	0,00050	41550
Pump_ch	2815	5,2	0,99000	0,01000	
Pump_exp	5050	9,8	0,99700	0,00300	
Reactor_ch	10400	1,3	0,99600	0,00400	14150
Reactor_exp	21500	1,9	0,99920	0,00080	14150
Separator	2300	0,2	0,99990	0,00010	41550
Separator1	2400	0,3	0,99995	0,00005	25000
Separator2	10500	0,9	0,99950	0,00050	
Separator3	9600	0,8	0,99990	0,00010	14150
Separator4	10500	0,9	0,99950	0,00050	
Separator5	1600	1,2	0,99995	0,00005	

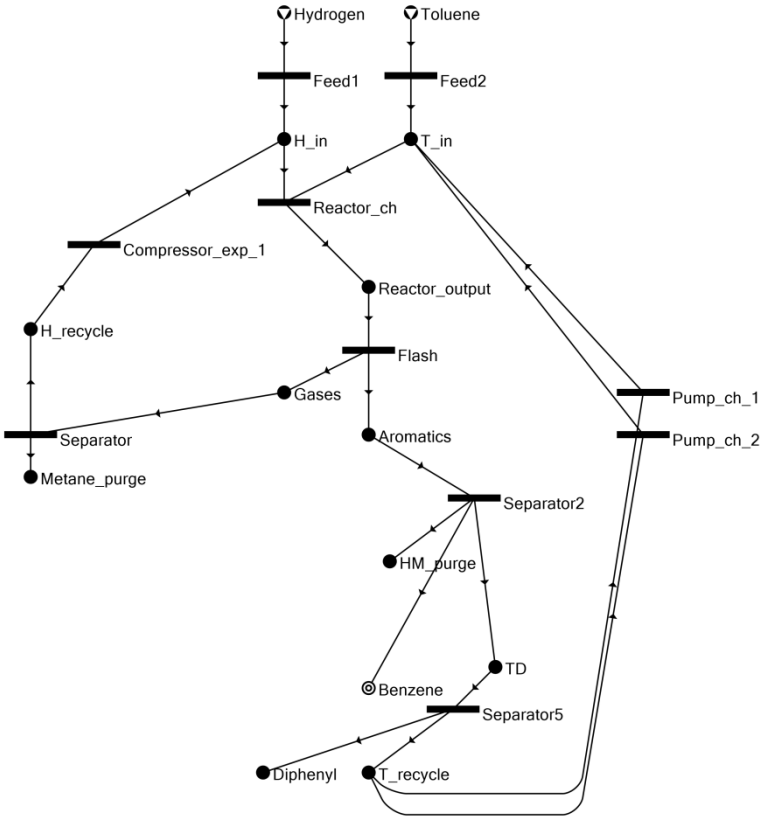
A megbízhatóság növelése érdekében a maximális hálózat kiegészült redundáns műveleti egységekkel a reaktorhoz, pumpához és a kompresszorhoz, a kiegészített maximális hálózat a 4.5. ábrán látható. A műveleti egységek modelljeit lineáris egyenletek írják le, a költségeik fix részt tartalmazó lineáris függvények. 28 551 kombinatorikusan lehetséges hálózat létezik, amelyeket az SSG algoritmus néhány másodperc alatt generál egy átlagos notebook-on. Ha figyelembe vesszük a redundáns egységek számára adott megkötéseket, akkor a generált hálózatok száma lecsökken. Ilyen feltételek mellett az SSG algoritmus 150 hálózatot generál, amelyek közül 96 teljesíti az anyagmérleghez és műveleti egység kapacitáshoz tartozó korlátozásokat. Ennek a 96 hálózatnak a költsége és meghibásodási valószínűsége a 4.6. ábrán látható. Mind a kettő indikátor fontos a megfelelő megoldás kiválasztásakor, ezért a Pareto megoldások kiemelve láthatóak. A 4.7. ábra mutatja az egyik Pareto megoldáshoz tartozó hálózatot. Ebben a hálózatban csak egy redundáns műveleti egység van, egy szivattyú. Szükség esetén mindegyik Pareto megoldás tovább elemezhető szimulációs program segítségével.



4.5. ábra: A 4.2. szemléltető példa maximális hálózata a redundáns opciókkal kiegészítve



4.6. ábra: A 4.2. szemléltető példa lehetséges hálózatai: költség és meghibásodási valószínűség



4.7. ábra: A 4.2. szemléltető példa egyik kiemelt, a 4.6. ábrán megjelölt Pareto megoldása

4.2 Megbízható rendszerek szintézise működési módok alapján

Az előző fejezetben ismertetett algoritmus megfelelően működik, amennyiben a generált megoldásokban a legfontosabb elem a struktúra, és a maximális hálózat szerkezete rendkívül sok működési lehetőséget tartalmaz (pl. szállító hálózat, kommunikációs hálózat). Vannak azonban olyan folyamatok, ahol a több kimeneti anyaggal rendelkező műveleti egységek miatt célszerű lehet több különböző és kisebb méretű műveleti egységet telepíteni, illetve ahol a beruházási vagy üzemeltetési költség nagyon érzékeny az anyagmennyiségre. Az így kapott hálózat csökkentheti a melléktermékek gyártását, ezzel hatékonyabbá téve a termelést. A folyamatban azonban lehetséges, hogy kevesebb, nagyobb kapacitású műveleti egységgel is kivitelezhető a termelés, de kevésbé hatékonyan. Ilyen folyamatok esetén a műveleti egységek kapacitásainak változtatásával különböző megbízhatósági szintek érhetőek el, amit az előző algoritmus nem tud megfelelően kezeli. Erre ismertet egy megfelelő algoritmust a jelen fejezet.

Ilyen feladatok esetében egy struktúra nem tud egyértelműen definiálni egy folyamatot. Matematikai modellek segítségével meghatározható a struktúra költség-optimális paraméterezése, azonban ez más indikátorok szempontjából nem feltétlenül a legkedvezőbb konfigurációt eredményezi. Például elképzelhető, hogy egyes műveleti egységek méreteinek növelésével a folyamat megbízhatóbbá válik, mert a nagyobb kapacitású műveleti egység redundanciát biztosít a folyamatnak.

A következő ismertetett módszer a folyamat lehetséges működési módjaiból indul ki. Az algoritmus főbb lépései a következőek:

- A maximális hálózat meghatározása
- A lehetséges működési módok meghatározása:
    - Strukturálisan lehetséges hálózatok generálása
    - A hálózatok lehetségességének kiértékelése LP modell, NLP modell, vagy szimulációs program segítségével
    - Egy lehetséges hálózat akkor lesz működési mód, ha minden műveleti egysége aktív
- A működési módok hálózatainak kombinálása:
    - Egy vagy több működési mód kiválasztása
    - A kiválasztottak közül a legkisebb működési költségű lesz az alap működési mód
    - A hálózat meghatározása a működési módok kombinációjával
- Költség és megbízhatóság meghatározása minden generált hálózatra
- A Pareto optimális megoldások meghatározása

Az algoritmus első lépése természetesen a maximális hálózat meghatározása (MSG algoritmus). A következő lépés az összes lehetséges működési mód generálása. A működési módok minden esetben minimálisak, azaz nem lehet belőlük műveleti egységet elhagyni, valamint nem lehet a műveleti egységek

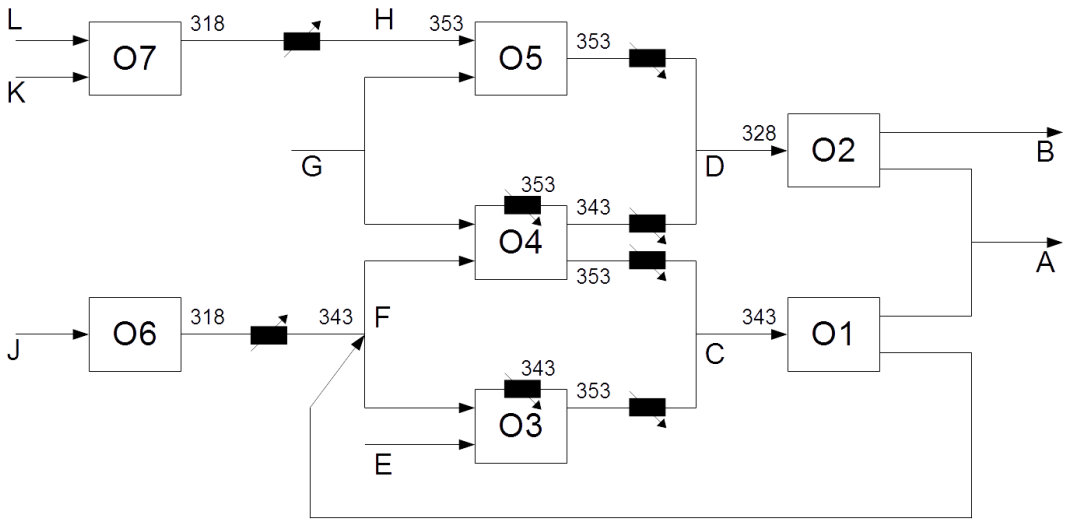
kapacitásait csökkenteni sem. Az SSG algoritmus adja a lépés alapját, amely generálja az összes lehetséges struktúrát. Minden struktúrát ellenőrizni kell a folyamat hálózat matematikai modelljének megfelelően (vagy szimulációs program segítségével), ami megadja a működéshez szükséges paramétereket. Az így kapott működőképes hálózatok közül azok lesznek a működési módok, amelyekben minden műveleti egység aktív, vagyis nem tartalmaznak olyan műveleti egységet, amely a hálózat működéséhez felesleges.

