

Kebelei Csaba

Fűrészüzem ütemezés: egy feladatspecifikus modell

Tudományos Diákköri Konferencia

Témavezető:

Dr. Hegyháti Máté

Egyetemi docens



Faipari Mérnöki és Kreatívipari Kar
Soproni Egyetem

2023



SOPRONI EGYETEM
Faipari Mérnöki és Kreatívipari Kar
H-9401 Sopron, Bajcsy-Zs. u. 4. Pf.: 132.
Tel: +36 (99) 518-101 Fax: +36 (99) 518-259

NYILATKOZAT

Alulírott **Kebelei Csaba** (neptun kód: **I0D34L**) jelen nyilatkozat aláírásával kijelentem, hogy a **Fűrészüzem ütemezés: egy feladat-specifikus modell** című (alábbi megfelelő rész aláhúzendó)

TDK dolgozat

MDK dolgozat

(a továbbiakban: dolgozat) **önálló munkám**, a dolgozat készítése során betartottam a szerzői jogról szóló 1999. évi LXXVI. tv. szabályait, különösen a hivatkozások és idézések tekintetében.

Hivatkozások és idézések szabályai:

Az 1999. évi LXXVI. tv. a szerzői jogról 34. § (1) és 36. § (1) első két mondata.

Kijelentem továbbá, hogy a dolgozat készítése során az önálló munka kitétel tekintetében a konzulens, ill. a feladatot kiadó oktatót **nem tévesztettem meg**.

Jelen nyilatkozat aláírásával tudomásul veszem, hogy amennyiben bizonyítható, hogy a dolgozatot **nem magam készítettem**, vagy a dolgozattal kapcsolatban szerzői jogsértés ténye merül fel, a Soproni Egyetem **megtagadja a dolgozat befogadását és ellenem fegyelmi eljárást indíthat**.

A dolgozat befogadásának megtagadása és a fegyelmi eljárás indítása nem érinti a szerzői jogsértés miatti egyéb (polgári jogi, szabálysértési jogi, büntetőjogi) jogkövetkezményeket.

Dátum: Sopron, 2023. 11. 17.

.....

hallgató aláírása

Absztrakt

Az elsődleges faipar kulcsfontosságú állomásai a fűrészüzemek, ahol a farönkökből különböző fűrészárúkat, mint például pallókat, gerendákat állítanak elő. A folyamat a rönkök osztályozásával, tisztításával, fémkereséssel kezdődik, ezt követi a tényleges fűrészelés, szabászat, majd a fűrészárú kezelése, mint a máglyázás, szárítás, gőzölés. A fűrészelés maga egy egylépéses folyamatnak tekinthető, melynek ütemezésével több cikk foglalkozik az irodalomban. Maturana és társai 2010-ben definiáltak egy egygépes feladatosztályt, melyben a cél a fűrészárú tárolásából valamint kései szállításából fakadó költségek minimalizálása. Az azóta eltelt bő évtizedben több cikk terjesztette ki a vizsgált feladatosztályt több gép, választható vágási minták, átállási idők, vagy robosztusság figyelembe vételével.

A jelenkor egyik legnagyobb kihívása a térség fűrészüzemeiben a munkaerőhiány, az irodalomban található modellek azonban általában nem foglalkoznak a fűrészelési folyamat emberi erőforrás igényével. Szintén nincsenek modell szinten megkülönböztetve a keretfűrész, valamint a szalagfűrész technológiák, melyek alkalmazása több ponton eltér egymástól, legyen szó például átállítási időkről, szükséges operátorok számáról, elvégezhető feladatokról. Bár több figyelembe vett szempont kezelhető általánosabb ütemezési modellekkel, a szerzők korábbi, harmadlagos faipari területen végzett vizsgálatai azt mutatták, hogy az általános modellek nem nyújtanak kellően hatékony megoldást. Jelen kutatás célja ezért egy olyan vegyes-egész lineáris programozási modell felírása volt, mely figyelembe veszi a fent említett paramétereket, valamint általánosság helyett törekszik a feladat specialitásait minél jobban kiaknázni.

Az elkészült modell könnyebb tesztelése érdekében készült egy feladatok szerkesztésére, megoldására alkalmas grafikus felhasználói felület. Ezenfelül a megoldás hatékonyságát faiparral foglalkozó szakemberekkel való konzultációk és a szakirodalom alapján meghatározott tesztiesen vizsgáltam mind rövid mind hosszú tervezési időtávra.

Tartalomjegyzék

1. Bevezetés	4
2. Szakirodalmi áttekintés	6
2.1. Ütemezési problémák	6
2.2. Fűrészüzemek	9
2.3. Fűrészüzemek ütemezési problémái	11
3. Probléma definíció	14
3.1. Informális probléma definíció	14
3.2. Formális probléma definíció	16
3.2.1. Halmazok	16
3.2.2. Paraméterek	17
4. Megoldó módszer	18
4.1. Változók	18
4.2. Korlátozások	20
4.3. Célfüggvény	22
5. Tesztelés	23
5.1. Tesztelés grafikus interfészen keresztül	23
5.1.1. A szoftver architektúrája és felépítése	23
5.1.2. Szoftver használata	26
5.2. Tesztelés ipari tapasztaltok alapján	28
5.2.1. A tesztelés környezete	28
5.2.2. Tesztadatok meghatározása	28
5.2.3. Teszteredmények	28
6. Konklúzió és a jövőbeli tervek	31
A. GMPL kód	37

1. fejezet

Bevezetés

A fűrészüzemek feladata, hogy a kivágott farönkökből a keresletnek megfelelő fűrészáru készletet gyártson. Ennek folyamata több lépcsőből áll, emiatt a feldolgozást egy komplex tervezés előzi meg. A termelés ütemterve számos módon kivitelezhető. A munkám során olyan megközelítésekkel foglalkozom, melyekkel meghatározható a fűrészüzemek folyamatainak optimális ütemezése.

Az irodalomban több megoldás is megtalálható, mely ilyen üzemek ütemtervének optimalizálásával foglalkozik. A témában publikált kutatások legtöbbször általánosabb megoldásokat javasol, azonban a világon a fűrészüzemeknek számos variánsával találkozhatunk. Ebből kifolyólag több specifikusabb tényezőt is megkülönböztethetünk egy üzem esetén, melyek figyelembevétele sokszor elengedhetetlen a hatékony megoldás meghatározásához. Jelen munkám során megvizsgáltam egy olyan specifikusabb problémaosztályt, mely jobban modellezi a térség kisebb fűrészüzemeit, mint az általánosabb megközelítések.

Jelenleg a faipar világszinten munkaerőhiánnyal küzd, ennek egy oka, hogy bizonyos pozíciókhoz speciális szaktudás szükséges, illetve az alkalmazottaknak gyakran rendkívül megterhelő fizikai feladatokat kell elvégezniük. Egy fűrészüzem esetén is jelentős problémát jelent, hogy a fűrészelést végző gépek üzemeltetéséhez külön szaktudás szükséges. A nagyobb automatizált fűrészüzemekkel szemben a kisebb üzemekben a farönkök, fűrészárak mozgatását is az alkalmazottak végzik, mely több ember bevonásával is jelentős fizikai terhelést jelent. Ebből kifolyólag az üzemeknek különös figyelmet kell fordítani a humán erőforrás menedzsmentre, mely az ütemezés során szintén fontos szerepet kap.

Az emberi erőforrások ütemezésén túlmenően sok szakirodalmi módszer nem veszi figyelembe a fűrészüzemekben alkalmazható rönkvágó technológiák közti különbségeket. A kutatásom során megvizsgáltam a két legelterjedtebb fűrészelést végző géptípus - a keretfűrész és a szalagfűrész - sajátosságait, és modellezés szinten is figyelembe vettem azokat.

Egy fűrészüzem ütemezése esetén szem előtt kell tartani a kereslet igényeit is. Ezáltal az üzemekben lévő alapgépeken döntést kell hozni arról, hogy mikor, milyen vágási mintát alkalmazzunk, melyek különböző fűrészáru hozamokat eredményeznek. Míg egy drágább

automatizált rönkvágó technológiánál nem jelent számottevő időt, de egy egyszerűbb keretfűrész technológia esetén elengedhetetlen a minták közti átállásokkal járó idővel való számolás.

A dolgozatomban egy olyan problémaosztályra javasoltam egzakt megoldást, mely a fentebb említett specifikusabb tényezőket is figyelembe veszi. A megoldásom egy olyan vegyes-egész lineáris programozási modell, mely a különböző paraméterek alapján több napos tervezési időtávra határozza meg az optimális megoldást.

A modell teszteléséhez készítettem egy grafikus interfészt, mely segítségével könnyen tesztelhető a modell a felületen rögzített adatokon keresztül. A megközelítés teszteléséhez meghatározásra került egy adathalmaz, melyet faiparral foglalkozó szakemberek tapasztalatai alapján állítottam össze. A megoldás hatékonyságának vizsgálatához külön tesztet készült a futási idők elemzésére, illetve a munkaerőhiány hatásának kiértékelésére.

A dolgozatom 2. fejezetében egy szakirodalmi áttekintés került kidolgozásra, ahol bemutatom az ütemezési problémák témakörét, a fűrészüzemek általános feldolgozásának folyamatát, illetve az irodalomban lévő javasolt megoldásokat specifikusan a fűrészüzemek ütemezésére. A feldolgozott szakirodalmak és az ipari tapasztalatok alapján azonosított problémát a 3. fejezetben informálisan és formálisan is definiáltam. Ezt követően a 4. fejezetben a feladatosztályra javasoltam egy MILP modellt, melynek a tesztelését az 5. fejezetben részleteztem. Végezetül a 6. fejezetben összefoglaltam a munkám eredményét, továbbá ismertettem a jövőbeli kutatási terveket.

2. fejezet

Szakirodalmi áttekintés

A fejezetben először a 2.1. alfejezetben áttekintést nyújtok az ütemezési problémák témaköréről. Ezt követően általánosan bemutatom a fűrészüzemeket és azok sajátosságait a 2.2. alfejezetben. A fejezet végén a 2.3. alfejezetben pedig összefoglalásra kerülnek a fűrészüzemek ütemezésével kapcsolatos kutatások és azok eredményei.

2.1 Ütemezési problémák

Az élet számos területén találkozhatunk ütemezési problémákkal, például órarendtervezésnél, a mozik műsorlistájának meghatározásakor vagy a számítógépek folyamatainak ütemezésénél. Ezek a problémák az optimalizálási problémák speciális változataiként tekinthetők. A terület sajátossága, hogy a feladatok során időzítéssel kapcsolatos döntéseket kell hozni, illetve dönteni kell arról, hogy az időzített feladatokat mely erőforrásokhoz rendeljük. Mint minden optimalizálási problémánál, szükséges egy cél megfogalmazása, mely alapján a sok alternatív megoldás közül kell meghatározni a legjobbat. Ütemezési feladatok esetén ilyen tipikus célok a befejezési idő, termelési költség minimalizálása vagy a termelés maximalizálása. A célok mellett figyelembe kell venni különböző megkötéseket is, ilyen lehet például a határidők betartása vagy a gyártásütemezéseknél gyakran előforduló raktárkészlet maximális kapacitásának betartása.

Az ütemezési feladatokat számos tulajdonság alapján lehet kategorizálni. Egyik legfontosabb tényező, amit figyelembe kell venni, hogy sztochasztikus [24] vagy determinisztikus [9] az adott ütemezési probléma. Az utóbbi esetén minden feladathoz tartozó paraméter előre meghatározott, míg egy sztochasztikus ütemezést részben vagy akár teljesen valószínűségi változók jellemeznek. Valószínűségi változó sok paraméterre megfogalmazható. Ilyen tipikusan a termékek iránti keresletnek a bizonytalansága, vagy a hibalehetőségek száma egy termelési folyamatban[19].

A szakirodalomban számos olyan optimalizálási megoldó módszer létezik, melyekkel az ütemezési problémák hatékonyan megoldhatók. A megfelelő megoldásnak a kiválasztása során az előzőekben említett tulajdonságokon kívül elengedhetetlen szempont a feladat

komplexitása, mérete. Kisebb méretű feladatoknál, vagy ahol fontosabb szempont az optimum megtalálása, mint a sebesség, gyakran alkalmaznak egzakt módszereket, melyek előnye, hogy minden esetben garantálják a legjobb megoldás megtalálását. Minél bonyolultabb vagy nagyobb egy feladat, annál több időt vesz igénybe az optimális megoldásnak a megtalálása, ami bizonyos esetekben kivitelezhetetlen a jelenkor eszközeivel. Ennek elkerülése érdekében sokszor alkalmaznak a nagyobb feladatoknál úgynevezett heurisztikákat. Az ilyesfajta eljárások nem garantálják a lehető legjobb megoldást, hanem elsősorban közelítik azt. Ezenfelül ezek az approximációs megoldások sokkal kedvezőbb futási idővel rendelkezhetnek, szemben az egzakt módszerekkel, melyek elsősorban a kevésbé komplex feladatoknál lehetnek előnyösebbek, ahol az optimum megtalálása az elsődleges cél. Gyakran előfordulhat, például egy online ütemezés során, hogy rövid időn belül szükségünk van egy megoldásra és ezen esetben is jobb választás lehet egy heurisztikus közelítés, mely általában gyorsabban biztosít jobb eredményt, mint egy egzakt megoldás.

