Bacheloroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for økonomi
Institutt for industriell økonomi og teknologiledelse

Blikås, Mikael Hatlenes
Pedersen, Sigmund
Vassbø, Lars Isachsen

Bruk av Diskret Hendelsessimulering for å Optimalisere Ytelsen av et Miniload Automatisk Lagring og Plukk System hos ASKO Sentrallager Kjøl AS

Application of Discrete Event Simulation to Optimize the Efficiency of a Miniload Automated Storage and Retrieval System at ASKO Sentrallager Kjøl AS

Bacheloroppgave i Logistikkingeniør

NTNU
Norwegian University of
Science and Technology

Blikås, Mikael Hatlenes
Pedersen, Sigmund
Vassbø, Lars Isachsen

Bruk av Diskret Hendelsessimulering for å Optimalisere Ytelsen av et Miniload Automatisk Lagring og Plukk System hos ASKO Sentrallager Kjøl AS

Bacheloroppgave i Logistikkingeniør
Veileder: Dr. Alireza Ashrafian, PhD
Mai 2021

Norges teknisk-naturvitenskapelige universitet
Fakultet for økonomi
Institutt for industriell økonomi og teknologiledelse

NTNU
Fakultet for økonomi
Institutt for industriell økonomi og teknologiledelse

Bacheloroppgave TLOG3001

Tittel (norsk og engelsk): Bruk av Diskret Hendelsessimulering for å Optimalisere Ytelsen av et Miniload Automatisk Lagring og Plukk System hos ASKO Sentrallager Kjøl AS Application of Discrete Event Simulation to Optimize the Efficiency of a Miniload Automated Storage and Retrieval System at ASKO Sentrallager Kjøl AS	Prosjektnr.: 002
Forfattere: Blikås, Mikael Hatlenes Pedersen, Sigmund Vassbø, Lars Isachsen	Dato: 20.05.2021 Gradering: Åpen
Studieretning: Logistikkingeniør	
Veileder internt: Dr. Alireza Ashrafian, PhD	
Oppdragsgiver: ASKO Sentrallager Kjøl AS	
Oppdragsgivers kontaktperson: Freyja Medewitz	

Sammendrag:
Analysering og optimalisering av helautomatisk lager- og plukksystem hos ASKO Sentrallager Kjøl AS. Bruk av diskret hendelsessimulering for å undersøke hvordan logiske parametere påvirker vareflyt og ytelse til deler av lageret.

Stikkord: Diskret Hendelsessimulering, Lager Logistikk, Automatlager, FEFO	Keywords: Discrete Event Simulation, Warehouse Logistics, Automated Warehouse, FEFO

Forord

Denne bacheloroppgaven er skrevet av Mikael Hatlenes Blikås, Sigmund Pedersen og Lars Isachsen Vassbø, og er vår avsluttende oppgave for bachelorstudiumet Logistikkingeniør, ved NTNU.

Det 3-årig studiumet har gitt oss forutsetninger for å bli gode problemløsere, som har vært en nødvendighet i denne oppgaven. Gjennom Lean produksjon og kvalitetsstyring, Operasjonsanalyse, og Logistikkteknologi og digitalisering, har vi lært oss et «mindset», fått forståelse for digitalisering og fått testet ut fremtidens forbedringsverktøy, 3D-simulering.

I vår oppgave er det analysert produksjonsdata, og laget en 3D-simulering av deler av ASKO Sentrallager Kjøl AS sitt anlegg på Vestby. Dette for å bedømme utfallet av endringer i logiske parametere, og gjennom analyser avdekke mulige flaskehalser.

Vi ønsker å rette en takk til alle som har bidratt til denne oppgaven fra ASKO Sentrallager Kjøl AS, spesielt Frejya Medewitz, Eivind Johan Folland, Ove-Eirik Wang Krogstad, og WITRON ansatt Chris Todler. Vi ønsker å rette en takk til Christian Wordenskjold Nørregaard som kom med innspill for hvordan vi kunne optimalisere hastigheten på 3D-simuleringen. Til slutt vil vi gi en stor takk til vår veileder ved NTNU, Alireza Ashrafian, for gode innspill, konstruktive tilbakemeldinger og veiledning med oppgaven.

Dato/ Mikael Hatlenes Blikås:

20/05 Mikael Blikås

Dato/ Sigmund Pedersen:

20/05 Sigmund Pedersen

Dato/ Lars Isachsen Vassbø:

20/05

Sammendrag

ASKO Sentrallager Kjøl AS distribuerer av kjølevarer, altså produkter som oppbevares ved 0 til 4 ºC. Lageret er nesten helautomatisert, og automatiseringsteknologien er levert av WITRON AS. Dette lageret distribuerer cirka 70% av alle kjølevarene som ASKO Norge AS i sin helhet distribuerer til sine sluttkunder, og rundt 25% av alle kjølevarene i Norge går innom dette sentrallageret. Lageret består av to hovedområder for plukking som omhandler automatisk og dynamisk plukk.

Det automatiske plukkområdet, Order Picking Machinery (OPM), består av komponenter som automatisk håndterer kolli, Automated Storage and Retrieval System (ASRS), og som automatisk stabler kolli på lastbærer, Case Order Machine (COM). I OPM har ASKO Sentrallager Kjøl AS observert at ved enkelte tilfeller venter COM med å stable kolli på lastbærer, da ikke får levert kolli. Det er ASRS som leverer kolli til COM og på grunn av ventingen i COM mistenkes det at ASRS er en flaskehals.

Det er implementert en prioriteringslogikk for hvilke kolli som skal plukkes først i ASRS, en logikk som bestemmer at kolli med tidligst holdbarhet plukkes først, First Expired, First Out (FEFO). Kolli som plukkes på grunn av FEFO må flyttes mellom forskjellige ASRS og dette tar kapasitet. Det er derfor laget en styringsparameter for om kolli skal flyttes kun ved hjelp av ASRS-kraner eller om kolli skal ta en alternativ rute. ASKO Sentrallager Kjøl AS ønsker å se på hvordan FEFO og endringer i styringsparameteren påvirker kapasitetsutnyttelsen i OPM.

Gjennom denne rapporten vil kapasitetsutnyttelsen av dagens system bli analysert og ved hjelp av 3D-simulering vil endringer i prioriteringslogikk og styringsparameteren bli utforsket.

Abstract

ASKO Sentrallager Kjøl AS distributes refrigerated goods, which are products that are kept at 0 to 4 °C. The warehouse is almost fully automated, where the automation technology is delivered by WITRON AS. This warehouse distributes approximately 70% of all the refrigerated goods that ASKO Norge AS delivers to its customers. Approximately 25% of all refrigerated goods distributed throughout Norway goes through this warehouse.

The automatic picking area, Order Picking Machinery (OPM), consist of components that can handle goods automatically, Automated Storage and Retrieval System (ASRS), and can automatically pick and place goods on a transport unit, Case Order Machine (COM). ASKO Sentrallager Kjøl AS has noticed that on some occasions, goods are not delivered to COM, and therefore it must wait before it can continue the picking process. Since ASRS delivers goods to COM, and because COM occasionally waits, ASRS is suspected to be the bottleneck.

There is implemented a prioritization logic for which products whom should be picked first in ASRS. A logic that determines that the product with the lowest expiry date should be picked first, First Expired, First Out (FEFO). Goods whom must be picked because of FEFO needs to be relocated between different ASRS and this reduces the available capacity from the ASRS-vehicles. As a result of this there is implemented a control parameter that determines whether goods should be relocated between ASRS-vehicles or through an alternative route. ASKO Sentrallager Kjøl AS wishes to analyze how FEFO and changes in the control parameter affect the performance of OPM .

Throughout this thesis the performance of OPM will be analyzed and with the help of 3D simulation, changes in the prioritization logic and control parameter will be explored.

Definisjoner

COM	Case Order Machine
OPM	Order Picking Machinery
ASRS	Automatic Storage and Retrieval System
SKU	Stock Keeping Unit – Artikkelnummer
ASKO SL Kjøl	ASKO Sentrallager Kjøl AS
FEFO	First Expired, First Out
Input	Innkommende kolli
Output	Utgående kolli

Innholdsfortegnelse

1.0 Innledning

1.1 Bakgrunn for oppgaven

Den 25 januar i år deltok vi, sammen med vår veileder Dr. Alireza Ashrafian i et første møte med representanter fra oppdragsgiver, Asko Sentrallager Kjøl AS (heretter ASKO SL Kjøl). ASKO SL Kjøl var representert av Freyja Medewitz, Ove-Eirik Wang Krogstad og Thomas Skjerden. På møtet presenterte de ulike problemstillinger og mulige forbedringspotensialer i forbindelse med drift av lageret sitt.

I uken som fulgte bestemte vi oss for å sette søkelys på en bestemt del av lageret, nemlig det helautomatiske «Order Picking Machinery» området (heretter OPM). OPM tar imot helpaller og omlaster disse etter kundeordre. OPM består i hovedsak av et stort brettlager der individuelle kolli lagres på brett til det sendes ut til en plukkmaskin. Etter flere møter med ASKO SL Kjøl og Dr. Alireza Ashrafian kom vi i felleskap frem til en problemstilling og en gjennomføringsplan for prosjektet.

Det automatiserte kjølelageret på Vestby ble åpnet i januar 2017, og har siden vært i kontinuerlig drift. Etter litt tid med innkjøring og normal drift har ASKO SL Kjøl jobbet kontinuerlig med å identifisere tiltak som kan gi en mer effektiv og sikker drift. ASKO SL Kjøl har gjennom dette arbeidet identifisert flere mulige flaskehalser og kapasitetsbegrensninger, men har ikke hatt ressurser til å undersøke dette nærmere. Brettlageret i OPM er en mulig flaskehals da enkelte komponenter videre i produksjonskjeden ofte blir stående å vente. Brettlageret er den delen som tar for seg lagring og henting av kolli lagret på brett ut til plukkmaskinene. Brettlageret er et Automated Storage and Retrieval System (heretter ASRS). ASRS er delt opp i 18 soner og tilhørende plukkmaskiner.

I OPM er det implementert en prioriteringslogikk basert på First Expired, First Out (heretter FEFO). Denne logikken bestemmer at brett, med kolli, som har tidligst utløpsdato skal plukkes først. Dette er en viktig logikk som bidrar til minst mulig svinn som følge av utgått holdbarhetsdato, men som ifølge ASKO SL Kjøl, også kan begrense ytelsen i plukkområdet.

FEFO-logikken krever at den laveste holdbarhetsdatoen plukkes først. For at denne logikken skal følges er det et behov for å reallokere brett mellom ASRS-soner. Reallokering er at brett må flyttes til en annen ASRS-sone, der ASRS-kranen utfører denne operasjonen.

ASKO SL Kjøl tror at FEFO-logikken og reallokeringer påvirker handlingsmønsteret til ASRS-kranene, og at dette kan redusere tiden tilgjengelig for Input og Output. Denne prioriteringslogikken, og måten kolli blir reallokert i lageret er noen av områdene ASKO SL Kjøl ønsker å se nærmere på i håp om å avdekke muligheter for forbedring.

1.2 Overordnet problemstilling

ASKO SL Kjøl ønsker å undersøke mulige flaskehalser og kapasitetsbegrensninger i lageret sitt. Gjennom møter med ASKO SL Kjøl har det blitt bestemt at denne oppgaven skal fokusere på utnyttelsen av det helautomatiske plukkområdet i lageret, OPM.

Produksjonsdata viser at COM-ene sjeldent plukker på maksimal kapasitet, og mistenker at dette forårsakes av flaskehalser tidligere i produksjonskjeden. ASRS er leddet i produksjonskjeden som er før COM, og ASKO SL Kjøl mistenker at enkelte logiske parametere kan være kapasitetsbegrensende for utnyttelsen av ASRS.

FEFO-logikken forårsaker reallokeringer som krever både tid og tilgjengelighet fra ASRS-kranene. Det er derimot uvisst i hvilken grad FEFO påvirker utnyttelsen av ASRS-kranene, og om det er noen gode løsninger for å redusere innvirkningen FEFO har på denne utnyttelsen. ASRS har også en styringsparameter for hva regnes som lange reallokeringer. Korte reallokeringer gjennomføres ved at ASRS-kranene sender kolli på et reallokeringsbånd som er plassert mellom hver ASRS-kran. Lange reallokeringer gjennomføres ved at ASRS-kranene sender kolli ut gjennom utgående bånd fra ASRS, hvor kolli vil ta en alternativ rute som fører til starten på inngående samlebånd i OPM. I dag regnes en reallokering som lang dersom kolliet må innom 8 ASRS-kraner eller mer for å komme til destinasjonen. ASKO SL Kjøl ønsker å undersøke hvordan endringer i styringsparameteren for lange reallokeringer påvirker utnyttelsen av ASRS.

Gjennom analyser av produksjonsdata og diskret hendelsessimulering vil denne oppgaven avdekke hvordan logiske parametere for ASRS påvirker ytelsen til OPM.

2.0 Automatlager

Dette kapittelet tar for seg hva et automatlager er. Det vil gi en grunnleggende forståelse av hvordan komponenter og system hos ASKO SL Kjøl fungerer.

Lager har en viktig rolle i verdikjeden, men bidrar derimot lite til direkte verdiskapning. Generelt sett er oppgavene til et lager å kunne ta imot, lagre, og sende ut produkt ved behov. Den viktigste rollen et lager har er evnen til å møte etterspørsel på kort varsel. Problemet er at i de fleste tilfeller krever dette en høy varebeholdning, noe som er kostbart, både i form av høy kapitalbinding, samt drift av vareflyten og bygget. For å kunne redusere disse kostnadene er automatisering av prosesser[1] en løsning.

Det er forskjellige typer prosesser i et lager med ulike krav for automatisering. Formålet er derimot alltid det samme (Ten Hompel & Schmidt, 2007, p. 137):

- Redusere omlastningstiden
- Redusere ledetiden
- Kunne overholde frister (deadlines)
- Opprettholde kontinuerlig kvalitet av produkt og produksjon
- Redusere eller eliminere gjentagende og anstrengende prosesser for ansatte
- Redusere driftskostnader
- Øke systemets totale ytelse

I et tradisjonelt lager er de fleste prosesser manuelle; fra inngående varer, lagring og utgående varer. Manuell arbeidskraft kan gjøre et lager mindre effektivt og ha høyere kostnader. Det er også en større risikofaktor rundt dette, både rundt håndtering av gods og for personell som utfører jobben. Ved å automatisere en større grad av disse prosessene vil en kunne redusere risikoen for feil og en kan effektivisere og optimalisere driften.

Automatiseringsteknologien gjør at lager kan benytte seg av roboter, maskiner og datasystemer som tar over prosessene og driften som tidligere har blitt gjennomført manuelt. Dette kan bidra til å redusere eller eliminere prosesser. Håndtering av materialflyten og

[1] En prosess defineres som en aktivitet i et system hvor materialer, energi eller informasjon blir reformert, transportert eller lagret. (Ten Hompel & Schmidt, 2007, p. 139)

informasjonsflyten til lageret ved hjelp denne teknologien vil en kunne oppnå en mer jevn og effektiv flyt, og færre risikofaktorer innenfor lagring.

Automatlager har en høy areal- og volumutnyttelse, og operer nøyaktig og hurtig døgnet rundt, dette ved hjelp av de datastyrte maskinene. Denne nøyaktigheten medfører færre feil, og dersom feil oppstår vil datasystemet kunne varsle om dette. Et automatlager reduserer behovet for antall lagerarbeidere som er nødvendig for driften, som i tur minsker risikoen for skade på personell i lageret. Det trengs kun operatører for å overvåke driften og noen servicearbeidere som kan rette opp feil. Dette resulterer i lavere totalkostnader, økt kapasitet og sikkerhet, samt bedre forutsetninger for å kunne levere et høyt volum av produkter innenfor tidsfrister. Implementering av et automatlager er en kostbar investering, men ettersom det reduserer bemanning og gir økt kapasitetsutnyttelse vil det på sikt øke inntjeningen. Det vil også gi et konkurransefortrinn ovenfor andre aktører ved at driften blir optimalisert, effektiv og nøyaktig.

Det finnes mange forskjellige automatlagre og enda flere forskjellige teknologier for drift av dette. Alt fra roboter som flytter på lastbærere automatisk og autonomt, til maskiner som automatisk plukker kolli på pall. Hvilken teknologi og utstyr som brukes er avhengig av hvilke arbeidsoppgaver som skal gjennomføres. Med dagens teknologi er det mulig å helautomatisere lagerdriften, ved hjelp av roboter og datamaskiner som snakker sammen.

I et automatlager har alle komponenter mulighet til å snakke sammen og samhandle, ved hjelp av et styringsprogram som holder oversikt over hele driften. Det er essensielt å kombinere et slikt program med maskinene for å oppnå best mulig utnyttelse av driften. Dette tar over mange av de manuelle operasjonene, som å ta valg og styre vareflyten.

Det finnes tre typer lager; produksjons-, kontrakts-, og distribusjonslager (van den Berg & Zijm, 1999, p. 520). Produksjonslager håndterer råvarer, påbegynt- og ferdig-produkt i en produksjon. Dette opptrer som en mellomlagring mellom prosesser i en produksjon. Kontraktslager Outsourcer lagertjenester til en eller flere aktører ved å tilby lagertjenester. Distribusjonslager er lager som tar imot produkt fra et spekter leverandører og distribuerer dette ut til neste ledd i verdikjeden. Et distribusjonslager kan også inneha funksjonen med å omlaste lastbærere fra å inneholde et produkt til å inneholde mange, fra Single Pallet til Mixed Pallet. Når lager nevnes heretter er det distribusjonslager det refereres til, ettersom at ASKO SL Kjøl er et distribusjonslager.

2.1 WMS – Warehouse Management System

Warehouse Management System er en programvare for å administrere og kontrollere vareflyten i et lager. Systemet holder oversikt over vareflyten og alle prosessene som skjer med enhver komponent og et hvert produkt. Systemet er i stand til å drifte et automatlager og det vil kunne ta beslutninger og valg rundt aktiviteter som skal forekomme.

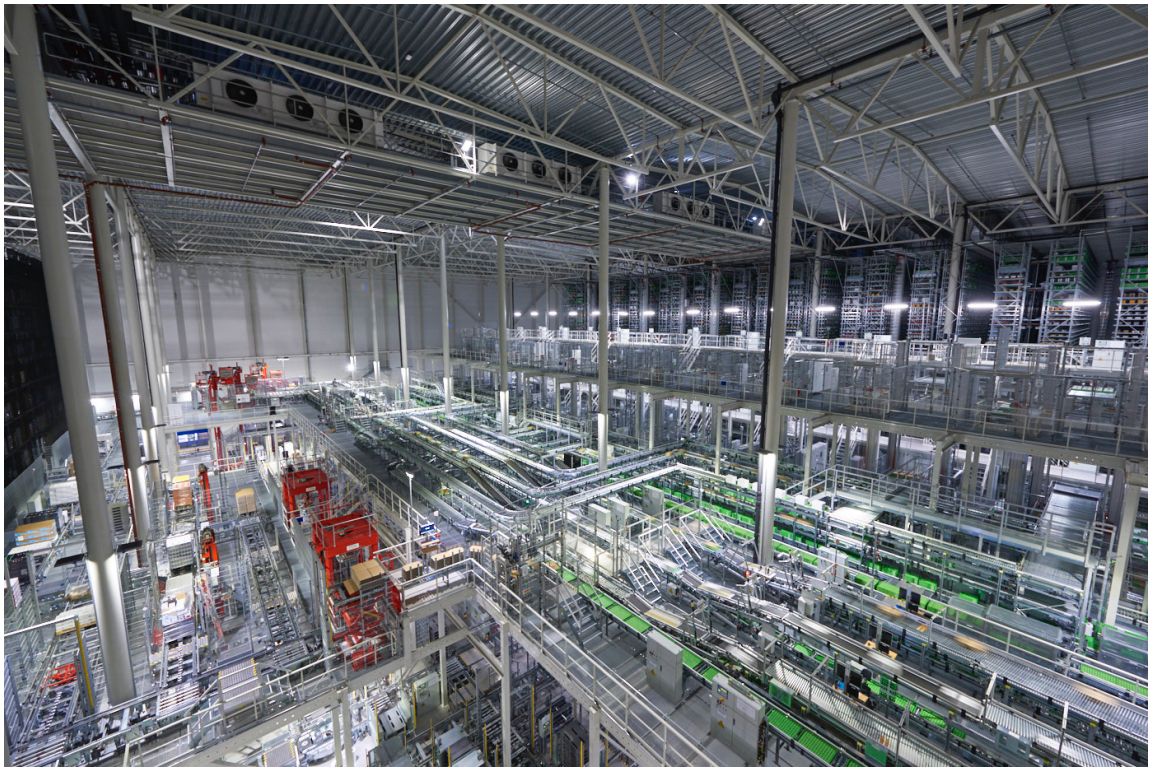
WMS forenkler overvåkningen av lageret og dens prosesser. Dette ved hjelp av en kombinasjon av sensorer og enhetsmerkinger (RFID, strekkode, QR kode) i anlegget. Systemet bruker dette til å spore alle varer og lastbærere. Feil eller blokkerte områder vil bli varslet om når de oppstår, noe som gjør det mulig å reagere og utbedre raskt. Det er også skannere i anlegget som måler dimensjoner på varene, og varsler ved avvik. På denne måten kan WMS ta en kvalitetskontroll av alle varene som er i lageret og forsikre god kvalitet. Enten før de skal plukkes eller mens de beveger seg gjennom lageret.

Informasjon om produksjon og produktivitet, samt oversikt over kundeordre og volum er viktig for å få en god forståelse for driftsbilde. Dette er noe et WMS holder styr på og denne informasjonen kan hentes ut herifra. Alt dette er essensielt for planlegging og beslutningstaking.

WMS gir altså forutsetning for å kunne styre et lager på den beste måten. Den vil sørge for at lageret har riktig belastning, riktig bemanning på riktig sted, og klarer å sende ut rett vare til rett tid, rett volum og rett kvalitet, rett sted til rett pris. (Ten Hompel & Schmidt, 2007)

2.2 OPM – Order Picking Machinery

OPM er et helautomatisert plukksystem, patentert av WITRON (WITRON, 2020a). Systemet brukes innenfor tørr-, frys- og kjølevarer, og regnes som et av de beste i verden. OPM består av flere komponenter som automatisk splitter paller inn til enkeltkolli, plukker enkeltkolli basert på ordre og plasserer enkeltkolli på lastbærer. OPM består i hovedsak av to forskjellige typer maskiner og et system; depalleterer, COM, og ASRS.



Bilde 1: OPM området hos ASKO Sentrallager Kjøl AS

OPM skiller seg ut fra andre plukksystemer på grunn av COM og dens Software. Dette er en plukkmaskin som plasserer kolli raskt og sikkert i riktig forhåndskalkulert rekkefølge på lastbærer. Kombinasjonen av ASRS og COM gir OPM muligheten til å håndtere et stort spekter av produkter og samtidig tilfredsstille etterspørsel. Systemet er modulært, alle soner er like, og størrelsen tilpasses etter behov. Derfor er systemet anvendelig for anlegg uavhengig av størrelse på sortiment og etterspørsel.

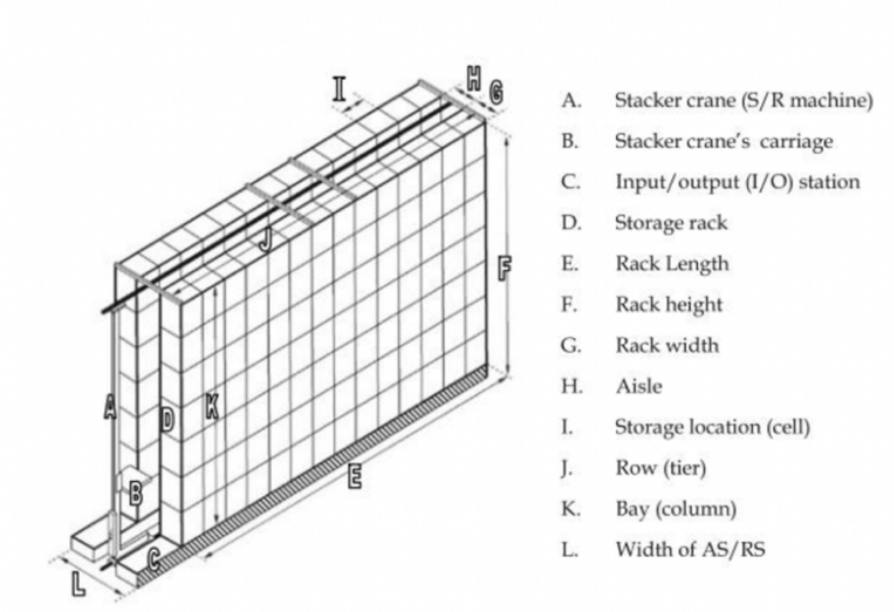
2.3 Depalleterer

Depalleterer er en maskin som splitter paller til enkle kolli, og klargjør dem til å kunne sendes inn til ASRS. Denne maskinen løfter opp pallen, fjerner det øverste laget og sender det ut på et nettverk av samlebånd. Her blir hvert kolli satt på hvert sitt brett. Denne prosessen gjentas fram til pallen er tom eller til systemet har tatt de lagene den trenger. Pallene kommer hit for å levere kolli til ASRS. Depalleterer får helpaller fra høytvarelager, High Bay Warehouse (heretter HBW). Helpallene ankommer med et beskyttelseslag av plastfolie. Den eneste manuelle prosessen i OPM foregår her; fjerning av plastfolie og en manuell sjekk av riktig artikkelnummer, Stock Keeping Unit (Heretter SKU), og antall.

Bilde 2: Depalleterer hos ASKO Sentrallager Kjøl AS

2.4 ASRS – Automatic Storage and Retrieval System

ASRS er et system som både håndterer og lagrer varer. Det er et lagersystem som automatisk tar imot og plasserer kolli eller lastbærer på en bestemt plass, samt henter ut riktig kolli når dette kreves. ASRS er et veldig allsidig lagersystem som brukes til å håndtere alt fra helpaller til små forbrukerpakninger med svært høy volumutnyttelse.

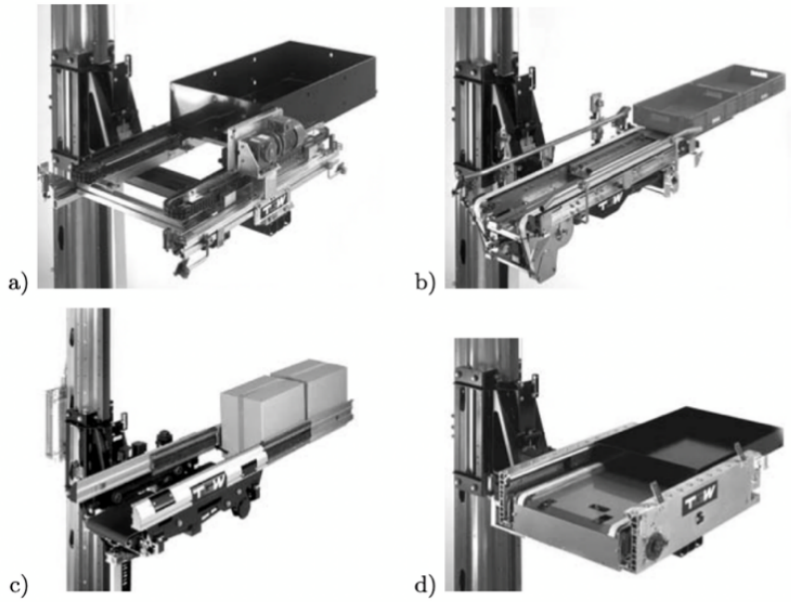


Figur 1: Illustrasjon av ASRS (Manzini, 2012, p. 203)

En ASRS-sone består av en Store and Retrieve enhet, to reoler, Input- og Outputlokasjon. Store and Retrieve enheten, også kalt kran, består av en mast, en vertikal heis med en eller flere lastenheter, samt en styringsenhet. Kranen er begrenset til to akser. Masten beveger seg langs den horisontale aksen, og heisen vertikalt. Den horisontale aksen er begrenset av en eller flere skinner som masten jobber langs. Heisen er låst til den vertikale aksen, masten. På denne måten, med vertikal heis og horisontal bevegelse av mast, kan selve lasteenheten bevege seg raskt og effektivt fra celle til celle i reolen. Størrelse på reoler avhenger av hvilken last som skal håndteres; vekt, volum og antall. Styringsenheten sørger for at lasteenheten er ved riktig lokasjon raskt og presist ved bruk av Programmerbar Logisk Styring (Heretter PLS). Input- og Outputlokasjon er der kranen tar imot kolli til å sette inn i reol eller sender kolli ut av reol og til neste steg i vareflyten.

PLS styrer bevegelsene til kranen og lasteenheten. Styringsenheten får informasjon fra WMS om hvilken celle som har det kolliet som er etterspurt, og henter denne. WMS vet alltid hvilken lokasjon alle kolli har, mens styringsenheten vet hvordan kranen skal komme seg til riktig posisjon.

En ASRS-kran har en lasteenhet som varierer ut ifra hvilket kolli som håndteres. ASRS som skal håndtere paller, gjerne HBW, har gafler som løfter opp pall for å så trekke dem ut. Pall har mye friksjon mot underlaget, og må derfor klareres opp fra underlaget før de kan tas ut av reol. Miniload bruker flere forskjellige lastenheter basert på hvilken lastbærer som brukes. Under er det en illustrasjon av fire forskjellige lastenheter som brukes ved Miniload i ASRS.