A működési módok kombinációja két lépésből áll. Az első lépés a kombinált struktúra meghatározása. Ehhez a kiválasztott működési módok műveleti egységeinek unióját kell venni. A kombinált hálózatban szerepelni fog minden olyan műveleti egység, amely legalább egy kiválasztott működési módnak része. A második lépés a műveleti egységek kapacitásainak kiválasztása. A kombinációnak egy olyan hálózatnak kell lennie, amely mindegyik kiválasztott működési módra képes. Ehhez arra van szükség, hogy a műveleti egységek kapacitása kellően nagy legyen. A kombinált hálózat minden műveleti egységéhez ki kell gyűjteni azok kapacitásait az egyes működési módokban, és ezen értékek közül a legnagyobbat állítani be a kombinált hálózatban.

A módszer futási ideje itt is exponenciális, jelentős eltéréssel az egyes feladatok között. Egyik tényezője az SSG algoritmus és a működőképességi teszt, ahol a domináns tényező a kombinatorikusan lehetséges hálózatok száma. A működési módok kiválasztása után azok kombinációja minden egyes részhalmazt figyelembe vesz, így a működési módok száma jelentős hatással van a futási időre. Jellemzően a működési módok csak egy kis részhalmazát adják az SSG által generált hálózatoknak, és számuk elsősorban a maximális hálózat komplexitásától függ, nem a méretétől. Minden kombinációra meg kell határozni a megbízhatóságot, amely a kombinált hálózat méretétől függő számítási igénnyel jár.

4.3. szemléltető példa

A módszer szemléltetésére egy példán keresztül kerül sor. Ez a példa egy összetettebb szintézis feladat része, amely a 2. fejezet 2.2. szemléltető példáját egészíti ki a folyamat hőcserélő hálózatának meghatározásával. A szuperstruktúra hagyományos rajzát a 4.8. ábra mutatja. Mivel a hőcserélő hálózat tervezését jelen dolgozat nem tárgyalja, azoknak a lépéseknek csak az eredményét közöljük, a részleteit nem. A számítások során a hőcserélő egységeket műveleti egységként kezeljük, így a megbízhatóságot meghatározó algoritmus szempontjából teljesen megegyeznek a többi műveleti egységgel.



4.8. ábra: A 4.3. szemléltető példa szuperstruktúrája hagyományos ábrázolással

A példához tartozó bemeneti adatokat a 4.4. – 4.8. táblázatok tartalmazzák. A 4.4. táblázatban találhatóak a műveleti egységek adatait, a 4.5. táblázat mutatja a nyersanyagok egységárát, illetve a maximális elérhető folyamot. A 4.6. táblázatban az anyagok fűtési-, illetve hűtési követelményei láthatóak, a 4.7. táblázatban a műveleti egységekhez tartozó rejtett hők, végül a 4.8. táblázatban a külső hőforrások adatai.

4.4. táblázat: A 4.3. szemléltető példa műveleti egységeinek adatai

Műveleti egység	Bemenetek (arány) (1000 t/y)	Kimenetek (arány) (1000 t/y)	Évesített beruházási költség (1000 USD)		Üzemeltetési költség (1000 USD)		Megbízhatóság
			Fix rész	Lineáris együttható	Fix rész	Lineáris együttható	
O1	C(3,0)	A(2,0),F(1,0)	1500	240	500	160	0,989
O2	D(1,5)	A(1,0),B(0,5)	760	140	640	350	0,994
O3	E(0,6),F(1,4)	C(2,0)	1600	200	500	270	0,985
O4	F(1,2),G(0,8)	C(1,0),D(1,0)	1000	120	500	100	0,990
O5	G(2,0),H(1,0)	D(3,0)	780	56	1500	670	0,993
O6	J(1,0)	F(1,0)	1300	130	200	100	0,975
O7	K(1,2),L(0,8)	H(2,0)	720	68	1400	500	0,990

4.5. táblázat: A 4.3. szemléltető példa nyersanyagainak adatai

Nyersanyag	Egységár (USD/t)	Maximális folyam (1000 t/y)
E	110,0	50
G	80,0	50
J	60,0	50
K	210,0	50
L	200,0	50

4.6. táblázat: A 4.3. szemléltető példához tartozó hűtési és fűtési követelmények

Anyagáram	Hőkapacitás (J/(g K))	Honnan	Hova	Kezdő hőmérséklet (K)	Végső hőméséklet (K)
C	0,6	O3	O1	353	343
C	0,6	O4	O1	353	343
D	0,7	O4	O2	343	328
D	0,7	O5	O2	353	328
F	1,4	O6	O3,O4	318	343
H	1,1	O7	O5	318	353

4.7. táblázat: A 4.3. szemléltető példához tartozó rejtett hők

Műveleti egység	Hőmérséklet (K)	Hőforrás együtthatója (GJ/1000 t)
O3	343	10,0
O4	353	5,5

4.8. táblázat: A 4.3. szemléltető példához tartozó külső források

Külső forrás	Típus	Hőmérséklet (K)	Egység ár (USD/MJ)
HU	Hot	373	6,0
CU	Cold	293	9,0

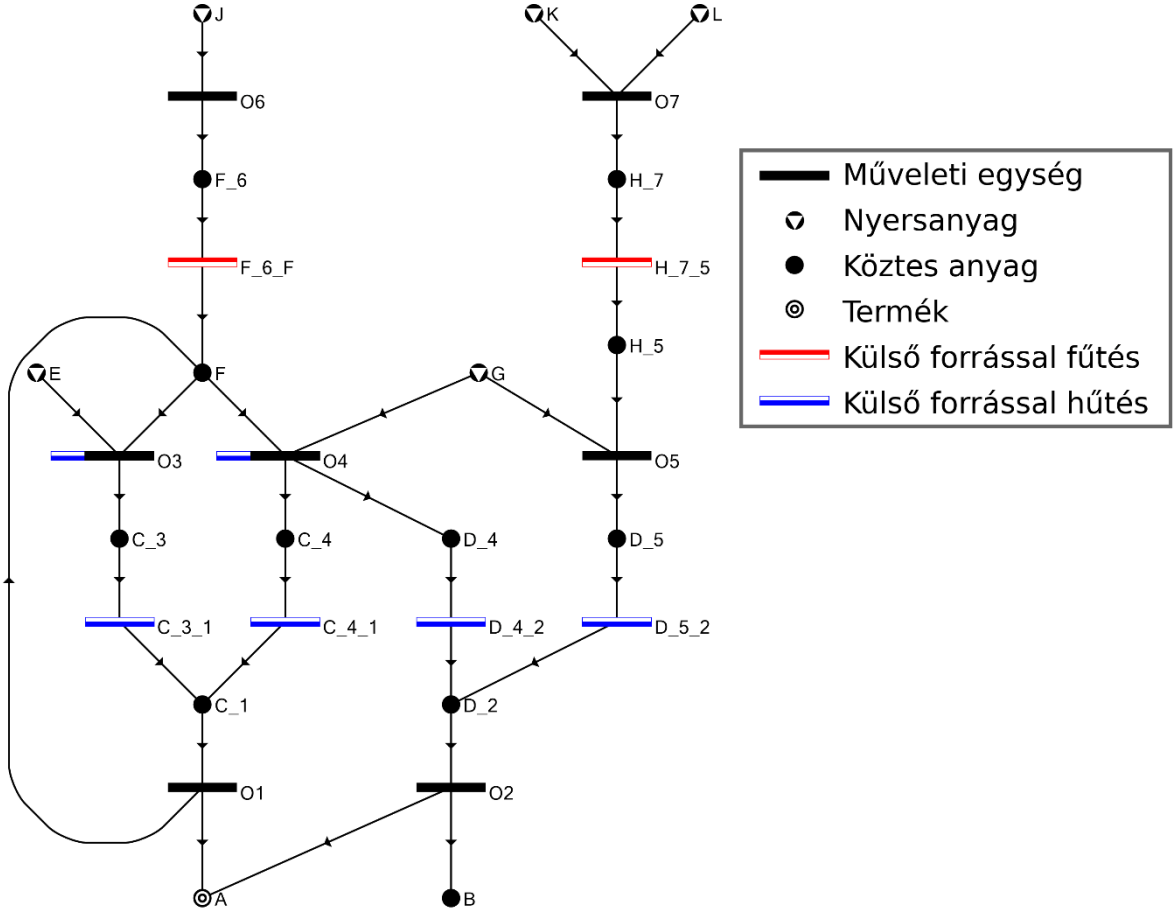
A folyamat elvárt teljesítménye, hogy az *A* jelzésű termékből 5 000 t/y gyártást teljesítsen 0,99-nél nem rosszabb megbízhatósággal. A hőcserélő egységek ugyanolyan műveleti egységként szerepelnek a feladatban, mint a többi egység, azzal a különbséggel, hogy a beruházási-, illetve üzemeltetési költségük kiszámítása plusz 1 lépést igényel. A hőcserélők költségét a hőcserélő felület alapján határozzuk meg, jelen feladatban lineáris függvényekkel. A P-gráf modellben azonban a hőcserélők mérete az általuk elcserélt hőmennyiséggel egyezik meg. A hőmennyiségből vissza kell számolni a hőcserélő felületet, ami a (4.1) képlet szerint tehető meg, ahol Q az elcserélt hőmennyiség, $LMTD$ a hőcserélő működése során a logaritmikus középhőmérséklet, U a hőcserélési együttható, A pedig a terület. Mivel jelen modellben $LMTD$ és U minden hőcserélőhöz egy konstans értékek, ez az átalakítás lineáris. Így, bár a hőcserélő egységek költsége egységes a hőcserélő felület függvényében, minden, a maximális hálózatán szereplő hőcserélő egység a saját $LMTD$ és U értékei alapján saját, átalakított költségfüggvénnyel rendelkezik. A hőcserélők általános adatait a 4.9. táblázat tartalmazza. Jelen esetben a hőcserélők meghibásodásával nem foglalkozik a feladat (a megbízhatóságuk 1), azonban ezek az értékek is részt vesznek a megbízhatóság kiszámításában, így nem kellene az algoritmust módosítani ahhoz, hogy figyelembe vegyük.