Az egzakt módszerek esetén az egyik leggyakoribb módszertan a vegyes-egész lineáris programozás (Mixed Integer Linear Programming - MILP). A gyártásütemezési problémáknál alkalmazott MILP modellezési technikákról részletesen Méndez és szerzőtársai [17], illetve újabban Harjunkoski és szerzőtársai [11] írnak. Az ütemezésekre használt MILP modelleket két nagy csoportba lehet sorolni, az egyik az időrács (time-grid) alapú modellek [8], illetve a másik csoport a precedencia alapú modellek [16]. A lineáris programozási módszereken kívül gyakran alkalmaznak kombinatorikus módszereket, ilyen például az S-gráf módszertana, melyet Sanmartí és szerzőtársai [25, 26] dolgoztak ki. Az S-gráf modell olyan gráf reprezentáción alapul, mely kihasználja az ütemezési feladatok speciális tulajdonságait, ezáltal hatékony megoldó algoritmusok kidolgozását teszi lehetővé [23] [21]. A lineáris programozási modellekkel ellentétben itt nem paraméterekből és változókból készül a modell, hanem csúcsokból és súlyozott, irányított élekből. Ezenfelül számos heurisztikus megoldással is találkozhatunk a szakirodalomban, mint például az evolúciós algoritmusok, valamint a lokális keresési technikák alkalmazása. Az evolúciós algoritmusok használata egy elterjedt módszertan, amelynek működési alapja az evolúció darwini elméletében gyökerezik, ami szerint a természetben a folyamatosan változó életkörülményekhez való alkalmazkodás a fő szempont. Ezt a megközelítést [18] [13] [22], valamint a lokális keresési módszereket [7] [28] is számos kutatásban alkalmazták ütemezési problémákra.

Az ütemezés kutatási területén belül, egy nagyon fontos kategória a gyártórendszerek ütemezése. A termelési rendszerekben léteznek folyamatos és szakaszos folyamatok. A legfőbb különbség közöttük az, hogy a bemenetük/kimenetük hogyan kerül felhasználásra/előállításra. A szakaszos termelési rendszerek ellentétben a folyamatossal, a termékeket meghatározott véges adagokban dolgozzák fel. Mindkét típusnak hatalmas irodalma van és különböző modellezési technikákat igényelnek.

A gyártási feladatoknál egy másik fontos tényező az erőforrások és azok működése. Ha

a termékek gyártása egy lépésből áll, akkor single stage problémáról van szó. A single stage problémák az alábbi alkategóriákba sorolhatók:

- **Single Machine:** Egyetlen gép elérhető csak, melyen minden feladatot el lehet végezni. A feladatoknak különböző feldolgozási ideje van.
- **Identical paralell machines:** m számú azonos gép áll rendelkezésre, melyeken bármelyik egylépéses feladat végrehajtható.
- **Paralell machines with different speed:** Hasonló az előző pontban említett típushoz, de itt minden gépnek meghatározott eltérő végrehajtási sebessége van.
- **Unrelated machines in paralell:** A Single stage általánosítása, ahol egy gépen végzett feladat feldolgozási ideje a bemeneti paraméter.

Több lépéses gyártási tervek esetén jellemzőek a szekvenciális folyamatok, melyek közül a legismertebbek a következők:

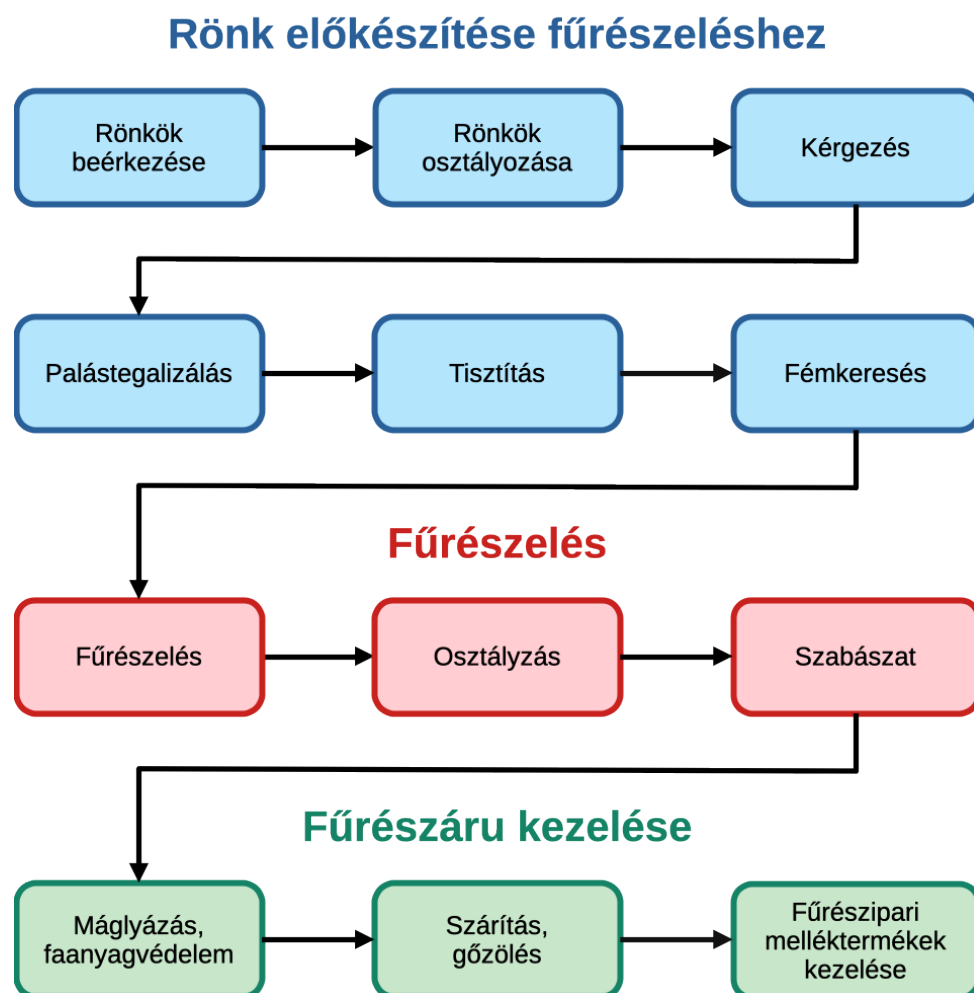
- **Flow shop:** A feladatoknak m lépése van, melyeket az m gépeken kell végrehajtani, minden feladat esetén ugyanabban a sorrendben.
- **Flexible flow shop:** Az Flow shop és a Identical paralell machines változatának az általánosítása, ahol minden egyes feladatnak c szakaszon kell ugyanabban a sorrendben végig mennie, ahol több párhuzamos gép áll rendelkezésre.
- **Job shop:** Az Flow shop általánosítása, ahol a feladatoknak különböző sorrendje lehet a gépeken, és nem szükséges egy feladatnak az összes gépen végig mennie.
- **Flexible job shop:** A Job shop és a Flexible flow shop általánosítása, amikor a feladatoknak az szakaszokon a feladattól függő sorrendben kell végig menniük, és minden szakaszban néhány azonos gép áll rendelkezésre.
- **Open shop:** Minden feladatnak végig kell mennie az összes gépen, de ezek sorrendje nincs előre meghatározva.

Számos cikk található a szakirodalomban, melyek az erdőgazdálkodás és az ahhoz tartozó tevékenységek optimalizálásával foglalkoznak. Ebben a témában nyújt áttekintést Weintraub és szerzőtársai által írt cikk [31]. Ennek egy válfaja a faipari termelési folyamatok ütemezése. Ide tartozhat egy fahulladék feldolgozó üzem ütemezése is, melyre Dávid és szerzőtársai [6] adnak egy hatékony megoldást. A jelen dolgozat témáját a fűrészüzemek képzik, mely szintűgy fontos területe a faipari termelésnek.

2.2 Fűrészüzemek

A faipari termékek gyártását egy összetett feldolgozási folyamat előzi meg, melynek a fa kivágása utáni első fázisa a fűrészüzemekben a farönkök megmunkálása. Az üzemek célja, hogy a beérkezett rönkökből a keresletnek eleget tevő fűrészárú keletkezzen. Az iparban nagyjából minden fűrészüzem más specifikációkkal rendelkezik, köszönhetően a technológiai fejlődésnek és a folyamatosan változó keresletnek. Léteznek nagyobb, jelentős mértékben automatizált fűrészüzemek, de találkozhatunk kisebb, kevésbé gépesített üzemekkel is, ahol más szempontok alapján zajlik feldolgozás. Az üzemek működését jelentősen befolyásolja még azok földrajzi elhelyezkedése is, mivel a különböző területeken található erdők eltérő faállománnyal rendelkeznek. Ezek a különböző fatípusok pedig eltérő feldolgozási módszereket igényelnek.

A fűrészüzemek általános folyamatát a 2.1. ábra szemlélteti.



2.1. ábra. Fűrészüzemek általános feldolgozási folyamata

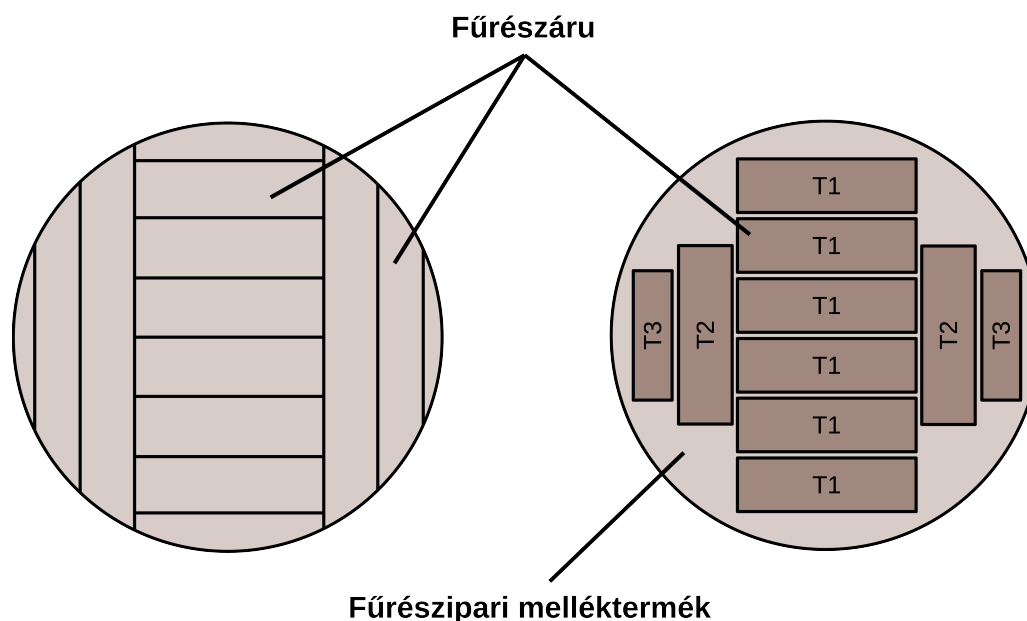
A különböző rönkökből másképpen és másfajta végtermékek állíthatók elő, ezért azokat

a beérkezést követően osztályokba sorolják olyan tulajdonságaik alapján, mint például fafaj, átmérő, hossz vagy minőség. A fűrészelés előtt egy előkészítési eljárás történik a farönkökkel. Az első ilyen lépés a kérgezés, ahol a kérget távolítják el a fáról. Ezt követi a palástegalizálás, melynek a célja a farönk durva külső felületének az eltávolítása, ami a felület egyenetlenségeinek csökkentésével elősegíti a fűrészelhetőséget. A tisztítás során eltávolítják a fafelületen lévő port, homokot és az egyéb szennyeződések. Fontos még figyelembe venni a faanyagban esetlegesen megbúvó fémrészeket, melyek könnyen okozhatnak sérüléseket a faanyagnak és az üzem gépeinek, ezért a rönköket át kell vizsgálni egy fémkereső géppel.