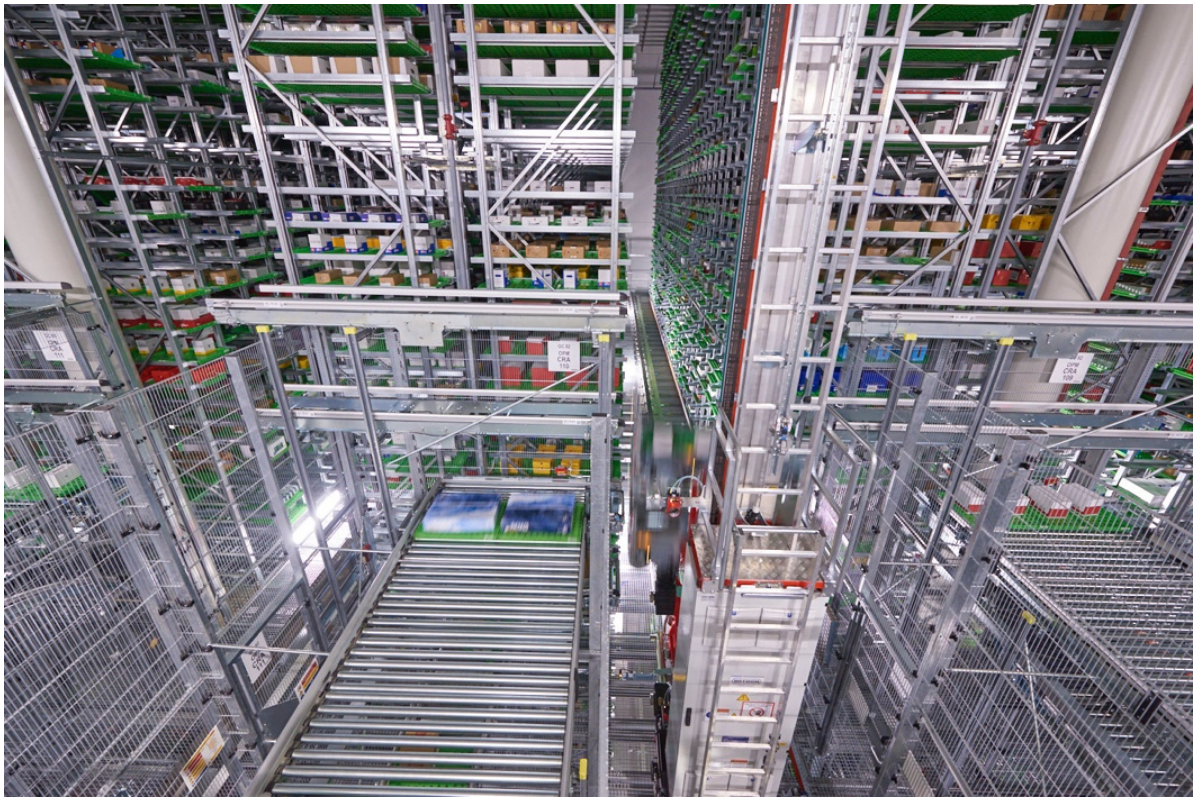


Bilde 3: Forskjellige lastenheter for Miniload ASRS (Ten Hompel & Schmidt, 2007, p. 123)

A	Pulling device	En trekkenhet som drar lastbærer til seg
B	Drive-in telescope	En teleskoparm som skyves under lastbærer og heises opp før den dras tilbake
C	Gripper	To armer som griper kolli og trekker til seg
D	Friction belt	To belter som trekker lastbærer til seg ved hjelp av friksjon

Tabell 1: Forklaring forskjellige lastenheter for ASRS (Ten Hompel & Schmidt, 2007, p. 123)

Et Miniload-lager skiller seg fra tradisjonelle lager ettersom beholdningen lagres i individuelle kolli fremfor på paller. Miniload brukes for lager med små forpakninger og ofte høy produktvariasjon. I motsetning til paller, som er store og har høy vekt, har kolliene liten egenvekt med små dimensjoner. Håndtering av Miniload ved bruk av ASRS gir god utnyttelse av tilgjengelig areal, da ASRS har høy volumutnyttelse og takler et vidt sortiment. (Ten Hompel & Schmidt, 2007)



Bilde 4: ASRS hos ASKO Sentrallager Kjøl AS

2.5 COM – Case Order Machine

COM er en plukkmaskin som plasserer kolli raskt og presist på lastbærer i en forhåndskalkulert rekkefølge. COM får kolli fra ASRS i en gitt rekkefølge basert på ordre. Rekkefølgen på ordren beregnes ved hjelp av komplekse algoritmer der dimensjonen på kolli, vekt og hvor mye tyngde forpakningen tåler er de viktigste parameterne. COM bruker en robotisert arm for å skyve kolli inn på lastbærer, og lastbæreren beveger seg vertikalt ettersom et nytt lag skal plasseres av armen. Når lastbærer er fylt opp blir denne flyttet og en ny tom lastbærer ankommer. En COM har en teoretisk kapasitet på 520 kolli per time. (WITRON, 2020a)

I OPM er det COM som blir tildelt ordren, her kalkuleres plukkrekkefølgen, ASRS forbereder seg med å sortere brettene. Når alle brett er i riktig sone ankommer en tom lastbærer til COM, og begynner å kalle på brett fra ASRS. Mellom ASRS og COM er det en Fine Sorter, et område der rekkefølgen på kolli kan sorteres. Denne funksjonen gjør at plukkrekkefølgen ikke trenger å være så streng, og ASRS kan dermed plukke mer effektivt. Selv om det finnes en Fine Sorter er COM avhengig av at alle ASRS som leverer til COM samarbeider og leverer kolli i riktig rekkefølge. Enkelte anlegg vil også ha en Sequence Buffer, en stor Fine Sorter. Det sørger for bedre utnyttelse av ASRS og antall plukk i COM.

Bilde 5: COM hos ASKO Sentrallager Kjøl AS

Bildet over illustrerer hvordan en COM ser ut. Kolli på brett kommer inn og disse blir separert. Kolli blir så skjøvet til siden hvor en arm plasserer kolli på riktig sted. Her brukes rullekonteiner som lastbærer, men COM stabler også på pall. Hver COM har hver sin ASRS-sone der den kan kalle på kolli fra. COM kan kun få levert kolli fra denne sonen. Hvis COM krever kolli fra andre soner må disse reallokeres. Det vil si at hver COM i prinsippet har tilgang til hele lageret. Det er likevel mest effektivt når COM forholder seg til sin tildelte sone ettersom reallokering er tidkrevende.

2.6 Andre komponenter i OPM

Navn	**Forkortelse**	**Beskrivelse**
Tray Merge		En punkt hvor kolli blir lagt på brett. Kolli transporteres på et samlebånd og blir sluppet ned på et samlebånd som går under hvor det ankommer brett.
Tray Item Collection	TIC	En ansamling av 12 brett som sørger for raskere transport av flere kolli.
Tray Item Collection Builder	TIC Builder	Et område hvor brett blir holdt igjen til det har ankommet nok til å sette sammen en TIC. Fire og fire brett blir satt sammen til et 4x3 rektangel. Når det er samlet en TIC blir den sendt videre til VC.
Vertical Conveyor	VC	En heis som har plass til en TIC. Får levert TIC fra TIC Builder og leverer TIC til TC.
Transfer Car	TC	En vogn som leverer brett inn til riktig ASRS sone. Vognen beveger seg horisontalt og kan frakte én TIC.
Packcorner		En beskyttelsesenhet. Et ramme bestående av tre vegger som sørger for bedre stabilitet i COM og under transport av ferdig stablet lastbærer.
Wrapper		En maskin som automatisk legger plastfoil rundt lastbærer slik at kolli står trygt og stabilt.

Tabell 2: Andre komponenter i OPM

3.0 ASKO Sentrallager Kjøl AS

Kapittelet vil kort forklare hva ASKO Norge er, samt ASKO SL Kjøl og deres leverandør av automatiseringsteknologi, WITRON. Deretter en gjennomgang av logikken og vareflyten hos ASKO SL Kjøl, før en detaljert problemstilling for oppgaven.

3.1 ASKO Norge

ASKO Norge AS (heretter ASKO) er en av Norges største distributører av matvarer. ASKO er logistikkavdelingen i NorgesGruppen med ansvar for vare- og informasjonsflyten fra leverandør til sluttkunde. «Vi forsyner Norge med mat» står det i logoen deres, og det er akkurat det de gjør. ASKO leverer mat til hele Norge, fra Lindesnes i sør til Nordkapp i nord. De distribuerer til alle dagligvarebutikker som ligger under NorgesGruppen og andre partnere, samt storhusholdninger, servicehandel, bensinstasjoner og kiosker. ASKO har totalt 13 regionslagere plassert rundt i hele Norge og to sentrallagre lokalisert i Vestby.

Figur 2: ASKO Norge AS Logo

ASKO importerer selv råvarer og matvarer. De bruker egne lastbærere og lasteenheter for å distribuere matvarer fra sør til nord. I de store sentrallagrene mottas varene fra produsentene. Her blir de sortert, pakket om og videre distribuert til de regionslagrene, disse kan også ta imot matvarer direkte fra produsent, men i mindre grad. Fra de regionale lagrene sendes varene så ut til kundene. ASKO tar med andre ord hånd om de fleste leddene i matforsyningskjeden. I dag leverer ASKO hele 84,1 % av matvarene som finnes i dagligvarebutikkene til NorgesGruppen.

I 2011 ble det bygget et nytt sentrallager på Vestby utenfor Oslo med enkelte automatiserte prosesser, som håndterer tørrvarer. I januar 2017 ble det ferdigstilt et til stort sentrallager for kjølevarer på Vestby, ASKO SL Kjøl. Dette sentrallageret et av de mest avanserte lagerhusene og distribusjonssenterene i Norge. I 2021 skal ASKO åpne et nytt helautomatisert regionslager i Sande, dette blir toppen av miljøvennlige bygg. I tillegg til miljøvennlig bygg, jobber ASKO kontinuerlig med å finne bedre og mer bærekraftige løsninger for å dekke energibehovet sitt. Mange av bilene til ASKO går i dag på

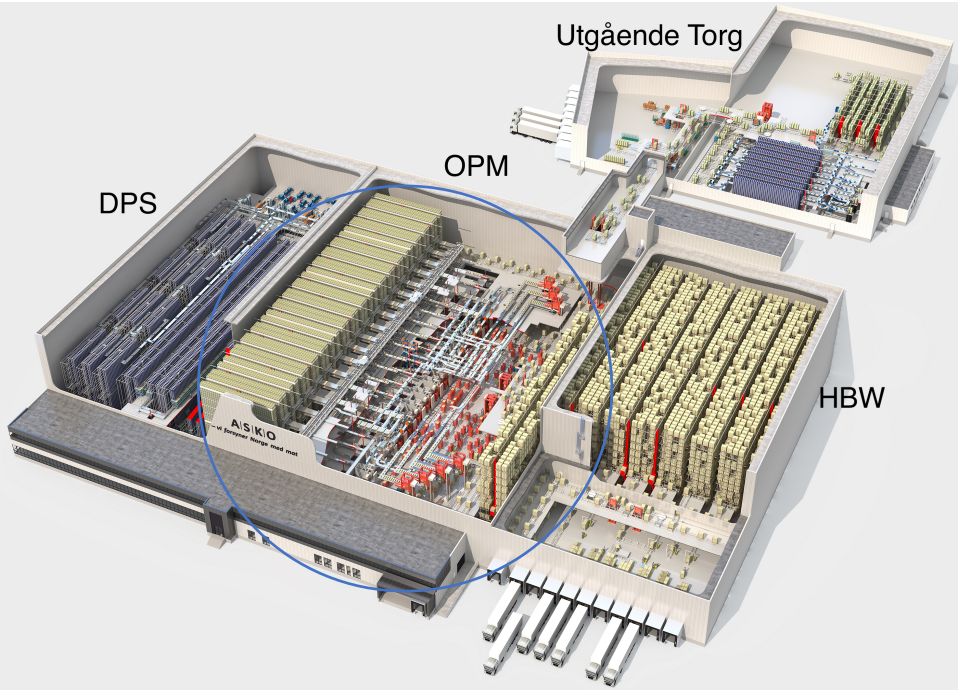
egenprodusert hydrogen og elektrisitet. Ved et av regionslagrene har de satt opp egne vindmøller som sørger for mesteparten av byggets energibehov.

ASKO er morselskapet til flere datterselskap, som alle deler ASKO-navnet. ASKO SL Kjøl, ASKO Sentrallager AS (Tørrvarer) og ASKO Transport AS er noen eksempler. ASKO har datterselskap som tar for seg hver enkelt del av distribusjonskjeden. (ASKO, 2020)

3.2 ASKO Sentrallager Kjøl AS

ASKO SL Kjøl er ASKO sitt sentrallager for kjølevarer, altså produkter som oppbevares ved 0 til 4 °C. Lageret er nesten helautomatisert og som tidligere nevnt lokalisert i Vestby. Dette lageret distribuerer cirka 70% av alle kjølevarene som ASKO i sin helhet distribuerer til sine sluttkunder. Rundt 25% av kjølevarene som distribueres i hele Norge går gjennom dette lageret. Lageret håndterer alle kjølevarene som leveres til ASKO sine kunder på Østlandet og breddesortimentet for resten av landet. Dermed må ASKO SL Kjøl være i stand til å håndtere en stor og variert vareflyt til enhver tid, både i antall produkter og volum.

Lageret til ASKO SL Kjøl sto ferdig i starten av 2017 med automatiske lagerløsninger levert av WITRON AS. Vareflyten er nesten helautomatisert fra lossing til lasting i alle avdelinger, med kun en håndfull manuelle operasjoner. (ASKO, 2018)



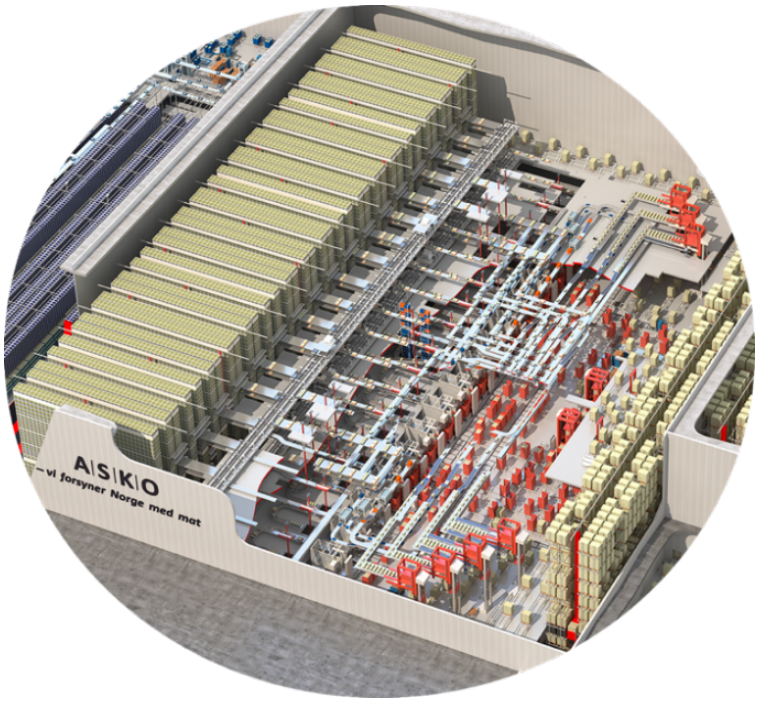
Figur 3: Illustrasjon av ASKO Sentrallager Kjøl AS

Lageret deles inn i fire hovedområder; inngående flyt fra varemottak til høylager (HBW), etterfulgt av to forskjellige plukksystemer, helautomatisk plukk (OPM) og kasseplukk (DPS – Dynamic Picking System), før ferdig kundeordre sendes ut på pall eller rullekonteiner til utgående torg. Produkter med pappemballasje blir sendt til OPM og varer uten pappemballasje eller som må plukkes manuelt blir sendt til DPS.

I dag består OPM av 18 COM, 36 ASRS og 7 depalleterere. Hver ASRS inneholder to reoler som alle består av en matrise med 34 x 29 celler, med plass til fire små brett i hver celle. Dette tilsvarer 3944 brett per reol, som utgjør cirka 280 000 plasser totalt. ASKO SL Kjøl operer med to forskjellige brettstørrelser; liten og stor. Et stort brett er dobbelt så langt som et lite brett og 97% av alle brett som brukes er små.

Lasteenheten som blir brukt er et Drive-in Telescope, se *Bilde 3: Forskjellige lastenheter for Miniload ASRS*. En plate som går inn under et brett, løfter dette opp, for å så ta det med seg. En kran har plass til fire små brett per etasje og har to etasjer. En kran bruker tilnærmet like mye tid på å plukke opp eller sette fra seg brett, uavhengig av om det er et eller fire brett. Ved bruk av begge etasjer vil det forekomme noe lengre tidsbruk da lasteenheten må flyttes.

OPM håndterer cirka 1800 forskjellige SKU-er. Etter at lageret ble ferdigstilt og som følge av at FEFO ble så streng som den er i dag, har det gitt et konkurransefortrinn med at varene som finnes ute i NorgesGruppens butikker har fått en økt holdbarhet med en til to dager.



Figur 4: Illustrasjon av OPM hos ASKO Sentrallager Kjøl AS

3.2.1 WITRON AS

WITRON er et tysk firma som spesialiserer seg innenfor lagerbygg, både de fysiske komponentene og Software i systemet. De leverer delvis- og helautomatiserte lagerløsninger med kompleks Hardware- og Softwareløsninger. WITRON sin spesialitet er OPM. Her brukes deres mest avanserte og mest suksessfulle maskin, COM, plukkmaskinen som kaller på kolli og stabler dette på lastbærer i en forhåndskalkulert rekkefølge. I tillegg til plukkmaskinen tilbyr de også løsninger som «Goods-to-Person», «Person-to-Goods» og «Pick-and-Pack». De fire plukkmåtene utnytter forskjellige lastbærere, blant annet paller, brett og kasser. WITRON leverer lagerløsninger til alle typer temperaturer; tørr-, kjøl- og frysevarer og til mange forskjellige industrier: mat, elektronikk, bil, produksjon og helse, med flere. (WITRON, 2020b)

3.2.2 Logikk innad i lageret

FEFO – First Expired, First Out

FEFO er en logikk som går ut på å velge varer med kortest holdbarhetsdato først. FEFO brukes når det er snakk om forgjengelige varer, varer som mister sin fulle verdi etter utløpsdato. Målet med en slik logikk er å forhindre svinn av utløpte produkter, samt forlenge tiden der produktet er tilgjengelig for kunden før utløpsdato

FEFO sørger for minst mulig svinn som følge av utgått holdbarhetsdato hos ASKO SL Kjøl. Logikken er så streng at den varen med kortest holdbarhet uansett plukkes først, men det forekommer noen unntak for produkter med lang holdbarhet. Dette skjer uavhengig av lokasjon. Dersom det kommer to bestillinger av samme vare, vil den med kortest holdbarhet prioriteres, selv om det samme produktet finnes i riktig sone.

Reallokering

Reallokering er et begrep for flytting av brett mellom ASRS-soner. En reallokering gjennomføres når COM trenger et brett som ikke finnes i sin ASRS-sone, på grunn av FEFO og hvis det er høy konsentrasjon av et produkt i en del av lageret.

Reallokering skjer mellom ASRS-kraner eller ved en alternativ lengre rute. ASRS-kranene kan sende brett mellom hverandre, altså fra en kran til neste. Dette skjer ved at kranen går til reallokeringsposisjonen, enten for å ta imot brett, gi fra seg brett eller fungere som et mellomledd mellom to omliggende kraner. Når kranen brukes som et mellomledd vil den ta fra den ene siden og sende direkte ut på andre siden, slik at neste kran kan gjøre det samme.

Skal brett reallokeres for lengre avstander krever det flere kraner, noe som tar opp kapasitet fra de involverte kranene. På grunn av denne kapasitetsbegrensningen finnes det en alternativ rute som ikke bruker like mange kraner. Den alternative ruten innebærer at brett blir sendt ut av ASRS-sonen og inn på et nettverk av samlebånd. Her blir brett omdirigert til samlebåndet for Input. Dermed kan brett bli plassert i hele lageret på nytt. Denne muligheten finnes kun på partallssoner og det kan ta lengre tid å sende brett denne ruten, avhengig av hvor langt brett skal reallokeres. Det finnes derfor en parameter som bestemmer avstanden et brett skal reallokeres før den skal ta den alternative ruten. En parameter som sier hvor mange soner et brett må reallokeres før det betegnes som en lang reallokering og må benytte den alternative ruten. Ved normal drift er denne satt til 8.

Ordre

ASKO SL Kjøl har Pull Production. COM får en ordre og kaller på brett fra ASRS, og eventuelt paller fra HBW hvis dette trengs. Når kundeordre kommer inn i systemet blir det sjekket hvilke SKU som er med og antall kolli ordren inneholder. En ordre får så et tidspunkt for når den tidligst kan plukkes, når den senest skal være ferdig plukket og når den skal være på utgående torg. En ordre blir plassert i den sonen som har tilgang til flest kolli som ordren inneholder. På denne måten må færre brett reallokeres og ordren kan plukkes raskere.

Så fort alle brett er i samme sone som ordren skal plukkes i, og ordren er den neste som skal plukkes, vil ASRS begynne å sende kolli i ordren til COM.

3.3 Vareflyt

Vareflyten i OPM kan sees i *Figur 5: Forenklet fremstilling av vareflyten i OPM hos ASKO Sentrallager Kjøl AS* på neste side. Helpaller kommer inn i HBW fra varemottak. Her sjekkes det at det er riktig SKU som er mottatt og riktig antall ved hjelp av sensorer, vekt og bilde med informasjon fra WMS. Varemottaket inneholder en av tre manuelle operasjoner tilknyttet OPM, frakte pall fra lastebil og til et samlebånd som sender pallen inn til HBW. Fra HBW blir helpaller sendt til Defoil. Defoil er den andre manuelle prosessen og den eneste som er lokalisert i selve OPM-området. Her blir plastikkfolie tatt av pallen og det blir sjekket at det er riktig vare og riktig antall. Deretter blir det sendt til depalleterer og hvert lag på pallen blir splittet til individuelle kolli.

Kolli går så til Tray Merge hvor de blir satt på hvert sitt brett. Hvert brett kan til enhver tid spores og systemet ved alltid hvilket kolli som står på brettet, samt all tilhørende informasjon

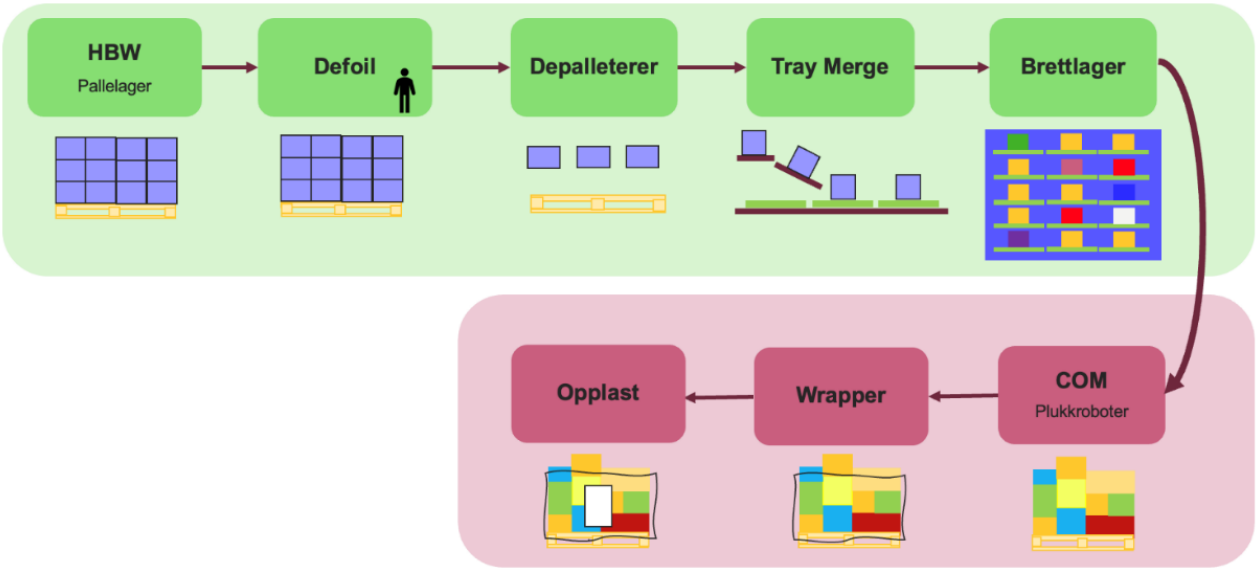
til hvert kolli. Etter Tray Merge går det gjennom et samlebåndsnettverk før det ender opp i en TIC Builder.

Her blir 12 brett satt sammen til en TIC. Fra TIC Builder går TIC enten opp eller ned, ved hjelp av en VC. VC-ene er her fordi brettlageret er fordelt over to etasjer. Her blir TIC-en tatt imot av en TC, som så sender riktig mengde brett inn til riktig ASRS sone, brettlageret. Brett blir sendt ut fra brettlageret og til COM når det trengs. Brett går ut fra ASRS, og til en ny heis som sender det ned mot COM. Rett foran COM er det en Fine Sorter. På grunn av denne kan ASRS gjøre enkelte avvik fra den strenge plukkrekkefølgen, aller siste sortering og kontroll av rekkefølgen skjer her.

Brett blir sendt gjennom en siste sjekk hvor dimensjoner, orientering, vekt og om kolli er skadet, blir kontrollert og varslet om hvis det forekommer avvik. Dette sikrer at COM opererer med minimalt antall feil. Etter at alt er kontrollert blir kolli separert fra brettet og COM plasserer kolli på riktig sted på pall eller i rullekonteiner. Brett går tilbake til Tray Merge og blir brukt på nytt med nye kolli. Pallen eller rullekontaineren som står i COM ankommer alltid i en Packcorner.

Når ordren er ferdig blir pallen sendt videre til en Wrapper. Her blir lastbærer separert fra Packcorner og den ferdige pallen blir plastret og sendt ut til opplast. I denne transportetappen får pallen en løpeseddel som viser hvilken lastebil den skal på og hvilken ordre det er. Til slutt skjer den tredje og siste manuelle operasjonen, lastbærer blir lastet i lastebil.

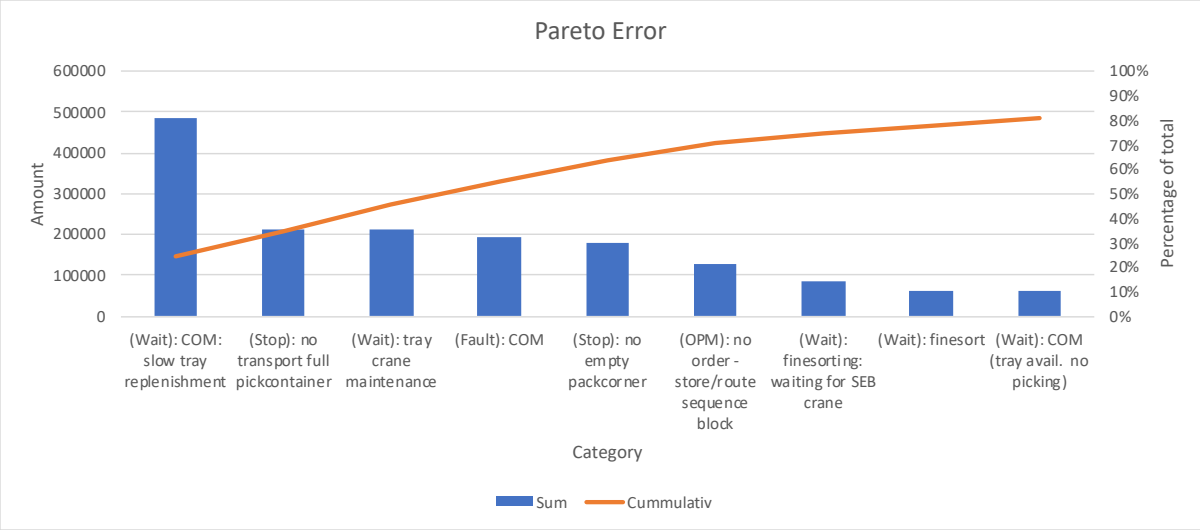
En del av lageret består av kasseplukk, DPS. Herifra kommer kasser stablet på pall, enten inn til OPM eller opp til opplast. Går pallen til OPM er det fordi ordren inneholder varer som fraktes både i kasser og på vanlig pall. COM legger da et papplokk på kassene og stabler deretter kolli på disse. Det at ASKO SL Kjøl har mulighet til å stable på kasser gjør systemet mye mer komplekst og avansert enn andre varianter av OPM.



Figur 5: Forenklet fremstilling av vareflyten i OPM hos ASKO Sentrallager Kjøl AS

3.4 Dagens situasjon

Lageret er velfungerende, men opererer med lavere ytelse enn ønskelig. Dette kommer tydelig frem når en ser på COM. COM kan ofte være inaktiv og venter da på at andre komponenter i lageret skal gjøre sin del eller fordi det ikke er flere ordre. Det er gjennomført en paretoanalyse for å se hvilke feil som opptrer mest og som forhindrer COM i å stable kolli på lastbærer. Ved hjelp av dette verktøyet ser en tydelig hva som forhindrer flest plukk i systemet og hvilke komponenter som må sees på. Denne analysen er gjort for å videre kunne spesifisere en mer konkret problemstilling og for å spisse arbeidet i dette prosjektet. *Figur 6* er en forenklet versjon av denne analysen. Her er det tatt med de ni feilene som opptrer oftest for å belyse disse. En fullkommen analyse kan sees i *Vedlegg 1: Full Paretoanalyse.* Det er tydelig at «(Wait): COM: Slow tray replenishment» er den feilen som trekker ned ytelsen til COM mest. Dette er at COM ikke stabler på pall, men venter på å få brett fra ASRS. I dette tilfellet stabler COM kolli fortere enn det ASRS klarer å levere. Dette bekrefter påstanden ASKO SL Kjøl kom med i starten, at ASRS-kranene er en av de største flaskehalsene i OPM.



Figur 6: Paretoanalyse av feilkilder for COM

«(Wait): COM: Slow tray replenishment» står for 24% av alle forhindrede plukk i COM. Det er flere feilkilder som trekker kapasiteten ned vesentlig:

- (STOP) no transport full pick container. Lastbærer er stablet ferdig i COM, men er ikke blitt hentet ennå
- (Wait): tray crane maintenance. Vedlikehold i ASRS kranene
- (FAULT): COM. Det har skjedd en feil i COM

Disse fire feilene står for 55% av totalt antall plukk forhindret.

«(Wait): COM: Slow tray replenishment» kommer av at ASRS gjør andre ting enn å levere brett direkte til COM. ASKO SL Kjøl tror dette kommer av styrings- og logikkparametere som påvirker ASRS.

3.5 Problemstilling

ASKO SL Kjøl ønsker å analysere kapasitetsutnyttelse for enkelte komponenter i den helautomatiske delen av lageret, OPM. Basert på genererte rapporter i deres WMS, vises det at de automatiske plukkrobotene, COM, sjeldent arbeider så fort som de er i stand til å gjøre. Fra *Figur 6*, som viser årsaker til at COM ikke plukker, er det tydelig at feilårsaken «(Wait): COM: Slow tray replenishment» er den største kilden til redusert plukking. Denne feilårsaken oppstår når COM er klar til å plukke, men det ikke har kommet kolli fra ASRS. Dette tyder på at ASRS er den største begrensende faktoren for høyere plukkrater.

ASRS har enkelte logiske parametere som har en ukjent påvirkning på ytelsen til lageret. FEFO-logikken er verdiskapende for sluttkunden, i og med at sluttkunden i snitt får én til to dager ekstra holdbarhet for kjøleprodukter, men kan ha en negativ innflytelse på ytelsen til OPM-området. Styringsparameteren for lengre reallokeringer påvirker vareflyten på inngående samlebånd til ASRS-lageret i OPM, samt kapasitetsutnyttelsen av ASRS-kranene.

Analyser av eksisterende produksjonsdata vil brukes til å definere ytelsen til OPM-delen av lageret slik det er i dag, i tillegg til å undersøke hvilke faktorer som påvirker denne ytelsen. For å kartlegge dagens driftsbilde vil OPM-området analyseres ved gjennomsnittlig drift, samt hvordan driftsbildet endres ved lav og høy pågang. Det vil bli undersøkt i hvilken grad reallokeringer påvirker ytelsen til ASRS, og hvordan FEFO-logikken påvirker antallet reallokeringer og distansen på reallokeringer.

For å undersøke hvordan endringer av logiske parametere påvirker vareflyten og ytelsen til OPM, vil oppgaven benytte seg av diskret hendelsessimulering i programmet FlexSim. Som følge av den høye kompleksiteten til denne delen av lageret vil det bli bygget en forenklet 3D-modell av lageret, med en tilnærmet logikk for vareflyt og styring av maskineri. Simuleringen vil benyttes til å undersøke konkret hva som skjer når styringsparameteren for lengre reallokeringer endres, i tillegg til å avdekke i hvilken grad FEFO påvirker ytelsen til ASRS.

Gjennom analyser av produksjonsdata og ved bruk av diskret hendelsessimulering skal følgende forskningsspørsmål besvares:

F1: I hvilken grad påvirker reallokeringer ytelsen til ASRS?

F2: I hvilken grad påvirker FEFO-logikken antallet reallokeringer og distansen på reallokeringer?

F3: Hvilke kapasitetsbegrensninger pålegges av FEFO?

F4: Hvordan påvirker endringer av styringsparameteren for lengre reallokeringer handlingsmønsteret for ASRS?

4.0 Driftsdata

All data og informasjon som er brukt til analyser og måling av ASRS og COM er gitt fra ASKO SL Kjøl. Tilgang til dette er gitt gjennom eksternt skrivebord og muligheten til å koble seg opp på deres WMS, samt gjennom mange møter med bedriften. I WMS-en ble det gitt tilgang til to programmer:

- BWOS, ABL Witron Operating System
- BMIS, Base-Layer Management Information System

BWOS er en todimensjonal visuell fremstilling av hele lagersystemet hvor de forskjellige etasjene er spredd utover. Her overvåkes systemet og status på de forskjellige områdene med real-time informasjon. BMIS er en stor database som samler data fra alle områdene i lageret. Her kan en trekke ut oppdatert informasjon og få det presentert i en rapport. Disse rapportene inneholder statistikk og informasjon om driften.