$$A = \frac{Q}{LMTD * U} \tag{4.1}$$

A 4.9. ábra mutatja a példához tartozó maximális hálózatot, ahol a hőcserélő egységek még nem szerepelnek, csak az egyes hűtési- és fűtési igényeket jelölik virtuális műveleti egységek. Ebből készül el a tényleges maximális hálózat, amin szerepel az összes potenciális hőcserélő egység. Ezek száma túl nagy ahhoz, hogy grafikusan ábrázolni lehessen.

4.9. táblázat: A 4.3. szemléltető példában a hőcserélő egységek általános adatai

Adat	Érték
Fix beruházási költség (1000 USD)	184,49
Proporcionális beruházási költség (1000 USD)	103,10
Fix üzemeltetési költség (1000 USD)	86,21
Proporcionális üzemeltetési költség (1000 USD)	63,75
Hőcserélési együttható (W/(m² K))	39,9

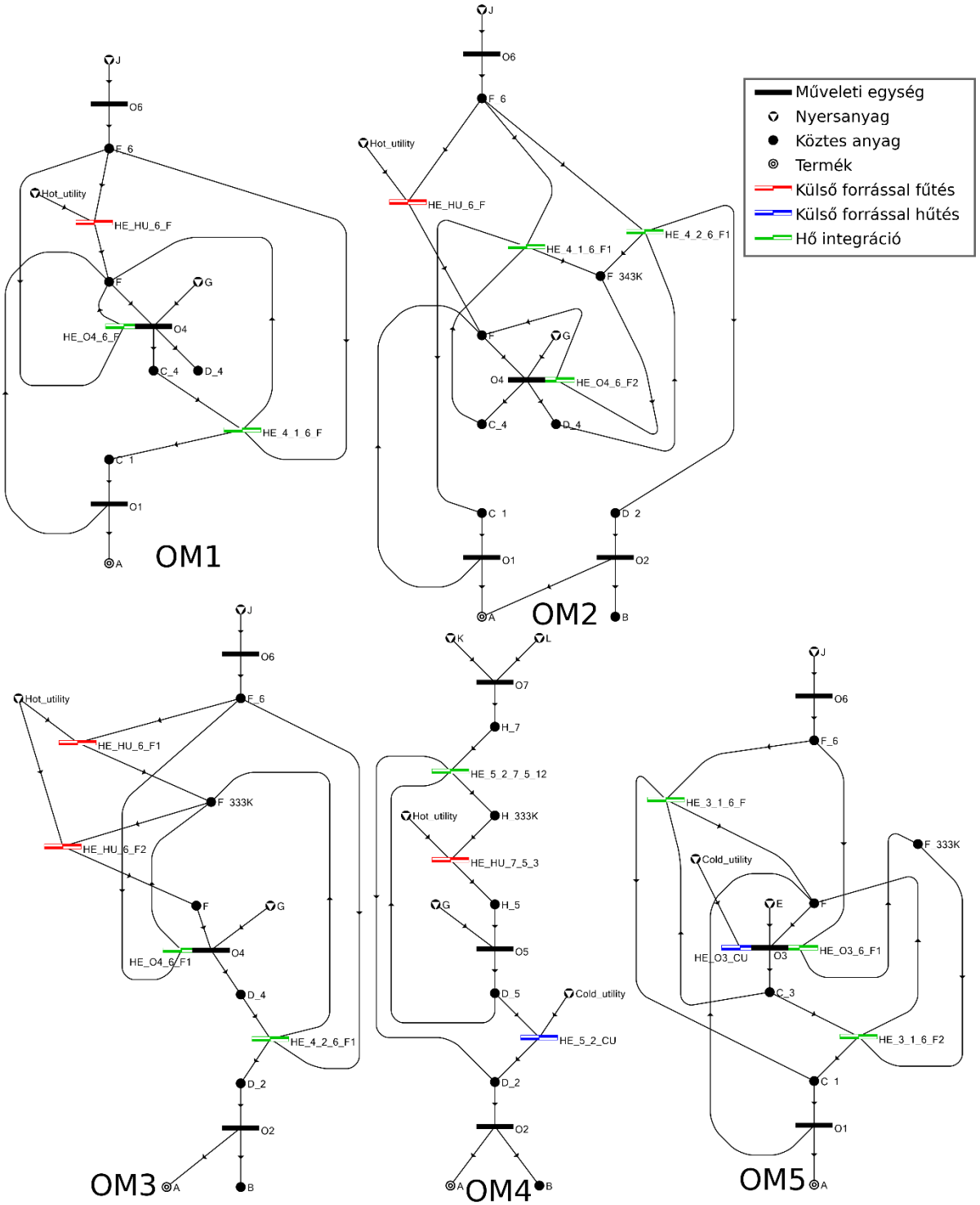


4.9. ábra: A 4.3. szemléltető példa maximális hálózata

A második lépés az algoritmusban a működési módok meghatározása. A példa folyamatához öt működési mód tartozik. Ezek egyben láthatóak a 4.10. ábrán. A színes műveleti egységek a hőcserélő egységeket jelölik. Minden működési módhoz csak az optimális hőcserélő hálózatot határoztuk meg, mivel a jelenlegi munka a megbízhatóságra koncentrál, és nem a hő cserélési lehetőségekre.