A fűrészelés során a rönköket hosszirányú vágással alakítják falécekké. A vágást különféle gépekkel végezhetik, melyeknek két leggyakoribb típusa a keretfűrész és a szalagfűrész. A két rönkvágó technológia között számos különbség van. A keretfűrészgép, avagy a gatter egy nagy teljesítményű alapgép, mely gyorsabb vágást biztosít a szalagfűrészekhez képest, azonban fix vágási mintával dolgozik, melyet csak a gép átállításával lehet megváltoztatni. Így különösen fontos a megfelelő vágási mintáról való döntés, hiszen egy ilyen átállítás plusz idővel jár. Ezenfelül általában jóval több humán erőforrást igényel egy ilyen technológia működtetése, mint a szalagfűrész esetén, mivel az egész folyamatot felügyelő szakemberen túl szükség van külön segédmunkásokra a feldolgozott fűrészárak, esetleg a rönkök mozgatásához. Ez az emberi erőforrás igény jelentős tényező lehet a térségeinkben megtalálható fűrészüzemekben, ahol a jelenkor egyik legnagyobb kihívása a munkaerőhiány. Egy szalagfűrészgép a rugalmasabb feldolgozás mellett jóval lassabb vágási sebességet biztosít, így tömegtermelésre kevésbé alkalmas. Előnye viszont többek között a vékonyabb szerszám használat, és az ezzel együtt járó kisebb fűrészelési résvesztés, vagy a kisebb energia-felvétel. Ezenfelül sok más technológia tényező is szerepet játszik a fűrészelési folyamatban, melyről részletesebben Hargitai [10] ír.

A különböző vágási minták különböző fűrészáru kombinációkat és fűrészipari melléktermék mennyiségeket eredményeznek. A legjobb minta meghatározásához fontos figyelembe venni, hogy melyik rönkosztályból kerülnek a rönkök feldolgozásra. A [20], [27], [32] kutatások megvizsgálták azt a problémát, hogy hogyan lehet kiválasztani az optimális vágási mintákat a különféle típusokhoz. [5], [12] cikkek megoldást is javasolnak erre a kérdésre. A rönkvágási módok közül az egyik legegyszerűbb az élesvágás, ahol a rönkből egyetlen vágással állítják elő a szélezetlen fűrészárakat. Ezenfelül gyakran alkalmaznak bonyolultabb vágásokat is, ahol többször feldolgozzák az adott rönköt a fűrészüzemi alapgépen. Ilyen például a prizma vágás, ahol a rönköt prizma alakúra vágják, majd ezután a prizmából állítják elő a fűrészárak egy részét, melyek közül a két szélső darabon kívül szélezetlen áruvá válnak. Erre egy példát mutat be a 2.2. ábra.

Az alapgépeken történő vágás után az áruk a szabászáron is áteshetnek, ahol méretre vágják, így a feldolgozott fűrészárakat a keresletnek megfelelő méretűre alakíthatják. A 2.2 ábrán a jobb oldali mintán látható, ahogy a szélezetlen fűrészárak is méretre lettek



2.2. ábra. Prizma vágás példa

vágva. Az ilyesfajta műveletekhez különféle technológiákat alkalmaznak, ilyen például a sorozatvágó körfűrészgép. A fűrészelés után a készterméket méret és minőség alapján osztályozzák. Az utóbbit meghatározhatják például elszíneződések, gombakárosítás, rovarkárosítás, repedések vagy fűrészelési hibák alapján. A legutolsó fázisa az eljárásnak a fűrészárak kezelése. A raktározás egyik általános módszere a termékek máglyákban történő tárolása. A faanyagok áteshetnek egy gőzölési folyamaton, melynek célja a csírátlanítás, a mechanikai tulajdonságok javítása, a fertőzések megszüntetése, a szárítási eljárás elősegítése, illetve ennek hatására a faanyag színe sötétedik, így esztétikailag is javítja a fűrészárut. A szárítás történhet a természetes módon a szabadlevegőn, de megvalósítható mesterséges szárítására alkalmazott szárítóberendezésekkel is. Ezenfelül a fűrészüzemeknek nem szabad figyelmen kívül hagyni az eljárás során keletkező fűrészipari melléktermék kezelését sem.

A faanyagok feldolgozásának ezen fázisa rendkívül összetett, sok döntéssel találkozhatunk benne, amit fontos mindig előre megtervezni, hogy hatékony működést és a magas minőségű fűrészárakat érjen el a fűrészüzem. A feldolgozás optimalizálása több kategóriára bontható, melyek közül az egyik legfontosabb a folyamatok megfelelő ütemezése.

2.3 Fűrészüzemek ütemezési problémái

Az erdészeti ellátási lánc esetén általában három döntéshozatali szint határozható meg, pontosabban a stratégiai, taktikai [14] és az objektív tervezés. A stratégiai döntéseket hosszú távra kell meghozni, ami jelenthet akár több évtizedet is. Ennél a szintnél olyan célokról beszélhetünk, mint például az erdei ökoszisztéma megőrzése vagy a fenntartha-

tó erdőgazdálkodás. A taktikai tervezésnél középtávú tervezésről beszélhetünk, melyek tipikusan néhány évet, hónapot jelentenek. Ilyen problémát vizsgálnak meg Pamela P. Alvarez és szerzőtársai a cikkükben [1]. Objektív tervezésnél valós idejű, napi, heti, vagy akár egy hónapos tervezéssel foglalkozunk. Ezek általában igen részletes megoldásokat jelentenek. Az ilyen fűrészüzem ütemezési terveknel a legtöbb kutatás arra összpontosít, hogy miként lehet meghatározni a farönkök feldolgozásának a legjobb módját a termelés optimalizálása érdekében.

A fűrészüzemi ütemezési megoldások során az egyik leggyakoribb technika, a matematikai programozás modellek alkalmazása. Sergio Maturana és szerzőtársai cikkükben [15] bemutatnak egy determinisztikus matematikai modellt egy chilei fűrészüzem ütemezésére, melynek a teljesítményét egy heurisztikával hasonlítják össze. A megoldás során 6 hetes periódusokkal dolgoztak, melyeket napi szintű részperiódusokra osztottak. A modell célfüggvénye a megrendelések elmaradásával, illetve a raktárkészletben maradó plusz termékekkel járó költségeket minimalizálja, illetve figyelembe veszi, hogy a rönkök kiválasztásánál elsősorban a jobb minőségűek kerüljenek feldolgozásra. Több tesztet összevetésében a cikk megmutatta, hogy a matematikai modell a hatékonyabb megoldás a vizsgált problémára.

Több szakirodalomban javasolt megoldás feltételezi, hogy az összes adat ismert, ami a valóságban sok esetben nem igaz. Az optimalizálási modellek során és a fűrészüzemek ütemezésénél is gyakran előjön a bizonytalanság kezelésének a problémája. Az irodalomban általában különféle megközelítéseket használnak az ilyesfajta problémára, például alkalmazhatnak érzékenységvizsgálatot, ami a döntési modellek kiértékelését jelenti, azaz megmutatja az eredmények változását a bemeneti paraméterek változtatásával. Megoldás lehet a sztochasztikus programozás, ahol azt feltételezzük, hogy a bizonytalan paraméterek valószínűségi eloszlása ismert vagy pontosan becsülhető. A megközelítés hátránya, hogy nagyon sok adatot igényel, és sokszor közülük néhányhoz nehéz hozzájutni, ezenfelül a problémák gyakran megoldhatatlanok, vagy túl sok időt igényelnek. Népszerű megközelítés még a robusztus optimalizálás, mely bizonytalansági tartományok alapján kezeli a bizonytalanságot, így kevesebb adatot és számítási erőforrást igényel, mint a sztochasztikus programozás. Mauricio Varas és szerzőtársai 2014-ben [30] bemutattak egy megoldást a chilei fűrészüzem problémájára, mely kezeli a farönkök készletének vagy a keresletnek a bizonytalanságát. Ez a modell Bertsimas és Sim [3] által kidolgozott megközelítésen alapul. A bizonytalanságot más kutatások is figyelembe veszik robusztus optimalizálással, például Masoumeh Kazemi Zanjani és szerzőtársainak [33] a megoldása a folyamatok hozamának és a termékek iránti keresletnek a bizonytalanságát kezeli. Ezenfelül a kutatás eredményei arra is utalnak, hogy a robusztus optimalizálási megközelítés bizonyos szituációkban előnyöket jelenthet a kétlépcsős sztochasztikus programozási megoldásokkal szemben.

A témában lévő kutatások legtöbbje az egy feldolgozási folyamat ütemezését vizsgálja.

Pamela P. Alvarez és Jorge R. Vera [2] olyan megoldást mutat be, mely több fűrészüzem folyamatát tervezi meg párhuzamosan. A feldolgozás ütemezése többek között kezeli az esetleges szárítást vagy az újrafeldolgozást és ennek költséget is.

A szakirodalomban több cikkben figyelembe veszik a különböző vágási minta és rönk kombinációkat, azonban ezekre a vágási mintákra külön döntést nem fogalmazznak meg a modellekben. Azok csak a feldolgozás hozamának paraméterét befolyásolják. 2018-ban Nicolás Vanzetti és szerzői [29] mutattak olyan modellt, mely figyelembe veszi a vágási minták beállításának idejét és költségét. 2019-ben Diego Broz és szerzőtársaik a cikkükben [4] a célprogramozás módszerét alkalmazták az előbbi problémára, azonban míg Vanzettiék [29] 5 napos periódusra határozták meg, ebben a cikkben egy napos ütemezést valósítottak meg. A megoldásukban figyelembe vették a rönkök különböző feldolgozási idejét, melyet a legtöbb másik irodalomban található módszer figyelmen kívül hagy, vagy csak egyetlen konstanssal írja le.

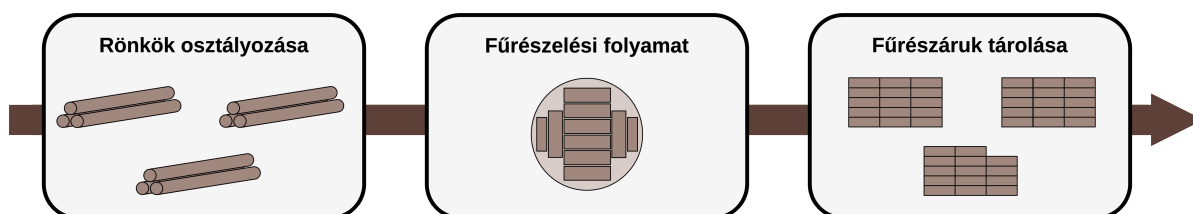
3. fejezet

Probléma definíció

Annak érdekében, hogy egy olyan megoldást biztosíthassunk, mely közelebb áll az ipari szektor kihívásaihoz, mint az irodalmi megoldások, olyan tényezőket kell figyelembe venni, melyek gyakran fűrészüzem specifikus kérdések. Ezeket a specifikus kérdéseket különböző faiparral foglalkozó szakemberrel konzultálva határoztam meg. Így a szakirodalomban lévő megoldások, illetve az előbb említett diskurzusok alapján fogalmaztam meg a probléma definíciót mind informálisan a 3.1 szövegrészben, mind formálisan a 3.2 alfejezetben.

3.1 Informális probléma definíció

A fűrészüzem ütemezése több napos tervezésként valósul meg. Továbbá adott egy műszakhossz, mely alatt történik a fűrészüzemi feldolgozás. Megoldásom ezen fűrészüzemi feldolgozás folyamatának az átgondolásával kezdtem. Az ütemezés szempontjából a műveletek egyik legkritikusabb pontja a fűrészelési lépés. Ebből kifolyólag bizonyos folyamatok elhagyhatók, melyek nem befolyásolják jelentősen az ütemezés hatékonyságát. A feladat során feltételezzük, hogy adott a rönkkészlet, illetve nem foglalkozunk a fűrészárak rak-tározásával, mint ütemezendő folyamat. Ezt az eljárást a 3.1 ábra szemlélteti.



3.1. ábra. Az ütemezendő folyamat

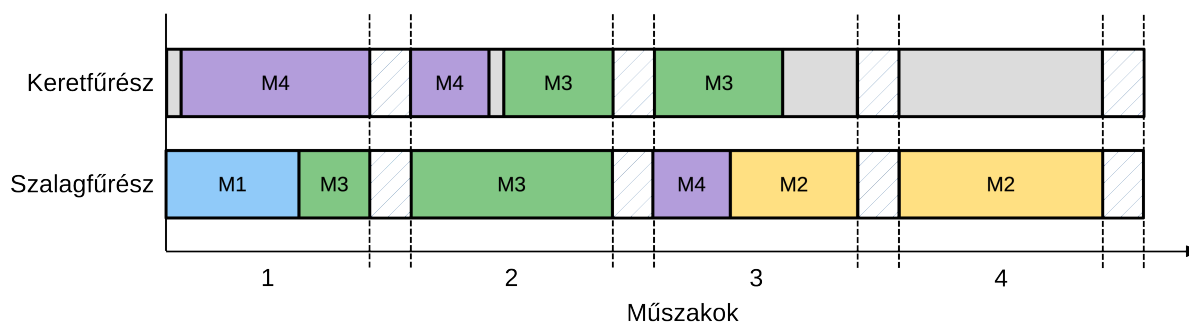
A farönkök az egész tervezési időszakra érkeznek, így a feldolgozás során ezzel kell gazdálkodni. A beérkezést követően a legfontosabb lépés a rönkök típusok alapján történő csoportosítása. A feladat megoldásához el kell különíteni azokat a farönk osztályokat, me-

lyeket a fák fizikai tulajdonságai alapján határoznak meg. A vizsgált problémaosztályban feltételezzük, hogy egyetlen fafajjal foglalkozunk, valamint a rönkök hossza is megegyezik, így az átmérőjük alapján kerülnek különböző osztályokba. Ezen csoportok alapján definiálhatóak a feldolgozást befolyásoló paraméterek.