Data er hentet ut fra disse rapportene og i korresponderende tidsintervall:

Rapport	**Tidsintervall**
OP13: COM Pick Statistics	01/03/2021 00:00 – 12/04/2021 10:00
OP13a: COM Pick Performance	01/03/2021 00:00 – 12/04/2021 10:00
OP08a: Crane Performance	11/03/2021 00:00 – 12/04/2021 10:00
OM55a: Cost Relocation Orders To COM	23/03/2021 00:00 – 06/04/2021 09:14
OM29: Order-Overview	21/03/2021 21:37 – 03/04/2021 17:04
OP48c: PAL INFO	19/03/2021 15:37 – 03/04/2021 16:38

Tabell 3: Rapporter hentet ut fra WMS med korresponderende tidsintervall

Alle rapportene inneholder forskjellig data:

- OP13: Hvor mange kolli hver COM har stablet på lastbærer.
- OP13a: Hvor mange kolli som ikke har blitt stablet på lastbærer og grunnen for dette.
- OP08a: Hvor mye tid ASRS-kranene har brukt på forskjellige operasjoner og antall brett som er håndtert.
- OM55a: Nøyaktig når en reallokering skjedde, fra sone til sone og en potensiell grunn for at reallokeringen ble gjennomført.
- OM29: Ordrenummer, SKU, antall, når ordren tidligst kan plukkes, når ordren ble ferdig stablet og når ordren skal sendes ut fra lageret.

- OP48c: Alle inngående kolli til OPM. SKU, antall og utløpsdato, samt hvilken depalleterer det kom fra.

Dette er et lite utdrag av ASKO SL Kjøl sine mange rapporter for produksjonsdata. Disse seks er de som har vært mest relevante for dette prosjektet. OP13, OP13a og OP08a viser data i timesintervaller. OM55a, OM29 og OP48c er hentet ut med tidsstempel for når hendelser inntraff. Det er hentet ut 637 timesintervaller med data fra OP13 og OP13a, og 421 timesintervaller med data fra OP08a.

Data fra når systemet ikke produserer er fjernet. Tider hvor det ikke produseres er fredag 22:00 til lørdag 08:00 og lørdag 17:00 til søndag 23:00. I disse tidsintervallene utføres det renhold og vedlikehold.

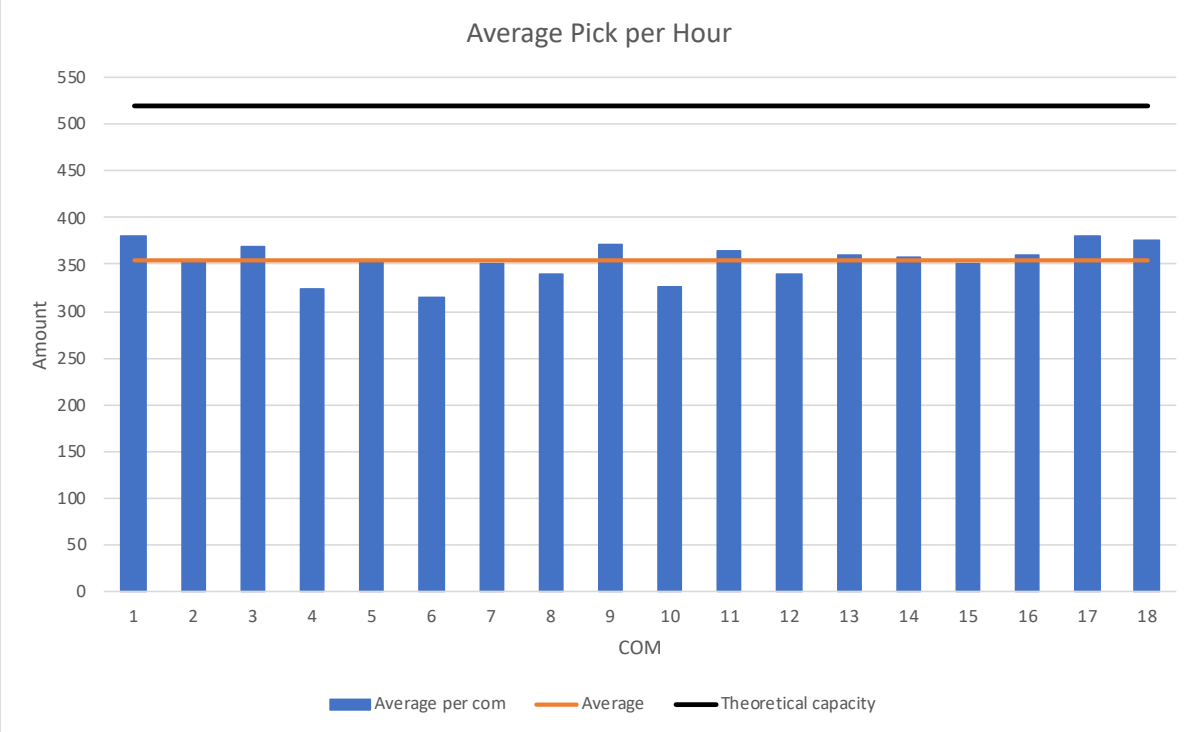
Data fra 26/03/2021 22:00 til og med 05/04/2021 01:00 (påskeuken) ble ikke hentet ut for OP13, OP13a og OP08a. Da dette ble oppdaget var dataen slettet fra WMS. Dette gjør validering av FlexSim-modellen noe vanskeligere da reell data for output ikke finnes for all simulert data.

OP13, OP13a, OP08a og OM55a er rapporter som er blitt brukt for å kartlegge dagens situasjon. OM29 og OP48c er brukt som simuleringsdata i FlexSim.

4.1 Dataanalyse

4.1.1 COM

Alle 18 COM jobber uavhengig av hverandre og tar kun en ordre om gangen. COM kan planlegge hvilke brett den trenger fra ASRS for flere ordrer samtidig, men kolli som blir stablet på en lastbærer er alltid i samme ordre. Det er implementert en regel som sier at alt kolli i en ordre må være i riktig sone før COM begynner å plukke. En COM har en teoretisk kapasitet på 520 kolli per time, som utgjør en total teoretisk kapasitet på 9360 kolli per time i OPM. Grafen under viser hvor mange kolli hver COM plukker i snitt per time.



Figur 7: Gjennomsnittlig plukk i COM

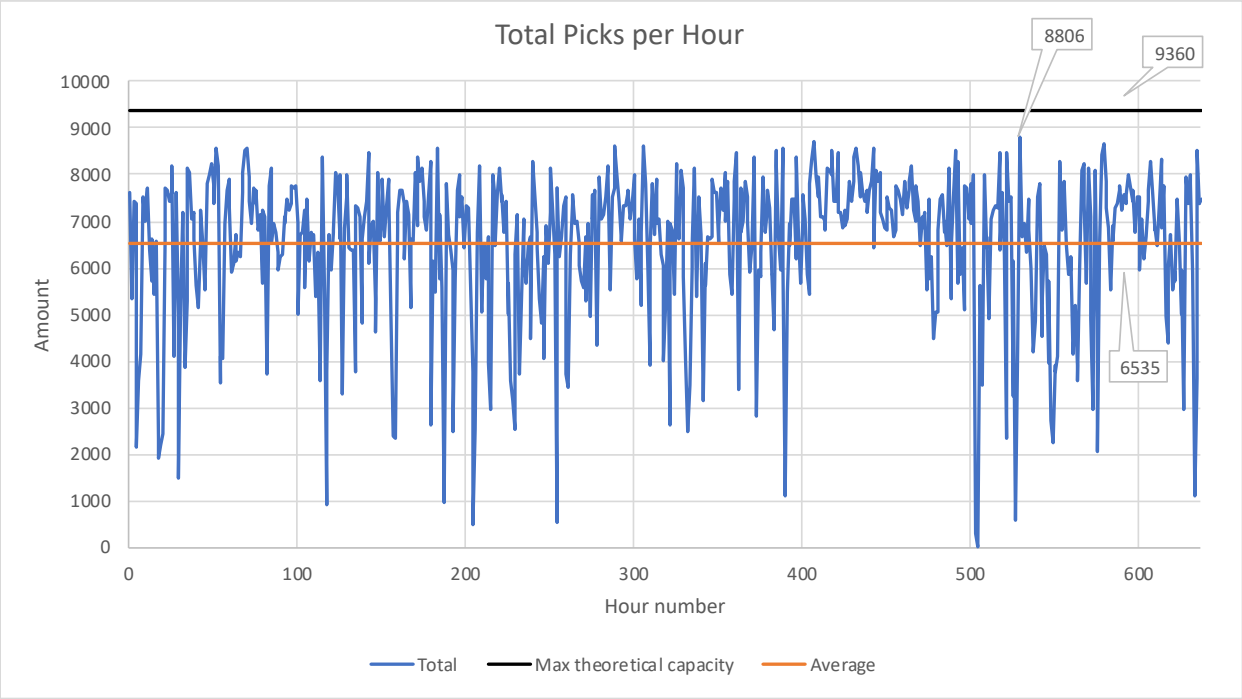
Grafen viser at det er forskjell i hvor mye hver COM plukker. COM 1, 17 og 18 er de som plukker mest og COM 4, 6 og 10 plukker minst. Partallssoner plukker minst fordi disse har tilgang til den alternative reallokeringsruten og mulighet til å sende brett tilbake til Tray Merge. På grunn av dette blir reallokerte brett som skal bruke den alternative ruten sendt til en av partallssonene. Sonene ved ytterkantene av lageret, 1,17 og 18, plukker mest fordi ASRS-kranene i disse sonene brukes ikke som mellomledd for reallokeringer. Disse sonene har heller ingen spesielle arbeidsoppgaver, dermed har ASRS-kranene mer tilgjengelig tid til å levere brett til COM.

I snitt plukkes det 354 kolli i timen per COM. nedenfor er en full oversikt over hvor mye hver COM plukker i snitt per time. Det er 17% forskjell mellom COM 1 og 6. Den høyeste verdien i det målte tidsintervallet er 588 plukk i løpet av en time i COM 18.

Com	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Average	381	354	369	323	353	314	351	339	372	327	364	339	361	359	352	360	380	376

Average
354

Tabell 4: Gjennomsnittlig plukk i COM

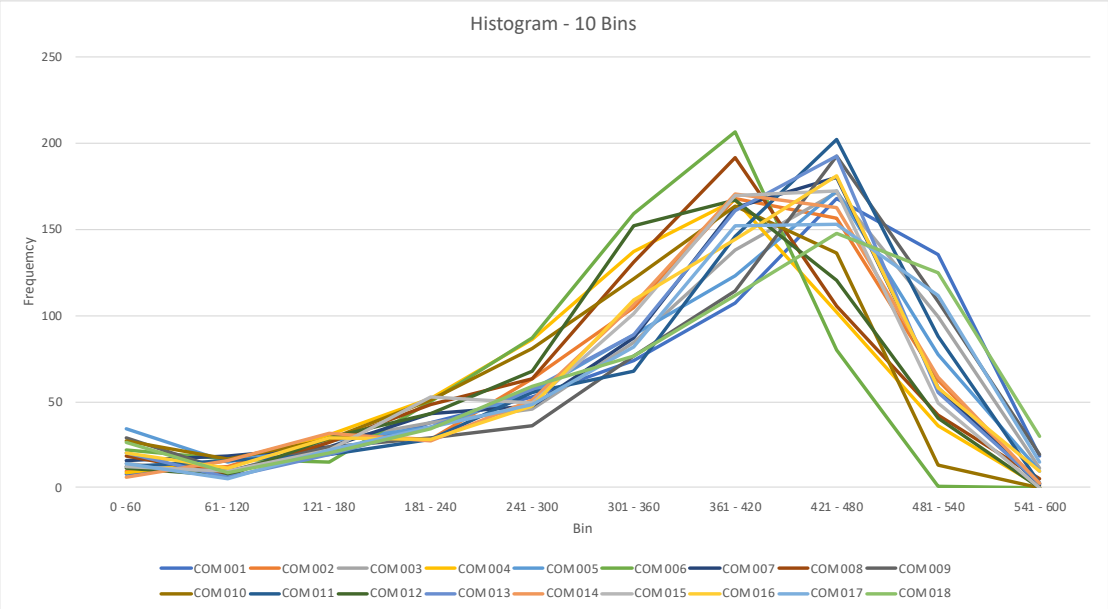


Figur 8: Totalt antall plukk per time

Figur 8 viser hvor mye som er plukket hver time i alle 18 COM. Hvert datapunkt er totalsummen av antall kolli plukket i gitt tidsintervall. Det er store svingninger og flere timer hvor det blir plukket under 2000 kolli. Maks teoretisk kapasitet, 9360, blir aldri nådd, men det høyeste som er målt i denne perioden er 8806, datapunkt 529. Dagsrekorden for antall plukk er 184 098, mens det høyeste målt i de eksporterte tidsintervallene er 182 724. Mellom 22:00 og 23:00 er det tidsintervallet det er høyest pågang mens 05:00 til 06:00 er den tiden på

døgnet hvor det er lavest pågang. Tirsdag er den dagen det er høyest pågang, men generelt sett er det jevnt.

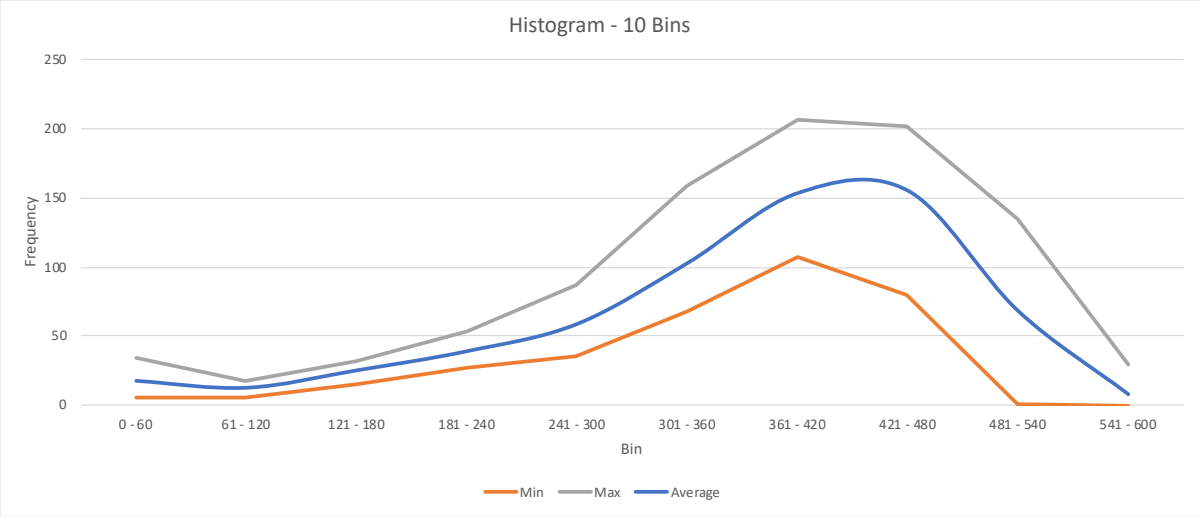
Antall plukk i COM har ingen statistisk fordeling, se *Figur 9* og *Figur 10* nedenfor og *Vedlegg 2: Histogram plukk i COM , 20/40 Bins* og *Vedlegg 3: @Risk Fit to distribution analyse*. *Figur 9* er et histogram for hver COM for hvor mye plukk som er blitt gjennomført, med Bin Size[2] på 10. *Figur 10* viser gjennomsnittet av alle 18 COM, samt alle topp- og bunnpunkt som finnes for hver bin. Se *Vedlegg 2* for grafer med flere Bins. I *Vedlegg 3* er det gjennomført en «Fit to Distribution» sjekk i @Risk[3]. Programvaren kom fram til at det ikke finnes noen statistisk fordeling for antall plukk per time per COM. Grafene har veldig lik fasong, men ingen har en kjent bestemt fordeling. Hovedgrunnen for at det ikke finnes noen statistisk fordeling som passer er fordi lageret er ordrestyrt og hva som skal gjennomføres bestemmes fra et kontrollrom. All plukk baserer seg på ordre og COM har kun oversikt over ordre i en gitt tidsperiode frem i tid. Kommer det en hastebestilling vil denne bli fremskjøvet og COM vil begynne på denne så fort det lar seg gjøre. På grunn av forskjellen i ordrestørrelse og uforutsette hendelser er det umulig å forutsi hvor mye COM skal plukke frem i tid.



Figur 9: Histogram - 10 Bins - Alt

[2] Verdier fra 0 og til maks antall plukk i en COM. Rundet maks opp til nærmeste hundre og delt på antall Bins. Dette gir 10 intervaller som inneholder 60 forskjellige plukkantall.

[3] Et Excel-tillegg levert av Palisade. Det er et statistikkprogram som brukes for å analysere statistiske distribusjoner og gjøre statistiske beregninger i Excel. *Fit to Distribution* sjekker alle datapunkter som blir fremstilt og om det finnes en statistisk fordeling for disse.



Figur 10: Histogram - 10 Bins - Min/Maks/Gjennomsnitt

4.1.2 Ytelse

COM er helautomatisk, men ytelsen er ikke optimal. Det er flere feil som oppstår, og disse reduserer plukkraten. Det vil si at det kommer færre brett til COM enn det den har kapasitet til å håndtere. Dette er ytelsesreduserende hendelser og aktiviteter som hovedsakelig skjer i ASRS eller i COM, men de kan også oppstå i depalletererne. På grunn av dette er det blitt beregnet at COM opererer på 67.7% av maks teoretisk kapasitet. Forholdet mellom plukk- og feilrate illustreres i *Tabell 5*. Dette er kalkulert ut ifra antall plukk gjennomført og antall plukk som ikke er gjennomført basert på tall beregnet av WMS. Listen over alle feilkilder som kan oppstå kan sees i *Vedlegg 1: Full Paretoanalyse*.

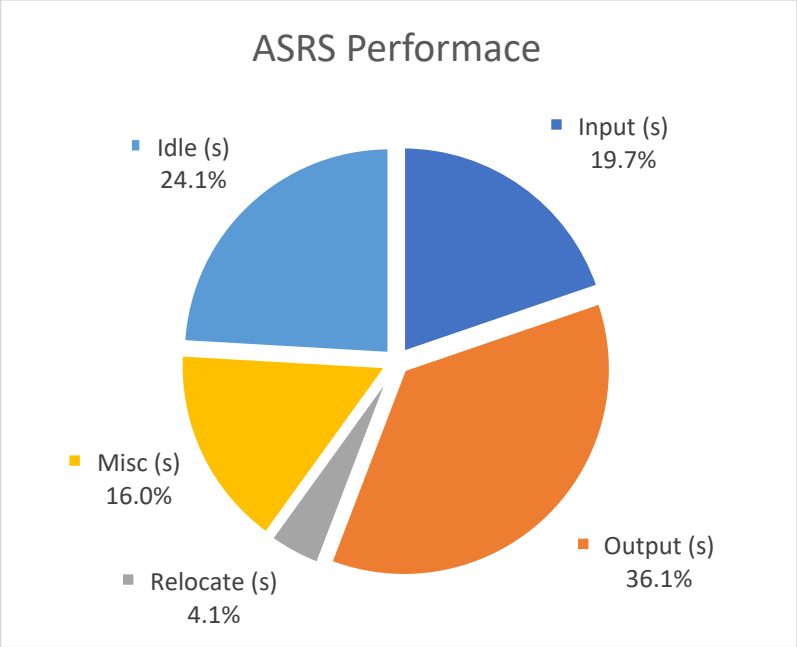
Plukkrate	0.67659	67.7%
Feilrate	0.32341	32.3%

Tabell 5: Plukk- og Feilrate i COM

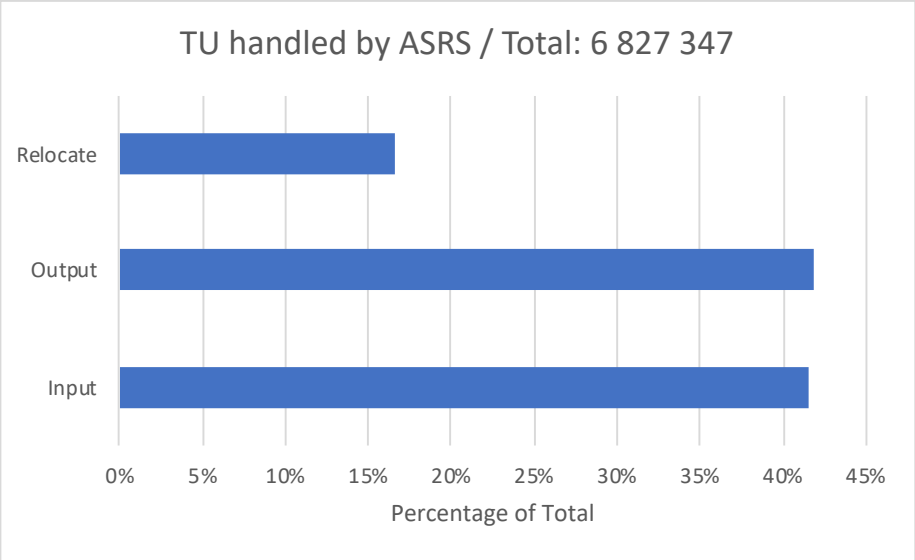
4.1.3 ASRS

Når COM trenger brett sendes disse ut fra ASRS. Siden ASRS sliter med å levere nok brett til å opprettholde kapasiteten til COM er det viktig å se på ytelsen til kranene. Kranene kan sende brett mellom hverandre, og partallssonene kan sende brett ut til en alternativ reallokeringsrute. Kranene kan også reallokere innad i egne reoler, både for å effektivisere distribusjonen for kommende ordre og for å frigjøre celler til nye brett. Dette for å optimalisere fyllingsgraden av celler som ikke er tomme.

Figur 11 viser hva ASRS kranene bruker tiden sin på. Her er totalen av all kran aktivitet summert og delt opp i de forskjellige aktivitetene. Output er den aktiviteten som tar lengst tid å gjennomføre, mens reallokering er den aktiviteten som tar minst tid. Det brukes mye tid på input og output, men 24% av tiden står de også rolig. Dette kommer av at kranene i flere tilfeller ikke har noe å gjøre, verken input, output eller reallokeringer. Input er den tiden det tar for kranen å hente et brett og sette det inn i brettlageret. Output er tiden det tar å hente brett fra brettlageret og sende det ut. Relocate er tiden kranene bruker på å reallokere brett. Misc er en egen kategori og den tar for seg alt annet som kranen gjør. Det er uvisst nøyaktig hva denne kategorien innebærer, men den inneholder en relativt stor andel av all aktivitet. Misc inneholder ikke tiden brukt på vedlikehold.



Figur 11: ASRS ytelse - totalt



Figur 12: Brett håndtert av ASRS - totalt

ASRS håndterer mange brett, både input, output og reallokeringer. Nøyaktig hvor mange og hvilken kategori disse tilhører kan sees i *Figur 12*. I den målte tidsperioden var det cirka 6,8 millioner interaksjoner med brett. En reallokering i dette tilfellet er for hver gang en kran er nær et brett som reallokeres. 17% av alle brett som ble håndtert var reallokeringer.

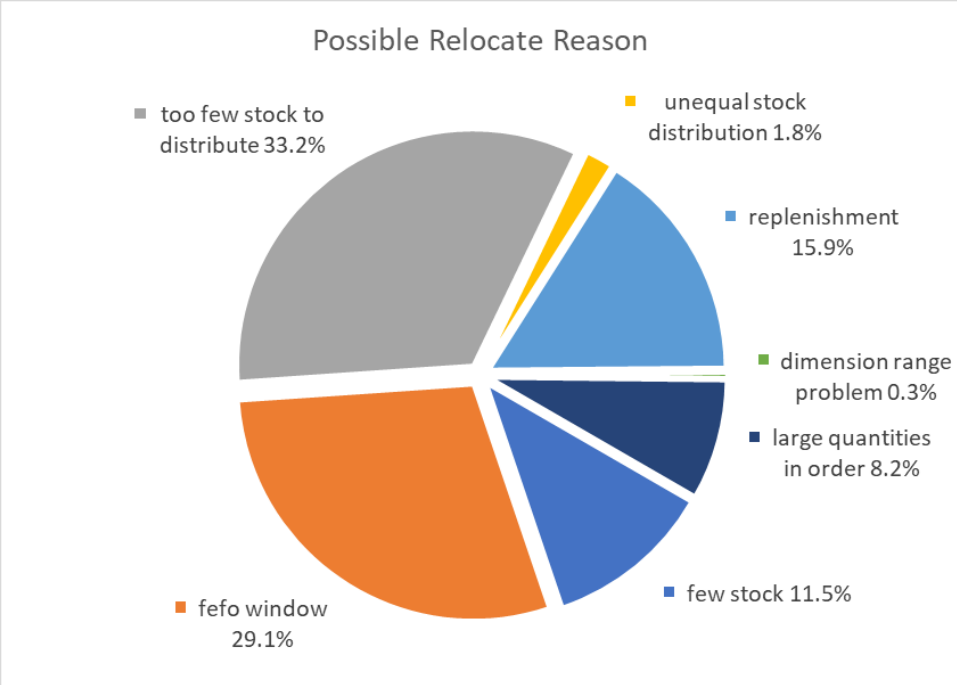
Kakediagrammet for ASRS viser ikke forskjellen mellom sonene eller kranene, bare hvordan hele bildet ser ut. I *Vedlegg 4: ASRS Performance / TU Handled by ASRS* er det en sammenlikning for hver sone og for hver kran, både tidsbruk og brett håndtert av ASRS, i prosentandeler. I sammenlikningen mellom soner er topp- og bunnverdiene markert med henholdsvis grønn og rød ramme. Hvor mye tid hver sone bruker på reallokeringer vises også. Det er lite forskjell i tidsbruk for input og output, men i de tre andre kategoriene er det sprik. Sone 11 bruker mest tid på reallokeringer, 6.7% og sone 16 bruker minst 1.4%. Dette samsvarer godt med hvor mange brett sone 16 reallokerer. Sone 18 er den sonen hvor kranene er minst i ro og som bruker mest tid på input.

4.1.4 Reallokering

Det er to måter å måle reallokeringer på:

- Hele distansen som brett reallokeres regnes som én reallokering. Hvis brett må fra sone 1 til sone 4 telles dette som en.
- Hver interaksjon med brett telles som én reallokering. Hver gang en kran håndterer et brett som reallokeres telles dette som et brett.

Det er tydelig at reallokering tar tid og kapasitet fra kranene. Dette er fordi de må gjøre andre ting enn kun input til brettlageret og output til COM. Reallokeringer er derimot en nødvendighet for å fullføre ordre, og sørger for minimalt med svinn.

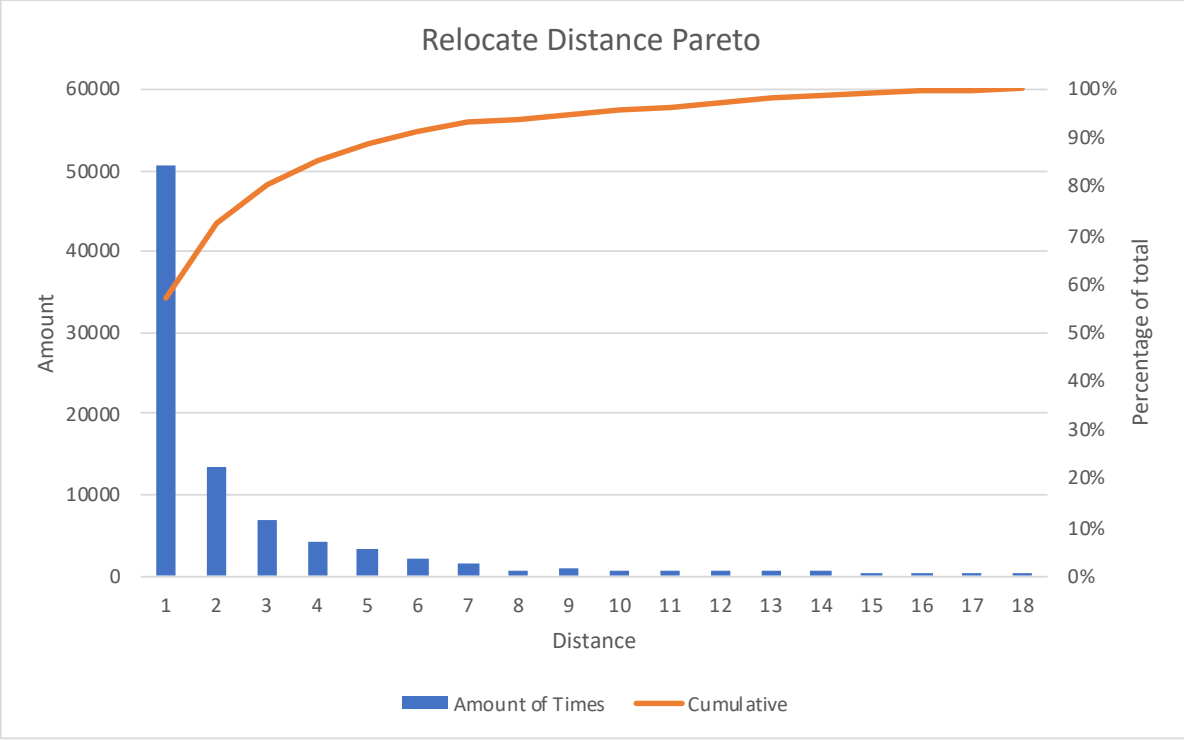


Figur 13: Potensielle årsaker for reallokering

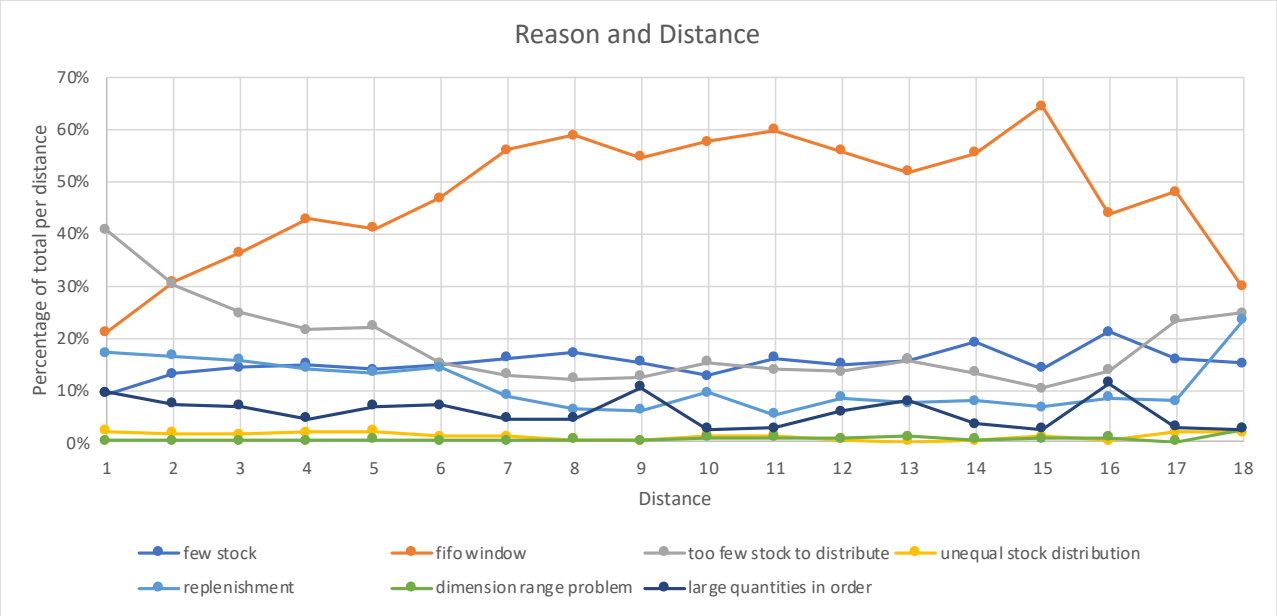
Dette er en oversikt over hvorfor brett har blitt reallokert, men det er viktig å nevne at dette kun er antagelser gjort av WMS. Hovedgrunnene er at bretten som COM trenger ikke finnes i dens sone (too few stock to distribute 33.2%) og på grunn av FEFO-logikken (fefo window 29.1%). Noe reallokering forekommer også fordi ordre har stort antall av enkelte SKU-er (large quantities in order 8.2%) og fordi det er ujevn mengde brett rundt i lageret (unequal stock distribution 1.8%). Avstanden for hvor mange soner et brett reallokeres har et markant flertall ved 1,2 og 3 soner, se *Figur 14: Distanse på reallokeringer og hvor ofte disse inntreffer.* Disse korte avstandene står for 80% av tilfellene, og det viser at alle SKU-er er

veldig jevnt fordelt i hele lageret, og det er sjeldent brett må reallokeres langt. Brett som må reallokeres med en distanse på én skjer ved 57% av tilfellene.