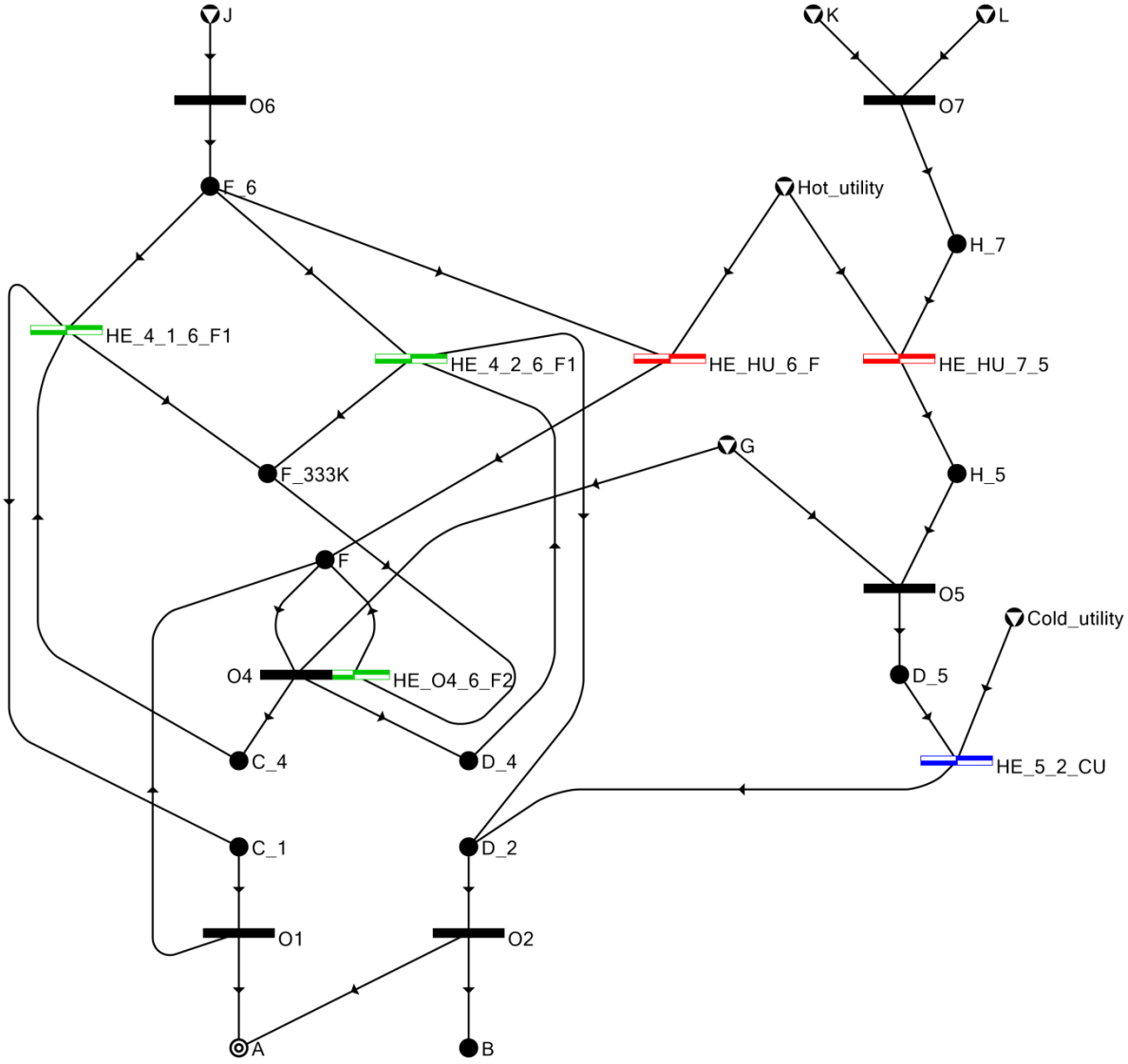
A következő lépés a működési módok kombinálása, és ez által a redundanciát tartalmazó folyamatok generálása. Ezt az összes lehetséges módon meg kell tenni, ami jelen esetben 31 lehetőséget ad, beleértve azt is, amikor nincs redundancia. A 2-es és 4-es (OM2 és OM4) működési módokon keresztül illusztráljuk a kombinálás menetét. A működési módok adatait a 4.10. és 4.11. táblázatok tartalmazzák. A két működési módnak csak 1 közös műveleti egysége van, O2. Ez az OM4 működési módban szerepel nagyobb kapacitással, ezért ezt a méretet kapja a kombinált folyamatban. Az összes többi műveleti (a hőcserélő egységeket

egyelőre figyelmen kívül hagyva) csak az egyik működési módban szerepel, ezért egyszerűen az ott meghatározott kapacitással kerülnek be a kombinált folyamatba. A két működési mód közül az OM2 rendelkezik a kisebb működési költséggel, ezért ez lesz a kombinált folyamatban az alap működési mód, míg az OM4 lesz a tartalék működési mód.



4.10. ábra: A 4.3. szemléltető példa 5 működési módja a hozzájuk tartozó hőcserélő hálózattal együtt

A kombinált hálózat hőcserélő egységeinek meghatározása egy kicsit eltér hagyományos műveleti egységekétől, ami a kombinálást illeti. Egy redundanciával rendelkező folyamatban a tartalék működési módra jellemzően csak az idő kis hányadában van szükség. Erre a kis időre viszont a legfontosabb tényező a működés fenntartása, a működési költség másodlagos. Ennek következtében a 4.3. szemléltető példa esetében egy olyan szabályt határoztunk meg, ami a robosztusságra törekszik a tartalék működési mód esetén. Ennek a szabálynak az értelmében a tartalék működési mód nem alkalmaz hő cserélést az áramok között, hanem minden hűtési-, illetve fűtési követelményt külső források használatával teljesít. Ez természetesen megnöveli a működési költséget, cserébe olcsóbbá és egyszerűbbé teszi a folyamat felépítését, valamint robosztusabb működést biztosít a tartalék működési mód számára. A teljes vizsgálat érdekében végig lehetne nézni az összes lehetséges hőcserélő hálózatot az összes működési módhoz, és megvizsgálni őket költség és megbízhatóság szempontjából, de jelen példa célja a működési módok kombinációjának ismertetése. Az OM2 és OM4 működési módok kombinációjából adódó folyamat végül a 4.11. ábrán látható.



4.11. ábra: Az OM2 és OM4 működési módok kombinációjából kapott folyamat

4.10. táblázat: Az OM2 és OM4 működési módok műveleti egységei

OM2 működési mód			OM4 működési mód		
Műveleti egység	Bemenetek (arány) (1000 t/y)	Kimenetek (arány) (1000 t/y)	Műveleti egység	Bemenetek (arány) (1000 t/y)	Kimenetek (arány) (1000 t/y)
O1	C(3,75)	A(2,5),F(1,25)	O2	D(7,5)	A(5),B(2,5)
O2	D(3,75)	A(2,5),B(1,25)	O5	G(5),H(2,5)	D(7,5)
O4	F(4,5),G(3,0)	C(3,75),D(3,75)	O7	K(1,5),L(1,0)	H(2,5)
O6	J(3,25)	F(3,25)			

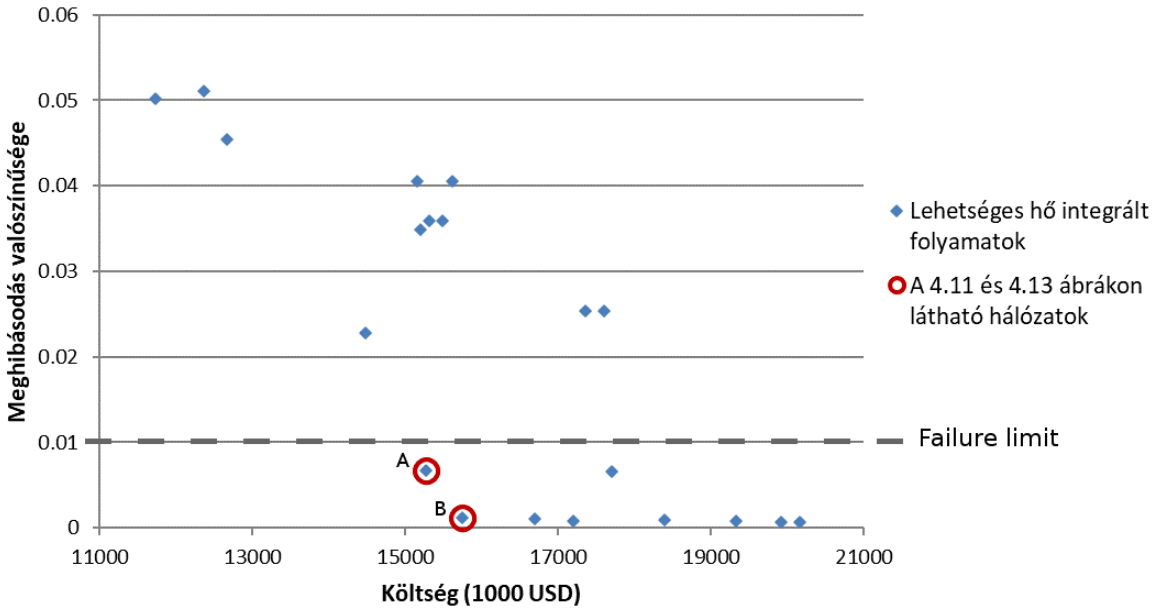
4.11. táblázat: Az OM2 és OM4 működési módok hőcserélő egységei

OM2 működési mód		OM4 működési mód	
Hőcserélő egység	Felület (m^2)	Hőcserélő egység	Felület (m^2)
HE_4_1_6_F1	0,80	HE_5_2_7_5_12	5,50
HE_4_2_6_F1	3,15	HE_5_2_CU	1,08
HE_O4_6_F2	2,29	HE_HU_7_5_3	0,89
HE_HU_6_F	0,21		

A lehetséges megoldások meghatározásának utolsó lépése a költség és megbízhatóság kiszámítása minden generált hálózatra. A 4.11. ábrán látható hálózat esetében ezek az értékek:

- Teljes évesített költség: 15 261,57 (1 000 USD)
- Megbízhatóság: 0,993237

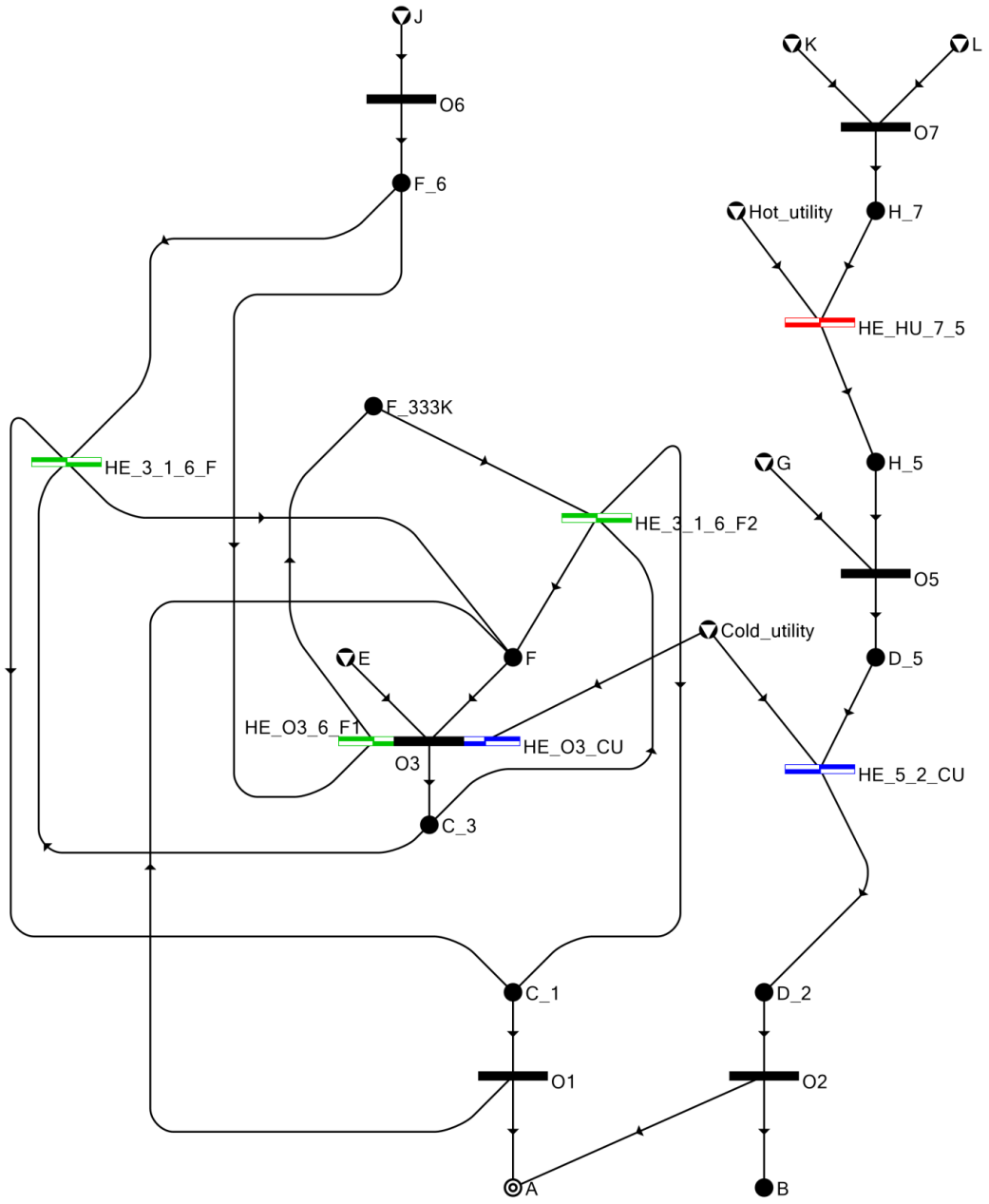
Az algoritmus végezetül a generált hálózatokat összehasonlítja költség és megbízhatóság szempontjából, amiket egy diagramon ábrázolva teljes képet kaphatunk a lehetséges folyamatokról. A hálózatok összehasonlítását a 4.12. ábra tartalmazza.



4.12. ábra: A 4.3. szemléltető példa lehetséges megoldása a költség és megbízhatóság ábrázolásával

Az ábrán az A-val jelöl folyamat a legolcsóbb olyan megoldás, amelyik teljesíti a megbízhatósági követelményt. Ez a folyamat pontosan a 4.11. ábrán bemutatott hálózat, ami az OM2 és OM4 működési módok kombinációiból jön össze. A 4.12. ábrán még egy megjelölt hálózat található, B jelzéssel. A diagramon lévő pontokat elemezve ez a B hálózat tűnik a tervező számára legkedvezőbbnek, hiszen az A-hoz képest a költség csak kicsivel nagyobb, cserébe a megbízhatóság jelentősen jobb. Ezt a hálózatot az OM4 és OM5 működési módok kombinációja adja, a hálózat maga a 4.13. ábrán látható. A B-vel jelölt hálózat paraméterei:

- Teljes évesített költség: 15 746,6 (1 000 USD)
- Megbízhatóság: 0,998854



4.13. ábra: A 4.3. szemléltető példa másik kiválasztott hálózata, az OM4 és OM5 működési módok kombinációja

4.3. A fejezethez kapcsolódó tézis

3. Tézis: Az a) és b) pontokban meghatározott módszereket dolgoztam ki adott megbízhatósági szintet teljesítő optimális folyamat hálózatok szintézisére.

a) A P-gráf keretrendszer algoritmusain alapuló módszer a generált lehetséges struktúrákat egyidejűleg értékeli gazdasági és megbízhatósági szempontból.
b) A lehetséges működési módokat generáló módszer a működési módok kombinációit értékeli gazdasági és megbízhatósági szempontból.
c) Az a) és b) pontban leírt módszerek meghatározzák a *Pareto* megoldásokat.

4.4. A fejezethez kapcsolódó publikációk

A. Orosz, Z. Kovacs, and F. Friedler, “Synthesis of processing systems taking into account reliability,” *Chemical Engineering Transactions*, vol. 70, pp. 1111–1116, 2018, doi: 10.3303/CET1870186.

Á. Orosz, F. Friedler, P. S. Varbanov, and J. J. Klemes, “Systems reliability, footprints and sustainability,” *Chemical Engineering Transactions*, vol. 63, pp. 121–126, 2018, doi: 10.3303/CET1863021.

A. Orosz, Z. Kovacs, and F. Friedler, “Processing Systems Synthesis with Embedded Reliability Consideration,” *Computer Aided Chemical Engineering*, vol. 43, pp. 869–874, 2018, doi: 10.1016/B978-0-444-64235-6.50152-2.

Á. Orosz, Z. Kovács, and F. Friedler, “Synthesis of heat integrated processing systems taking into account reliability,” *Energy*, vol. 181, pp. 214–225, 2019, doi: 10.1016/j.energy.2019.05.173.

5. fejezet

Összefoglalás

A disszertáció három területen mutat be új tudományos eredményeket. Az első témakör egy általános, P-gráf-alapú megbízhatóság számító algoritmus bevezetése, amellyel tetszőleges folyamat hálózat megbízhatósága meghatározható. A leszámláláson alapuló algoritmus bármilyen, P-gráffal ábrázolt folyamat hálózatra alkalmazható. A dolgozat ismertette a megbízhatósági formulát, példákon szemléltette a működését, majd bemutatott néhány módszer a számítási hatékonyság növelésére.

A második témakör tartalma a megbízhatóság növelése érdekében a folyamatba integrálható kétféle redundancia: lokális és globális. Példán keresztül szemléltetésre került, hogy a legkedvezőbb rendszer megtervezése érdekében általában mindkettő redundancia típust figyelembe kell venni.

A harmadik témakör kettő módszert mutat be a megbízhatóságot is figyelembe vevő szintézisre. Az egyik módszer a P-gráf keretrendszer SSG algoritmusát alkalmazza az összes lehetséges hálózat generálására, és mindegyiket kiértékeli költség és megbízhatóság szempontjából. A másik először meghatározza az összes lehetséges működési módot a feladat maximális struktúráján, majd ezen alap hálózatokat kombinálva készíti el a lehetséges folyamatokat.

A dolgozathoz kapcsolódó tézisek

1. **Tézis:** Általános formulát dolgoztam ki folyamat hálózatok megbízhatóságának meghatározására, feltéve, hogy a folyamatot alkotó műveleti egységek megbízhatósága adott.
   a) A formula a működőképes hálózatok P-gráf algoritmusokon alapuló kombinatorikus leszámlálására épül.
   b) A formulával meghatározható a strukturális megbízhatóság, ha a leszámlálás a kombinatorikusan lehetséges struktúrák alapján történik, valamint az általános megbízhatóság, ha a leszámlálás a lehetséges struktúrák alapján történik
   c) Gyorsító eljárásokat javasoltam a formula meghatározásához szükséges számítási igény csökkentésére.
   d) Bemutattam a formula alkalmazását több esettanulmánnyal.

1. **Tézis:** Globális strukturális redundanciát megengedő folyamat hálózatok esetére igazoltam, hogy a gyakorlatban általában alkalmazott műveleti egység szintű (lokális) redundancia gyakran kizárja a globálisan optimális hálózatot.

2. **Tézis:** Az a) és b) pontokban meghatározott módszereket dolgoztam ki adott megbízhatósági szintet teljesítő optimális folyamat hálózatok szintézisére.
   a) A P-gráf keretrendszer algoritmusain alapuló módszer a generált lehetséges struktúrákat egyidejűleg értékeli gazdasági és megbízhatósági szempontból.
   b) A lehetséges működési módokat generáló módszer a működési módok kombinációit értékeli gazdasági és megbízhatósági szempontból.
   c) Az a) és b) pontban leírt módszerek meghatározzák a *Pareto* megoldásokat.