A legtöbb eddigi kutatásban javasolt megoldások elsősorban a nagyobb automatizált fűrészüzemek igényeihez igazodnak. Így a feladatosztály megfogalmazásánál a célom egy olyan fűrészüzem reprezentálása volt, ahol a rönkök fűrészelését a jóval egyszerűbb technológiák végzik. A feladat során két rönkvágó alapgép áll rendelkezésre az üzemben, egy keretfűrész és egy szalagfűrész. A munkám során szeretnék különös hangsúlyt fektetni a két technológia között lévő különbségekre, mivel a két megoldás jelentősen eltér. Így a feladat során egyik fő döntés az, hogy mikor mely gépet üzemeltetjük és hogyan.

Egy fontos tényező továbbá a vágási mintákról való döntés, melyről sok irodalmi megoldás feltételezi, hogy az adott esetben a leghatékonyabb van használatban. Ezenfelül nem foglalkoznak azzal, hogy egy keretfűrészgépen a minta váltása sokszor jelentős átállási időt eredményez. A probléma esetén minden rönk típushoz különböző vágási mintákat definiáltam, melyeket a fűrészelési folyamat során alkalmazhatunk. A gyakorlati tapasztalatok alapján feltételezzük, hogy egy műszak alatt csak egy vágási minta csere történhet a keretfűrészgépen. Ezenfelül figyelembe vesszük, hogy ha egy műszak kimarad, akkor az üres nap alatt megtörténhet az átállás egy másik vágási mintára, így, ha váltás történik ez idő alatt, akkor az azt követő napon nem számolunk az azzal járó plusz idővel. Szalagfűrész esetében a minták közötti váltás a gyakorlatban nem jár számottevő késleltetéssel, így ezen a gépen tetszőleges számú váltás megengedett egy nap során.

A fűrészelési folyamatok ütemezésére egy példamegoldás a 3.2 Gantt diagrammon látható, ahol 4 műszakra történt a tervezés. Azt, hogy mikor, milyen mintával történik az adott gépen a feldolgozás az M1, M2, M3 és az M4 jelöli, melyeket eltérő színekkel ábrázoltam. Szürke színnel jelöltem a gépek üresjáratát, melyek a minták közti átállási időt is magába foglalja.



3.2. ábra. Az fűrészüzem ütemezésének egy példája

A folyamat alatt csak szélezetlen fűrész árukat gyártunk élesvágás technológiával. Így minden rönk csak egyszer kerül feldolgozásra mindkét rönkvágó alapgépre vonatkozóan.

A két technológia esetén a fűrészelési idők is különbözőek, így a rönk átbocsátóképessége is eltérő a két gépnél. Ezenfelül minél több fűrészárut eredményez egy vágási minta, annyiival tovább tart ez a fűrészelés. Hiszen egy szalagűrész esetében ennyiszor kell végig fűrészelni a rönköt, illetve a keretfűrésznel lassabb előtolási és szerszámsebességet kell alkalmazni a több fűrészlap esetén.

Az automatizált üzemekkel szemben a kisebb fűrészüzemekben jelentős kihívást jelent az eszközök üzemeltetéséhez szükséges humán erőforrás. Két képzettséget különböztetünk meg az üzem alkalmazottjaira vonatkozóan, melyek a szakemberek és a segédmunkások. A két rönkvágó alapgép eltérő emberi erőforrás igényekkel rendelkezik. Ahhoz, hogy döntéseket hozzassunk, hogy mikor melyik alkalmazott, melyik gépen dolgozzon, definiálni kell a dolgozók beosztását. Előfordulhat, hogy több szakember áll rendelkezésre, mint amennyi az adott gép vagy a gépek működtetéséhez szükséges lenne. Ebben az esetben a szakemberek segédmunkásként is dolgozhatnak, azonban ez fordítva nem történhet meg, mivel egy segédmunkás nem tudja ellátni a szakember feladatait. A feladatosztály megengedi továbbá, hogy extra segédmunkásokat alkalmazzunk. Ezt egy plusz paraméter szabályozza, reprezentálva a munkaerőhiányt és az azzal járó nehézségeket.

A tervezési időszak minden napjára vonatkozóan specifikált mennyiségű fűrészáru előállításra van szükség. Amennyiben egy adott napon túltermelés történik, a felhalmozott fűrészárut egy raktárban tárolhatjuk el későbbi felhasználásra. A célfüggvény megfogalmazása során szintén figyelembe vettem az emberi munkaerőhiány problémáját. Mivel az üzem szeretné minden nap teljesíteni a keresletet, de ha nem tud minden gépet üzemeltetni, akkor gyakran hiánnyal kell számolnia. Így a cél ennek a hiánynak a minimalizálása, melyhez késedelmi díjat rendelünk.

3.2 Formális probléma definíció

A problémaosztály az alábbiakban felsorolt halmazokkal és paraméterekkel definiálható formálisan.

3.2.1 Halmazok

$\mathcal{D} = \{1, 2, \dots, |D|\}$ a napok halmaza.

$\mathcal{L}$ a farönk osztályok halmaza.

$\mathcal{T}$ a fűrészáru termékek halmaza.

$\mathcal{P}$ az összes vágási minták halmaza.

$\mathcal{P}_l \subseteq \mathcal{P}$ a vágási minták halmaza minden $l \in \mathcal{L}$ farönk osztályra.

3.2.2 Paraméterek

Általános paraméterek

$H \in \mathbb{R}^+$ a műszak hossza. $[s]$

$V_l \in \mathbb{R}^+$ a térfogat minden $l \in \mathcal{L}$ farönk osztályra. $[m^3]$

$C \in \mathbb{R}^{0,+}$ a raktár kapacitása. $[m^3]$

Fűrészelési technológiákhoz köthető paraméterek

$l_p \in L$ a $p \in \mathcal{P}$ vágási mintához tartozó farönk osztály.

$ST_p^F, ST_p^B \in \mathbb{Z}^+$ a keretfűrész és a szalagfűrész vágási ideje minden $p \in \mathcal{P}$ vágási mintára.
 $[sec/db]$

$CT^F \in \mathbb{Z}^+$ a keretfűrészben a vágási paraméter cserélésével járó átállási idő. $[sec]$

Humánerőforrás menedzsmenthez köthető paraméterek

$RR^{SP,F}, RR^{SP,B} \in \mathbb{N}_0$ a keretfűrész és a szalagfűrész működtetéséhez szükséges szakember száma. $[-]$

$RR^{AW,F}, RR^{AW,B} \in \mathbb{N}_0$ a keretfűrész és a szalagfűrész működtetéséhez szükséges segéd-munkások száma. $[-]$

$HR_d^{SP} \in \mathbb{N}_0$ minden $d \in \mathcal{D}$ napon a rendelkezésre álló szakember száma. $[-]$

$HR_d^{AW} \in \mathbb{N}_0$ minden $d \in \mathcal{D}$ napon a rendelkezésre álló segédmunkások száma. $[-]$

$X^{AW} \in \mathbb{N}_0$ a maximálisan megengedett extra segédmunkások száma. $[-]$

Raktározáshoz és a kereslethez köthető paraméterek

$I_l \in \mathbb{R}^{0,+}$ a rendelkezésre álló rönkök száma minden $l \in \mathcal{L}$ farönk osztályra. $[-]$

$DM_{t,d} \in \mathbb{R}^{0,+}$ a kereslet minden $t \in \mathcal{T}$ fűrészáru típusra és $d \in \mathcal{D}$ napra. $[m^3]$

$Y_{t,p} \in [0, 1]$ a hozamarány minden $t \in \mathcal{T}$ fűrészáru típusra és $p \in \mathcal{P}$ vágási mintára. $[-]$

$LF_t \in \mathbb{N}_0$ a késedelmi díj minden $t \in \mathcal{T}$ fűrészáru típusra. $[HUF/m^3]$

4. fejezet

Megoldó módszer

Ebben a fejezetben az előzőekben definiált optimalizálási problémára javaslok megoldást. A feladat komplexitása részben függ a paraméterezéstől is, így érdemes lehet heurisztikus megoldásokat egyaránt megvizsgálni. Azonban erre a feladatosztályra egzakt módszerek ugyancsak hatékonyan alkalmazhatók. A probléma egy egyszerűbb változatára Maturana és szerzőtársai cikkükben [15] javasoltak hatékony megközelítést. A megoldásom egy olyan vegyes-egész lineáris programozási, azaz MILP modell, amely az említett cikkben tárgyaltakon túlmenően kezeli a keretfűrész és a szalagfűrész közti különbségeket, beleértve a vágási minták cseréjét és az esetlegesen azzal járó plusz átállási időt is. A megoldásomban külön modellezek egy keretfűrész és egy szalagfűrész technológiát. Ezenfelül figyelembe vettem a különböző erőforrások emberi erőforrás igényét.

A megközelítés az alábbiakban taglalt változókból (4.1. alfejezet), korlátozásokból (4.2. alfejezet) és célfüggvényből (4.3. alfejezet) áll.

4.1 Változók

Melyik napon dolgozzon a keretfűrész, illetve a szalagfűrész:

$$w_d^F, w_d^B \in \{0, 1\} \quad \forall d \in \mathcal{D} \quad (4.1)$$

A megoldásban feltételezzük, hogy egy műszak alatt maximum egy vágási minta csere történhet a keretfűrészen. Ennek az időpontját a következő változó rögzíti:

$$t_d \in [0, H - CT^F] \quad \forall d \in \mathcal{D} \quad (4.2)$$

Annak meghatározása érdekében, hogy mikor milyen mintával dolgozzon a keretfűrész, a műszakot két szakaszra bontjuk. Külön döntést rendelünk a csere előtt lévő vágási

mintára és az utána lévőre:

$$s_{d,p}^{F,-}, s_{d,p}^{F,+} \in \{0, 1\} \quad \forall d \in \mathcal{D}, p \in \mathcal{P} \quad (4.3)$$

A keretfűrész napi rönkfeldolgozására adott a következő változó:

$$q_{d,p}^F \in \mathbb{N}_0 \quad \forall d \in \mathcal{D}, p \in \mathcal{P} \quad (4.4)$$

$q_{d,p}^F$ a többi változó értékéből az alábbi egyenlettel lenne kiszámítható:

$$q_{d,p}^F = s_{d,p}^{F,-} \cdot \frac{t_d}{ST_p^F} + s_{d,p}^{F,+} \cdot \frac{H - t_d - CT^F}{ST_p^F} \quad \forall d \in \mathcal{D}, p \in \mathcal{P} \quad (4.5)$$

A 4.5 egyenletben azonban a $s_{d,p}^{F,-} \cdot t_d$ és $s_{d,p}^{F,+} \cdot t_d$ változók szorzata szerepel, mely a lineáris programozási modell esetén nem megengedett. A quadratikus probléma elkerülése érdekében külön változók kerültek bevezetésre vágási minta csere előtti és utáni rönkfeldolgozásra:

$$q_{d,p}^{F,-}, q_{d,p}^{F,+} \in \mathbb{N}_0 \quad \forall d \in \mathcal{D}, p \in \mathcal{P} \quad (4.6)$$

A szalagfűrész napi rönkfeldolgozása:

$$q_{d,p}^B \in \mathbb{N}_0 \quad \forall d \in \mathcal{D}, p \in \mathcal{P} \quad (4.7)$$

Az adott napon alkalmazott extra segéd munkások száma:

$$x_d \in \mathbb{N}_0 \quad \forall d \in \mathcal{D} \quad (4.8)$$

Fűrészáru termelés minden napra vonatkozóan:

$$y_{d,t} \in \mathbb{R}^{0,+} \quad \forall d \in \mathcal{D}, t \in \mathcal{T} \quad (4.9)$$

Keresleten felül termelt és emiatt eltárolt fűrészárú minden napra vonatkozóan:

$$o(p)_{d,t} \in \mathbb{R}^{0,+} \quad \forall d \in \mathcal{D} \cup \{0\}, t \in \mathcal{T} \quad (4.10)$$

A termelésben keletkezett napi hiány a kereslethez képest:

$$u(p)_{d,t} \in \mathbb{R}^{0,+} \quad \forall d \in \mathcal{D}, t \in \mathcal{T} \quad (4.11)$$