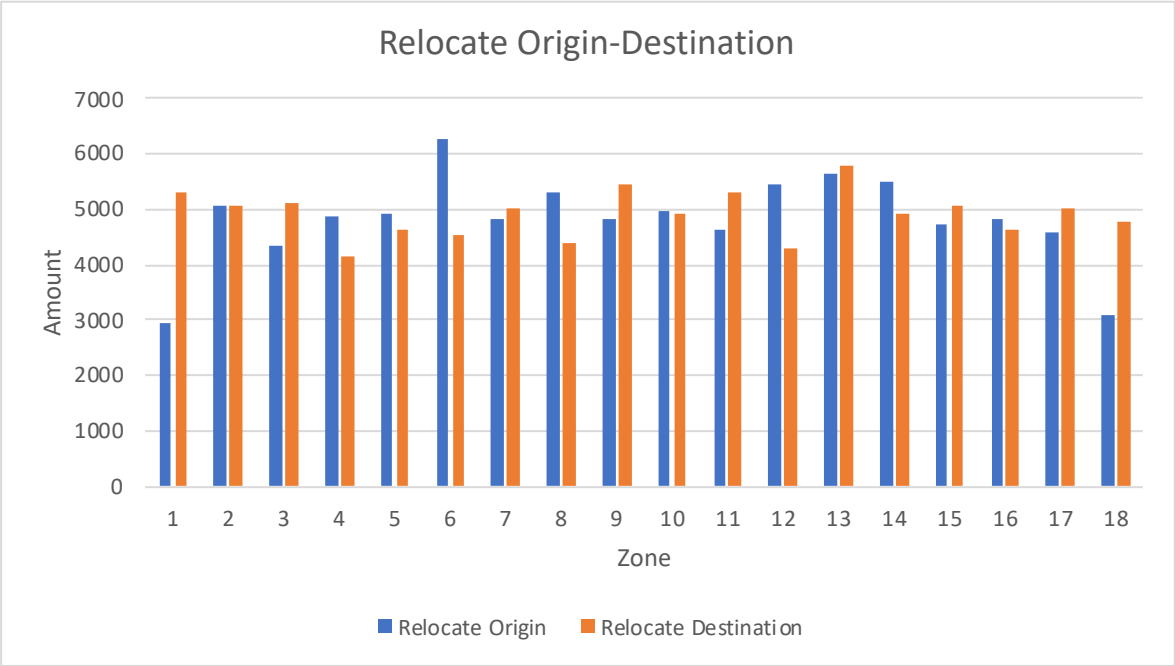
Reallokeringer med en distanse på én kommer hovedsakelig av at COM ikke har riktig brett i sin ASRS-sone, se *Figur 15: Distanse på reallokering med tilsvarende potensiell årsak - Prosentvis for hver distanse.* Dette kommer av at nabosonene som regel har riktig brett. FEFO er grunnen til flest reallokeringer for alle avstander lengre enn én. For alle avstander større enn én er 40% av reallokeringer på grunn av FEFO. 55% av reallokeringer med en avstand på sju eller mer er på grunn av FEFO.



Figur 14: Distanse på reallokeringer og hvor ofte disse inntreffer



Figur 15: Distanse på reallokering med tilsvarende potensiell årsak - Prosentvis for hver distanse



Figur 16: Reallokeringer startsone - sluttsone

Det er en markant forskjell i hvilke soner som sender ut flest brett til reallokering og hvilke soner som kaller på flest brett. Sone 1 og 18 tar imot mange flere brett enn de sender ut, de tar imot 82% og 54% mer. Dette fordi de ligger i ytterkant av lageret og ASRS kranene har ikke mulighet til å være mellomledd i reallokeringene. Sone 6 sender ut mer enn den tar imot, 38% flere brett.

4.1.5 FEFO

FEFO medfører reallokeringer, hele 29% av alle som gjennomføres. Dette trekker ned kapasiteten og forhindrer brett ut til COM. Siden FEFO er så streng som den er kan det hende at kolli må reallokeres fra en side av lageret til den andre, selv om et produkt med lengre holdbarhet er lokalisert i riktig sone. Dette er hovedgrunnen til at FEFO står for flest lange reallokeringer. Logikken er essensiell for å forhindre svinn, men det tar opp kapasitet hos ASRS og det fører til at COM må vente med å plukke.

Det er beregnet at en kran bruker cirka 43 sekunder i løpet av en time på å reallokere på grunn av FEFO. Dette tilsvarer 1.2% av tilgjengelig tid per kran. I løpet av en dag har hver kran i snitt brukt 17 minutter på reallokering av brett på grunn av FEFO. Dette er beregnet ut i fra tid brukt på reallokeringer og hvor mange reallokeringer som gjennomføres på grunn av FEFO.

4.2 Sammenlikning

De dataene som er presentert tidligere er systemets ytelse på generell basis. Videre er det gjennomført en sammenlikning av ytelsen ved lavt og høyt plukkantall. Lavt plukkantall er definert som 20% eller mer under gjennomsnitt og høyt plukkantall er definert som 20% eller mer over gjennomsnitt. Dette er gjort for å få nok datapunkter for å se en trend. Mer enn 20% gir få tidsintervaller og under 20% gir veldig mange.

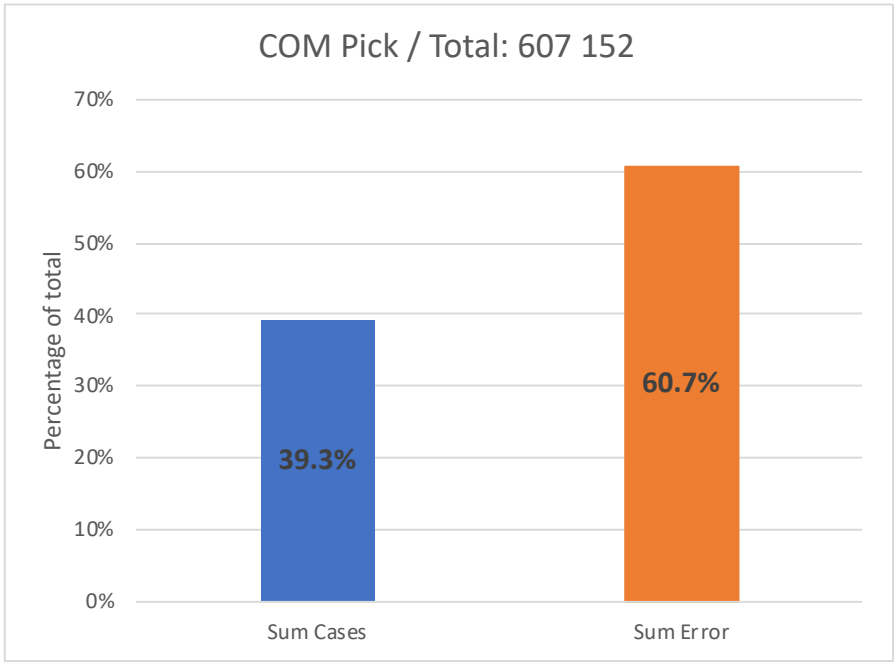
Distanser på reallokering og hvorfor brett er blitt reallokert er ikke tatt med her. Rapporten som inneholder dette ble gjort tilgjengelig etter at sammenlikningsanalysen var gjennomført. Dataene i disse tidsintervallene var slettet da denne rapporten ble gjort tilgjengelig.

4.2.1 Lavt plukkantall

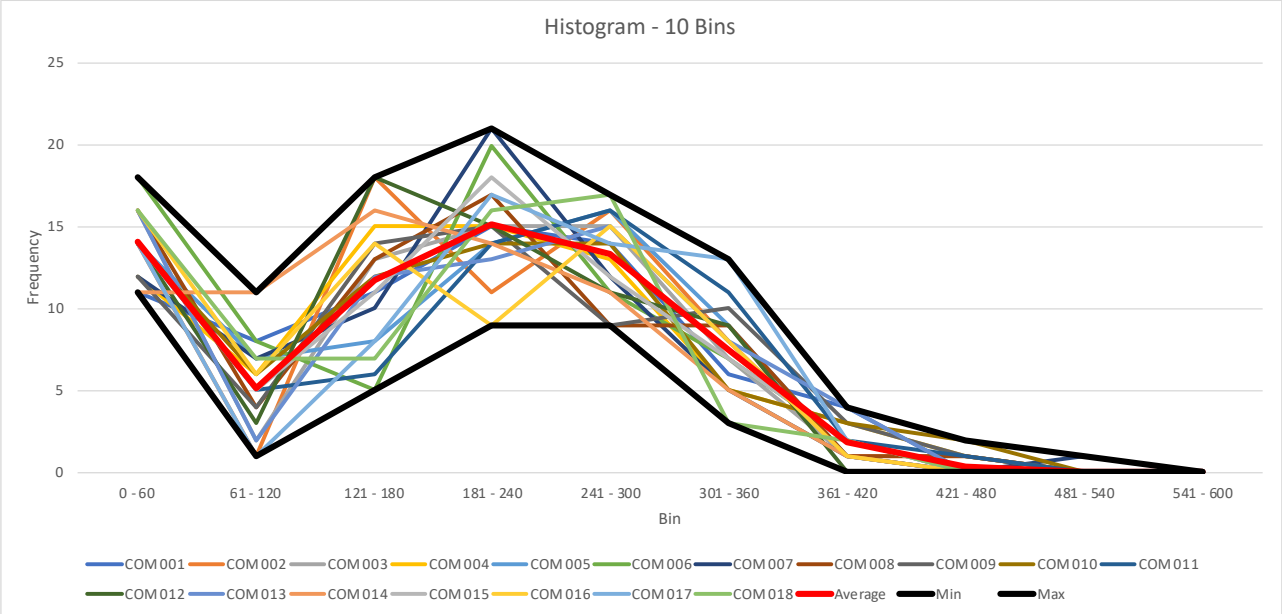
Ved lavt plukkantall stabler COM færre kolli på lastbærer og det er en del annen aktivitet som foregår enn det gjennomsnittlige aktivitetsbildet. Flere av tidsintervallene skjer om natten og når det er få ordre. Det er totalt 69 timer av dataene som betegnes som lavt plukkantall.

I disse tidsintervallene stables det i snitt 184 kolli på lastbærer i COM. Dette er en nedgang på 28% fra gjennomsnittlig drift. Det tilsvarer en utnyttelser på 35% av hva COM har kapasitet til. I stolpediagrammet kan en se at antall forhindrete plukk er mye høyere enn

antall plukk som er gjennomført. 23% av alle brett som ikke er blitt levert til COM er på grunn av «(Wait): COM: Slow tray replenishment» og 25% av all forhindret plukk er fordi det ikke er ordre. Det er heller ikke her noen statistisk fordeling på antall plukk i COM. Naturligvis er det en stor konsentrasjon av datapunkter ved lave bins.

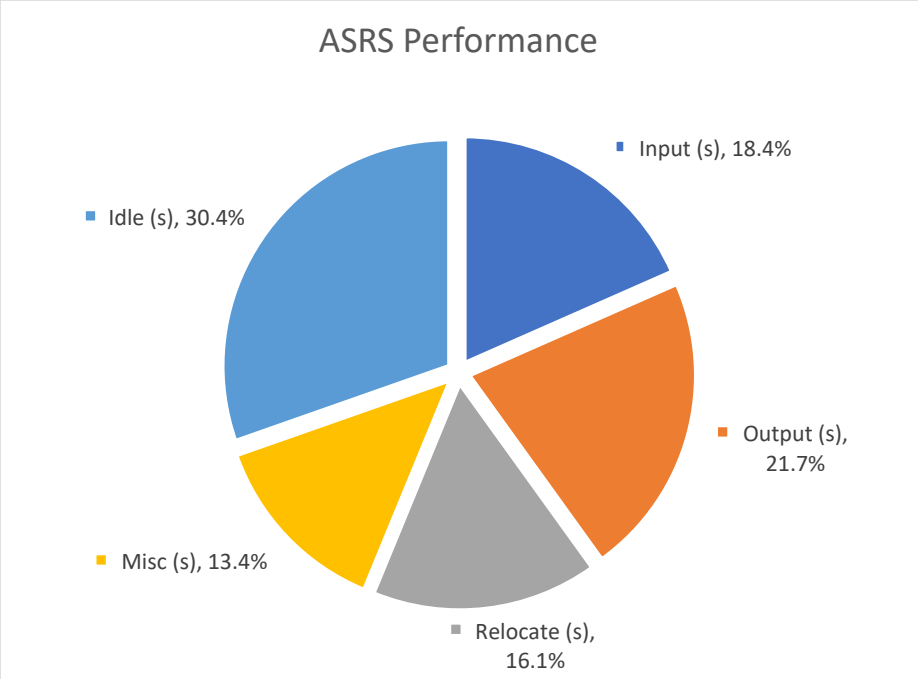


Figur 17: Pukk i COM - lavt plukkantall

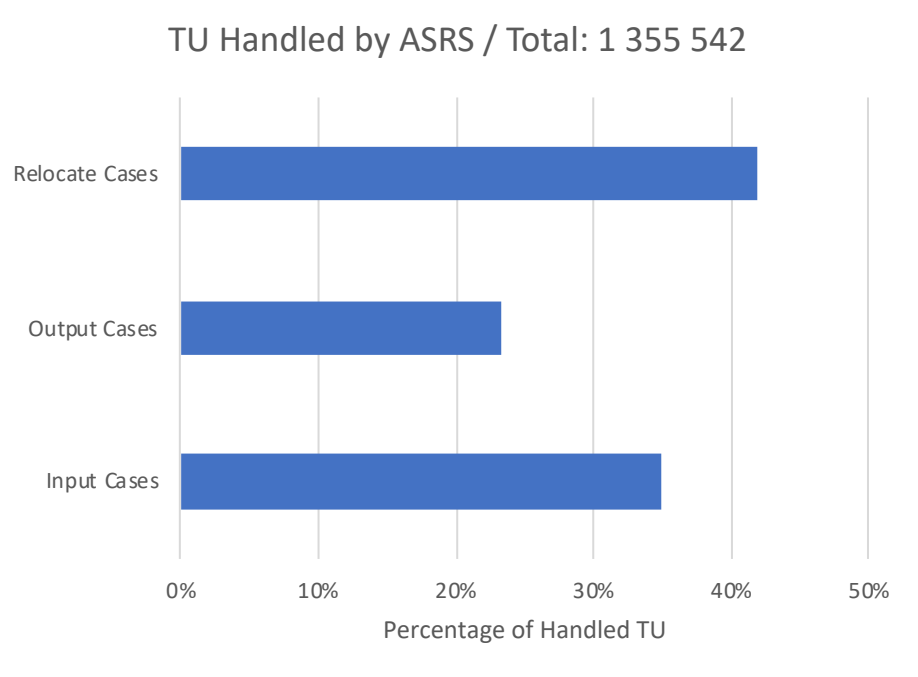


Figur 18: Histogram plukk i COM - 10 Bins - lavt plukkantall

I snitt bruker alle ASRS 15% mindre tid på å levere brett til COM, de er mer i ro og bruker nesten like mye tid på input av brett. De bruker fire ganger så mye av tiden sin til å reallokere mellom sonene, for å klargjøre nye ordre og for å jevne ut beholdningen i lageret. ASRS håndterer færre brett til output, men flere brett til reallokering. 40% av alle håndterte brett var reallokeringer, en økning på 23% fra snittet.



Figur 19: ASRS ytelse - lavt plukkantall

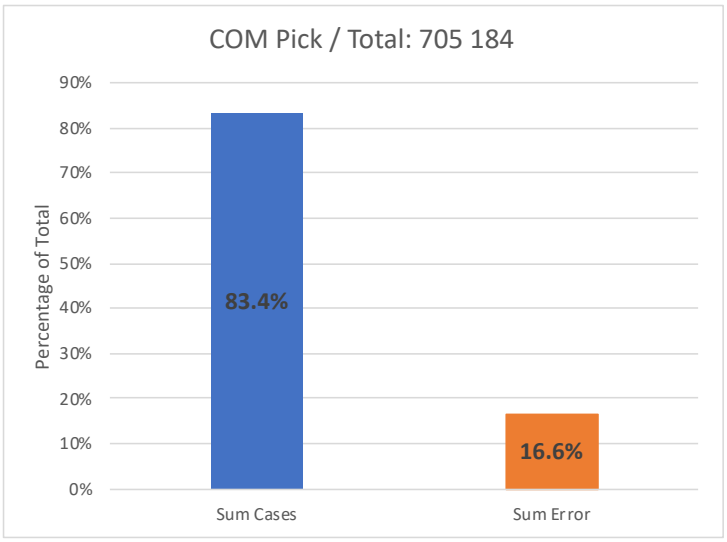


Figur 20: Brett håndtert av ASRS - lavt plukkantall

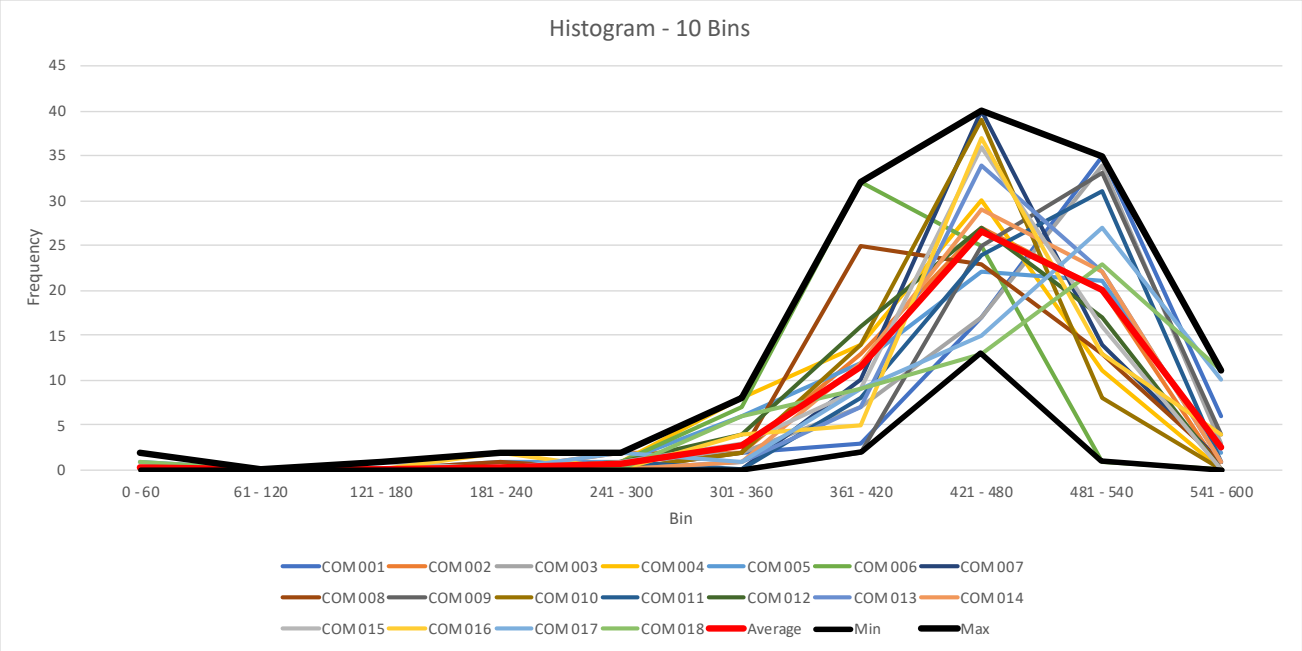
4.2.2 Høyt plukkantall

Ved høyt plukkantall stabler COM mer kolli på lastbærer enn ved gjennomsnittlig drift. De fleste tidsintervallene her er om ettermiddagen og om natten. Det er totalt 72 timer av dataene som betegnes som høyt plukkantall

Det stables i snitt 453 kolli per time per COM, en økning på 28% fra gjennomsnittlig drift. I stolpediagrammet ser en tydelig at antall plukk er mye høyere enn antall forhindrete plukk. Ved høyt plukkantall utnyttes 87% av kapasiteten til COM. Av de cirka 117 000 kolli som COM ikke har fått, er det «(Wait): COM: Slow tray replenishment» som står for 23% av disse og feil i COM står for 17% av tilfellene. Plukk har ingen statistisk distribusjon ved høyt plukkantall, men toppen av histogrammet er forskjøvet kraftig mot høyre.

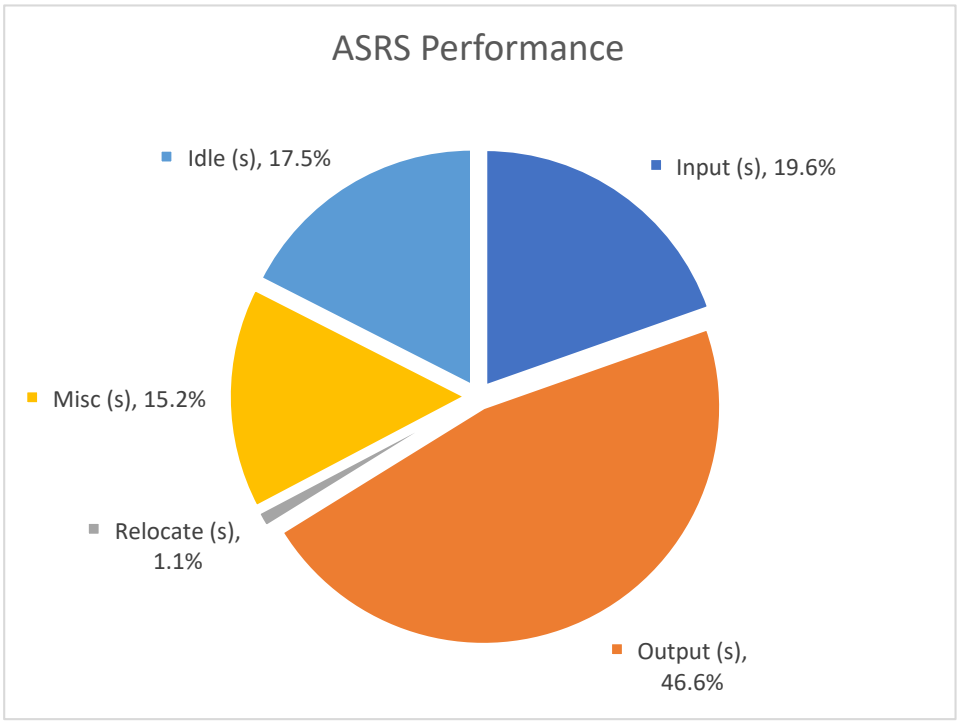


Figur 21: Plukk i COM - høyt plukkantall

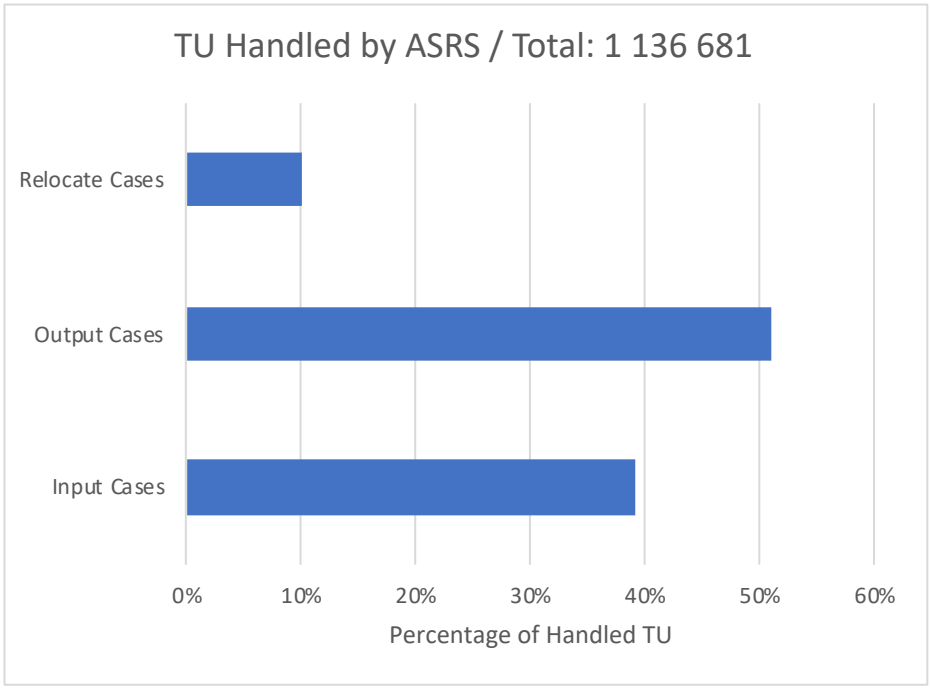


Figur 22: Histogram plukk i COM - 10 Bins - høyt plukkantall

ASRS bruker mye tid på å levere brett til COM, 10% mer enn snittet, og nesten like mye tid på input som snittet. Kranene er mye mindre i ro og reallokerer mye mindre, 1.1%. Dette er ¼ av gjennomsnittlig tid brukt på reallokeringer. Naturligvis er kranene mindre i ro da de leverer mye mer brett ut til COM. 51% av alle håndterte brett ble sendt ut til COM i disse tidsintervallene.



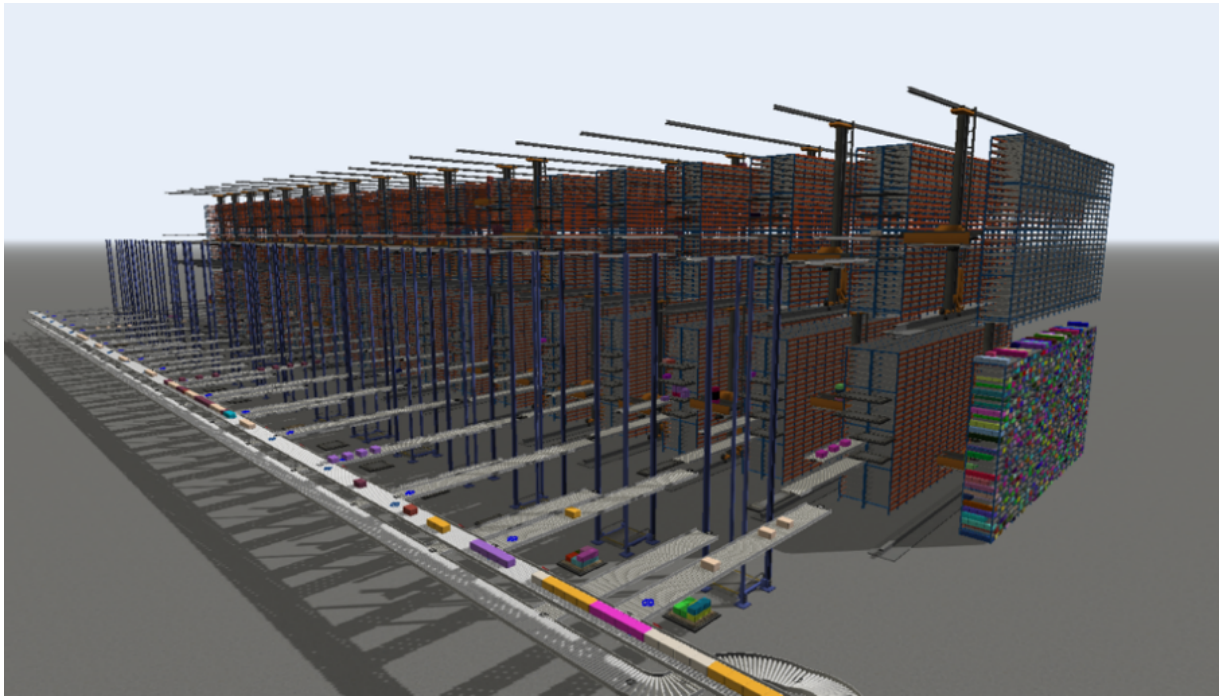
Figur 23: ASRS ytelse - høyt plukkantall



Figur 24: Brett håndtert av ASRS - høyt plukkantall

5.0 3D-Computer Simulation

3D-simulering er en måte å imitere hendelsesutfall i en grafisk 3D-modell, basert på fysiske parametere og tekniske spesifikasjoner. Ved hjelp av simulering kan det bli enklere å ta viktige avgjørelser basert på det som visuelt skjer, ved hjelp av grafer som oppdateres underveis i simuleringen, eller annen statistikk som genereres fra simuleringen. Simulering brukes blant annet for å analysere materialflyt, avsløre flaskehalser, planlegge nye fabrikker eller for å undersøke forskjellige variasjoner av et system. En simulering viser klart og tydelig hvordan et konsept kan se ut i virkeligheten, hvordan det presterer under forskjellige forutsetninger og hvordan endringer påvirker prosessene.



Figur 25: Eksempel på en 3D-simulering

Innenfor materialhåndtering er det avanserte og komplekse systemer som ikke lar seg modelleres i regneark eller ved lineær programmering, men dette er noe en simulering klarer utmerket. I komplekse regneark og lineær programmering kan det være utfordrende å inkludere alle forskjellige variasjoner og tilfeldigheter som kan oppstå. En matematisk modell inneholder ofte feil, og det er vanskelig å beregne hvordan hendelsesutfallet fra en komponent påvirker den neste komponenten. Et komplekst regneark klarer ikke å ta for seg alle tilfeldigheter, og aktivitetene som oppstår på grunn av dette. En simulering kan kjøres så mange ganger det er nødvendig for å få med alle utfall som kan oppstå. En kan kjøre samme

simulering så mange ganger at den dekker alle tilfeldigheter som oppstår og alle variasjoner som finnes.

I en simulering settes fysiske komponenter sammen og det vises hvordan disse påvirker og samhandler med hverandre. Alle komponentene blir sett på hver for seg og det blir implementert hvordan disse opptrer og hvilke interaksjoner de skal ha. En simulering ser på en komponent og hvordan hendelsesutfallet fra denne påvirker neste steg i modellen.

3D-visualiseringen av en modell viser hvordan komponenter opptrer og hvordan de samhandler med hverandre. I tillegg til denne fremstillingen av et system brukes det ofte grafer for å vise hvordan systemets prosesser yter. Dette gir en god visuell og klar oppfatning av ytelsen til modellen. Grafer gir også et godt innblikk i hvilke komponenter som er flaskehalser og hvilke prosesser som gjøres oftest. Slike grafer oppdateres kontinuerlig mens simuleringen gjennomføres og det gjør det mulig å se når eventuelle stopp, bunn- eller toppunkt oppstår.

Det er lett å endre måten en simulering opererer, fungerer og ser ut. Endringer i en simulering tar ikke mye tid og medfører ingen kostnader. Ved å kunne gjøre endringer tidseffektivt og kostnadsfritt er det enkelt å analysere forskjellige variasjoner av et system. Det er enkelt å se hvilke komponenter som er kritiske for systemet, hvilke som ikke trengs og hva det eventuelt trengs mer av. På denne måten er det enkelt å planlegge bygging av nye anlegg eller utbygging av et eksisterende anlegg. Ved å bruke 3D-simulering i planleggingsfasen av et prosjekt vil det være lettere å gjøre endringer og optimalisere systemer før byggingen har startet.