A dolgozathoz kapcsolódó publikációk

A. Orosz, Z. Kovacs, and F. Friedler, “Synthesis of processing systems taking into account reliability,” *Chemical Engineering Transactions*, vol. 70, pp. 1111–1116, 2018, doi: 10.3303/CET1870186.

Á. Orosz, F. Friedler, P. S. Varbanov, and J. J. Klemes, “Systems reliability, footprints and sustainability,” *Chemical Engineering Transactions*, vol. 63, pp. 121–126, 2018, doi: 10.3303/CET1863021.

A. Orosz, Z. Kovacs, and F. Friedler, “Processing Systems Synthesis with Embedded Reliability Consideration,” *Computer Aided Chemical Engineering*, vol. 43, pp. 869–874, 2018, doi: 10.1016/B978-0-444-64235-6.50152-2.

Á. Orosz, Z. Kovács, and F. Friedler, “Synthesis of heat integrated processing systems taking into account reliability,” *Energy*, vol. 181, pp. 214–225, 2019, doi: 10.1016/j.energy.2019.05.173.

Á. Orosz, J. Pimentel, and F. Friedler, “Simultaneous synthesis of processes with its heat exchanger networks: P-graph Approach,” *Chemical Engineering Transactions*, vol. 76, pp. 1219–1224, 2019, doi: 10.3303/CET1976204.

Z. Kovacs, A. Orosz, and F. Friedler, “Synthesis algorithms for the reliability analysis of processing systems,” *Central European Journal of Operations Research*, vol. 27, no. 2, pp. 573–595, 2019, doi: 10.1007/s10100-018-0577-0.

Á. Orosz and F. Friedler, “Synthesis technology for failure analysis and corrective actions in process systems engineering,” *Computer Aided Chemical Engineering*, vol. 46, pp. 1405–1410, 2019, doi: 10.1016/B978-0-12-818634-3.50235-6.

Irodalomjegyzék

[1] S. A. Papoulias and I. E. Grossmann, “A structural optimization approach in process synthesis—I,” *Comput Chem Eng*, vol. 7, no. 6, pp. 695–706, Jan. 1983, doi: 10.1016/0098-1354(83)85022-4.

[2] G. R. Kocis and I. E. Grossmann, “Global optimization of nonconvex mixed-integer nonlinear programming (MINLP) problems in process synthesis,” *Ind Eng Chem Res*, vol. 27, no. 8, pp. 1407–1421, Aug. 1988, doi: 10.1021/ie00080a013.

[3] F. Friedler, K. Tarjan, Y. W. Huang, and L. T. Fan, “Combinatorial algorithms for process synthesis,” *Comput Chem Eng*, vol. 16, pp. S313–S320, May 1992, doi: 10.1016/S0098-1354(09)80037-9.

[4] H. Kumamoto, K. Tanaka, and K. Inoue, “Efficient Evaluation of System Reliability by Monte Carlo Method,” *IEEE Trans Reliab*, vol. R-26, no. 5, pp. 311–315, Dec. 1977, doi: 10.1109/TR.1977.5220181.

[5] S. J. Prowell and J. H. Poore, “Computing system reliability using Markov chain usage models,” *Journal of Systems and Software*, vol. 73, no. 2, pp. 219–225, Oct. 2004, doi: 10.1016/S0164-1212(03)00241-3.

[6] B. Bertok, K. Kalauz, Z. Sule, and F. Friedler, “Combinatorial Algorithm for Synthesizing Redundant Structures to Increase Reliability of Supply Chains: Application to Biodiesel Supply,” *Ind Eng Chem Res*, p. 121029141124003, Oct. 2012, doi: 10.1021/ie301393d.

[7] A. Birolini, *Reliability Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017. doi: 10.1007/978-3-662-54209-5.

[8] J. J. Siirola, “The computer-aided synthesis of chemical process designs,” The University of Wisconsin, Madison, 1970.

[9] J. J. Siirola and D. F. Rudd, “Computer-Aided Synthesis of Chemical Process Designs. From Reaction Path Data to the Process Task Network,” *Industrial & Engineering Chemistry Fundamentals*, vol. 10, no. 3, pp. 353–362, Aug. 1971, doi: 10.1021/i160039a003.

[10] J. M. Douglas, “A hierarchical decision procedure for process synthesis,” *AIChE Journal*, vol. 31, no. 3, pp. 353–362, Mar. 1985, doi: 10.1002/aic.690310302.

[11] I. E. Grossmann and J. Santibanez, “Applications of mixed-integer linear programming in process synthesis,” *Comput Chem Eng*, vol. 4, no. 4, pp. 205–214, Jan. 1980, doi: 10.1016/0098-1354(80)85001-0.

[12] P. Floquet, L. Pibouleau, and S. Domenech, “Mathematical programming tools for chemical engineering process design synthesis,” *Chemical Engineering and Processing: Process Intensification*, vol. 23, no. 2, pp. 99–113, Mar. 1988, doi: 10.1016/0255-2701(88)80004-7.

[13] L. K. E. Achenie and L. T. Biegler, “A superstructure based approach to chemical reactor network synthesis,” *Comput Chem Eng*, vol. 14, no. 1, pp. 23–40, Jan. 1990, doi: 10.1016/0098-1354(90)87003-8.

[14] H. Yeomans and I. E. Grossmann, “A systematic modeling framework of superstructure optimization in process synthesis,” *Comput Chem Eng*, vol. 23, no. 6, pp. 709–731, Jun. 1999, doi: 10.1016/S0098-1354(99)00003-4.

[15] F. Friedler, K. Tarján, Y. W. Huang, and L. T. Fan, “Graph-theoretic approach to process synthesis: axioms and theorems,” *Chem Eng Sci*, vol. 47, no. 8, pp. 1973–1988, Jun. 1992, doi: 10.1016/0009-2509(92)80315-4.

[16] Z. Kovács, Z. Ercsey, F. Friedler, and L. T. Fan, “Separation-network synthesis: global optimum through rigorous super-structure,” *Comput Chem Eng*, vol. 24, no. 8, pp. 1881–1900, Sep. 2000, doi: 10.1016/S0098-1354(00)00568-8.

[17] R. Al, C. R. Behera, K. v. Gernaey, and G. Sin, “Stochastic simulation-based superstructure optimization framework for process synthesis and design under uncertainty,” *Comput Chem Eng*, vol. 143, p. 107118, Dec. 2020, doi: 10.1016/j.compchemeng.2020.107118.
[18] J. Ryu, L. Kong, A. E. Pastore de Lima, and C. T. Maravelias, “A generalized superstructure-based framework for process synthesis,” *Comput Chem Eng*, vol. 133, p. 106653, Feb. 2020, doi: 10.1016/j.compchemeng.2019.106653.
[19] C. A. Méndez, J. Cerdá, I. E. Grossmann, I. Harjunkoski, and M. Fahl, “State-of-the-art review of optimization methods for short-term scheduling of batch processes,” *Comput Chem Eng*, vol. 30, no. 6–7, pp. 913–946, May 2006, doi: 10.1016/j.compchemeng.2006.02.008.
[20] K. P. Papalexandri and E. N. Pistikopoulos, “Synthesis and Retrofit Design of Operable Heat Exchanger Networks. 1. Flexibility and Structural Controllability Aspects,” *Ind Eng Chem Res*, vol. 33, no. 7, pp. 1718–1737, Jul. 1994, doi: 10.1021/ie00031a012.
[21] J. M. Douglas, M. F. Malone, and M. F. Doherty, “The interaction between separation system synthesis and process synthesis,” *Comput Chem Eng*, vol. 9, no. 5, pp. 447–462, 1985, doi: 10.1016/0098-1354(85)80022-3.
[22] D. C. Y. Foo, “State-of-the-Art Review of Pinch Analysis Techniques for Water Network Synthesis,” *Ind Eng Chem Res*, vol. 48, no. 11, pp. 5125–5159, Jun. 2009, doi: 10.1021/ie801264c.
[23] H. L. Lam, “Extended P-graph applications in supply chain and Process Network Synthesis,” *Curr Opin Chem Eng*, vol. 2, no. 4, pp. 475–486, Nov. 2013, doi: 10.1016/j.coche.2013.10.002.
[24] Q. Chen and I. E. Grossmann, “Recent Developments and Challenges in Optimization-Based Process Synthesis,” *Annu Rev Chem Biomol Eng*, vol. 8, no. 1, pp. 249–283, Jun. 2017, doi: 10.1146/annurev-chembioeng-080615-033546.
[25] M. Skiborowski, “Process synthesis and design methods for process intensification,” *Curr Opin Chem Eng*, vol. 22, pp. 216–225, Dec. 2018, doi: 10.1016/j.coche.2018.11.004.
[26] F. Friedler, K. B. Aviso, B. Bertok, D. C. Foo, and R. R. Tan, “Prospects and challenges for chemical process synthesis with P-graph,” *Curr Opin Chem Eng*, vol. 26, pp. 58–64, Dec. 2019, doi: 10.1016/j.coche.2019.08.007.
[27] Y. Tian and E. N. Pistikopoulos, “Synthesis of operable process intensification systems: advances and challenges,” *Curr Opin Chem Eng*, vol. 25, pp. 101–107, Sep. 2019, doi: 10.1016/j.coche.2018.12.003.
[28] L. Mencarelli, Q. Chen, A. Pagot, and I. E. Grossmann, “A review on superstructure optimization approaches in process system engineering,” *Comput Chem Eng*, vol. 136, p. 106808, May 2020, doi: 10.1016/j.compchemeng.2020.106808.
[29] F. Friedler, J. B. Varga, and L. T. Fan, “Decision-mapping: A tool for consistent and complete decisions in process synthesis,” *Chem Eng Sci*, vol. 50, no. 11, pp. 1755–1768, Jun. 1995, doi: 10.1016/0009-2509(95)00034-3.
[30] F. Friedler, K. Tarjan, Y. W. Huang, and L. T. Fan, “Graph-theoretic approach to process synthesis: Polynomial algorithm for maximal structure generation,” *Comput Chem Eng*, vol. 17, no. 9, pp. 929–942, Sep. 1993, doi: 10.1016/0098-1354(93)80074-W.
[31] F. Friedler, J. B. Varga, E. Fehér, and L. T. Fan, “Combinatorially Accelerated Branch-and-Bound Method for Solving the MIP Model of Process Network Synthesis,” 1996, pp. 609–626. doi: 10.1007/978-1-4613-3437-8_35.
[32] F. Friedler, P. Varbanov, and J. Klemes, “Advanced HENs Design for Multi-Period Operation Using P-graph,” *Chem Eng Trans*, vol. 18, pp. 457–462, May 2009, doi: 10.3303/CET0918074.
[33] I. Heckl, L. Halász, A. Szlama, H. Cabezas, and F. Friedler, “Process synthesis involving multi-period operations by the P-graph framework,” *Comput Chem Eng*,