4.2 Korlátozások

A keretfűrész napi működését az alábbiakban részletezett egyenletek szabályozzák, melyek a tervezési időszak minden napjára meghatározzák a gép aktivitását. A 4.12 korlátozás a vágási minta csere előtti keretfűrész munkafolyamatokat írja le, míg a 4.13 a csere utáni folyamatokat rögzíti. E két kifejezés határozza meg, hogy ha bármely vágási mintával működik a keretfűrész az adott napon, akkor a gép üzemel, illetve ha a gép üzemel, akkor pontosan egy vágási mintát használ a váltás előtt és után.

$$\sum_{p \in \mathcal{P}} s_{d,p}^{F,-} = w_d^F \quad \forall d \in \mathcal{D} \quad (4.12)$$

$$\sum_{p \in \mathcal{P}} s_{d,p}^{F,+} = w_d^F \quad \forall d \in \mathcal{D} \quad (4.13)$$

A műszak kezdésekor figyelembe kell venni, hogy mi volt az előző nap végén lévő vágási minta. A 4.14 és a 4.15 korlátozások biztosítják, hogy amennyiben a keretfűrészgép két egymást követő munkanapon is üzemel, a műszak kezdetén azonos vágási mintát használjon, mint az előző munkanap végén. Ez a kikötés azonban nem vonatkozik az olyan esetekre, ahol a két műszak között van olyan nap mikor nem üzemel a gép, ilyenkor feltételezzük, hogy a szünet alatt megtörténhet az átállás két minta között.

$$s_{d,p}^{F,+} \geq s_{d+1,p}^{F,-} - 1 \cdot (2 - w_d^F - w_{d+1}^F) \quad \forall d \in \mathcal{D}, p \in \mathcal{P} \quad (4.14)$$

$$s_{d,p}^{F,-} \leq s_{d+1,p}^{F,+} + 1 \cdot (2 - w_d^F - w_{d+1}^F) \quad \forall d \in \mathcal{D}, p \in \mathcal{P} \quad (4.15)$$

A keretfűrész rönkfeldolgozásának modellezésekor úgynevezett "big-M" korlátozásokat alkalmaztam, melyek figyelembe veszik, hogy a gép mikor milyen mintával dolgozik. Az $\mathbf{M}$ paraméter bevezetése helyett a $\frac{H}{ST_p^F}$ kifejezést használtam, mely az alábbi korlátozásoknál kellően nagy konstans a relaxáció biztosítására. A napi vágási minta csere előtti rönkfeldolgozás kifejezésére a 4.16 és a 4.17 egyenletek szolgálnak, míg az átállás utáni feldolgozás számítását a 4.18 és a 4.19 korlátozások biztosítják.

$$q_{d,p}^{F,-} \leq \frac{t_d}{ST_p^F} + \frac{H}{ST_p^F} \cdot (1 - s_{d,p}^{F,-}) \quad \forall d \in \mathcal{D}, p \in \mathcal{P} \quad (4.16)$$

$$q_{d,p}^{F,-} \leq \frac{H}{ST_p^F} \cdot s_{d,p}^{F,-} \quad \forall d \in \mathcal{D}, p \in \mathcal{P} \quad (4.17)$$

$$q_{d,p}^{F,+} \leq \frac{H - t_d - CT^F \cdot (1 - s_{d,p}^{F,-})}{ST_p^F} + \frac{H}{ST_p^F} \cdot (1 - s_{d,p}^{F,+}) \quad \forall d \in \mathcal{D}, p \in \mathcal{P} \quad (4.18)$$

$$q_{d,p}^{F,+} \leq \frac{H}{ST_p^F} \cdot s_{d,p}^{F,+} \quad \forall d \in \mathcal{D}, p \in \mathcal{P} \quad (4.19)$$

A szalagfűrész technológia esetén a 4.20 korlátozás szabályozza a napi rönkfeldolgozást,

kezelve, hogy adott napon üzemel-e az eszköz.

$$\sum_{p \in \mathcal{P}} q_{d,p}^B \cdot ST_p^B \leq H \cdot w_d^B \quad \forall d \in \mathcal{D} \quad (4.20)$$

A következőkben részletezett korlátozások kezelik az üzem humánerőforrás menedzsmentjét. Mind a szalagfűrész, mind a keretfűrész esetén teljesíteni kell a feltételt, miszerint az adott gép csak akkor üzemeltethető, ha tudunk elég emberi erőforrást biztosítani a működéséhez. Ezt a korlátot foglalja magába a 4.21 kifejezés a szakemberekre vonatkozóan, ahol figyelembe vesszük az adott napon rendelkezésre álló dolgozók számát. A segéd munkások korlátját a 4.22 egyenlet írja le, mely során megengedjük, hogy a rendelkezésre szakemberek is alkalmazhatók legyenek segéd munkásként. Ezenfelül a korlátozás lehetőséget biztosít extra segéd munkások behívására is.

$$RR^{SP,F} \cdot w_d^F + RR^{SP,B} \cdot w_d^B \leq HR_d^{SP} \quad \forall d \in \mathcal{D} \quad (4.21)$$

$$\begin{aligned} RR^{AW,F} \cdot w_d^F + RR^{AW,B} \cdot w_d^B &\leq HR_d^{AW} + x_d + \\ (HR_d^{SP} - RR^{SP,F} \cdot w_d^F - RR^{SP,B} \cdot w_d^B) &\quad \forall d \in \mathcal{D} \end{aligned} \quad (4.22)$$

Az extra segéd munkás műszakok maximális számát egy paraméter határozza meg, mely összességében szabályozza a teljes tervezési időszakban alkalmazható segéd munkások számát. A modell a 4.23 egyenlettel menedzseli a napok között a pluszban megengedett segéd munkásokat.

$$\sum_{d \in \mathcal{D}} x_d \leq X^{AW} \quad (4.23)$$

Az üzem az egész tervezési időszakra egy előre meghatározott rönkkészletből gazdálkodik. Ezt a megoldást a 4.24 egyenlettel kezeli, ahol a rönkvágó alapgépek feldolgozását szabályozzuk.

$$\sum_{d \in \mathcal{D}} \sum_{\substack{p \in \mathcal{P} \\ l_p = l}} (q_{d,p}^{F,-} + q_{d,p}^{F,+} + q_{d,p}^B) \leq I_l \quad \forall l \in \mathcal{L} \quad (4.24)$$

Annak kiszámításáért, hogy melyik nap mennyi fűrészáru hozam keletkezett, a 4.25 kifejezés felel. Itt a vágási mintához tartozó hozamokat összegezzük minden gépre vonatkozóan.

$$y_{d,t} = \sum_{p \in \mathcal{P}} Y_{t,p} \cdot V_{l_p} \cdot (q_{d,p}^{F,-} + q_{d,p}^{F,+} + q_{d,p}^B) \quad \forall d \in \mathcal{D}, t \in \mathcal{T} \quad (4.25)$$

Minden tervezési napra adott egy kereslet, melyet az adott műszakban teljesítenie kellene a fűrészüzemnek. Azonban muszáj számolni, hogy egy adott műszak alatt hiány

is keletkezhet a kereslethez képest, ekkor pótolható a fűrészáru a raktárból, ha az előző napokban a kereslet igényei felett termelt az üzem. Ha erre nincs lehetőség, akkor a hiányként kell kezelni. Ennek korlátozásáért a 4.26 egyenlet felel.

$$y_{d,t} + o(p)_{d-1,t} - o(p)_{d,t} + u(p)_{d,t} \geq DM_{t,d} \quad \forall d \in \mathcal{D}, t \in \mathcal{T} \quad (4.26)$$

A modellben a 4.27 egyenlettel határozzuk meg a tervezési időszakjára vonatkozóan a raktár kezdeti állapotát, melyet alapértelmezetten minden fűrészáru típusra nulla értékre állítunk.

$$o(p)_{0,t} = 0 \quad \forall t \in \mathcal{T} \quad (4.27)$$

A fűrészáruk raktározásának egy maximum kapacitás paraméterrel szabunk határt, melyet a 4.28 kifejezés vesz figyelembe minden napra vonatkozóan.

$$\sum_{t \in \mathcal{T}} o(p)_{d,t} \leq C \quad \forall d \in \mathcal{D} \quad (4.28)$$

4.3 Célfüggvény

A célfüggvény definiálása során feltételezzük, hogy nem tudja az üzem teljesíteni a megrendeléseket, így mindenképp hiány keletkezik, melyhez hiányköltséget rendelünk. A modell futtatása során ennek a minimalizálása a cél, melyet a 4.29 formulával definiáltam.

$$\min \sum_{d \in \mathcal{D}} \sum_{t \in \mathcal{T}} LF_t \cdot u(p)_{d,t} \quad (4.29)$$

5. fejezet

Tesztelés

A 4. fejezetben javasolt megoldásom két különböző szemszögből teszteltem. Először az 5.1 alfejezetben mutatok be egy grafikus interfészt, melyet a tesztadatok felhasználóbarát megadására és a megoldás kényelmes futtatásának céljából implementáltam. Továbbá a saját tesztesetek meghatározását és az azokra kapott eredményeket az 5.2 alfejezetben ismertetem.

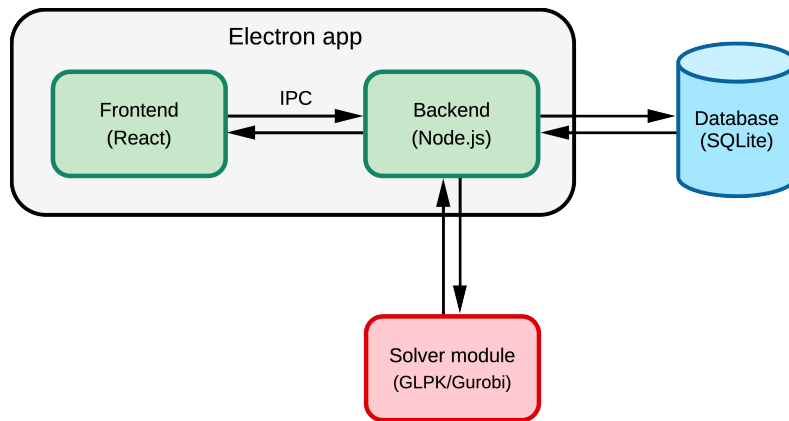
5.1 Tesztelés grafikus interfészen keresztül

A fejezetben bemutatásra kerül egy tesztelési környezet, melynek a szoftveres felépítését az 5.1.1 szövegrészben ismertetem, míg felhasználói szemszögből való használatát az 5.1.2 alfejezetben.

5.1.1 A szoftver architektúrája és felépítése

A szoftver implementálása során egy olyan felhasználóbarát interfész volt a célom, mely egy asztali számítógépen futtatható és egyszerű, hatékony kezelést biztosít. A megoldásnak a felhasználói felületen túlmenően funkciót kell biztosítania a felhasználó által bevitt adatok tárolására, illetve képes kell, hogy legyen futtatni a lineáris programozási modell megoldót. A szoftverrendszer architektúrája az 5.1 ábrán látható, mely a főbb komponenseket és azok kapcsolatát mutatja be.

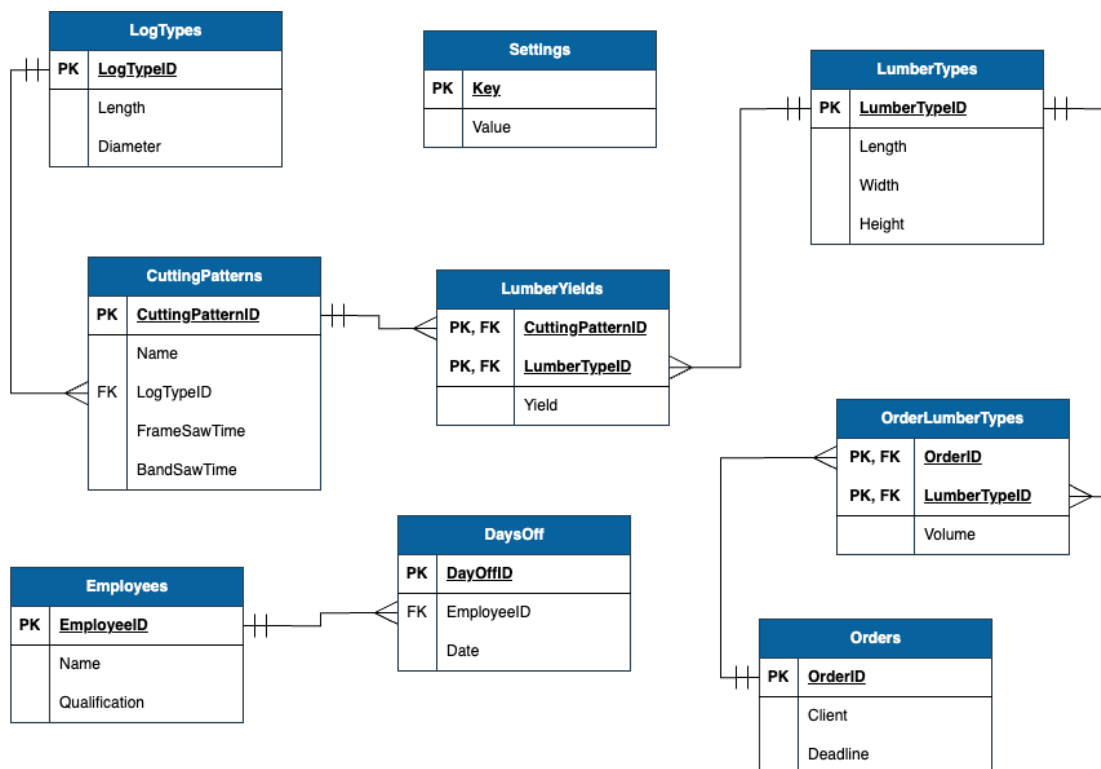
A grafikus interfész megvalósításához az Electron keretrendszert használtam, mely lehetővé teszi a modern webes technológiák alkalmazását asztali alkalmazások fejlesztésére. Az applikáció frontend részéhez React JavaScript-keretrendszert és Tailwind CSS nyílt-forráskódú CSS-keretrendszert használtam, míg a backend folyamatok implementálásához Node.js-t választottam. Ezen két komponens közti kommunikációt az úgynevezett IPC (Inter Process Communication) csatorna teszi lehetővé. A MILP modell futtatását egy Node.js által kezelt Python szkript végzi, melyet az ábrán "Solver module" elnevezéssel jelöltem. A szkript képes mind a GLPK, mind a Gurobi optimalizáló szoftver használatára.