Det er ikke bare fysiske endringer som er lett å gjennomføre, logikk er også enkelt å endre på. Ved endring av logikk vil en kunne se hvordan logiske parametere påvirker systemet i sin helhet, uten at det går utover det reelle systemet. Logiske parametere er en viktig del av et hvert system da det påvirker vareflyten direkte. Logikken i en simulering brukes for å bestemme regler og vareflyten til objekter som må gå gjennom de fysiske komponentene. Logikk bestemmer også hvordan fysiske objekter skal oppføre seg. For eksempel prosesstid på en maskin eller hvor ofte en feil oppstår.

Ved å kunne simulere hvor ofte feil oppstår, vedlikehold, pauser og andre hendelser som oppstår tilfeldig eller regelmessig, vil en få god forståelse for hvordan en reell versjon av systemet vil fungere. (Malinovskaya, 2021)

Flere cases fra virkeligheten viser godt hvordan simulering kan brukes og hva utfallet ved bruk av et slikt verktøy er. I (Malinovskaya, 2021, pp. 13-22) snakkes det om to cases hvor simulering er brukt:

- Bruk av simulering for å bestemme hvordan et nytt lager skal bygges. Hvilken teknologi som skal brukes og hvordan denne yter basert på reelle hendelser.
- Bruk av simulering for å finne flaskehalser og områder med lav effektivitet hos en metallprodusent.

Disse to eksemplene viser godt hvordan simulering er et effektivt og nøyaktig verktøy for å underbygge viktige beslutninger og for å løse komplekse oppgaver. I begge casene ble resultatet fra simuleringen brukt for å implementere nye løsninger.

5.1 FlexSim

FlexSim er et avansert 3D-simuleringsprogram som inneholder en rekke ulike geometriske verktøy, objekter og preprogrammerte aktiviteter som gjør det enkelt for brukeren å bygge tilpassede modeller. FlexSim er et godt verktøy for å simulere og analysere prosesser innenfor blant annet produksjon, materialhåndtering, helsevesenet, lagring, fabrikker, flyplasser og containerterminaler. Layouts i modellen kan baseres på Computer Aided Design (Heretter CAD) som gir muligheten til å gjenskape nøyaktige replikaer av det som skal modelleres. Reell data kan importeres til datatabeller i FlexSim, som for eksempel kan styre inngående varer i en modell. Som følge av programmets høye grad av tilpasningsmuligheter, kan modeller konstrueres slik at brukeren får akkurat den informasjonen som trengs. (FlexSim, 2021a)

FlexSim bruker et objekt-orientert design, som vil si at programmet baseres rundt objekter som innehar egenskaper og prosedyrer. Forskjellige objekter har forskjellige egenskaper og prosedyrer. Enkelte egenskaper og prosedyrer er justerbare for brukeren, slik at brukeren kan tilpasse objektene etter eget ønske. Andre egenskaper og prosedyrer er vanskeligere for brukeren å endre for å sikre funksjonaliteten til objektene. Tilpasningsmulighetene for brukeren handler i stor grad om å endre egenskapene og prosedyrene til objektene for å oppnå ønsket resultat.

Simuleringer av prosesser inneholder ofte statistiske distribusjoner. FlexSim har implementert Random Streams for alle statistiske distribusjoner, som vil si at en statistisk

distribusjon vil gi de samme tallene hver eneste gang en simulering blir kjørt, så lenge Random Streams er skrudd på. Dette medfører at simuleringer er repeterbare, som gjør det enklere å feilsøke en modell.

5.1.1 3D-Modellering

3D-modelleringsdelen av programmet består blant annet av (FlexSim, 2021b):

- ressurser som kan lage, endre og behandle flytobjekter
- objekter som kan utføre oppgaver og samhandle med flytobjekter og ressurser
- samlebånd og verktøy for å kontrollere vareflyten på samlebånd
- lagerobjekter for modellering av lager
- trafikknettverk som kontrollerer hvordan bevegende objekter oppfører seg.

Det er også mulig å lage egendefinerte objekter som kan programmeres slik brukeren ønsker det. Den visuelle fremstillingen av objekter kan tilpasses ved å endre eksisterende fremstillinger i FlexSim, eller ved å importere grafiske modeller fra andre programmer. Oppbygningen av ressurser og objekter som kan utføre arbeid, gjør det enkelt å gjenskape handlinger til ekte maskiner i programmet. Individuelle prosesstider for maskiner kan baseres på reell data, blant annet gjennom importerte datatabeller eller ved statistisk distribusjon. Det er kun brukerens ferdigheter og kreativitet, som begrenser hva som kan modelleres i FlexSim.

5.1.2 Statistikk

Gjennom en simulering generer FlexSim en stor mengde statistikk som kan visualiseres i Dashboard verktøyet (FlexSim, 2019a), eller eksporteres til andre programmer for analyse, som for eksempel til regneark programmer. Dashboard verktøyet inneholder en rekke ulike standard formater for visualisering av statistikk, samt mange muligheter for å tilpasse fremvisningen av statistikk for brukeren. Det er også mulig å gjennomføre analyser av den genererte statistikken ved å bruke kalkulerte tabeller som kan utføre avanserte matematiske kalkulasjoner.

5.1.3 Experimenter

Når en modell er ferdig utviklet og er klar for å testes har FlexSim et Experimenter verktøy som gjør det mulig å teste flere scenarier samtidig, i tillegg til å teste et scenario flere ganger for å undersøke hvordan fysiske endringer, logiske endringer og statistisk distribusjon påvirker prosessen. Denne funksjonen gjør det mulig å raskt analysere hvilken verdi for foranderlige parametere som gir en optimal løsning for en prosess. (FlexSim, 2019b)

5.1.4 Manuell Programmering

FlexSim støtter programmeringsspråkene C++ og FlexSims eget programmeringsspråk FlexScript, i tillegg til SQL spørringer for enkelte funksjoner i FlexSim. Som følge av FlexSims integrerte muligheter for tilpasning av modeller og logikk, er det som regel ikke nødvendig med koding fra brukerens side, men muligheten er der dersom brukeren ønsker enda mer fleksibel funksjonalitet. (FlexSim, 2019c)

5.2 Process Flow

Process Flow verktøyet i FlexSim gir i likhet med andre prosessflytverktøy, muligheten til å enkelt visualisere logikken og hendelsesforløpet til en prosess. I motsetning til vanlige prosessflytverktøy derimot, kan Process Flow også styre logikken til modellen direkte gjennom aktiviteter som kan tilpasses etter brukerens behov. Process Flow kan altså brukes som både et visuelt verktøy, og for å programmere logikk for en 3D-modell ved å koble ressurser og aktiviteter i Process Flow til 3D-objekter.

Formålet med Process Flow i FlexSim er å gjøre det enklere og raskere å utvikle avansert logikk for modeller. For eksempel kan det enkelt lages en felles, tilpasset logikk for 10 like 3D-objekter i Process Flow, i stedet for å tilpasse logikken for hvert enkelt objekt i 3D-modellen. Aktivitetene i Process Flow er kodeblokker, hvor brukeren enkelt kan kontrollere variablene som aktiviteten inneholder. Det er i utgangspunktet programmering uten behovet for å skrive koden selv, og åpner for flere tilpasningsmuligheter enn hva som er tilgjengelig i 3D-modelleringsdelen av programmet. Process Flow kan anses som hjernen til en 3D-modell, hvor aktivitetene og ressursene i Process Flow kontrollerer logikken i 3D-modellen.

Prosess Flow gjør det også mulig å simulere prosesser uten å bygge en 3D-modell. Det er fullt mulig å utvikle identiske simuleringer gjennom 3D-modellering og modellering i Process Flow. 3D-modellering har derimot den fordelen at det visuelt viser hva som skjer i en simulering, og gjør det enklere å tilpasse en modell etter en virkelig prosess. I FlexSim er Process Flow godt integrert i byggingen av 3D-modeller, og kombinasjonen av 3D-modellering og logikktilpasning i Process Flow resulterer i et robust simuleringsprogram.

5.3 Modellens oppbygging

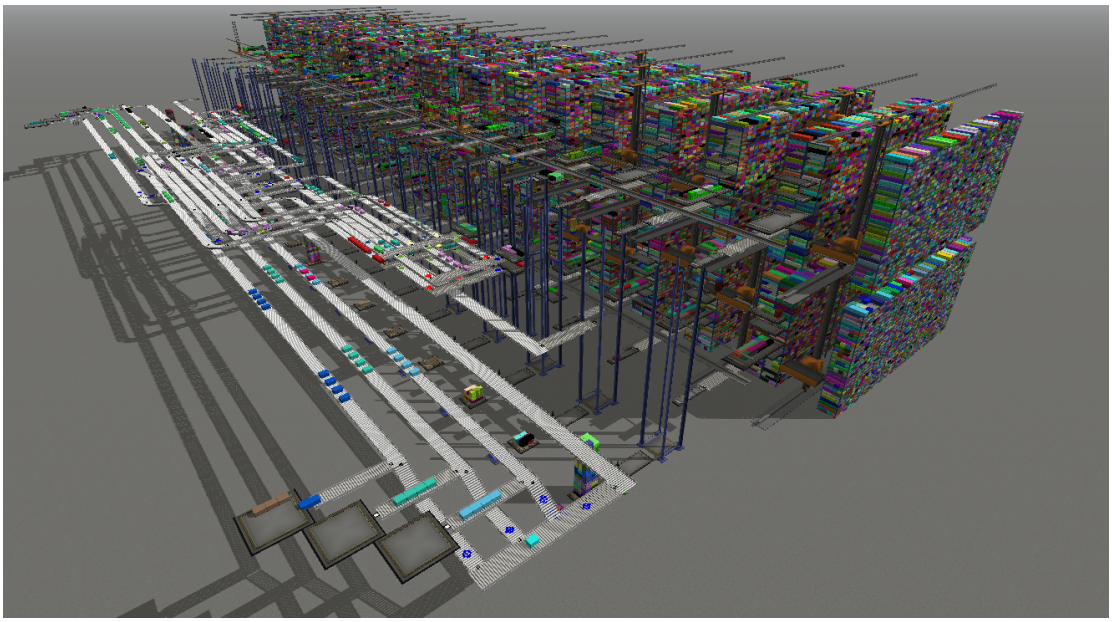
Simuleringsmodellen som er bygget i FlexSim er en forenklet modell av store deler av OPM-området til ASKO SL Kjøl. Modellens oppbygging er fokusert rundt de objektene og den logikken som kreves for at endringer i logiske parametere i modellen skal gi et realistisk resultat av hvordan endringer av logiske parametere hos ASKO SL Kjøl vil påvirke systemet deres. 3D-modelleringen er basert på CAD fra ASKO SL Kjøl, som vil si at 3D-modellens dimensjoner er relativt like lagerets dimensjoner. 3D-modellen inkluderer alle de fysiske komponentene av lageret som er sentrale for å gjenskape en god replika av lageret.

Modellen bruker datatabeller som er hentet fra ASKO SL Kjøl sitt WMS for å styre inngående kolli, startbeholdning og generering av ordre. All annen logikk i modellen, som hvilket kolli av en gitt SKU som skal velges for en ordre, er programmert ut i fra tilegnet kunnskap gjennom analyser av produksjonsdata, og samtaler med kontaktpersoner og ansatte ved ASKO SL Kjøl og WITRON. Som følge av at inngående kolli og ordregeneringer i modellen er reelle produksjonsdata fra ASKO SL Kjøl, vil det være relativt enkelt å validere modellen.

Generelt sett er modellens objekter og geometri bygget i 3D-modelleringsdelen av FlexSim, og logikken for vareflyt, informasjonsflyt, maskiner og lagring bygget i Process Flow.

5.3.1 3D-Modellen

3D-modellen inkluderer de sentrale fysiske komponentene fra ASKO SL Kjøl sitt lager som kreves for å gjenskape en god simulering av hvordan logiske parametere påvirker vareflyten og ytelsen til OPM-området.

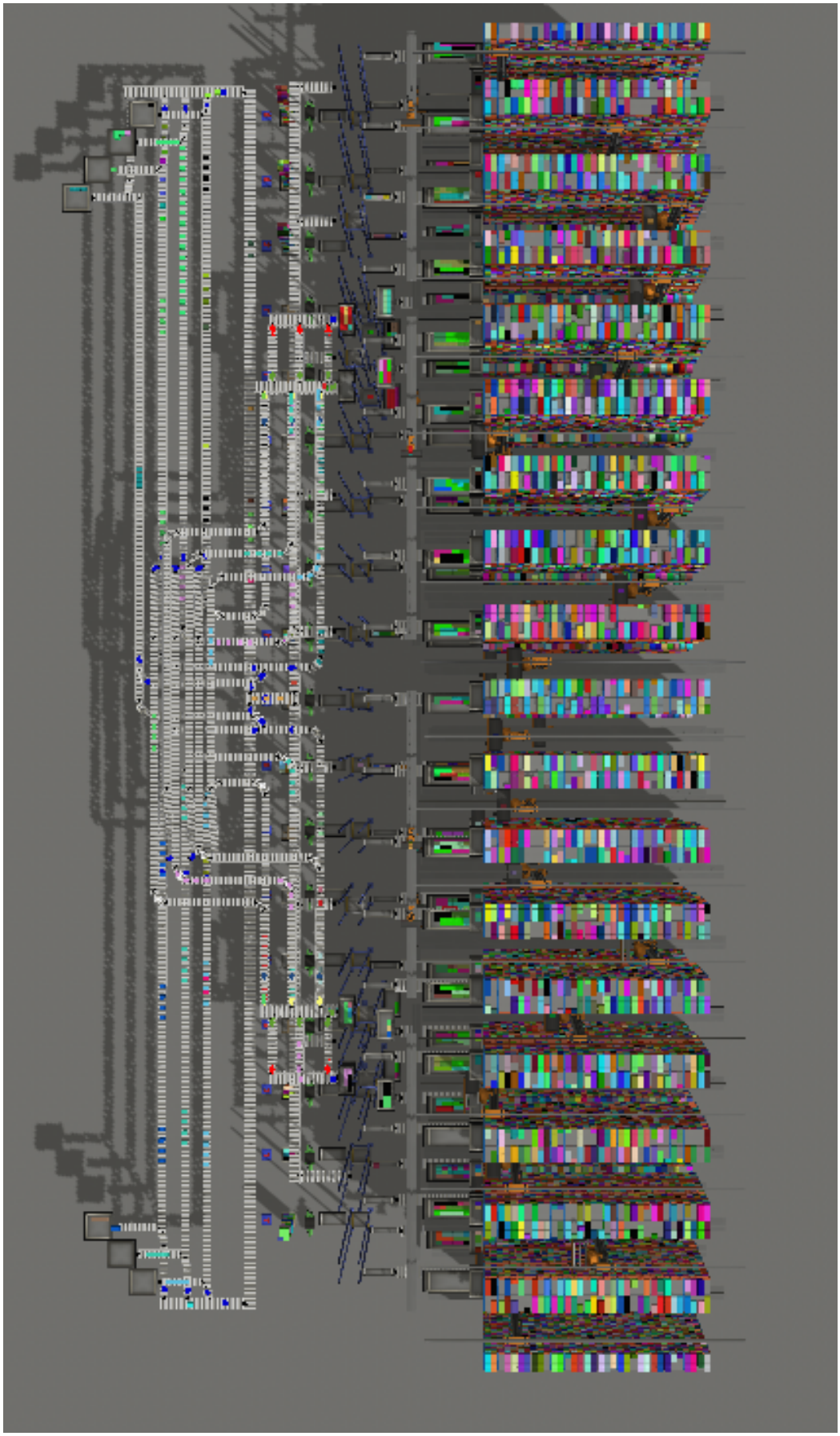


Figur 26: 3D-Modellen i sin helhet

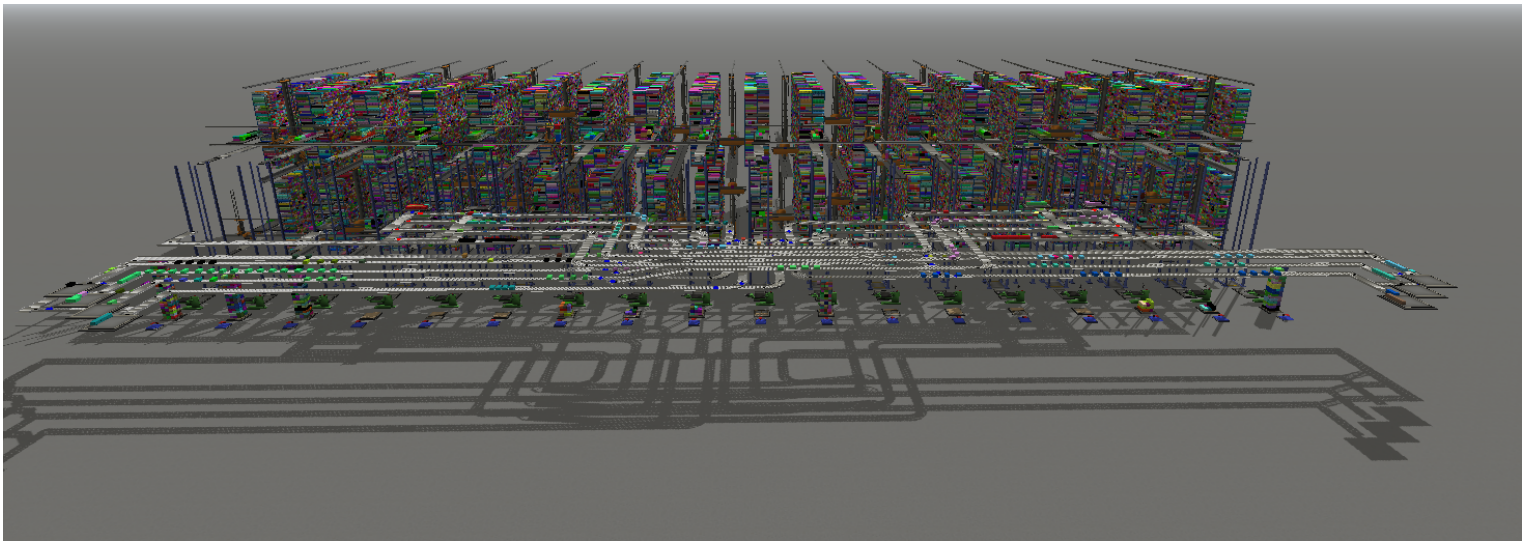
3D-modellen består av:

- 7 forenklede depalleterere som representerer starten for inngående vareflyt i simuleringen.
- Inngående samlebåndsnettverk fra depalleterere til TIC Builder.
- 8 forenklede TIC Builders med tilhørende heiser og TC.
- 36 Miniload ASRS
- Heiser for utgående varer fra alle ASRS-soner, med utgående samlebånd til COM, og utgående samlebånd til alternativ rute for lange reallokeringer for annenhver sone.
- 18 COM

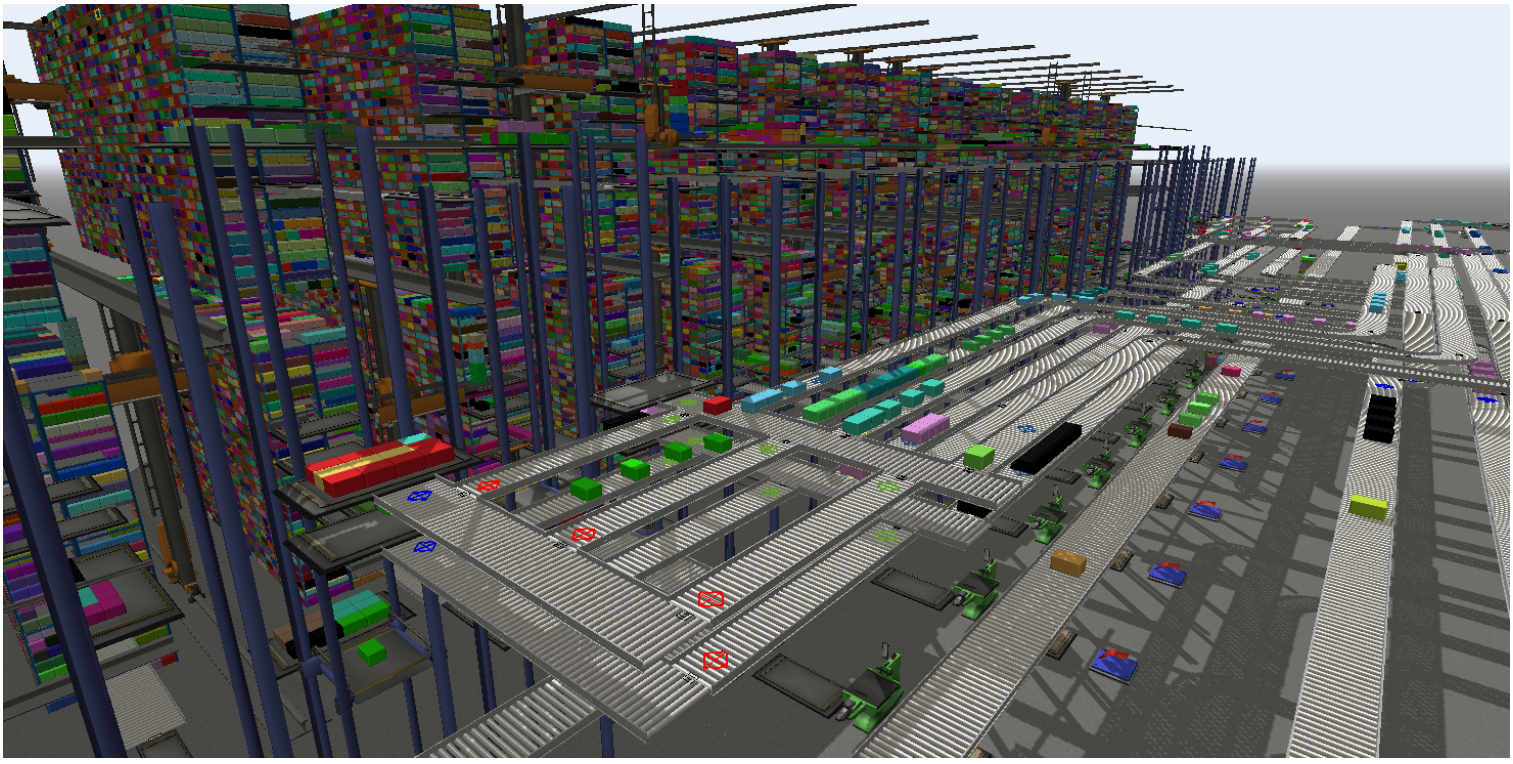
Modellen er delt inn i 18 soner. Hver sone består av en COM, to Miniload ASRS, en heis for utgående kolli, og samlebånd mellom nevnte objekter. Samlebåndsnettverket for inngående kolli ble inkludert i modellen, fordi reallokerte kolli som tar den lange alternative ruten går inn i det inngående samlebåndsnettverket, og vil dermed kunne forårsake flaskehalser for inngående vareflyt dersom for mange kolli tar den alternative reallokeringsruten samtidig.



Figur 27: 3D-Modellen Ovenfra



Figur 28: 3D-Modellen Forfra



Figur 29: 3D-Modellens TIC Builder

5.3.2 Importert produksjonsdata

Modellens startbeholdning, innkommende kolli og ordre generering er basert på enkelte av rapportene som ble beskrevet i kapittel 4. Innkommende kolli i modellen er direkte hentet fra rapport OP48c PAL INFO, som inneholder data for alle produkter som har gått ut av depalleterne. Innkommende kolli i modellen skal dermed i utgangspunktet være helt likt innkommende kolli hos ASKO SL Kjøl for samme periode. Ordre generering i modellen er direkte hentet fra OM29 Order-Overview, slik at ordrene som genereres i modellen er helt like ordrene som har blitt produsert hos ASKO SL Kjøl for samme periode. Tidspunktene for når ordre er ferdig i modellen vil variere fra de reelle produksjonsdataene basert på ytelsen til modellen. Det er importert nok data for å kunne simulere 10 dager. Forklaring på hva disse rapportene inneholder kan ses i kapittel *4.0 Driftsdata.*

Det har ikke vært mulig å eksportere lagerbeholdning for et gitt tidspunkt fra ASKO SL Kjøl sitt WMS, og dermed er startbeholdningen for simuleringen kalkulert fra OM29 og OP48c. Startbeholdningen er regnet ut ved å ta data for innkommende kolli fra OP48c for cirka 6.5 dag minus kolli i ordre fra OM29 for fem dager. Det er så blitt sett på utgående kolli minus inngående kolli for de resterende 10 dagene som blir simulert for å se om alle produkter som går ut også går inn. De produktene som ikke finnes i tidsintervallet men som går ut er plusset på i startbeholdningen. Dette vil si at alle produkter som har ligget i lageret lengre enn 6.5 dager før simuleringen begynner ikke er med i startbeholdningen. Avvik fra ordre vil dermed forekomme ved enkelte tilfeller hvor ordre skal ut før alle kolli som trengs for den ordren er kommet inn i systemet.

5.3.3 Modellens Logikk

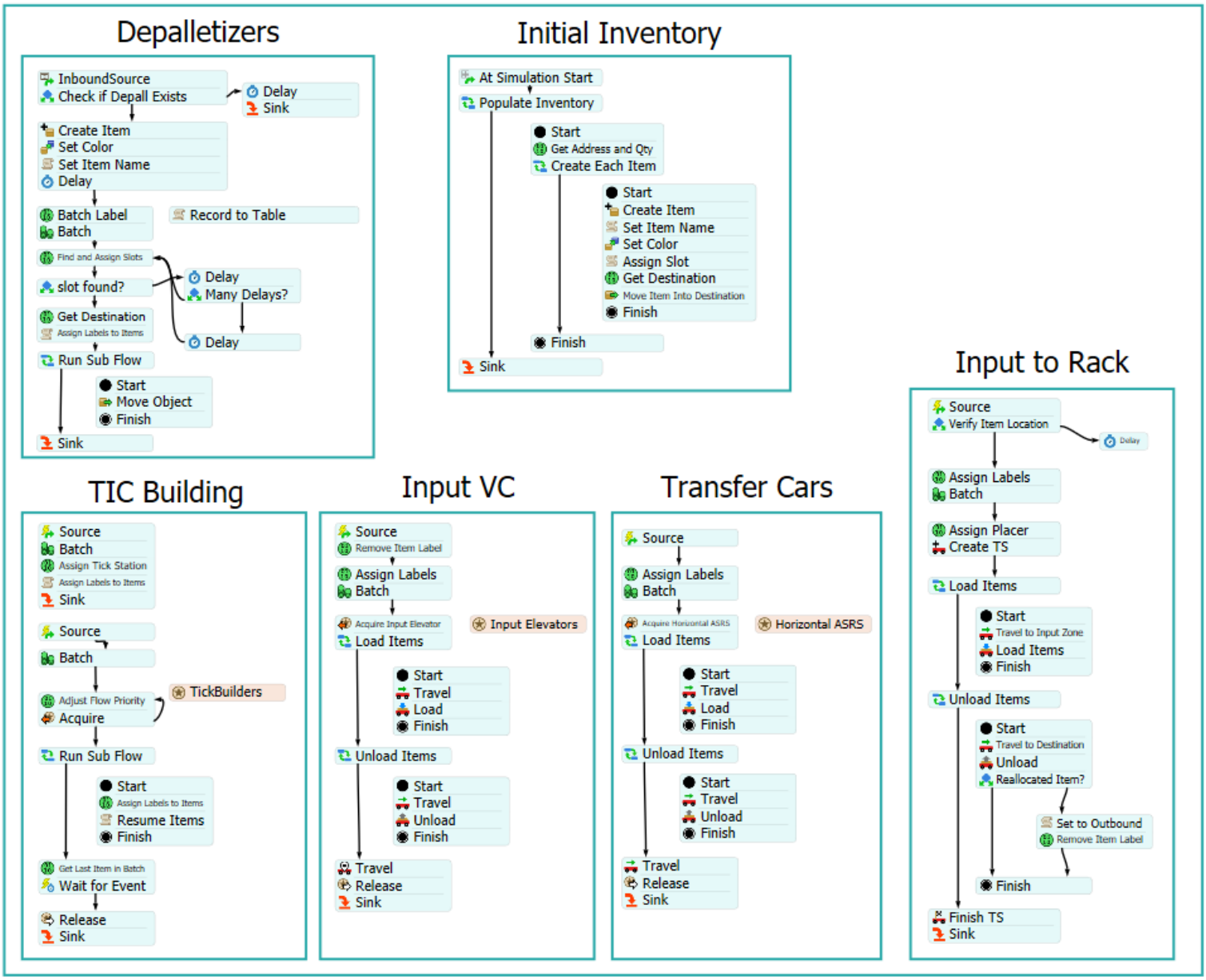
Modellens logikk styres i all hovedsak ved bruk aktiviteter og ressurser i Process Flow som er koblet til objekter i 3D-modellen. Det er tatt i bruk manuelt kodede FlexScript scripts for tilpasset logikk der det har vært bruk for det. I tillegg er det brukt Decision Points, som er et verktøy for å styre vareflyt på samlebånd, sammen med Process Flow for å styre vareflyten på samlebånd.

I denne modellen er Process Flow delt inn i to hovedkategorier, Inbound og Outbound, samt flere underkategorier. Inbound styrer vare- og informasjonsflyt for inngående kolli, og Outbound styrer vare- og informasjonsflyt for utgående kolli. Underkategoriene og deres respektive oppgaver er vist i *Tabell 6.*

Hovedkategori	Underkategori	Hensikt/Oppgave
Inbound	Initial Inventory	Leser av tabell med lageraddresse, SKU, antall og utløpsdato for kolli som er i startbeholdningen, og lager disse kolliene på riktig sted.
Inbound	Depalletizers	Lager inngående kolli i riktig depalleterer til riktig tid. Finner ledig lageraddresse til kolli ved bruk av scriptet findSlotsForBatch().
Inbound	TIC Building	Styrer vareflyten på samlebåndene i området hvor TIC-er lages. Process Flow kommuniserer her med Decision Points.
Inbound	Input VC	Styrer inngående heiser, slik at heisene laster riktig kolli på riktig sted, og leverer kolli på riktig sted.
Inbound	TC	Styrer transfer cars på samme måte som i Input VC.
Inbound	Input to Rack	Styrer ASRS-kranenes logikk for innkommende kolli.
Outbound	Order Generation	Ordre leses av fra tabell, med hvilke SKU som skal være i ordren og antall av hver SKU. Ordren blir tildelt en COM basert på lagerbeholdningen ved det tidspunktet med scriptet findCOM().
Outbound	Fill Out Individual Picks	Ordre fylles med kolli basert på SKU, antall, utløpsdato og tildelt COM. Bruker scriptet findItemNearCOM(). Dersom kolli mangler sjekkes det om ordren kan utsettes med checkInboundItems() scriptet.
Outbound	Reallocations	Dersom et eller flere kolli i en ordre ikke er i samme sone som den tildelte COM-en, må disse kolliene reallokeres. Denne underkategorien finner ut hvor langt et kolli skal reallokeres, hvilken reallokeringsrute som må benyttes, i tillegg til å styre alle kranene som involveres i en reallokering.
Outbound	Order Picking	Styrer plukking av ordre når en ordre er klar til å plukkes. Begrenser plukking til maksimalt én ordre per COM til enhver tid.
Outbound	COM Elevators	Styrer utgående heiser. Sjekker om kolli skal til COM eller til lang reallokeringsrute.