vol. 83, pp. 157–164, Dec. 2015, doi: 10.1016/j.compchemeng.2015.04.037.
[34] K. B. Aviso, J.-Y. Lee, and R. R. Tan, “A P-Graph Model for Multi-period Optimization of Isolated Energy Systems,” *Chem Eng Trans*, vol. 52, pp. 865–870, Aug. 2016, doi: 10.3303/CET1652145.
[35] B. Bertok and A. Bartos, “Algorithmic Process Synthesis and Optimisation for Multiple Time Periods Including Waste Treatment: Latest Developments in P-graph Studio Software,” *Chem Eng Trans*, vol. 70, pp. 97–102, Aug. 2018, doi: 10.3303/CET1870017.
[36] M. Frits and B. Bertok, “Scheduling custom printed napkin manufacturing by P-graphs,” *Comput Chem Eng*, vol. 141, p. 107017, Oct. 2020, doi: 10.1016/j.compchemeng.2020.107017.
[37] M. Frits and B. Bertok, “Routing and scheduling field service operation by P-graph,” *Comput Oper Res*, vol. 136, p. 105472, Dec. 2021, doi: 10.1016/j.cor.2021.105472.
[38] R. Adonyi, I. Heckl, and F. Olti, “Scheduling of bus maintenance by the P-graph methodology,” *Optimization and Engineering*, vol. 14, no. 4, pp. 565–574, Nov. 2013, doi: 10.1007/s11081-013-9240-8.
[39] E. Konig and B. Bertok, “Process graph approach for two-stage decision making: Transportation contracts,” *Comput Chem Eng*, vol. 121, pp. 1–11, Feb. 2019, doi: 10.1016/j.compchemeng.2018.07.011.
[40] Z. Ercsey, A. Nagy, J. Tick, and Z. Kovács, “Bus Transport Process Networks with Arbitrary Launching Times,” *Acta Polytechnica Hungarica*, vol. 18, no. 4, pp. 125–141, 2021, doi: 10.12700/APH.18.4.2021.4.7.
[41] L. Vance, I. Heckl, B. Bertok, H. Cabezas, and F. Friedler, “Designing sustainable energy supply chains by the P-graph method for minimal cost, environmental burden, energy resources input,” *J Clean Prod*, vol. 94, pp. 144–154, May 2015, doi: 10.1016/j.jclepro.2015.02.011.
[42] C. H. Lim *et al.*, “Synthesis of Resource Conservation Networks with P-Graph Approach—Direct Reuse/Recycle,” *Process Integration and Optimization for Sustainability*, vol. 1, no. 1, pp. 69–86, May 2017, doi: 10.1007/s41660-017-0005-2.
[43] B. S. How and H. L. Lam, “PCA Method for Debottlenecking of Sustainability Performance in Integrated Biomass Supply Chain,” *Process Integration and Optimization for Sustainability*, vol. 3, no. 1, pp. 43–64, Mar. 2019, doi: 10.1007/s41660-018-0036-3.
[44] Y. van Fan, J. J. Klemeš, T. G. Walmsley, and B. Bertók, “Implementing Circular Economy in municipal solid waste treatment system using P-graph,” *Science of The Total Environment*, vol. 701, p. 134652, Jan. 2020, doi: 10.1016/j.scitotenv.2019.134652.
[45] A. B. Nagy, R. Adonyi, L. Halasz, F. Friedler, and L. T. Fan, “Integrated synthesis of process and heat exchanger networks: algorithmic approach,” *Appl Therm Eng*, vol. 21, no. 13–14, pp. 1407–1427, Oct. 2001, doi: 10.1016/S1359-4311(01)00033-3.
[46] Á. Orosz, B. S. How, and F. Friedler, “Multiple-solution heat exchanger network synthesis using P-HENS solver,” *J Taiwan Inst Chem Eng*, vol. 130, 2022, doi: 10.1016/j.jtice.2021.05.006.
[47] B. H. Y. Ong, M. J. Atkins, T. G. Walmsley, and M. R. W. Walmsley, “Total Site Heat and Mass Integration and Optimisation using P-graph: A Biorefinery Case Study,” *Chem Eng Trans*, vol. 61, pp. 727–732, Oct. 2017, doi: 10.3303/CET1761119.
[48] H. L. Lam, P. S. Varbanov, and J. J. Klemeš, “Optimisation of regional energy supply chains utilising renewables: P-graph approach,” *Comput Chem Eng*, vol. 34, no. 5, pp. 782–792, May 2010, doi: 10.1016/j.compchemeng.2009.11.020.
[49] R. Lakner, P. S. Varbanov, and F. Friedler, “Systems Analysis of Electricity Transmission Networks for Improved Sustainability,” *Chem Eng Trans*, vol. 76, pp. 619–624, Oct. 2019, doi: 10.3303/CET1976104.
[50] Éles, Halász, Heckl, and Cabezas, “Evaluation of the Energy Supply Options of a Manufacturing Plant by the Application of the P-Graph Framework,” *Energies*