5.1. ábra. Rendszerarchitektúra diagram

Ezen túlmenően a backend felelős az adatkezelési folyamatokért is, beleértve az adatok tárolását és lekérdezését.

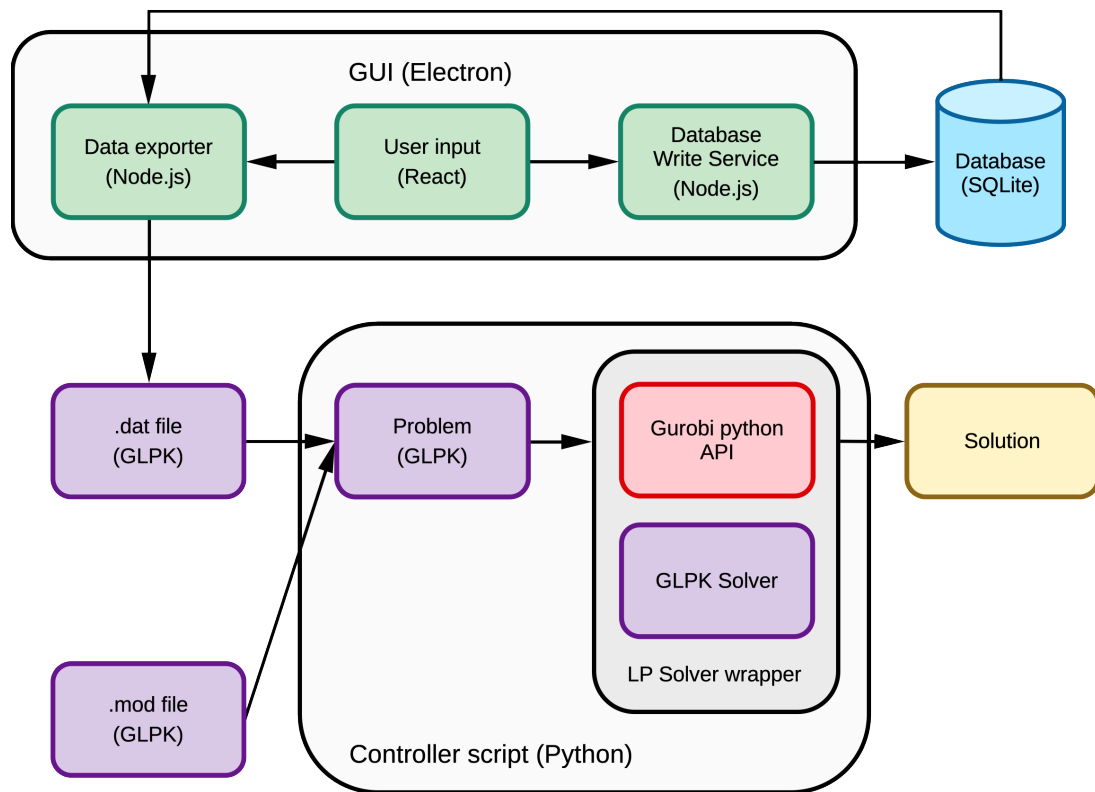
Az adattárolás egy SQLite adatbázisban történik, mely egy egyszerű és könnyen kezelhető megoldást nyújt az információk tárolására és kezelésére. Az adatok struktúráját és azok közötti kapcsolatokat az 5.2 ábrán látható egyed-kapcsolat diagram szemlélteti.



5.2. ábra. Egyed-kapcsolat diagram

A diagramon szereplő struktúra néhol más logikát követ, mint a 3.2 fejeztben definiált paraméterek. A tervezés során fő szempont volt, hogy felhasználói szemszögből rögzítsük az adatokat nem pedig a modell szempontjából. Ez többek között kiterjed a humán-

erőforrás menedzsmentre, ahol külön tároljuk az alkalmazottakat és a hozzájuk tartozó szabadságokat, vagy a keresletre, ahol külön megrendeléseket rögzítünk. Ezenfelül vannak olyan ütemezés specifikus információk, melyeket nem tárolunk el az adatbázisban, de modell futtatásához szükségesek, így a felületen keresztül ugyanúgy megadjuk azokat. Efféle funkcionalitások miatt egy komplexebb folyamatot kell kezelni a szoftver megoldásában, melyet az 5.3 ábra szemléltet.



5.3. ábra. A szoftver működésének folyamata

A folyamat diagrammon külön komponensként szerepel egy úgynevezett "exporter", mely adatbázisban tárolt adatokat és a felhasználói felületről származó információkat együttesen alakítja át a modell számára értelmezhető formátumra. Bizonyos adatokat az adatbázisból és bizonyos információkat felhasználói felületről közvetlenül konvertálja át a megfelelő adattípusra. A lineáris programozási modellt GMPL formátumban kezeljük, melyet az ábrán a "Problem" komponens reprezentál. A megoldás futtatását az "LP solver wrapper" doboz szemlélteti, mely magába foglalja a GLPK nyílt forráskódú megoldójának a lehetőségét, illetve döntést enged az Gurobi Python API használatára is. Ezt követően megoldó meghatároz egy megoldást, melyből a szoftver egy Gantt diagrammot készít a Python Matplotlib könyvtár alkalmazásával.

5.1.2 Szoftver használata

A szoftver lehetőséget biztosít, hogy az üzem ütemezéséhez szükséges adatokat különböző beviteli mezők segítségével megadhassuk és elmenthessük. Ennek eredménye képpen az alkalmazás újra indítását követően se vesznek el az előzőleg megadott adatok. A felület megjelenése az 5.4 és az 5.5 képernyőképen látható.

The screenshot shows the 'Sawmill Scheduler' application window. On the left is a sidebar with navigation options: 'Sawmill Scheduler', 'Sawmill Settings', 'Employees', 'Orders', and 'Scheduling'. The main area is divided into three sections: 'Logs', 'Lumbers', and 'Cutting Patterns'.

Logs: This section has input fields for 'Log' (value: 5) and 'Diameter (m)' (value: 0,5), followed by an 'Add' button. Below is a table with columns 'Length (m)', 'Diameter (m)', and 'Volume (m³)'. It contains one row with values 4.00, 0.50, and 0.79, respectively, with edit and delete icons on the right.

Lumbers: This section has input fields for 'Length (m)', 'Width (m)', and 'Height (m)', followed by an 'Add' button. Below is a table with columns 'Length (m)', 'Width (m)', 'Height (m)', and 'Volume (m³)'. It contains two rows: (4.00, 0.30, 0.20, 0.24) and (4.00, 0.40, 0.20, 0.32), each with edit and delete icons.

Cutting Patterns: This section has a 'New Cutting Pattern' button. Below is a table with columns 'Name' and 'Log'. It contains one row with 'CP' and 'Length 4m - Diameter 0.5m', with edit and delete icons.

5.4. ábra. Az adatrögzítés egy példája a grafikus felületen keresztül

A felület beviteli mezőit és a hozzá tartozó funkcióit külön csoportokba osztottam, melyek között a bal oldalon lévő menürendszer segítségével navigálhatunk, ezzel befolyásolva az alkalmazás jobb oldalán található tartalmat. Ezen csoport első pontja a "Sawmill settings", melyben a fűrészüzemhez tartozó alapvető információkat rögzíthetjük és szerkeszthetjük, ennek egy példáját az 5.4 képernyőkép szemlélteti. Ezen az oldalon többek között lehetőség van a rönkökre meghatározott vágási minták kezelésére is. Ilyen funkció megvalósítása látható az 5.5 képen, ahol a felhasználó egy új mintát vehet fel.

Ezenfelül elkülönítettem az alkalmazottakkal kapcsolatos funkciókat az "Employees" menüpont alatt és a kereslethez vagy rendelésekhez köthető beviteli mezőket az "Orders" fül alatt. Az utolsó menüponton keresztül pedig az ütemezés specifikus paramétereket határozhatjuk meg, majd futtathatjuk a bevitt adatokra a megoldást.

A megoldó futtatása eredményeképp születik egy Gantt diagram, melynek egy példája az 5.6 ábrán található. A diagramról leolvasható, mind a keretfűrész mind a szalagfűrész működésének ütemezése. Mindkét rönkvágó géphez tartozik egy-egy sáv, mely azt

Sawmill Scheduler

New Cutting Pattern

Name
Name

Log
Length 4m - Diameter 0.5m

Frame Saw Machine Cutting Time (mm:ss)
0 : 00

Band Saw Machine Cutting Time (mm:ss)
0 : 00

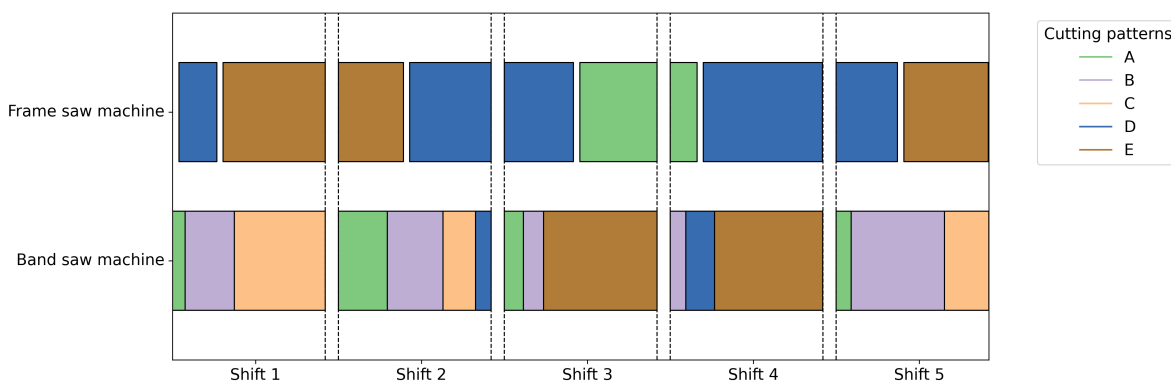
Lumber Yields

4m x 0.2m x 0.2m:	0	%
4m x 0.3m x 0.2m:	0	%
4m x 0.4m x 0.4m:	0	%
4m x 0.4m x 0.2m:	0	%

Save

5.5. ábra. Új vágási minta rögzítése a grafikus felületen

szemlélteti, hogy mikor melyik vágási mintával üzemeljen az adott eszköz. Ezek a sávok kisebb téglalapokra lettek bontva, melyek közötti fehér üres rész jelöli az üresjáratokat, illetve a vágási minták közti átállásokat. A mintákat különböző színekkel azonosíthatjuk be, ehhez egy segédlet a diagramm mellett található. A műszakokat az ábra x tengelyén különítettük el, melyek közti üres rész jelöli a napoknak azon szakaszát mikor nem üzemel a fűrészüzem.