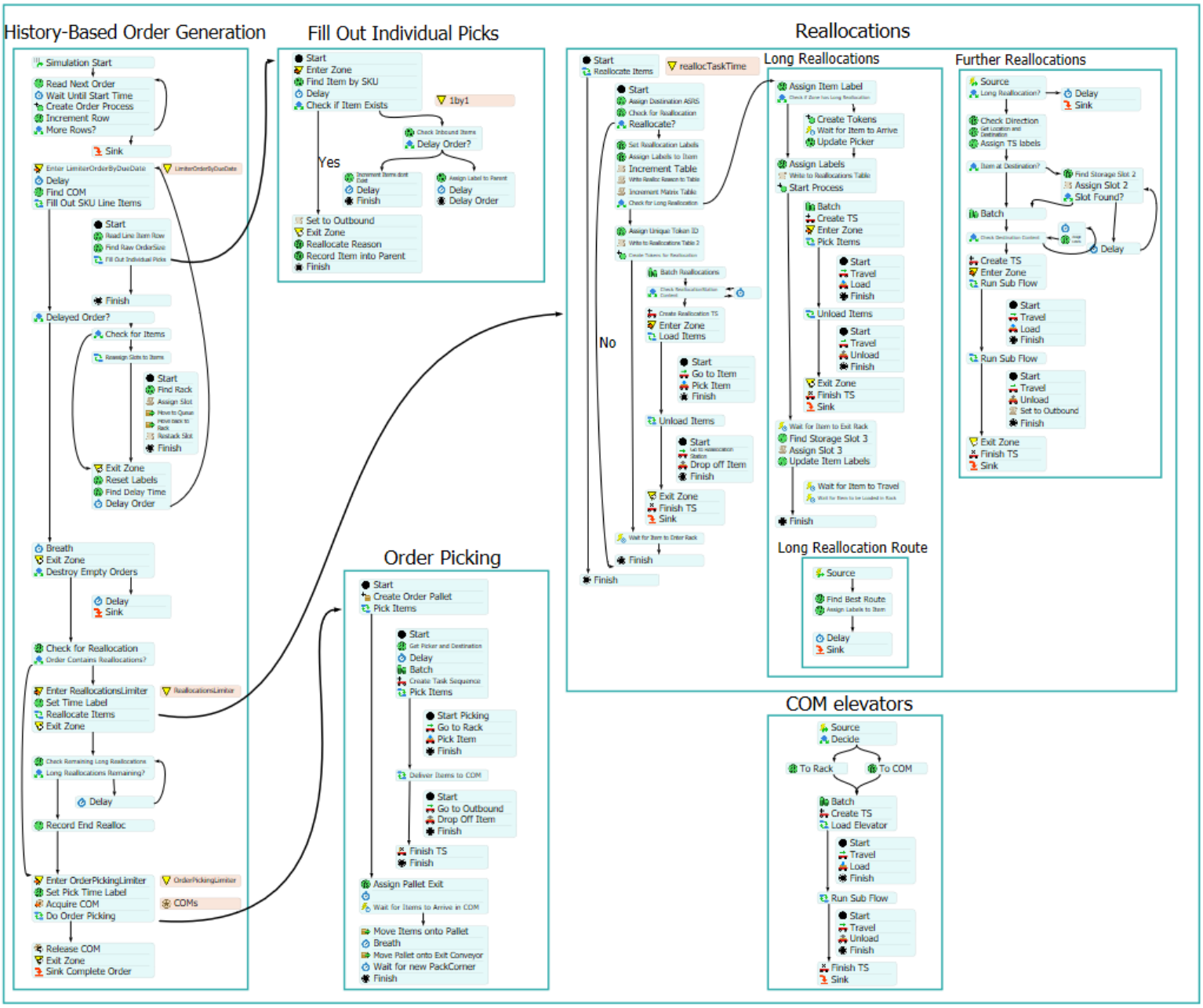
Tabell 6: Process Flow , Inbound og Outbound logikk

Inbound



Bilde 6: Process Flow - Inbound

Outbound



Bilde 7: Process Flow - Outbound

Manuell Programmering

Underkategorien Depalletizers forsøker å finne en optimal plassering for innkommende kolli basert på nåværende lagerbeholdning. For å utjevne vareflyten er det sentralt at SKU-er fordeles så jevnt som mulig mellom sonene. Innkommende kolli blir gruppert etter SKU i grupper på fire så langt det lar seg gjøre. Deretter sjekker scriptet findSlotsForBatch (*Vedlegg 5*) eksisterende lagerbeholdning, og prøver å finne en celle i en av sonene som har færrest adresser med den SKU-en. Hvis en gruppe med innkommende kolli inneholder fire kolli, vil scriptet kun søke etter helt ledige adresser i denne sonen. Dersom en gruppe innkommende kolli inneholder færre kolli enn fire, vil scriptet først forsøke å finne en celle som allerede har denne SKU-en og som har plass til alle kolliene i gruppen. Dersom en slik celle ikke eksisterer på det tidspunktet, vil denne gruppen få tildelt celle på samme måte som grupper på fire. Denne metoden for å finne celle til innkommende kolli sikrer en jevn fordelt lagerbeholdning gjennom hele simuleringen, og reflekterer ASKO SL Kjøl sin jevne fordeling av kolli i deres lager.

Når en ordre skal få tildelt en COM før ordren fylles ut med kolli, sjekker scriptet findCOM (*Vedlegg 6*) lagerbeholdningen i hver sone for å finne ut hvilken sone som inneholder flest av varene ordren skal ha. Scriptet lager tre prioriteringer, hvor sonen med flest antall av varene er første prioritet, sonen med nest flest av varene er andre prioritet, etc. Dersom flere soner har et likt antall varer blir de prioritert likt. Deretter sjekker scriptet om de prioriterte COM-ene har ordre som er i kø for plukking. Scriptet velger den høyeste prioriterte COM-en som ikke har ordre i kø. Dersom alle de prioriterte COM-ene har ordre i kø for plukking, velger scriptet den COM-en med høyest prioritering, eller en tilfeldig COM av de med høyest prioritering dersom det er flere. Denne metoden sikrer at ordre blir jevnt fordelt mellom COM-ene, samtidig som det blir et minimalt antall reallokeringer per ordre.

Etter at en ordre har fått tildelt COM, fylles ordren ut med kolli. Scriptet findItemNearCOM (*Vedlegg 7**Error! Reference source not found.***) sjekker først den laveste utløpsdatoen for hver SKU i ordren. Dersom en SKU med laveste utløpsdato finnes i samme sone som COM blir denne valgt for ordren. Hvis ikke, søker scriptet i alternerende rekkefølge ut fra sonen som tildelt COM er i, og finner SKU-en med laveste utløpsdato i nærmeste sone. Dersom det er flere celler i samme sone som inneholder SKU-en med gitt utløpsdato, vil cellen med lavest antall kolli prioriteres, slik at celler blir frigjort for innkommende kolli. ASRS-kranene i modellen er ikke programmert til å rydde opp i reolene, slik som ASRS-kranene til ASKO

SL Kjøl gjør. Denne prioriteringen sikrer at lageret ikke blir fylt opp med halvfulle celler i modellen.

Hvis en SKU i en ordre ikke finnes i lageret, er det implementert logikk for utsettelse av ordrer. Scriptet checkInboundItems (*Vedlegg 8*) sjekker om denne SKU-en er en innkommende vare før ordren må plukkes. Dersom SKU-en kommer inntil fem timer før ordren har avgangstid, vil ordren bli utsatt til det tidspunktet SKU-en er planlagt å komme inn på, pluss en time for å sikre at kolliet har kommet inn i lagerbeholdningen. Dette medfører at en ordre aldri blir startet på senere enn fire timer før avgangstid, og at ordre så sjeldent som mulig mangler kolli. Dersom en SKU ikke finnes i lagerbeholdningen, og SKU-en ikke kommer inn i løpet av dette tidsintervallet, blir SKU-en slettet fra ordren, og ordren blir markert med antall kolli som mangler.

For kolli som reallokeres forsøker simuleringen å finne årsaken til reallokeringer i scriptet ReallocReason (*Vedlegg 9*). Scriptet sjekker om SKU-en til kolliet som reallokeres finnes i samme sone som ordrens tildelte COM. Dersom SKU-en finnes i denne sonen vil reallokeringsgrunnen alltid være FEFO, og dersom SKU-en ikke finnes i denne sonen vil reallokeringsgrunnen alltid være Closest Item. Dette medfører at i enkelte tilfeller hvor SKU-en til det reallokerte kolliet ikke finnes i den gitte sonen, så vil reallokeringsgrunnen være Closest Item. Dette uavhengig om kolliet som skal reallokeres er fra en sone som er lengre unna en det aller nærmeste kolliet som følge av FEFO. Noen kolli vil altså ha reallokeringsgrunnen Closest Item selv om det egentlig ikke er det nærmeste kolliet på grunn av FEFO.

Overordnede prioriteringer i modellen

I TIC Building området er det laget et dynamisk prioriteringssystem basert på trafikken på samlebåndene før TIC Builder båndene. De innkommende samlebåndene med flest antall kolli får høyest prioritering i et forsøk på å redusere oppbygning av kø på disse samlebåndene. Prioriteringen oppdateres hver gang en ny gruppe kolli er klare for TIC Building eller hvert minutt for å sikre at prioriteringssystemet er oppdatert til enhver tid.

Ordre som skal fylles med kolli, reallokeres eller plukkes blir til enhver tid prioritert etter plukktid. Plukktid er hentet fra ASKO SL Kjøl sine produksjonsdata, og er tiden en ordre ble ferdigplukket hos dem. Denne prioriteringen sikrer at ordre blir plukket før avgangstiden, og medfører at ordre blir plukket i samme rekkefølge som hos ASKO SL Kjøl, med unntak av

ordre som blir utsatt. Utsatte ordre havner også dermed først i prioriteringslisten for ordre som skal reallokeres og plukkes, slik at ordren går raskt gjennom systemet.

ASRS-kranene prioriterer å gjennomføre arbeidsoppgaver relatert til reallokeringer i modellen. Starten av en reallokering, når en kran flytter et kolli fra en lagringsplass til en reallokerinsstasjon, har fått prioriteringen 1. Videre reallokeringer, når en kran flytter et kolli fra en reallokeringsstasjon til en ny reallokeringsstasjon eller til en ny lagringsplass, har fått prioriteringen 2. Resten av arbeidsoppgavene til kranene, som å håndtere innkommende kolli og utgående kolli, har prioritering 0. Høyere prioritering blir prioritert først. Dette ble implementert for å oftere ha ordrer klare for plukking til hver COM. Videre reallokeringer er prioritert høyest for at kolli skal ha lavest mulig oppholdstid på reallokeringsstasjonene, både for at reallokeringer skal bli gjennomført raskere, og for å gjøre reallokeringsstasjonene raskere tilgjengelig for nye reallokeringer.

5.3.4 Statistikk fra simulering

Det blir hentet ut flere datasett fra simuleringen som inneholder statistikk:

Navn	Hva måles	Når
TotalInput	Hvor mange kolli som kommer inn i modellen i hver depalleterer	Oppdateres hver time med antall for den timen
PicksPerHour	Hvor mange kolli blir plukket i hver COM per time	Oppdateres hver time med antall for den timen
ItemCount	Hvor mange kolli som er i hele systemet. Alt som er i rack, på samlebånd, i input og i COM	Oppdateres for hver time med antall på målte tidspunkt
RackContent	Hvor mange kolli er i hver rack	Oppdateres for hver time med antall på målte tidspunkt
CraneState	Hvilke aktiviteter hver kran gjør og hvor lenge de har gjort det	Oppdateres for hver time med antall sekunder per aktivitet for den timen
Reallocation_Staytime	Antall sekunder en ASRS-kran bruker på en arbeidsoppgave relatert til reallokeringer	Måles når aktiviteten er gjennomført, men summeres opp per time per kran
FinishedOrders	Ordrenummer, når reallokering for ordren ble startet, når reallokeringer var ferdig, når plukk startes og når ordren er ferdig. Sjekker om kolli ble reallokert og hvor mange kolli som ble slette fra ordren	Oppdateres så fort en ordre blir gjennomført
Reallocations	Hvordan: lang, kort og totalt	Måles for hvert reallokerte kolli
ReallocReasons	Årsak og distansen til reallokeringer	Måles for hvert reallokerte kolli
ReallocMatrix	To/From matrise med avstander på reallokeringer og antall	Måles for hvert reallokerte kolli
SlotDistribution	Antall celler som er ledige, har 1 til 3 kolli og fulle	Oppdateres hver time for hver sone og totalt
EndingInventory	Lagerbeholdning ved slutten av en simulering	Ved slutten av simuleringen

Tabell 7: Statistikk som blir hentet ut fra simuleringen

5.4 Verifisering

Verifisering av en simuleringsmodell innebærer å teste at modellen gjør det den skal gjøre uten at det skjer noen feil, men ikke nødvendigvis at modellen er lik den reelle prosessen som simuleres. For simulering av et lager kan verifisering gjennomføres ved å undersøke om lagerbeholdningen til enhver tid faktisk er det den skal være, at kolli ikke blir borte underveis i simuleringen og at programvaren ikke returnerer feilmeldinger i løpet av simuleringen. FlexSim gir gode tilbakemeldinger dersom feil oppstår, slik at det er enkelt å sjekke om modellen er programmert riktig.

Ved å fullføre en hel simulering uten feilmeldinger fra FlexSim ble modellen verifisert for programmeringsfeil. For å verifisere at kolli ikke blir borte i modellen er det gjennomført tester for å sjekke at alle innkommende varer minus alle utgående varer er lik lagerbeholdningen ved simuleringens slutt.

Initial Inventory	Input	Output	Calculated End Item Count
216944	763713	761595	219062

End Item Count
219062

Difference
0

Tabell 8: Verifisering av antall kolli

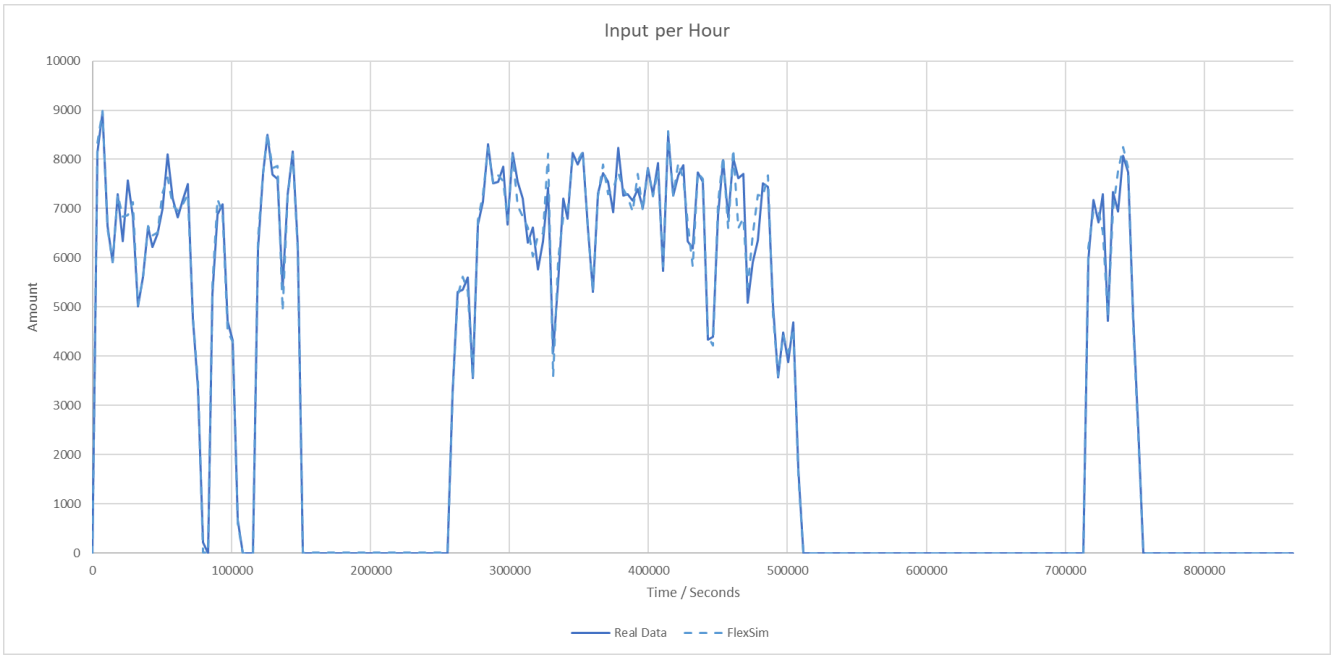
Det er også gjennomført visuelle undersøkelser av modellens oppførsel under simuleringer, for å verifisere at maskiner oppfører seg slik de skal, og at det ikke er noen uforutsette hendelser underveis i simuleringen. Verifisering av modellen har vært en kontinuerlig prosess gjennom hele prosjektet, hvor feil har blitt fikset underveis så fort de har blitt oppdaget.

5.5 Validering

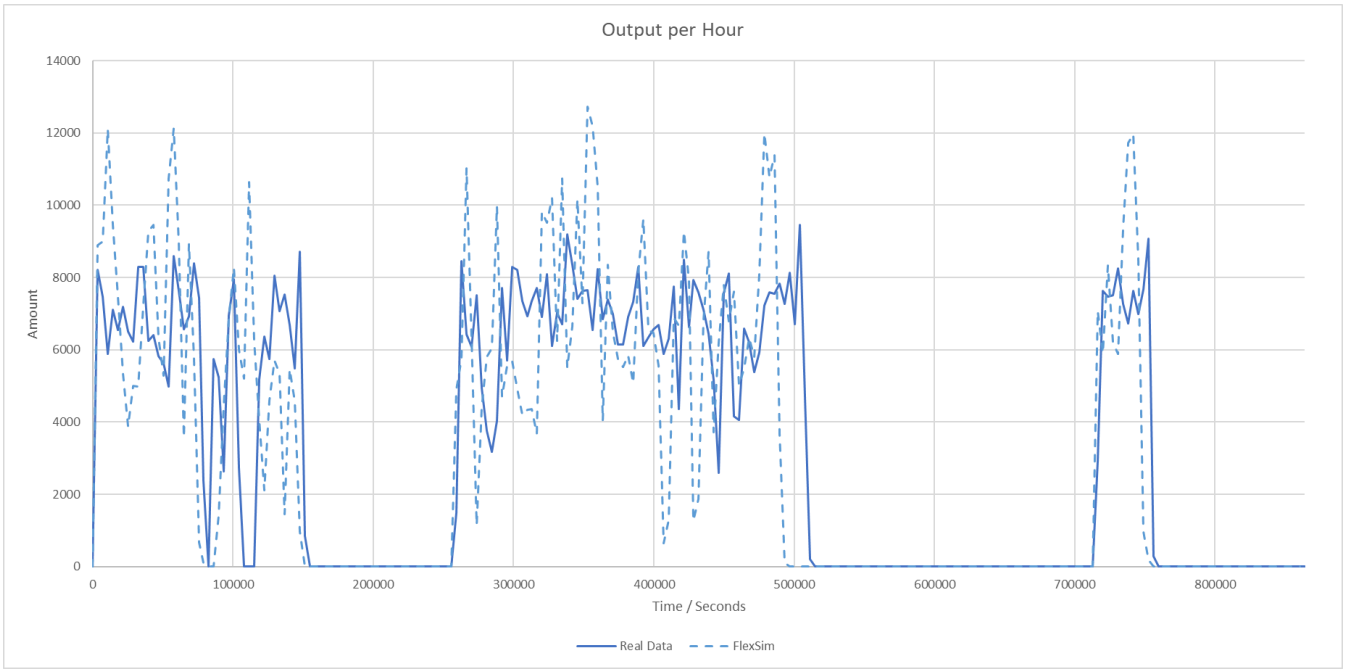
I valideringsfasen sammenliknes data fra simuleringen med produksjonsdata for å se hvor reell simuleringen er. Ettersom at alle inngående kolli og ordre genereres fra reell produksjonsdata fra ASKO SL Kjøl, kan modellen valideres ved å tilnærme produksjonsdata fra FlexSim mot produksjonsdata fra ASKO SL Kjøl gjennom logikkjusteringer i modellen. Gjennom flere simuleringsløp og sammenligninger med reell produksjonsdata blir modellen trinnvis mer tilnærmet lageret til ASKO SL Kjøl. Produksjonsdataen som er fokusert på for validering av modellen er plukk per time, lagerbeholdning per time og innkommende kolli per time.

Figur 30: Validering av simulering - input, som sammenligner innkommende kolli i modellen med innkommende kolli fra reelle data, viser at modellen er reell for input. Det er noen få avvik, hvor modellen sliter med å finne lagringsplass for innkommende kolli på grunn av høy lagerbeholdning, som medfører at innkommende kolli blir holdt igjen i depalleterne til modellen har funnet en lagringsplass.

Fra *Figur 31: Validering av simulering - output* er det tydelig at modellen plukker i samme tidsintervaller som i reelle produksjonsdata. Modellen har derimot større svingninger i plukkingen, som følge av at modellen plukker raskere enn i virkeligheten. Raskere plukking fører til lavere bunnpunkter, siden tilgjengelige ordrer blir ferdig raskere, og det kan være litt ventetid før de neste ordrene kan begynnes på. I enkelte perioder hvor det er mange ordrer som reallokeres samtidig, vil plukkraten også synke, ettersom at plukking er lavere prioritert enn reallokeringer. Det er også tydelig i grafen at plukkingen blir tidligere ferdig i modellen enn fra reelle data i de tre aktive tidsintervallene. Dette gjenspeiler også at modellen generelt plukker raskere.



Figur 30: Validering av simulering - input



Figur 31: Validering av simulering - output

FROM ZONE \ TO ZONE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	0	4047	1198	638	454	402	277	274	187	200	189	204	181	229	150	119	194	221
2	3655	0	3611	954	600	366	275	246	187	179	138	125	140	145	153	125	194	166
3	1874	1738	0	3521	997	549	393	329	218	222	138	154	126	122	100	134	152	169
4	1128	1011	1746	0	3481	1155	569	454	224	244	157	197	170	195	128	178	144	202
5	833	635	759	1684	0	3633	1069	643	382	324	266	192	181	173	98	161	185	222
6	600	508	474	616	1461	0	3634	1252	600	409	269	240	161	161	120	168	183	198
7	417	374	364	417	647	1606	0	4244	1050	559	412	229	197	224	104	215	202	196
8	359	261	315	264	410	748	1570	0	3554	1058	485	281	265	200	102	214	196	252
9	348	287	323	293	292	379	654	1809	0	3712	988	611	349	240	172	257	250	247
10	324	231	270	280	229	259	396	713	1745	0	3674	1035	519	291	186	276	256	326
11	283	198	215	215	217	212	283	442	751	1913	0	3952	1001	590	197	333	340	367
12	235	215	186	172	155	230	197	305	410	705	1763	0	3547	959	293	344	339	434
13	234	173	208	190	148	186	234	291	278	508	792	1967	0	4077	726	546	469	686
14	203	158	161	159	142	183	151	173	239	329	494	861	1739	0	2581	1097	685	855
15	238	219	202	215	208	205	176	192	235	310	455	650	910	2156	0	4095	1253	1400
16	160	154	199	155	171	176	161	168	162	215	253	383	475	909	1302	0	3849	1982
17	223	165	162	177	182	169	135	177	130	187	257	323	361	575	565	1787	0	4783
18	196	168	186	142	157	201	226	168	185	212	227	274	365	389	374	889	2032	0

Tabell 9: Validering av simulering - To/From matrise for reallokeringer mellom soner

For å validere at lagerbeholdningen er jevnt fordelt mellom alle sonene, har det blitt brukt en To/From matrise med antall reallokeringer som er sendt fra en sone til en sone. Som vist i *Tabell 9* er det flere reallokeringer jo kortere distansen er. Dette tyder på at SKU-er er jevnt fordelt, fordi modellen som regel finner en SKU i en nærliggende sone når det kreves reallokering.

Som følge av at dette er en forenklet modell av det komplekse lageret til ASKO SL Kjøl, har enkelte parametere blitt justert for å gjøre modellen tregere slik at produksjonsdata fra modellen er likere produksjonsdata fra ASKO SL Kjøl. I modellen har det blitt lagt inn en begrensning på hvor mange ordrer som kan reallokeres samtidig, både for å redusere trafikk på reallokeringsstasjonene mellom sonene, og for å indirekte redusere hastigheten på plukking av ordre. Modellen kan reallokere opptil 20 ordrer samtidig, men kun to ordrer per COM. Denne begrensningen sikrer også at ordrene som jobbes på er relativt jevnt fordelt over COM-ene til enhver tid.

ASRS-kranene har også fått reduserte hastigheter på lasting og lossing av kolli. Kranene i modellen vil totalt sett ha mindre å gjøre enn kranene hos ASKO SL Kjøl, som følge av at kranene i modellen ikke trenger å rydde i reolene, og at kranene i modellen kan ta ut et vilkårlig kolli fra en celle uten å måtte flytte på de andre kolliene i cellen først. Kranene bruker cirka et sekund på å laste hvert kolli, som vil tilsvare ASKO SL Kjøl sine kraner som bruker cirka fire sekunder på å laste fire kolli. ASKO SL Kjøl sine kraner vil derimot bruke cirka like lang tid på å laste et kolli som fire kolli, noe som ikke er programmert i modellen.

For å gjøre opp for denne forskjellen, har det blitt lagt på et halvt sekund ekstra lossetid per kolli når kranene leverer fra seg kolli.

6.0 Simulering av scenarier

Det er blitt gjennomført syv simuleringer med variasjoner av parameteren for lange reallokeringer. De syv simuleringene tar for seg en variasjon av parameteren hver, fra 5 til 11. ASKO SL Kjøl sin parameter for lange reallokeringer er 8. Det er kun testet nære variasjoner av parameteren. Et for lavt parameter vil gi for mange lange reallokeringer, som vil skape for mye trafikk på inngående samlebånd. En for høy parameter vil skape for mange korte reallokeringer, som vil redusere tilgjengelig tid for ASRS for andre aktiviteter.

Det er også simulert et scenario hvor FEFO-logikken er helt ekskludert, for å undersøke hvordan FEFO påvirker vareflyten og ytelsen til lageret. Scenariet uten FEFO bruker den standard parameteren for lange reallokeringer, 8, og sammenlignes med det tilsvarende scenariet hvor FEFO er inkludert.

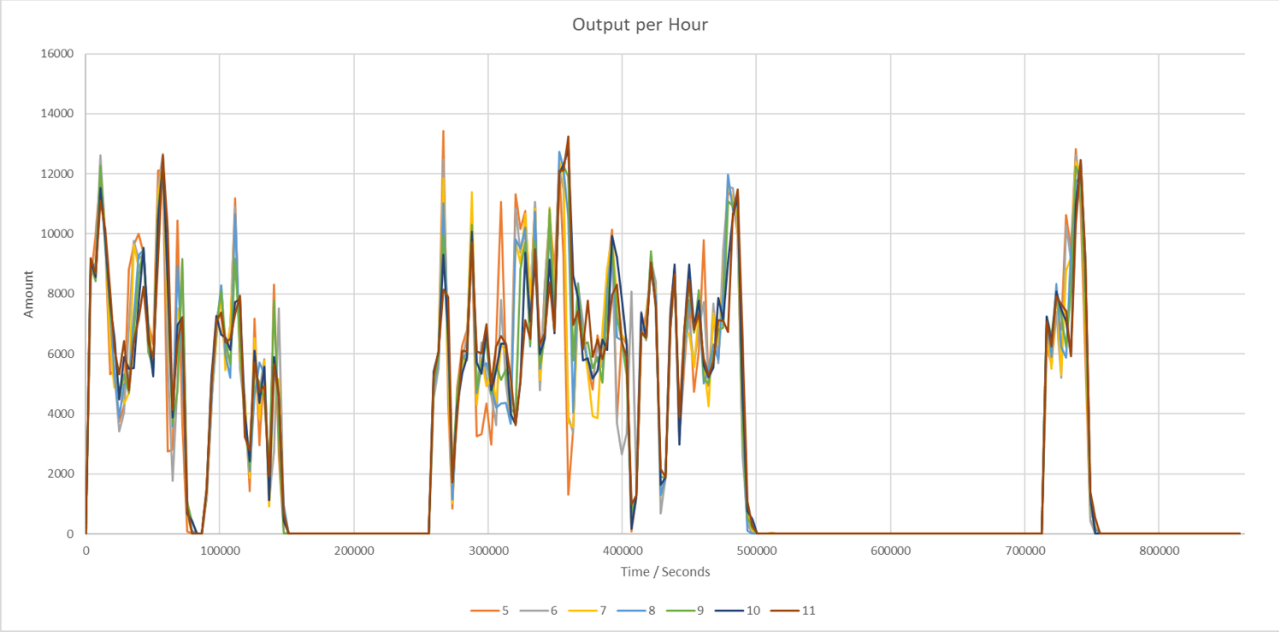
Hvert scenario bruker samme Random Stream og det er kun endringen i parameteren som utgjør en forskjell. Samme Random Stream vil si at alle tilfeldigheter som oppstår og statistiske distribusjoner er like for alle scenarioer. Alle inngående kolli vil få samme plass i lageret, med enkelte unntak som påvirkes av hvilket kolli som har blitt plukket og om det er mange lange reallokeringer. Hvilken COM ordre blir plassert i er alltid den samme.

6.1 Styringsparametere for lange reallokeringer

Endringer av styringsparameteren for lange reallokeringer påvirker vareflyten i OPM og handlingsmønsteret til ASRS.