(Basel), vol. 12, no. 8, p. 1484, Apr. 2019, doi: 10.3390/en12081484.
[51] K. M. Yenkie, J. Pimentel, Á. Orosz, H. Cabezas, and F. Friedler, "The P-graph approach for systematic synthesis of wastewater treatment networks," *AIChE Journal*, vol. 67, no. 7, 2021, doi: 10.1002/aic.17253.
[52] R. Lakner, B. Bertok, and F. Friedler, "Synthesis of Startable Reaction Pathways," *Chem Eng Trans*, vol. 70, pp. 1129–1134, Aug. 2018, doi: 10.3303/CET1870189.
[53] Z. Süle, J. Baumgartner, G. Dörgő, and J. Abonyi, "P-graph-based multi-objective risk analysis and redundancy allocation in safety-critical energy systems," *Energy*, vol. 179, pp. 989–1003, Jul. 2019, doi: 10.1016/j.energy.2019.05.043.
[54] Á. Orosz, J. Pimentel, and F. Friedler, "General formulation for the resilience of processing systems," *Chem Eng Trans*, vol. 81, pp. 859–864, 2020, doi: 10.3303/CET2081144.
[55] R. R. Tan, K. B. Aviso, J. J. Klemes, H. L. Lam, P. S. Varbanov, and F. Friedler, "Towards Generalized Process Networks: Prospective New Research Frontiers for the P-graph Framework," *Chem Eng Trans*, vol. 70, pp. 91–96, Aug. 2018, doi: 10.3303/CET1870016.
[56] F. Friedler, Á. Orosz, and J. Pimentel Losada, *P-graphs for Process Systems Engineering*. Cham: Springer International Publishing, 2022. doi: 10.1007/978-3-030-92216-0.
[57] R. E. Barlow and L. C. Hunter, "Reliability Analysis of a One-Unit System," *Oper Res*, vol. 9, no. 2, pp. 200–208, Apr. 1961, doi: 10.1287/opre.9.2.200.
[58] K. B. Misra and T. S. M. Rao, "Reliability Analysis of Redundant Networks Using Flow Graphs," *IEEE Trans Reliab*, vol. R-19, no. 1, pp. 19–24, Feb. 1970, doi: 10.1109/TR.1970.5216374.
[59] E. J. Henley and S. L. Gandhi, "Process reliability analysis," *AIChE Journal*, vol. 21, no. 4, pp. 677–686, Jul. 1975, doi: 10.1002/aic.690210406.
[60] H. D. Goel, J. Grievink, P. M. Herder, and M. P. C. Weijnen, "Integrating reliability optimization into chemical process synthesis," *Reliab Eng Syst Saf*, vol. 78, no. 3, pp. 247–258, Dec. 2002, doi: 10.1016/S0951-8320(02)00167-9.
[61] A. Naess, B. J. Leira, and O. Batsevych, "System reliability analysis by enhanced Monte Carlo simulation," *Structural Safety*, vol. 31, no. 5, pp. 349–355, Sep. 2009, doi: 10.1016/j.strusafe.2009.02.004.
[62] A. Roy, P. Srivastava, and S. Sinha, "Risk and reliability assessment in chemical process industries using Bayesian methods," *Reviews in Chemical Engineering*, vol. 30, no. 5, Jan. 2014, doi: 10.1515/revce-2013-0043.
[63] U. Abubakar, S. Sriramula, and N. C. Renton, "Reliability of complex chemical engineering processes," *Comput Chem Eng*, vol. 74, pp. 1–14, Mar. 2015, doi: 10.1016/j.compchemeng.2014.12.013.
[64] S. Soh and S. Rai, "An Efficient Cutset Approach for Evaluating Communication-Network Reliability With Heterogeneous Link-Capacities," *IEEE Trans Reliab*, vol. 54, no. 1, pp. 133–144, Mar. 2005, doi: 10.1109/TR.2004.842530.
[65] W.-C. Yeh, W. Zhu, and C.-L. Huang, "Reliability of Social Networks on Activity-on-Node Binary-State with Uncertainty Environments," *Applied Sciences*, vol. 12, no. 19, p. 9514, Sep. 2022, doi: 10.3390/app12199514.
[66] M. J. Zuo, Z. Tian, and H.-Z. Huang, "An efficient method for reliability evaluation of multistate networks given all minimal path vectors," *IIE Transactions*, vol. 39, no. 8, pp. 811–817, May 2007, doi: 10.1080/07408170601013653.
[67] E. Ruijters and M. Stoelinga, "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools," *Comput Sci Rev*, vol. 15–16, pp. 29–62, Feb. 2015, doi: 10.1016/j.cosrev.2015.03.001.
[68] P. Badida, Y. Balasubramaniam, and J. Jayaprakash, "Risk evaluation of oil and natural gas pipelines due to natural hazards using fuzzy fault tree analysis," *J Nat Gas Sci Eng*, vol. 66, pp. 284–292, Jun. 2019, doi: 10.1016/j.jngse.2019.04.010.
[69] H. Mohri and J. Takeshita, "Graph reliability evaluation via random *K* -out-of- *N*

systems," *Commun Stat Theory Methods*, pp. 1–11, Jul. 2022, doi: 10.1080/03610926.2022.2099897.
[70] S. Eryilmaz, "Review of recent advances in reliability of consecutive *k* -out-of- *n* and related systems," *Proc Inst Mech Eng O J Risk Reliab*, vol. 224, no. 3, pp. 225–237, Sep. 2010, doi: 10.1243/1748006XJRR332.
[71] G. Levitin, A. Lisnianski, H. Ben-Haim, and D. Elmakis, "Redundancy optimization for series-parallel multi-state systems," *IEEE Trans Reliab*, vol. 47, no. 2, pp. 165–172, Jun. 1998, doi: 10.1109/24.722283.
[72] Y. Chen, Y. Ye, Z. Yuan, I. E. Grossmann, and B. Chen, "Integrating stochastic programming and reliability in the optimal synthesis of chemical processes," *Comput Chem Eng*, vol. 157, p. 107616, Jan. 2022, doi: 10.1016/j.compchemeng.2021.107616.
[73] R. Nath and P. K. Muhuri, "Evolutionary Optimization based Solution approaches for Many Objective Reliability-Redundancy Allocation Problem," *Reliab Eng Syst Saf*, vol. 220, p. 108190, Apr. 2022, doi: 10.1016/j.ress.2021.108190.
[74] K. K. Aggarwal, "Redundancy Optimization in General Systems," *IEEE Trans Reliab*, vol. R-25, no. 5, pp. 330–332, Dec. 1976, doi: 10.1109/TR.1976.5220030.
[75] D. W. Coit, "Cold-standby redundancy optimization for nonrepairable systems," *IIE Transactions*, vol. 33, no. 6, pp. 471–478, 2001, doi: 10.1023/A:1007689912305.
[76] Z. Sule, J. Baumgartner, and J. Abonyi, "Reliability - Redundancy Allocation in Process Graphs," *Chem Eng Trans*, vol. 70, pp. 991–996, Aug. 2018, doi: 10.3303/CET1870166.
[77] A. Peiravi, M. A. Ardakan, and E. Zio, "A new Markov-based model for reliability optimization problems with mixed redundancy strategy," *Reliab Eng Syst Saf*, vol. 201, p. 106987, Sep. 2020, doi: 10.1016/j.ress.2020.106987.
[78] M. Marseguerra, E. Zio, L. Podofillini, and D. W. Coit, "Optimal Design of Reliable Network Systems in Presence of Uncertainty," *IEEE Trans Reliab*, vol. 54, no. 2, pp. 243–253, Jun. 2005, doi: 10.1109/TR.2005.847279.
[79] B. Suman, "Simulated annealing-based multiobjective algorithms and their application for system reliability," *Engineering Optimization*, vol. 35, no. 4, pp. 391–416, Aug. 2003, doi: 10.1080/03052150310001597765.
[80] E. Zio and V. Zille, "Multi-Objective Optimization of Network Systems by Using ANT Algorithms," *Qual Technol Quant Manag*, vol. 4, no. 2, pp. 211–224, Jan. 2007, doi: 10.1080/16843703.2007.11673146.
[81] M. Rausand and A. Hoyland, *System reliability theory*, 2nd ed. Nashville, TN: John Wiley & Sons, 2003.
[82] S. E. Chang, R. T. Eguchi, and H. A. Seligson, "Estimation of the economic impact ofmultiple lifeline disruption: Memphis light, gas and water division case study. Technical Report NCEER-96-001," Buffalo (NY), 1996.
[83] L. Chang and J. Song, "Matrix-based System Reliability Analysis of Urban Infrastructure Networks: A Case Study of MLGW Natural Gas Network ," in *Proceedings of the 5th China-Japan-US trilateral symposium on lifeline earthquake engineering*, 2007.
[84] Y. Kim and W.-H. Kang, "Network reliability analysis of complex systems using a non-simulation-based method," *Reliab Eng Syst Saf*, vol. 110, pp. 80–88, Feb. 2013, doi: 10.1016/j.ress.2012.09.012.
[85] C. Holló, Z. Blázsik, B. Imreh, and Z. Kovács, "On a merging reduction of the process network synthesis problem," *Acta Cybernetica*, vol. 14, no. 2, Jan. 1999, [Online]. Available: https://cyber.bibl.u-szeged.hu/index.php/actcybern/article/view/3526
[86] R. Drechsler and B. Becker, *Binary Decision Diagrams*. Boston, MA: Springer US, 1998. doi: 10.1007/978-1-4757-2892-7.
[87] E. N. Pistikopoulos, T. V. Thomaidis, A. Melin, and M. G. Ierapetritou, "Flexibility, reliability and maintenance considerations in batch plant design under uncertainty," *Comput Chem Eng*, vol. 20, pp. S1209–S1214, Jan. 1996, doi: 10.1016/0098-1354(96)00209-8.

[88] S. Terrazas-Moreno, I. E. Grossmann, J. M. Wassick, and S. J. Bury, “Optimal design of reliable integrated chemical production sites,” *Comput Chem Eng*, vol. 34, no. 12, pp. 1919–1936, Dec. 2010, doi: 10.1016/j.compchemeng.2010.07.027.
[89] M. F. D. Benjamin, C. D. Cayamanda, B. A. Belmonte, R. R. Tan, and L. F. Razon, “A Risk-based Criticality Analysis in Bioenergy Parks using P-graph Method,” *Chem Eng Trans*, vol. 52, pp. 1243–1248, Aug. 2016, doi: 10.3303/CET1652208.
[90] M. F. D. Benjamin, “P-graph Approach to Criticality Analysis in Bioenergy Parks under Uncertainty,” *Chem Eng Trans*, vol. 61, pp. 619–624, Oct. 2017, doi: 10.3303/CET1761101.
[91] R. R. Tan, K. B. Aviso, K. D. S. Yu, M. A. B. Promentilla, and J. R. Santos, “P-Graph Approach to Allocation of Inoperability in Urban Infrastructure Systems,” *Chem Eng Trans*, vol. 45, pp. 1339–1344, Sep. 2015, doi: 10.3303/CET1545224.