5.6. ábra. Példa megoldás

5.2 Tesztelés ipari tapasztaltok alapján

5.2.1 A tesztelés környezete

Egy lineáris programozási modell megoldására számos szoftvercsomag létezik. Ilyen optimalizálási eszköz például a GLPK nyílt forráskódú megoldója. Azonban az iparban igen elterjedt a CPLEX, illetve a Gurobi szoftvere, melyek rendkívül összetettek, de jóval eredményesebbek a komplexebb modelleknél, mint az előbb említett GLPK. A javasolt modellt minden esetben a Gurobi 10.0.1-es verziójával oldottam meg, a Gurobi Python API-t használva. A modellt GMPL nyelven implementáltam, mely az A függelékben tekinthető meg. A megírt modellt Python alapú szoftverrel fordítottam lp típusú fájlra, melyet a Gurobi megoldóval futtattam. A Gurobi szoftver esetén megadható egy futási időkorlát, melyet 3600 másodpercre paramétereztem fel a tesztelés során. Előfordulhat olyan eset, mikor ezen az időkorláton belül nem sikerül megtalálni a megoldónak az optimális megoldást, ekkor az addig talált legjobb megoldás kerül meghatározásra, mint a leghatékonyabb megtalált ütemezés. Az eredményeket Python szkripttel dolgoztam fel, majd a Matplotlib külső könyvtár segítségével ábrázoltam. A teszteseteket egyetlen számítógépen futtattam, Apple M2 modellű processzorral és 16 GB-os memóriával.

5.2.2 Tesztadatok meghatározása

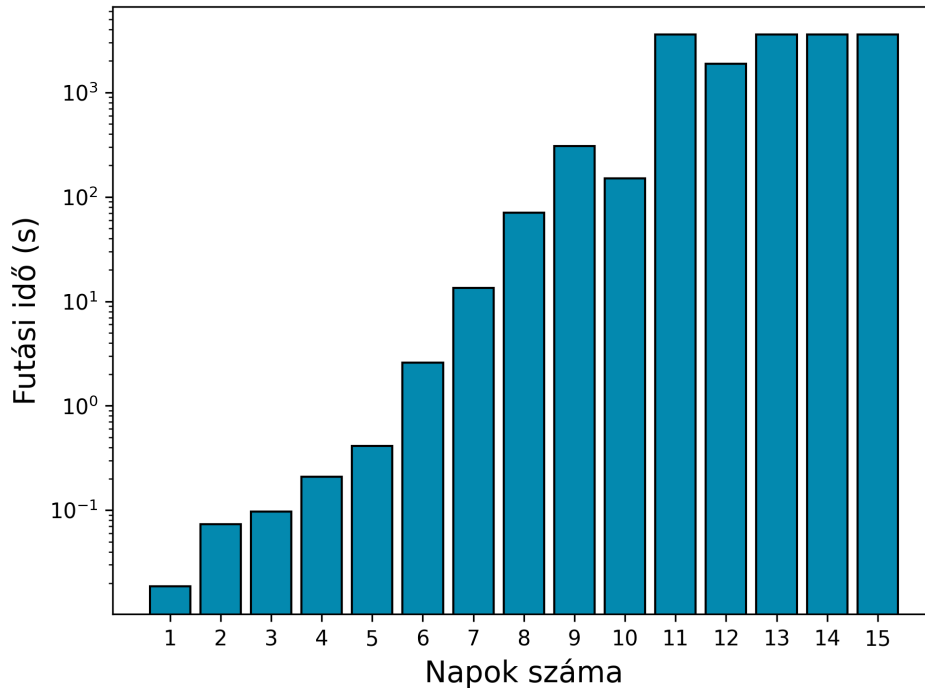
A tesztelés során saját teszteseteket kellett meghatároznom, mivel a munkám során vizsgált problémaosztályra tesztadatok sem a szakirodalomból, sem ipari partnertől nem álltak a rendelkezésemre. Annak érdekében, hogy a tesztesetek minél közelebb álljanak a valósághoz, faiparral foglalkozó szakemberekkel konzultáltam. Ezenfelül Hargitai László Fűrészáru [10] című könyvét használtam fel a reális adathalmaz deklarációjához. A tesztelés során az ipari tapasztalatok alapján meghatározott tesztadatokat konstansként rögzítettem, míg bizonyos paramétereket a tesztek specifikus igényeihez igazítottam.

5.2.3 Teszteredmények

Munkám során két fő teszteset került kidolgozásra. Először MILP modell hatékonyságának vizsgálatához annak gyorsaságát néztem meg. Ennek érdekében különböző tervezési időtávokat vizsgálva futtattam a megoldásom.

A modell futási sebességét 1-től egészen 15 napig terjedő tervezési időtávokra vizsgáltam, ahol a paraméterezés során a 3.2 definiált $\mathcal{D}$ halmaz méretét növeltem. Annak érdekében, hogy a napok számán túlmenően más tényező ne befolyásolja a feladat komplexitását, minden műszakra vonatkozóan megegyező keresletet és munkaerő paramétereket rögzítettem. További paramétereket nem befolyásol a napok halamaza, így a bemenet többi részét konstansként kezeltem.

A teszteredmények az 5.7 oszlop diagrammon láthatóak, ahol a magasságok jelölik a futási időt másodpercben mérve, melyet a különböző tervezési időtávok eredményeztek. A könnyebb értelmezhetőség érdekében az ábrázolás során logaritmikus skálázást használtam.

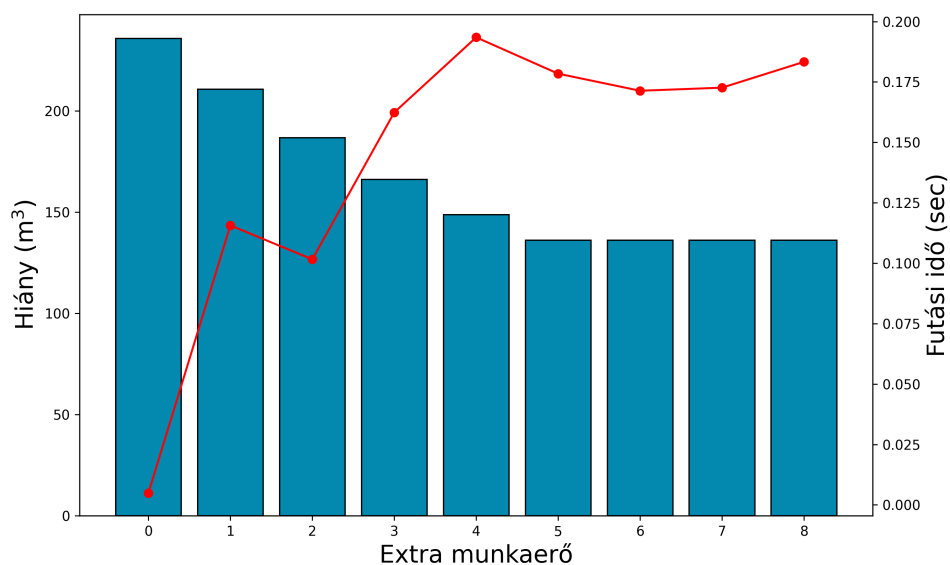


5.7. ábra. Futási idő eredmények több napos időszakokra

Az oszlop diagrammon jól látható, hogy a futási idők általában exponenciálisan növekednek a napok számának növelésével. Bizonyos nap számokra kivételek is megfigyelhetők, mivel előfordult olyan eset is, ahol gyorsabban talált megoldást a modell, mint az előző napoknál, melyet vélhetően a Gurobi szoftver heurisztikus megoldásai eredményezhetnek. Ezenfelül kiderül, hogy a modell 11 napos ütemezésig minden példát képes volt megoldani egy órán belül. A 11 napos ütemezésnél már azonban nem találta meg az optimumot, és az előre rögzített 3600 másodperces időkorlát miatt leállt a megoldó szoftvert.

A második tesztet során a munkaerőhiány hatását vizsgáltam. Megfogalmaztam egy olyan tesztet is, melyben a munkaerőhiányt szimuláltam a maximálisan rendelkezésre álló segéd munkások számának variálásával. A futtatások során a 3.2 fejezetben megfogalmazott X^{AW} paramétert növeltem 0-tól kezdődően. Így az első esetben nem engedélyeztük az üzem számára a plusz munkások behívását és az előre meghatározott beosztás alapján kellett ütemezni az alkalmazottakat. A tesztelés során azt vizsgáltam meg, hogy mennyivel jobb megoldás érhető el a munkaerőhiány csökkentésével. Az eredményeket az 5.8 Pareto-diagrammon ábrázoltam, ahol a kék oszlopok a célfüggvény értékét jelölik, azaz a termelésben keletkező hiány mértékét, míg a piros vonal a megoldások futási idejét

reprezentálja.



5.8. ábra. Extra munkaerő hatásai

Az ábráról leolvasható, hogy bizonyos mértékig javított az eredményeken az előzőekben megfogalmazott munkaerő befolyásolása mindaddig, amíg el nem értük azt az emberi erőforrás készletet, hogy minden műszakban minden gépet üzemeltetni tudjunk. Ez a vizsgált paraméterezés során 5 plusz segéd munkást jelentett. A futási időkön látható, hogy plusz számolási igényt jelentett a paraméter növelése, így növekedés azonosítható a paraméter növelésével, amíg el nem értük az előzőekben említett határt, mely után nem volt szignifikáns változás a megoldási időkre vonatkozóan sem.

6. fejezet

Konklúzió és a jövőbeli tervek

Dolgozatomban megvizsgáltam a fűrészüzemek ütemezését és annak több sajátosságát. Az irodalomban lévő tanulmányokban rengeteg megoldás létezik, és különféle kihívásokkal foglalkoznak. Munkám eredménye egy módszer, mely megoldást nyújt a fűrészüzemek ütemezésére olyan specifikusabb tényezőket figyelembe véve, mint a fűrészüzemekben alkalmazható rönkvágó technológiák közti különbségek. Ez magába foglal olyan kérdéseket, melyeket az eddigi szakirodalomban lévő megoldások általánosabban vagy egyáltalán nem is kezeltek. Ilyen például vágási minták közti átállások vagy adott gép üzemeltetéséhez szükséges humánerőforrás igény. A probléma megoldásához kidolgozásra került egy MILP modell, melyet több szempont szerint is teszteltem. Elkészült egy külön tesztelésre alkalmas környezet, mely magába foglalja a modell egyszerűbb futtatása érdekében implementált grafikus interfészt is. A tesztelés alatt különféle scenáriókat vizsgáltam, mint például a különböző tervezési időszakokra vonatkozó modell futási sebességének elemzése, valamint a munkaerőhiány hatásának vizsgálata a fűrészüzem ütemezési hatékonyságára.

A téma rengeteg lehetőséget rejthet még a jövőben, melyeket későbbi kutatásomban szeretném még részletesebben feltérképezni és körüljárni. Ennek érdekében további feladat specifikációkat szeretnék meghatározni. A kutatásom jövőbeli céljai közé tartozik konkrét ipari partnerrel való együttműködés, melynek keretében a modell további tesztelése és finomítása történne.

Néhány, már most felvetődött kiegészítés:

Melléktermékek Egy ilyen lehetséges bővítés lehet a fűrészelések során keletkező fűrészipari melléktermékek kezelése, mely mint második célfüggvény jelenhet meg a modellben.

Rönkvágási módok Az élesvágáson túlmenően sok más rönkvágási mód is létezik, mint például a prizmavágás, ahol egy rönk több fűrészelési lépést igényel, ezáltal az ütemezési feladatra is kihatással van.

Farönk tulajdonságok Fűrészelési sebességet, termékminőséget jelentősen befolyásolhatják a farönkök olyan fizikai tulajdonságai is, mint például a fafaj, szárazság, stb.

Kötelező karbantartás Gyártási gépek számára gyakran előírt a karbantartási munkák rendszeres elvégzése. Ezek figyelembevétele és megfelelő időzítése szintén fontos lehet egy ilyen ütemterv esetén.

Ezenfelül tovább bővíthető a feladatosztály több feldolgozási lépéssel, melyek alkalmazását egyaránt befolyásolhatják a farönkök különféle jellemzői. Az ilyesfajta továbbfejlesztések egy lényegesen komplexebb modellt eredményeznek, melynek a futási ideje számottevően nagyobb. Ennek érdekében célszerű olyan megoldást is kidolgozni, mely hatékonyabb futási idővel rendelkezik a komplexebb problémák esetén is, ilyen lehet egy heurisztikus algoritmus. Ez magával vonja, mint tesztelési szempont, az újonnan kidolgozott algoritmus és a már meglévő MILP modell összehasonlítását.

Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani témavezetőmnek, dr. Hegyháti Máténak a támogatásáért és szakmai tanácsaiért. Köszönettel tartozom továbbá a Soproni Egyetem és a szlovéniai InnoRenew CoE kutatóintézet munkatársainak, akik időt nem sajnálva az értékes szakmai tapasztalataikkal segítettek a munkámat. Továbbá szeretném megköszönni a dolgozatom elkészítésében nyújtott segítségét és támogatását családomnak és barátaimnak.

Végül szeretném megköszönni az alábbi ösztöndíj támogatását: A Kulturális és Innovációs Minisztérium ÚNKP-23-1-I-SOE-127 kódszámú Új Nemzeti Kiválóság Programjának a Nemzeti Kutatási, Fejlesztési és Innovációs Alapból finanszírozott szakmai támogatásával készült.