6.1.1 Input, Output og Rack Content

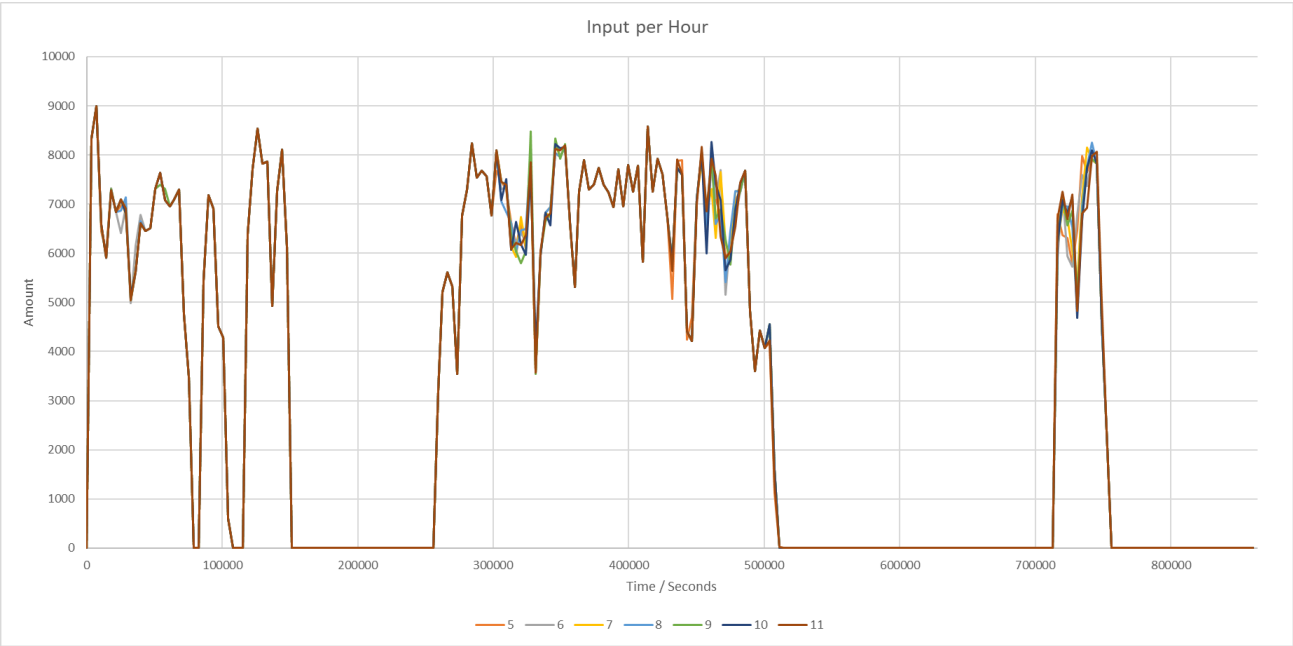
Figur 32: Simulering - Output per time - Alle reallokeringsparametere viser at alle simuleringene følger de samme trendene for plukking, men at forskjellige variasjoner av parameteren for lange reallokeringer gir noe variasjon i når ordre plukkes. Lange reallokeringer tar generelt lengre tid å gjennomføre enn korte reallokeringer, som vil påvirke når ordre er klare for å plukkes. Lange reallokeringer vil derimot påvirkes i mindre grad av trafikken på reallokeringsbåndene enn korte reallokeringer. Ved kortere parameter blir standardavviket høyere, noe som vil gi større spredning i Output. Ved høyere parameter blir standardavviket lavere, Output er jevnere over tid, se *Tabell 10: Simulering - Standardavvik - Alle reallokeringsparametere.*



Figur 32: Simulering - Output per time - Alle reallokeringsparametere

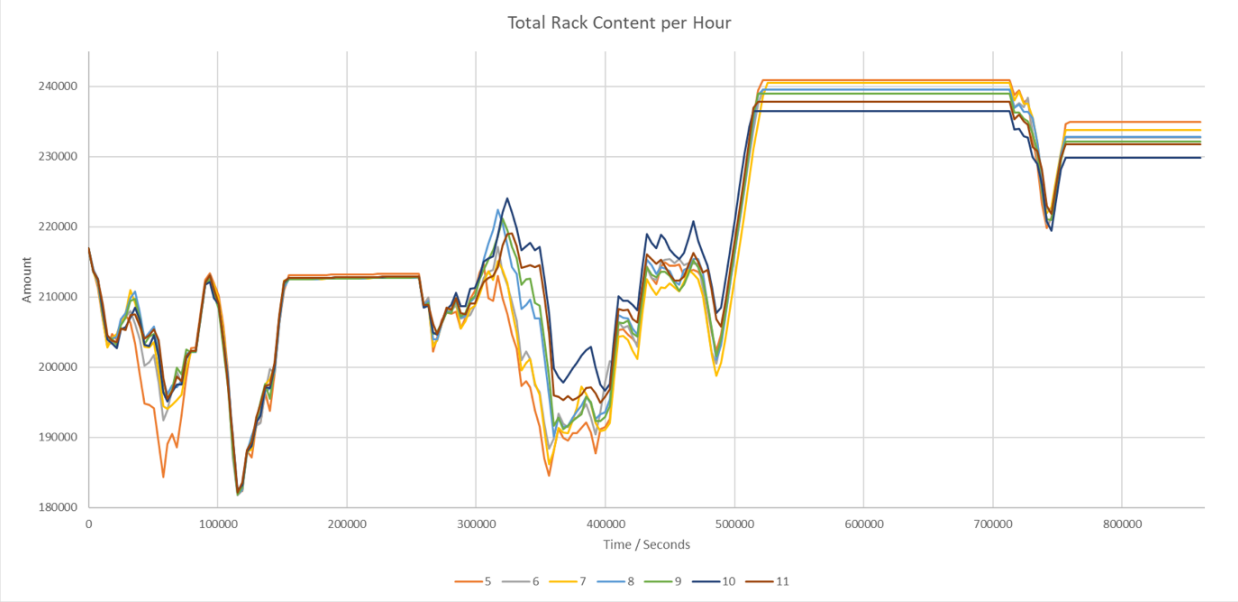
Parameter	5	6	7	8	9	10	11		Real Data
Standard Deviaton	3981	3915	3877	3856	3813	3792	3728		3505

Tabell 10: Simulering - Standardavvik - Alle reallokeringsparametere



Figur 33: Simulering - Input per time - Alle reallokeringsparametere

Figur 33: Simulering - Input per time - Alle reallokeringsparametere er svært lik for alle scenariene. Variasjonen i denne grafen er som følge av variasjoner i lagerbeholdningen, som vises i *Figur 34: Simulering - Total lagerbeholdning - Alle reallokeringsparametere*, hvor modellen ved enkelte tilfeller sliter med å finne optimale lagerplasser for innkommende kolli. De forskjellige scenariene følger veldig like trender i *Figur 34*, med noe variasjon som følge av forskjeller i tidspunkter for plukking av ordrer. Scenariene med lavere parametere for lange reallokeringer har høyere lagerbeholdning mot slutten av simuleringen. Det betyr at modellen finner færre kolli for enkelte ordre i disse scenariene. Grunnen til dette er at inngående samlebånd får mye trafikk, som følge av flere lange reallokeringer, som medfører at innkommende kolli bruker lengre tid på å komme til brettlageret. Ordrer vil kun søke etter kolli som er i brettlageret, og dermed vil det i enkelte tilfeller slettes kolli fra en ordre, selv om kolliet er på vei inn.



Figur 34: Simulering - Total lagerbeholdning - Alle reallokeringsparametere

6.1.2 ASRS

Oppførselen til kranene er veldig like i de forskjellige scenarioene. I parameter 5 er kranene mest i ro og samtidig bruker de minst tid på de fem aktivitetene som måles. Dette da kranene sjeldent trenger å være mellomledd for reallokeringer og de trenger hovedsakelig kun å konsentrere seg om output. Siden kranene håndterer færre brett tilsammen er det naturlig at de også bruker minst tid på aktivitetene. Scenario har høyere total aktiv tid fordi noen ordre blir plukket i tidsrommet hvor det ikke skal bli plukket. Dette kommer av at reallokeringer ved bruk av den alternative ruten tar lengre tid enn ved bruk av kran til kran.

I parameter 10 er kranene mest aktive prosentvis, men den har minst tid brukt totalt. Parameter 11 er den som brukes mest tid på reallokeringer, dette da de fleste reallokeringer forgår fra kran til kran.

Scenario	Total Time	Idle	TravelOffset Empty	TravelOffset Loaded	Loading	Unloading	Sum	Time used for relocation
5	17334000	33.85 %	8.80 %	39.94 %	5.80 %	11.60 %	100.00 %	15.92 %
6	17179200	32.63 %	8.97 %	40.56 %	5.94 %	11.89 %	100.00 %	17.02 %
7	17287200	32.47 %	9.04 %	40.54 %	5.99 %	11.97 %	100.00 %	17.80 %
8	17208000	31.52 %	9.16 %	41.02 %	6.10 %	12.21 %	100.00 %	18.75 %
9	16963200	29.88 %	9.41 %	41.86 %	6.28 %	12.56 %	100.00 %	19.84 %
10	16880400	28.94 %	9.57 %	42.33 %	6.39 %	12.77 %	100.00 %	20.62 %
11	17110800	29.55 %	9.51 %	41.89 %	6.35 %	12.70 %	100.00 %	20.87 %

Tabell 11:Simulering - ASRS ytelse – Alle reallokeringsparametere

Aktivitet	**Betydning**
Total Time	Totalen av alle timesintervallene hvor kranene brukes
Idle	Tiden hvor kranene står i ro
TravelOffset Empty	Tiden kranene beveger seg uten kolli
TravelOffset Loaded	Tiden kranene beveger seg med kolli
Loading	Tiden kranene bruker på å plukke kolli
Unloading	Tiden kranene bruker på å sette fra seg kolli
Time used for relocation	Tiden en kran bruker på aktiviteter relatert til reallokeringer

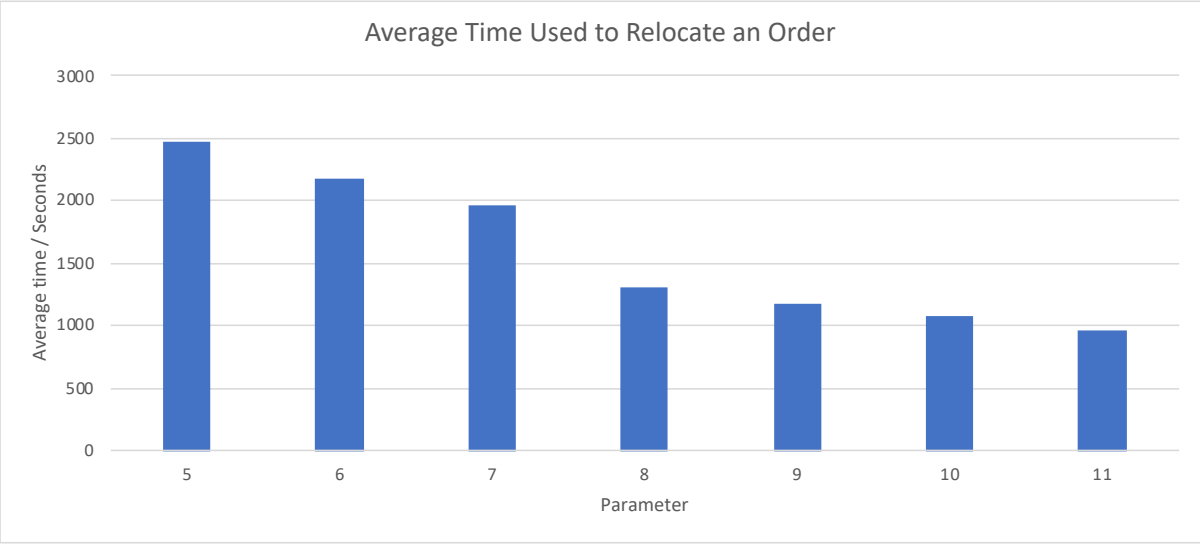
Tabell 12: Forklaring av aktiviteter for ASRS i simuleringen

Et intervall hvor kranene brukes er definert som en time hvor tiden brukt i Idle-tilstanden er mindre enn 3600 sekunder. Altså hver eneste time hvor kranene har minst én arbeidsoppgave. Dette er implementert for å unngå å inkludere tidsperioder hvor lageret ikke opererer, ettersom at disse timene ville forskjøvet de gjennomsnittlige verdiene. På grunn av variasjonen scenarioene påfører er det ulikt antall tid som blir brukt for produksjon. Rådataen

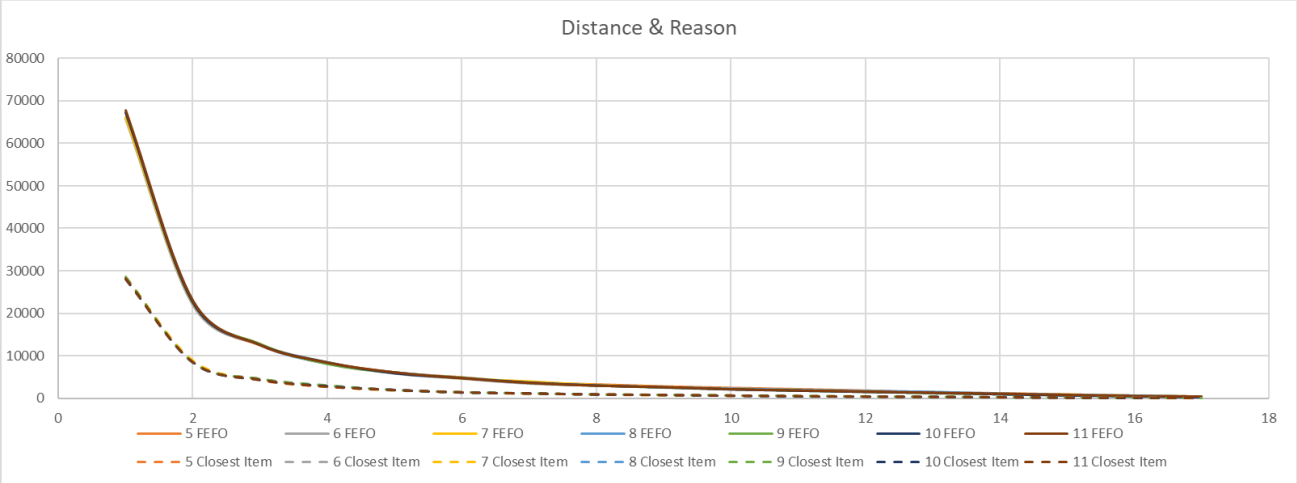
for Time Used for Relocation er total tid brukt på arbeidsoppgaver relatert til reallokeringer for ASRS. Dette inkluderer TravelOffset Empty, TravelOffset Loaded, Loading og Unloading for reallokeringer. Prosentverdiene for Time Used for Relocation er deretter beregnet ved å ta total tid brukt på reallokeringer delt på Total Time. I scenariene med høyere parametere for lange reallokeringer, brukes det mer tid på reallokeringer ettersom at det er flere korte reallokeringer.

6.1.3 Reallokeringer

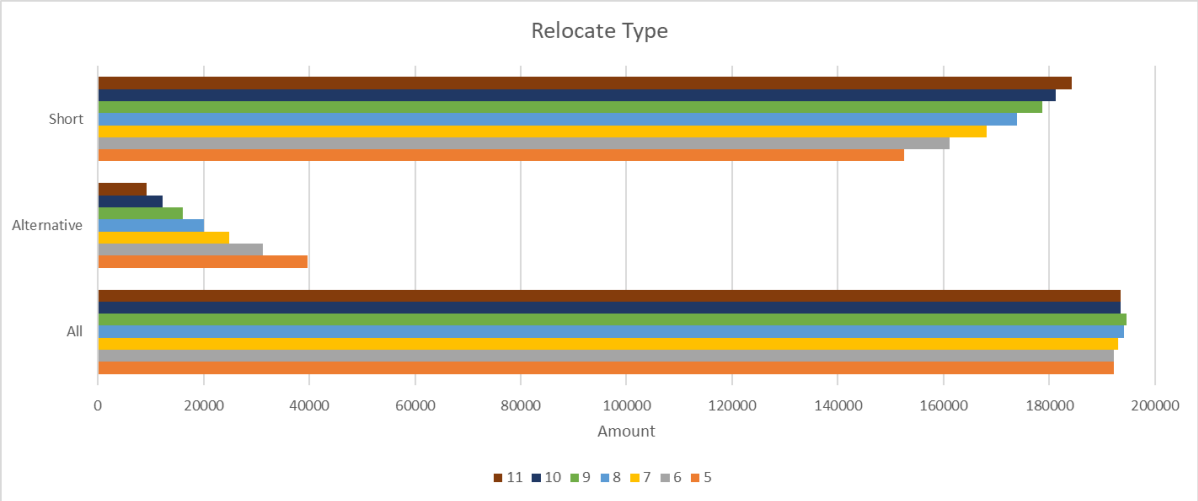
Figur 35 viser at jo høyere parameteren for lange reallokeringer er, jo kortere tid i snitt tar det å reallokere ferdig en ordre. Dette gjenspeiler at kortere reallokeringer tar kortere tid å gjennomføre enn lange reallokeringer. Her regnes en reallokering som startet på når modellen begynner å prosessere reallokering av ordren, og en reallokering regnes som avsluttet når alle reallokerte kolli er i en lagringsplass i riktig sone. En ordre kan altså vente i kø på å bli reallokert uten at dette teller i tiden for reallokering, dersom det er mange andre ordrer som reallokeres på det tidspunktet.



Figur 35: Simulering - Gjennomsnittlig tid brukt for å reallokere en ordre - Alle reallokeringsparametere



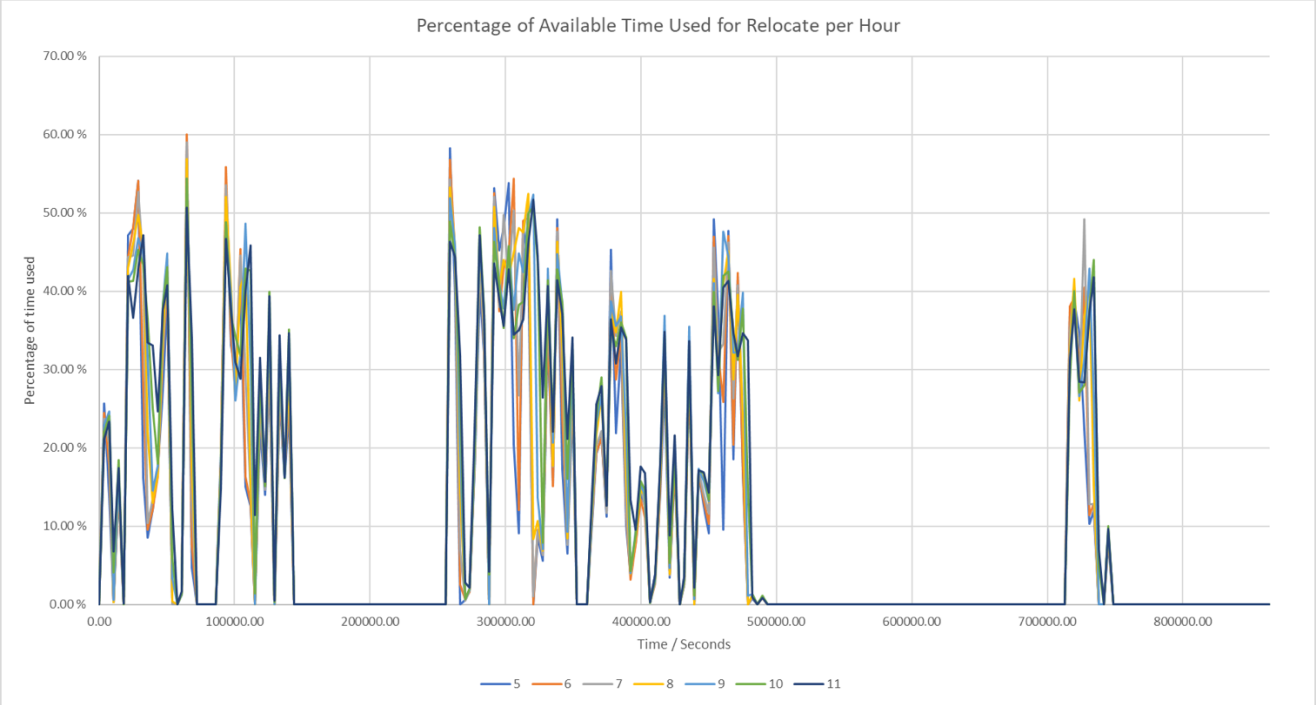
Figur 36: Simulering – Distanse på reallokeringer og årsak - Alle reallokeringsparametere



Figur 37: Simulering – Valg av reallokeringsrute – Alle reallokeringsparametere

Antall reallokeringer for hver parameter er veldig like. Dette da modellen bruker samme Random Streams for alle scenarioene. Grunnen til at kolli blir reallokert er de samme, med marginale forskjeller mellom parameterne, se *Figur 36*. Ved høyere parameter vil antallet alternative reallokeringer reduseres. Dette samsvarer med tid brukt på reallokeringer, se ***Error! Reference source not found.***.

Tid brukt på reallokeringer per time kan sees i *Figur 38: Simulering - Tid brukt på reallokeringer - Alle reallokeringsparametere*. Alle parameterne følger samme trend med toppunkter ved enkelte tidsintervaller. Parameter 11, som i snitt bruker mest tid på reallokeringer, har ingen timer hvor den har brukt mest tid på dette, men den er mer jevn gjennom hele perioden.



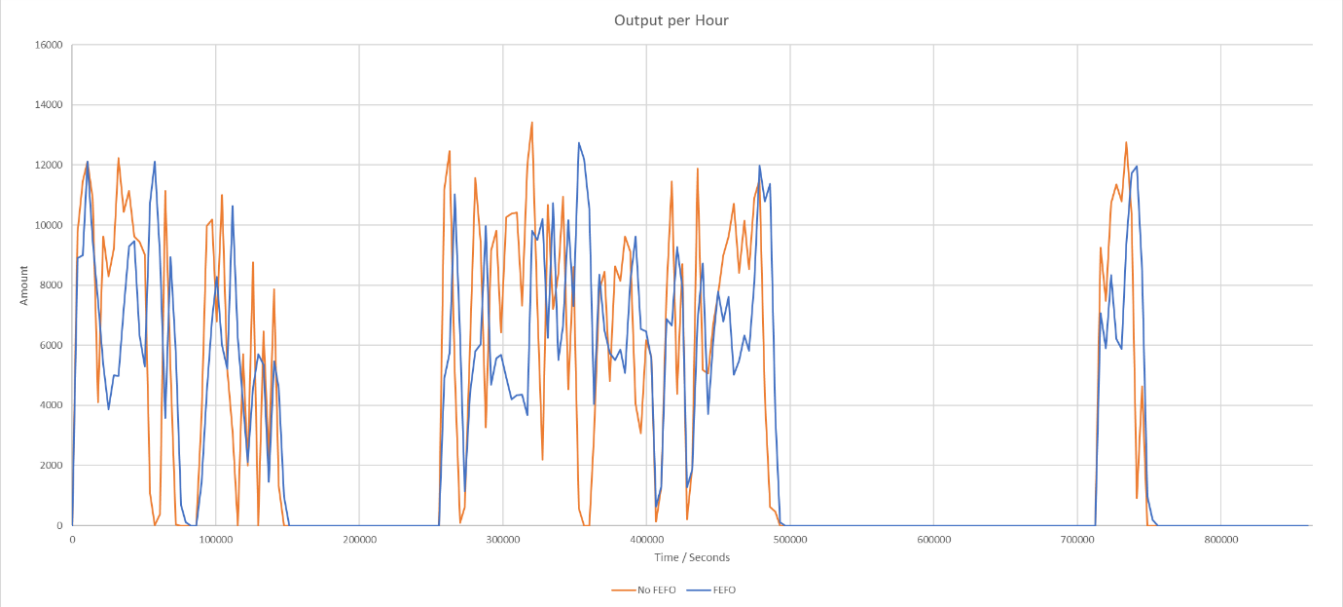
Figur 38: Simulering - Tid brukt på reallokeringer - Alle reallokeringsparametere

6.2 FEFO

Når FEFO fjernes fra simuleringen blir handlingsmønsteret helt annerledes. ASRS bruker mindre tid på reallokeringer, antall reallokeringer går ned og ordre blir plukket fortere. Sammenlikningene her baserer seg på parameter 8, med og uten FEFO.

6.2.1 Input, Output og Rack Content

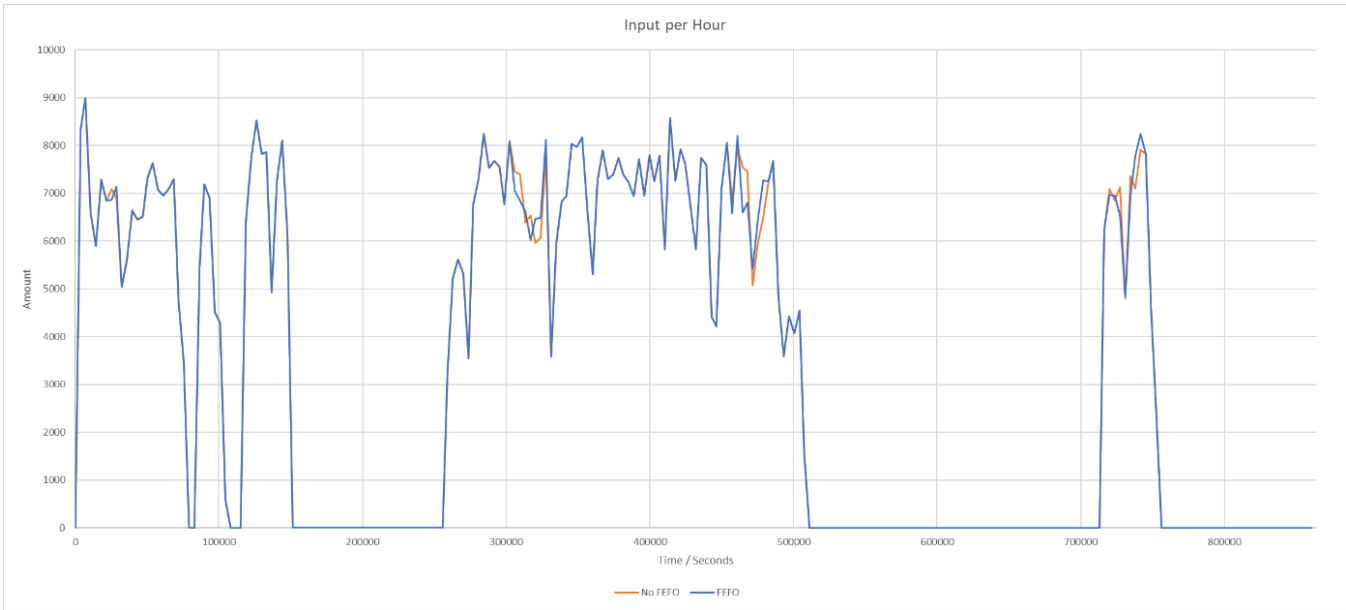
Figur 39: Simulering - Output per time - Uten FEFO viser hvordan forskjellen i plukk er for disse scenarioene. Det er tydelig at når FEFO fjernes så plukkes det fortere. Dette da kolli som regel finnes i riktig sone. Det reallokeres mindre og reallokeringene er som regel korte. Uten FEFO er plukk mer variert per time, se *Tabell 13: Simuleirng - Standardavvik - Uten FEFO.* Dette er beregnet ved å ta gjennomsnittet av alle plukk per COM per time, sett bort i fra alle verdier som er 0. Ved slutten av simuleringen lå det igjen 3179 kolli som hadde gått ut på dato men som ikke var plukket som følge av at FEFO var fjernet.



Figur 39: Simulering - Output per time - Uten FEFO

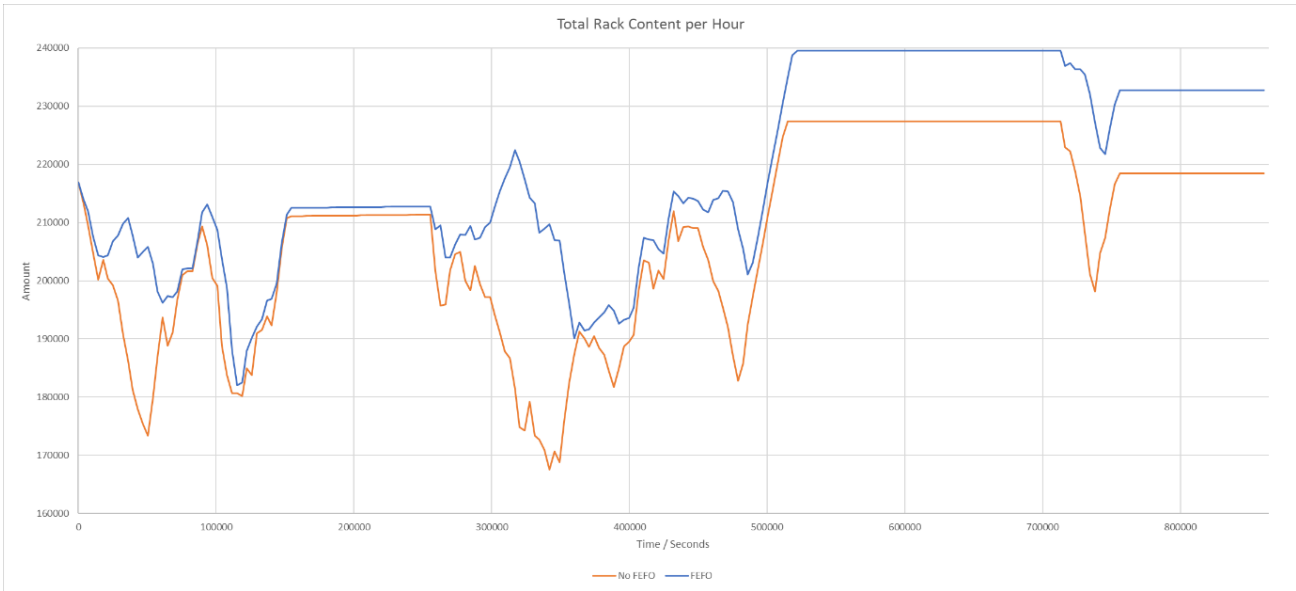
Scenario	No FEFO	FEFO		Real
Standard Deviation	4363	3856		3505

Tabell 13: Simuleirng - Standardavvik - Uten FEFO



Figur 40: Simulering - Input - Uten FEFO

Figur 40: Simulering - Input - Uten FEFO viser at input er tilnærmet lik hverandre. Uten FEFO har noe forskjell enkelte steder da simuleringen finner lettere plass fordi Rack Content er mindre. Variasjonen i input grafen følger variasjonen i *Figur 41.* Når FEFO er fjernet er lagerbeholdningen generelt lavere enn når FEFO er implementert, se *Figur 41: Simulering - Total lagerbeholdning - Uten FEFO*. Den lille forskjellen i input kommer av at lagerbeholdningen til tider er høy og da sliter simuleringen med å finne plass til nye kolli. Den store variasjonen i enkelte tidsrom kommer av at ordre blir fortere plukket og kolli går fortere ut. Når FEFO er fjernet finner simuleringen flere kolli og det fører til at lagerbeholdningen blir lavere på slutten. De områdene hvor lagerbeholdningen går opp, der FEFO er implementert, kommer av at det foregår mange reallokeringer som tar tid og kapasitet fra kranene. Det kommer tydelig frem at når ordre slippes så må det reallokeres mye når FEFO er implementert da lagerbeholdningen øker. Uten FEFO synker lagerbeholdningen kraftig da ordre blir plukket med en gang.



Figur 41: Simulering - Total lagerbeholdning - Uten FEFO

6.2.2 ASRS

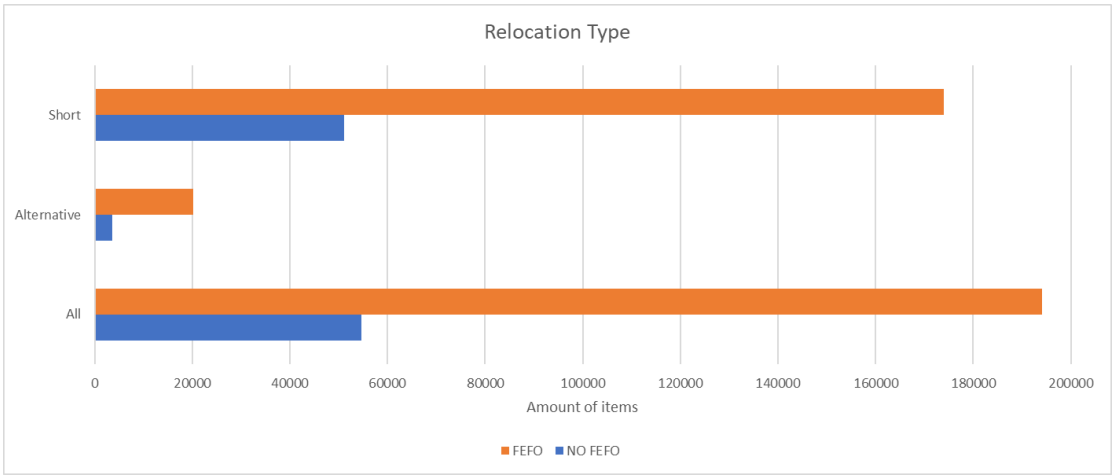
I disse to scenarioene er det stor forskjell i hvordan ASRS jobber. Uten FEFO er kranene generelt mindre i aktivitet og de er mer Idle prosentvis enn når FEFO er inkludert. Det er stor forskjell i hvor mye tid som blir brukt til reallokering. For en forklaring av momentene i *Tabell 14,* se *Tabell 12: Forklaring av aktiviteter for ASRS i simuleringen.*

Scenario	Total	Idle	TravelOffset Empty	TravelOffset Loaded	Loading	Unloading	Sum	Time used for relocation
No FEFO	16880400	43.12 %	7.40 %	34.55 %	4.98 %	9.95 %	100.00 %	5.47 %
FEFO	17208000	31.52 %	9.16 %	41.02 %	6.10 %	12.21 %	100.00 %	18.75 %

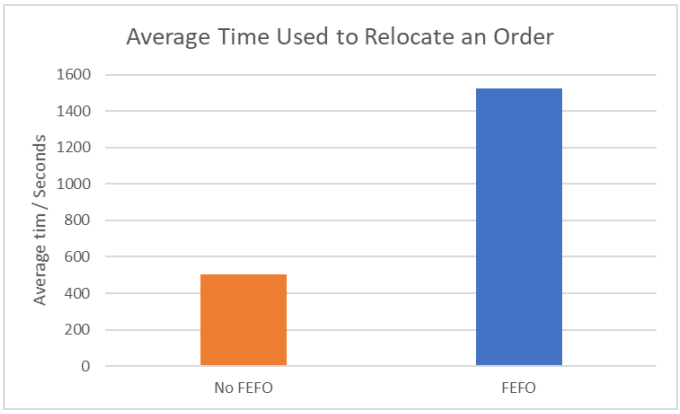
Tabell 14: Simulering - ASRS ytelse - Uten FEFO

6.2.3 Reallokeringer

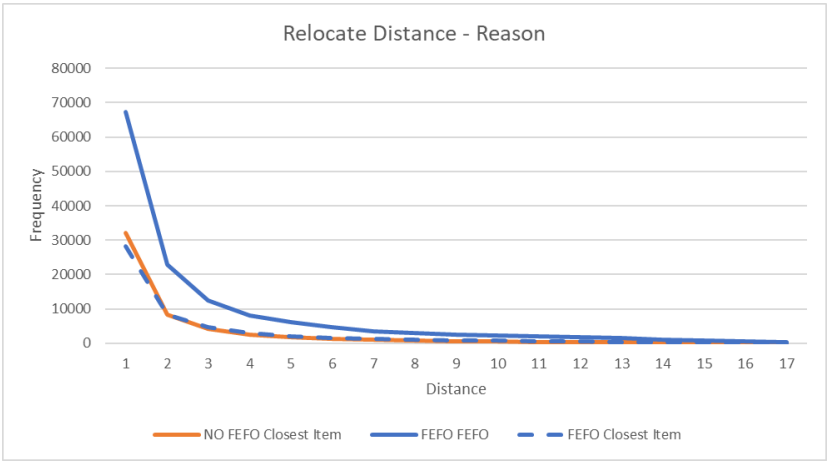
Figur 42 viser at totalt antall reallokeringer går betydelig ned i scenariet uten FEFO. Det er svært få reallokeringer som er så lange at de må benytte seg av den alternative ruten når FEFO fjernes. I *Figur 43: Simulering - Gjennomsnittlig tid brukt for å reallokere en ordre - Uten FEFO* kommer det frem at reallokeringer som følge av FEFO betraktelig øker den gjennomsnittlige tiden som brukes på å reallokere en ordre.