Irodalomjegyzék

- [1] Pamela P Alvarez, Alejandra Espinoza, Sergio Maturana, and Jorge Vera. Improving consistency in hierarchical tactical and operational planning using robust optimization. *Computers & Industrial Engineering*, 139:106112, 2020.
- [2] Pamela P Alvarez and Jorge R Vera. Application of robust optimization to the sawmill planning problem. *Annals of Operations Research*, 219:457–475, 2014.
- [3] Dimitris Bertsimas and Melvyn Sim. The price of robustness. *Operations research*, 52(1):35–53, 2004.
- [4] Diego Broz, Nicolás Vanzetti, Gabriela Corsano, and Jorge M Montagna. Goal programming application for the decision support in the daily production planning of sawmills. *Forest Policy and Economics*, 102:29–40, 2019.
- [5] Celso Carnieri and Guillermo A Mendoza. A fractional algorithm for optimal cutting of lumber into dimension parts. *Annals of Operations Research*, 95:83–92, 2000.
- [6] Balázs Dávid, Olivér Ősz, and Máté Hegyháti. Robust scheduling of waste wood processing plants with uncertain delivery sources and quality. *Sustainability*, 13(9):5007, 2021.
- [7] Luca Di Gaspero, Andrea Schaerf, Marco Cadoli, Wolfgang Slany, and Moreno Falaschi. Local search techniques for scheduling problems: algorithms and software tool. Forum, 2003.
- [8] Christodoulos A Floudas and Xiaoxia Lin. Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. *Computers & Chemical Engineering*, 28(11):2109–2129, 2004.
- [9] Mario J Gonzalez Jr. Deterministic processor scheduling. *ACM Computing Surveys (CSUR)*, 9(3):173–204, 1977.
- [10] László Hargitai. *Fűrészáru*. Szaktudás Kiadó Ház Rt., Budapest, 2003.
- [11] Iiro Harjunoski, Christos T Maravelias, Peter Bongers, Pedro M Castro, Sebastian Engell, Ignacio E Grossmann, John Hooker, Carlos Méndez, Guido Sand, and

- John Wassick. Scope for industrial applications of production scheduling models and solution methods. *Computers & Chemical Engineering*, 62:161–193, 2014.
- [12] Ignacio Hinostroza, Lorena Pradenas, and Víctor Parada. Board cutting from logs: Optimal and heuristic approaches for the problem of packing rectangles in a circle. *International Journal of Production Economics*, 145(2):541–546, 2013.
- [13] John Lancaster and Mustafa Ozbayrak. Evolutionary algorithms applied to project scheduling problems—a survey of the state-of-the-art. *International journal of production research*, 45(2):425–450, 2007.
- [14] Li Luo, Jim O’Hehir, Courtney M. Regan, Li Meng, Jeffery D. Connor, and Christopher W.K. Chow. An integrated strategic and tactical optimization model for forest supply chain planning. *Forest Policy and Economics*, 131:102571, 2021.
- [15] Sergio Maturana, Enzo Pizani, and Jorge Vera. Scheduling production for a sawmill: A comparison of a mathematical model versus a heuristic. *Computers & Industrial Engineering*, 59(4):667–674, 2010.
- [16] CA Méndez, GP Henning, and J Cerdá. An milp continuous-time approach to short-term scheduling of resource-constrained multistage flowshop batch facilities. *Computers & Chemical Engineering*, 25(4-6):701–711, 2001.
- [17] Carlos A Méndez, Jaime Cerdá, Ignacio E Grossmann, Iiro Harjunkski, and Marco Fahl. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers & chemical engineering*, 30(6-7):913–946, 2006.
- [18] Khaled Mesghouni, Slim Hammadi, and Pierre Borne. Evolutionary algorithms for job-shop scheduling. *International Journal of Applied Mathematics and Computer Science*, 14(1):91–104, 2004.
- [19] Josefa Mula, Raul Poler, Jose P García-Sabater, and Francisco Cruz Lario. Models for production planning under uncertainty: A review. *International journal of production economics*, 103(1):271–285, 2006.
- [20] Luis G Occena and Daniel L Schmoldt. Grasp: a prototype interactive graphic sawing program. *Forest Products Journal*, 46(11/12):40, 1996.
- [21] Olivér Osz. *Methods for scheduling industrial processes*. PhD thesis, Széchenyi István University, Hungary, 2022.
- [22] Olivér Ósz, Balázs Dávid, József Garab, and Máté Hegyháti. Long-term plant-level scheduling with uncertainty in the plywood industry. 2022.

- [23] Olivér Ósz, Dominic CY Foo, and Máté Hegyháti. Minimizing freshwater usage in batch process scheduling: S-graph approach. *Process Integration and Optimization for Sustainability*, 5(1):31–42, 2021.
- [24] Michael Pinedo. *Overview of Stochastic Scheduling Problems*, pages 607–610. 01 2012.
- [25] E Sanmarti, F Friedler, and L Puigjaner. Combinatorial technique for short term scheduling of multipurpose batch plants based on schedule-graph representation. *Computers & chemical engineering*, 22:S847–S850, 1998.
- [26] E Sanmarti, L Puigjaner, T Holczinger, and F Friedler. Combinatorial framework for effective scheduling of multipurpose batch plants. *AIChE Journal*, 48(11):2557–2570, 2002.
- [27] CL Todoroki and EM Rönnqvist. Combined primary and secondary log breakdown optimisation. *Journal of the Operational Research Society*, 50(3):219–229, 1999.
- [28] Edward Tsang and Chris Voudouris. Fast local search and guided local search and their application to british telecom’s workforce scheduling problem. *Operations Research Letters*, 20(3):119–127, 1997.
- [29] Nicolás Vanzetti, Diego Broz, Gabriela Corsano, and Jorge M Montagna. An optimization approach for multiperiod production planning in a sawmill. *Forest Policy and Economics*, 97:1–8, 2018.
- [30] Mauricio Varas, Sergio Maturana, Rodrigo Pascual, Ignacio Vargas, and Jorge Vera. Scheduling production for a sawmill: A robust optimization approach. *International Journal of Production Economics*, 150:37–51, 2014.
- [31] Andrés Weintraub and Carlos Romero. Operations research models and the management of agricultural and forestry resources: a review and comparison. *Interfaces*, 36(5):446–457, 2006.
- [32] Matthew F Winn, Randolph H Wynne, Philip A Araman, et al. Alog: A spreadsheet-based program for generating artificial logs. *Forest products journal*, 54(1):62–66, 2004.
- [33] Masoumeh Kazemi Zanjani, Daoud Ait-Kadi, and Mustapha Nourelfath. Robust production planning in a manufacturing environment with random yield: A case in sawmill production planning. *European Journal of Operational Research*, 201(3):882–891, 2010.

A. függelék

GMPL kód

```
set Logs;
param logVolume{Logs};

set Lumbers;
set AllCuttingPatterns;
param logParam{AllCuttingPatterns} symbolic in Logs;
set CuttingPatterns{l in Logs} := setof{cp in AllCuttingPatterns:
    logParam[cp] == l} cp;

param maxExtraAssistantWorker;

param frameSawCuttingTime{AllCuttingPatterns};
param bandSawCuttingTime{AllCuttingPatterns};
param frameSawSpecialistRequirement;
param frameSawAssistantWorkerRequirement;
param bandSawSpecialistRequirement;
param bandSawAssistantWorkerRequirement;
param frameSawSetupTime;
param numberOfDays;
set Days := 1..numberOfDays;
param shiftLength;
param SpecialistAvailability{Days};
param AssistantWorkerAvailability{Days};
param logInventory{Logs};
param demand{Lumbers, Days};
param yield{Lumbers, AllCuttingPatterns};
param extraCost{Lumbers} default 1;
```

```

param storageCapacity;

var frameSawBefore{Days, AllCuttingPatterns} binary;
var frameSawAfter{Days, AllCuttingPatterns} binary;
var changeTime{Days} >= 0, <= shiftLength-frameSawSetupTime;
var frameSawWorking{Days} binary;
var bandSawWorking{Days} binary;
var extraAssistantWorker{Days} >= 0;
var countFrameSawBefore{Days, AllCuttingPatterns} >= 0, integer;
var countFrameSawAfter{Days, AllCuttingPatterns} >= 0, integer;
var countBandSaw{Days, AllCuttingPatterns} >= 0, integer;
var production{day in Days, lumber in Lumber} >= 0;
var shortage{day in Days, lumber in Lumber} >= 0;
var storage{day in Days union {0}, lumber in Lumber} >= 0;

s.t. InitializeStorage{lumber in Lumber}:
    storage[0, lumber] = 0;

s.t. FrameSawWorkingBeforeConstraint{day in Days}:
    sum{cp in AllCuttingPatterns} frameSawBefore[day, cp] =
        frameSawWorking[day];

s.t. FrameSawWorkingAfterConstraint{day in Days}:
    sum{cp in AllCuttingPatterns} frameSawAfter[day, cp] =
        frameSawWorking[day];

s.t. NoChangeIfBothDaysWork1{day in Days, cp in AllCuttingPatterns: day
    != numberOfDays}:
    frameSawAfter[day, cp] >= frameSawBefore[day+1, cp] - 1 * (2 -
        frameSawWorking[day] - frameSawWorking[day + 1]);

s.t. NoChangeIfBothDaysWork2{day in Days, cp in AllCuttingPatterns: day
    != numberOfDays}:
    frameSawAfter[day, cp] <= frameSawBefore[day+1, cp] + 1 * (2 -
        frameSawWorking[day] - frameSawWorking[day + 1]);

s.t. BeforeChangeCount1{day in Days, cp in AllCuttingPatterns}:
    countFrameSawBefore[day, cp] <= changeTime[day] /
        frameSawCuttingTime[cp] + shiftLength / frameSawCuttingTime[cp] *

```

```

(1 - frameSawBefore[day, cp]);

s.t. BeforeChangeCount2{day in Days, cp in AllCuttingPatterns}:
    countFrameSawBefore[day, cp] <= shiftLength / frameSawCuttingTime[cp]
        * frameSawBefore[day, cp];

s.t. AfterChangeCount1{day in Days, cp in AllCuttingPatterns}:
    countFrameSawAfter[day, cp] <= (shiftLength - changeTime[day] -
        frameSawSetupTime * (1-frameSawBefore[day, cp])) /
        frameSawCuttingTime[cp] + shiftLength / frameSawCuttingTime[cp] *
        (1 - frameSawAfter[day, cp]);

s.t. AfterChangeCount2{day in Days, cp in AllCuttingPatterns}:
    countFrameSawAfter[day, cp] <= shiftLength / frameSawCuttingTime[cp]
        * frameSawAfter[day, cp];

s.t. MaxBandSawCutting{day in Days}:
    sum{cp in AllCuttingPatterns} countBandSaw[day, cp] *
        bandSawCuttingTime[cp] <= shiftLength * bandSawWorking[day];

s.t. SpecialistAvailabilityConstraint{day in Days}:
    frameSawSpecialistRequirement * frameSawWorking[day] +
        bandSawSpecialistRequirement * bandSawWorking[day] <=
        SpecialistAvailability[day];

s.t. ExtraWorkerAvailability{day in Days}:
    frameSawAssistantWorkerRequirement * frameSawWorking[day] +
        bandSawAssistantWorkerRequirement * frameSawWorking[day] <=
        AssistantWorkerAvailability[day] + extraAssistantWorker[day] +
        (SpecialistAvailability[day] - frameSawSpecialistRequirement *
        frameSawWorking[day] + bandSawSpecialistRequirement *
        bandSawWorking[day]);

s.t. MaxExtraWorkersHired:
    sum{day in Days} extraAssistantWorker[day] = maxExtraAssistantWorker;

s.t. NextDayStorageLimit{day in Days}:
    sum{lumber in Lumbers} storage[day, lumber] <= storageCapacity;

```

```

s.t. MaxInventory{l in Logs}:
    sum{day in Days, cp in AllCuttingPatterns: logParam[cp]==1}
        (countFrameSawBefore[day, cp] + countFrameSawAfter[day, cp] +
         countBandSaw[day, cp]) <= logInventory[l];

s.t. ProductionCalculation{day in Days, lumber in Lumbers}:
    production[day, lumber] = sum{cp in AllCuttingPatterns} yield[lumber,
        cp] * logVolume[logParam[cp]] * (countFrameSawBefore[day, cp] +
        countFrameSawAfter[day, cp] + countBandSaw[day, cp]);

s.t. ShortageCalculation{day in Days, lumber in Lumbers}:
    production[day, lumber] + storage[day-1, lumber] - storage[day,
        lumber] + shortage[day, lumber] >= demand[lumber, day];

minimize ShortageCost:
    sum{day in Days, lumber in Lumbers} extraCost[lumber] * shortage[day,
        lumber];

```