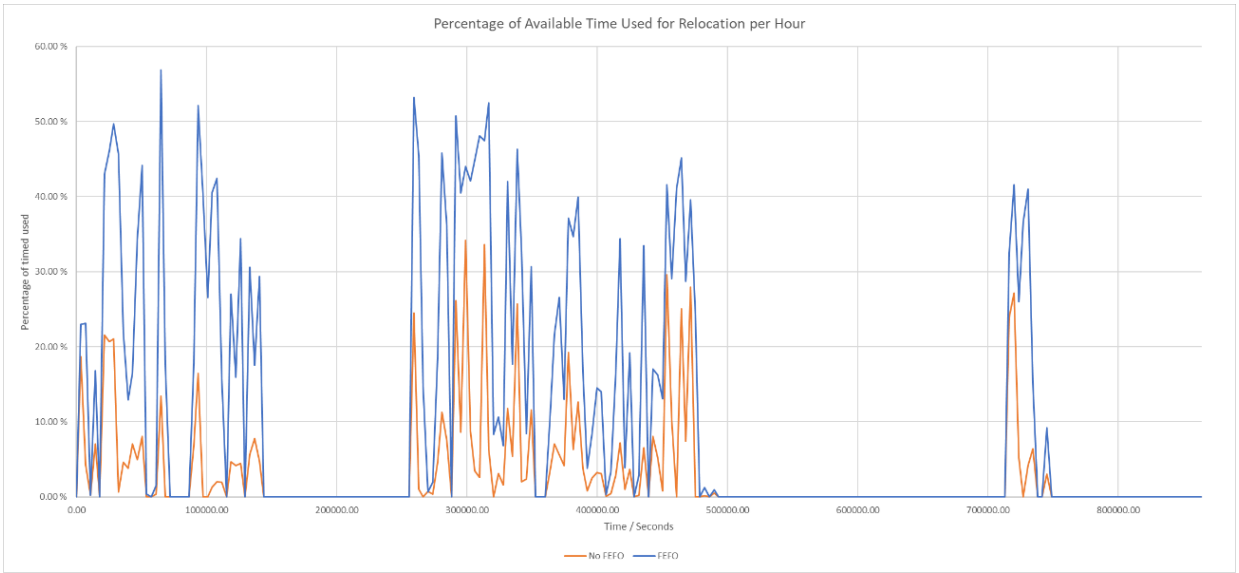
Figur 42: Simlering - Valg av reallokeringsrute - Uten FEFO



Figur 43: Simulering - Gjennomsnittlig tid brukt for å reallokere en ordre - Uten FEFO



Figur 44: Simulering – Distanse på reallokeringer og årsak - Uten FEFO



Figur 45: Simulering - Tid bruk på reallokering per time - Uten FEFO

Når FEFO fjernes er det kun en grunn til at reallokeringer gjennomføres, fordi kolli som COM trenger ikke er i dens sone. Det er 22% forskjell i antall reallokeringer som gjennomføres på grunn av at kolli ikke finnes i riktig sone. Enkelte reaollokeringer når FEFO er implementert vil telles som FEFO og ikke Closest Item hvis begge kriteriene inntreffer.

I *Figur 45: Simulering - Tid bruk på reallokering per time - Uten FEFO* sammenliknes tid brukt på reallokeringer, med og uten FEFO. Det er tydelig at FEFO påvirker hvor mye tid som blir brukt på reallokeringer. Uten FEFO blir det aldri brukt mer enn 35% av total tid per time på reallokeringer. Med FEFO er det høyeste som blir brukt per time 57%. I alle timesintervallene brukes det mer tid på reallokeringer når FEFO er implementert

7.0 Diskusjon

Her skal det diskuteres resultatene fra dataanalysene og de ulike scenariene fra simuleringen.

7.1 Dataanalyse

Gjennom dataanalysen er det blitt gjort funn som må diskuteres. Flere beregninger inneholder feilkilder og enkelte resultater må belyses og videre diskutert.

7.1.1 COM ytelse

Det er tydelig at ytelse av COM på generell basis ikke er optimal og output er varierende for hver sone. COM opererer i snitt ved 67% kapasitet, dette kommer av feil i COM, tidligere ledd i produksjonskjeden eller mangel på ordre.

Sone 1, 17 og 18 plukker i snitt mest da disse er i ytterkantene av lageret, tilhørende ASRS trenger ikke å reallokere like mye og de kan derfor hovedsakelig konsentrere seg om å levere brett til COM. COM 6, som ligger i midten av lageret, plukker minst. Dette kommer av at ASRS må bruke mer tid på reallokering med nabosonene og ASRS kan sende tomme brett direkte til depalleterer. Det er tydelig forskjell i antall plukk per dag og time, på grunn av mengden av ordre og størrelsene på disse. Dette gjør det vanskelig å holde produksjonen jevn for alle perioder.

Måten en ordre får tildelt en COM påvirker også output. Siden en ordre blir plassert i den sonen som har mest kolli som skal på ordren, kommer det an på hvordan fordelingen av SKU-er er mellom sonene. Dette er for å minimere antall reallokeringer, slik at ASRS kan sende flere brett. Eventuelle hastebestillinger og prioriteringsrekkefølgen på ordre vil også påvirke ytelsen til COM.

Alle disse faktorene påvirker output for hver COM, men måten ASRS yter påvirker mest. Det er ASRS som leverer COM med brett og sørger for at COM får gjort den verdiskapningen den skal, sette kolli på lastbærer. Når ASRS må bruke tid på andre oppgaver enn kun å levere brett til COM går ytelsen ned. Disse andre oppgavene er dog viktige for å kunne opprettholde driften og varebeholdningen, samt for at COM skal kunne fullføre ordre.

I måten gjennomsnittlig plukk har blitt regnet ut er det tatt et gjennomsnitt av alle datapunkter som er blitt hentet ut. Ved å bruke denne metoden er det mange ekstremalpunkter og verdier som ikke blir belyst og det er vanskelig å se trender i datasettet. Dette skaper usikkerhet i hvor nøyaktig resultatet er. På en annen side er gjennomsnitt den beste måten å kartlegge hvordan systemet opererer på generell basis, da det er brukt såpass store datasett.

For å unngå å bare vise gjennomsnittlig drift og påfølgende Flaw of Averages, er det sett på ytelsen ved lavt og høyt plukkantall. Her kommer det tydelig frem hvor mye COM plukker og hvilke feilkilder som forhindrer mest. Ved lavt plukkantall fremgår det at det er lite ordre, men også samtidig at ASRS ikke leverer brett da de er opptatt med andre arbeidsoppgaver. Ved høyt plukkantall stabler COM 28% mer enn ved gjennomsnittlig drift. De 16.6% av brett som ikke ble plukket, men som COM har kapasitet til, kommer av at ASRS ikke opprettholder den høye etterspørselen av brett og på grunn av feil som forekommer i COM. Dette gir godt grunnlag for å videre se på hvordan ASRS yter, både ved generell drift og høyt og lavt plukkantall.

Antallet brett COM ikke får levert er vanskelig å beregne, og tallet på hvor mange brett som blir forhindret er kalkulert av systemet. Det er usikkert hvor nøyaktig disse grunnene er da det er vanskelig å bestemme nøyaktig hvor mange plukk som forhindres i COM, på bakgrunn av hvilke feilkilder som opptrer. Ser en på total plukk pluss antall plukk forhindret er dette tallet varierende og det er sjeldent det blir nøyaktig teoretisk maks kapasitet, 9360. I dataene forekommer 9360 kun når det ikke produseres i lageret og ved et unntak. Det er flere den kalkulerte plukkmengden overstiger 9360.

7.1.2 ASRS Performance

ASRS sin hovedoppgave er å sørge for at COM får levert brett slik at ordre kan bli fullført. I tillegg til denne viktige oppgaven må ASRS også sørge for å ta hånd om input fra depalleterer slik at varebeholdningen holder seg jevn og COM får fylt ut nye ordre, samt reallokere mellom soner. Alle disse arbeidsoppgavene tar opp mye tid, men samtidig er kranene en del Idle, både ved gjennomsnittlig drift og ved høy pågang.

Det er urovekkende at kranene er Idle i 24% av tiden de produserer, men samtidig kan dette indikere at kranene er effektive. Det er flere grunner til at de er Idle; At COM ikke trenger brett eller har ordre, at det ikke er noe input og at det er ingen reallokeringer som må gjennomføres. På bakgrunn av dette er det fortsatt uutnyttet tid når COM har mye å gjøre og ordre kan bli plukket. Årsaken til mye Idle kan også bety at det ikke er kranene som er flaskehalsen. Det kan hende at:

- Depalliterer, samlebåndsnettverket, TIC Builder eller VC ikke klarer å levere nok input til ASRS
- Det er for lite plass i brettlageret i forhold til sortimentet og volumet av ordre

Det er usikkerhet i hvor nøyaktig snittet er da det er basert på alle 36 kranene for den totale tidsperioden. Dette vil dog gi et godt bilde på hvordan systemet opererer i løpet av en lengre perioder, med opp- og nedgang i aktivitet.

Nøyaktig hvor lang tid en kran bruker på å håndtere et brett er uvisst da håndteringen av et brett tar like lang tid som fire. Dette gjør det vanskelig å beregne nøyaktig hvor mange brett som ikke blir levert til COM på grunn av andre arbeidsoppgaver enn output. Hvis dette skal beregnes må det brukes mer nøyaktig data og ikke kun tidsbruk på timesintervaller på forskjellige aktiviteter og antall brett håndtert.

Misc er en aktivitet som tar en del tid fra kranene, det tar mindre tid ved høy og lav pågang enn ved gjennomsnittlig drift. Det er uvisst nøyaktig hvilke aktiviteter denne kategorien innebærer. Tidsbruken på denne kategorien er mindre urovekkende enn Idle. Misc kan være forflytting uten brett og eventuelle mørketall for reallokeringer. Når kranene skal reallokere og ikke har et brett, fra verken input, output eller er i reallokeringsposisjon, kan det være at aktiviteten Misc er den som brukes for å få kranen til reallokeringsposisjon.

Kranene jobber relativt jevnt, men det er alltid noen unntak. Kranene i ytterkant bruker mer tid på output og sender flere brett ut. Dette da det ikke er nødvendig for disse å reallokere like mange brett som de i midten. Sonene i ytterkant bruker mindre tid på reallokeringer, men de får levert en del brett på denne måten. Dette hjelper med å øke utnyttelsen av COM.

7.1.3 Reallokeringer

Reallokeringer er den aktiviteten ASRS kranene utfører for å flytte brett mellom sonene. Det er en aktivitet som er nødvendig for at COM skal kunne fullføre ordre og for at varer med lav holdbarhet skal gå ut av lageret. Det tar opp potensiell output fra ASRS til COM og jo lengre brett må reallokeres, jo mer tid tar det fra kranene.

Grunnene til at brett blir reallokert er flere. De to grunnene som opptrer oftest er fordi brett som COM trenger ikke er i dens sone og på grunn av FEFO. Disse står for 62% av alle reallokeringer som blir gjennomført. I dataene er grunnen til reallokering oppført som Possible Relocate Reason. Dette betyr at grunner for reallokeringer kun er en antagelse generert fra WMS, men det er ingen grunn til å tro at det oppstår feil her. Dermed er det ikke sikkert at disse grunnene er helt nøyaktige og dette beregnede tallet kan ha avvik fra hvorfor brett faktisk blir reallokert.

Avstanden brett blir reallokert blir målt fra start sone til slutt sone. Dette vil si at de lengste reallokeringene skal ha en avstand på 17, men dataene viser at noen reallokeringer har en avstand på 18. Det er uvisst hvorfor dette forekommer da ikke skal være mulig å ha en avstand på 18. De fleste brett reallokeres ved korte avstander, noe som gjør at færre ASRS-kraner må være innblandet for å reallokere. Dette viser at varebeholdningen er veldig jevn gjennom hele lageret og at ordre blir riktig plassert i forhold til logikken bak dette.

Sammenlikningen for hvilke soner som tar imot og sender ut flest brett er beregnet ut i fra hvilken kran som sender ut brettet og hvilken sone brettet skal til. Sone 1 og 18 tar imot flest brett og sender ut færrest. Sone 1 bruker en del tid på reallokeringer mens sone 18 bruker lite. Her kommer det tydelig frem at tidsbruk per brett er vanskelig å beregne da det tar nesten like mye tid uavhengig om det er et eller fire brett.

Hver kran bruker i snitt 2.5 minutt per time på reallokeringer, 4.1% av tilgjengelig tid. Ved lav pågang bruker kranene mer tid på reallokeringer, da de har de tid til å jevne ut varebeholdningen, rydde opp slik at hver celle inneholder samme SKU og klargjøre for eventuelle ordre. Ved høyt plukkantall reallokeres det betydelig mindre og output til COM er betydelig høyere. Det blir reallokert mindre fordi ASRS prioriterer output til COM og fordi de kolli som trengs på ordren allerede er i riktig sone.

7.1.4 FEFO

FEFO reduserer ytelsen til OPM, da logikken gjør at brett må reallokeres, og som regel over lengre avstander. Det er blitt beregnet at FEFO tar opp 1.2% av all tiden til kranene. Dette vil si at en kran mister 17 minutter med potensiell output hver dag. Nøyaktig hvor mange brett dette tilsvarer i output er vanskelig å beregne da denne dataen ikke har vært tilgjengelig og som nevnt tidligere, mer nøyaktig data trengs.

For å regne ut 1.2% er det brukt gjennomsnittlig tid kranene bruker på reallokeringer, 4.1%, og sett på hvor mange av disse som er FEFO reallokeringer. FEFO tilsvarer 29% av alle reallokerte brett. Her kan det være feilkilder, denne beregeningen er bare en hypotese. Variansen som forsvinner ved å ta gjennomsnitt blir ikke belyst og usikkerheten bak hvorfor brett blir reallokert gjør denne beregningen lite legitim.

Selv om FEFO reduserer tiden ASRS kan bruke på output er logikken essensiell for å forhindre svinn. Det er uvisst ut fra dataanalysen nøyaktig hvordan FEFO påvirker systemet og hvordan systemet hadde oppført seg uten denne logikken.

7.1.5 Forbedringer av data analyse

For å tilstrekkelig kunne beregne hvor mye tid reallokeringer og FEFO påvirker systemet krever det mer data enn det som er sett på. Mer data fra OM55a skulle vært sett på for å bedre kunne beregne hvor mange brett som gikk ut av ASRS og ikke kun håndteringer av brett i ASRS. Ved hjelp av dette kan en sjekke om output fra ASRS tilsvarer plukk i COM pluss brett reallokert ut fra sonen. Denne rapporten gjør det også mulig å beregne hvor lang tid en reallokering tar.

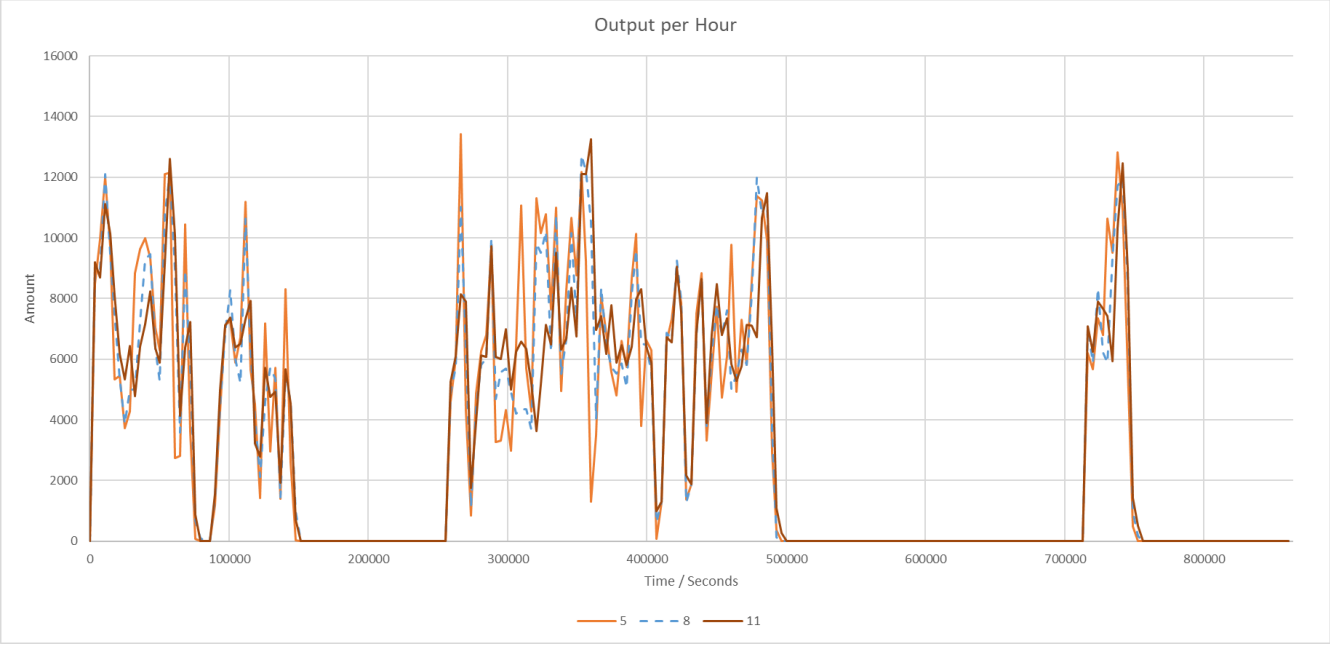
Data for hvor lenge en kran gjør en aktivitet og når denne ble gjort, samt nøyaktig hvor mange brett håndtert under denne aktiviteten vil styrke legitimiteten bak analysen. Ved bruk av dette sammen med OM55a vil det mest sannsynlig være mulig å beregne hvor mye de forskjellige aktivitetene påvirker output til COM, kun basert på data fra WMS.

7.2 FlexSim

7.2.1 Parameter for alternative reallokeringer

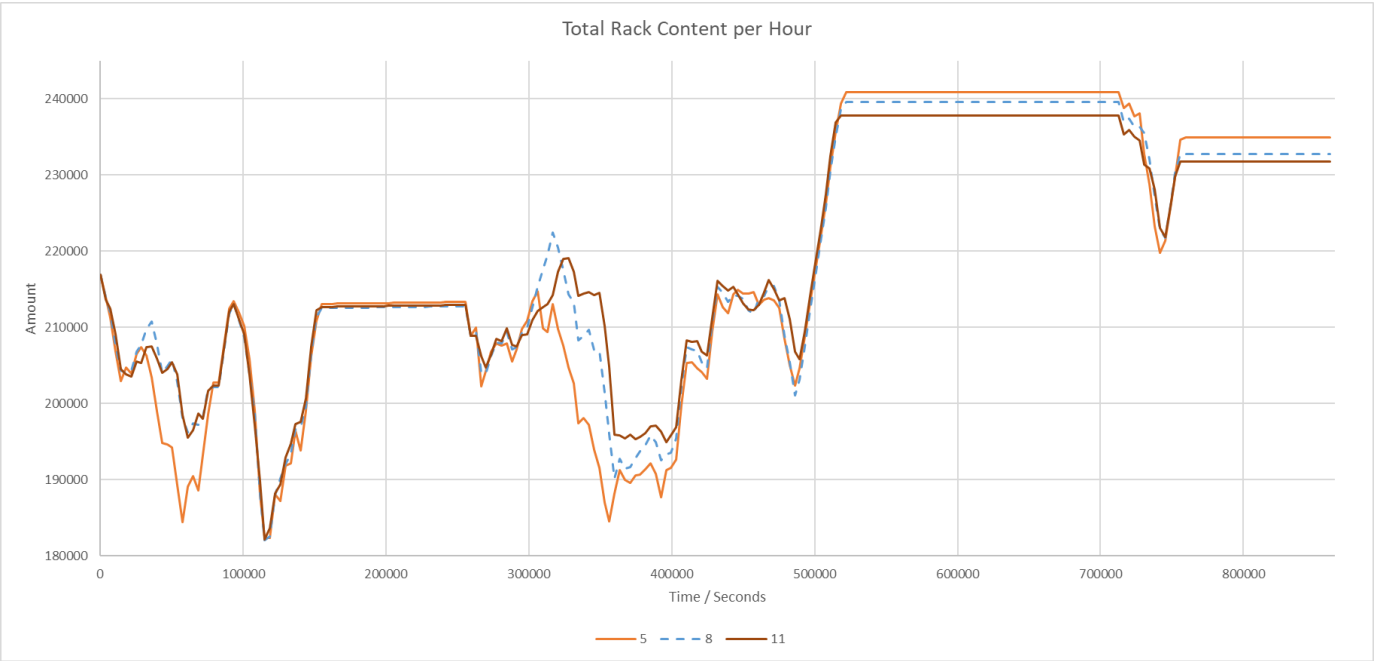
Endringer i reallokeringsparameteren påvirker handlingsbildet til ASRS. Ved å sette denne parameteren lavt vil flere kolli bli sendt gjennom den alternative ruten og hvis den settes høyt vil det ta tid og kapasitet fra ASRS, slik som forventet. Hvis ASRS må bruke mye tid på å reallokere vil det gå utover antall kolli som blir sendt ut til COM, men hvis mye blir sendt gjennom den alternative ruten vil det ta opp kapasitet fra Input.

Det er forskjell i hvor mye COM plukker per time. Det er en trend ved lav parameter at svingningene i output er større enn ved høye parametere. Dette kommer av at ASRS kan levere flere kolli da de ikke trenger å bruke tid på å reallokere. Ved høye parametere har svingningene lavere amplitude og Output er jevnere. Alle parameterne følger de samme trendene da ordre blir tilgjengelig på samme tidspunkt, men antall plukk per time varierer.



Figur 46: Diskusjon - Output per time - Parameter 5, 8 og 11

Når Output har mye svingninger går lagerbeholdningen opp og ned i takt med dette. Når det er lite Output vil lagerbeholdningen bli høyere og det vil bli vanskeligere for Input å finne plass. Dette vil si at ved lavere reallokeringsparameter vil Output ha større svingninger som vil gi negative innvirkninger på lagerbeholdningen og Input. Kranene vil også være mer Idle i disse tilfellene. Ved å ha en lav parameter vil kranene brukes mindre og flere kolli vil bli prioritert ved den alternative ruten.



Figur 47: Disuksjon - Lagerbeholdning per time - 5, 8 og 11

Selv om kranene bruker mindre tid på reallokeringer og er mer Idle, kan kolli bruke lengre tid på å bli reallokert da den alternative ruten vil ta lengre tid i de fleste tilfeller, som vises i *Figur 35: Simulering - Gjennomsnittlig tid brukt for å reallokere en ordre - Alle reallokeringsparametere*. Dette medfører at ordre kan bruke lengre tid på å bli klare til plukking. Ved å sende mange kolli gjennom den alternative ruten kan dette påvirke Input ved å ta opp kapasitet på samlebåndene.

Hvis en bruker en høyere reallokeringsparameter vil Output bli jevnere og amplitudene mindre. Hvis Output er jevnere vil lagerbeholdningen være jevnere og det blir lettere for Input å finne plass. Jevnere lagerbeholdning over tid vil sørge for at flere kolli blir funnet og færre kolli blir slettet fra ordre.

Ved å sende alle reallokeringer gjennom ASRS vil en utnytte kranene bedre på sikt, men Output til COM vil slite ved høy pågang da mye tid blir brukt på å reallokere. Input båndene vil ikke bli overbelastet av reallokeringer, men det kan hende at ASRS ikke får tatt i mot de kolli den skal. Ved å ha høyere parameter ser en at tiden det tar å reallokere en ordre går ned. Dette da reallokering mellom ASRS tar mindre tid enn ved den alternative ruten.

De fleste reallokeringer skjer ved korte avstander og det blir reallokert like mye for hver parameter. Lav parameter gir naturligvis flere alternative reallokeringer og dette øker mer og mer jo lavere den er. De fleste reallokeringer skjer ved korte avstander og oftest mellom nabo soner. En for lav styringsparameter for lange reallokeringer vil dermed eksponentielt øke antall kolli som tar den alternative reallokeringsruten, og raskt skape kø på inngående samlebånd.

7.2.2 FEFO

Det er tydelig at FEFO påvirker systemet, både positivt og negativt. Det blir høyere output, færre reallokeringer og lavere lagerbeholdning, men svinn i form av utgått holdbarhetsdato vil forekomme dersom logikken fjernes helt.

Det er store svingninger i Output og forskjellen mellom med og uten FEFO er store. Fjerning av FEFO reduserer antallet reallokeringer, og dermed kan ASRS sende ut hvilket som helst kolli til COM og ordrer blir ferdig fortere. Siden det plukkes mye ordre i starten er det noen intervaller hvor det plukkes mindre da alle ordrer blir ferdig plukket. Dette gir store svingninger i Output grafen.

ASRS jobber mer effektivt og har derfor mindre total aktiv tid. Siden det er færre reallokeringer blir det brukt mindre tid på dette og mer tid på Output. Ved å fjerne FEFO bruker kranene kun 5.5% av tiden sin på reallokeringer. De reallokeringene som oppstår er kun fordi riktig kolli ikke finnes i riktig sone. Avstanden kolli blir reallokert på har et markant flertall rundt lave distanser og det er et mindretall som blir sendt gjennom den alternative ruten.

Siden færre kolli trengs å reallokeres bruker systemet mindre tid på å klargjøre en ordre enn hvis FEFO er implementert. Dette gjenspeiles i Output grafen da mer Output gjennomføres tidligere.

Selv om FEFO påvirker systemet ved at det må gjennomføres mange flere reallokeringer og det tar kapasitet fra ASRS-kranene er det en nødvendig logikk. FEFO sørger for minimalt med svinn i form av at varer går ut på dato og at de varene som har lavest holdbarhet blir plukket først. FEFO er også verdiskapende for kunden fordi produktene som leveres har en til to dager lengre holdbarhet.

7.2.3 Utfordringer ved simulering

Det har vært en utfordrende prosess å prøve å gjenskape en god 3D-modell og simulering av det komplekse lageret til ASKO SL Kjøl.

Det har ikke vært mulig å hente ut den fullstendige lagerbeholdningen hos ASKO SL Kjøl for et gitt tidspunkt. Som følge av dette måtte Initial Inventory i modellen beregnes ut ifra Input og Output i lageret over en tidsperiode. Denne metoden for å beregne en startbeholdning ga en ufullstendig startbeholdning i modellen, som medførte at mange ordrer i løpet av simuleringen ikke fant alle kolliene de skulle ha. I løpet av de 10 simulerte dagene manglet totalt cirka 40 000 kolli i ordrer. Dette er cirka 5% av alle utgående kolli i simuleringen, som ikke er en ubetydelig mengde. Problemet ble litt redusert ved å undersøke hvilke SKU-er som manglet, for så å legge til cirka 12 000 kolli i den manglende startbeholdningen. Dette løste derimot ikke hele problemet, og det hadde ikke vært en god løsning å legge til flere manglende kolli i startbeholdningen, ettersom at dette vil føre til andre problemer i simuleringen på grunn av en for høy lagerbeholdning.

En annen løsning som bidro ytterligere til å redusere dette problemet, var å implementere utsettelse av ordrer, som forklart i

Manuell **Programmering** under kapittel 5.3.3. Utsettelse av ordrer viste seg derimot å skape nye problemer i simuleringen, hvor alt for mange ordrer ble utsatt som følge av manglende kolli. Dette medførte at lagerbeholdningen ved enkelte tidspunkt ble for høy, slik at innkommende kolli ikke fikk plasser i lageret. Som følge av en for mangelfull startbeholdning, og de negative konsekvensene utsettelse av ordrer medførte, ble logikken for utsettelse av ordrer skrudd av i de endelige simuleringene som ble brukt for oppgaven.

Et annet aspekt som har vært utfordrende er tilnærmingen av plukkraten i simuleringen mot plukkraten hos ASKO SL Kjøl. I valideringsfasen av simuleringen har flere logiske parametere blitt justert, som blant annet laste og lossetider for ASRS-kraner, antall ordrer som kan arbeides på samtidig og prioritering av arbeidsoppgaver for ASRS. Simuleringen har som regel hatt en høyere plukkrate enn lageret til ASKO SL Kjøl, så det har blitt eksperimentert med forskjellige løsninger for å redusere hastigheten på plukking uten å innføre større begrensninger.

Antall reallokeringer som kan jobbes på samtidig ga det beste resultatet ved rundt maksimalt 20 reallokeringer om gangen. Denne begrensingen medførte at COM-ene til tider venter på at reallokeringer må fullføres, uten at det ble en for stor begrensning. I tillegg hjalp denne begrensingen til å regulere trafikken på reallokeringsbåndene mellom soner, som kunne være problematisk for høyere verdier av denne parameteren.

ASRS-kranene til ASKO SL Kjøl bruker cirka like lang tid på å laste og losse et kolli som fire kolli. Implementering av dette krever manuell programmering i FlexSim, og har ikke vært prioritert. En implementering av denne funksjonaliteten vil gi en bedre replika av det reelle lageret. ASRS-kranene i modellen har en konstant tidsbruk for lasting og lossing av kolli, hvor lasting og lossing av fire kolli tar noe lengre tid i modellen enn i virkeligheten i et forsøk på å balansere dette.

Hos ASKO SL Kjøl har kranene en dynamisk prioritering, hvor henting og plassering av innkommende kolli prioriteres når det er mye Input, og reallokeringer og ordreplukking prioriteres når det er mye som skal ut av lageret eller hvis det er kort tid til en ordre har avgangstid. Å gjenskape et slikt intelligent og dynamisk prioriteringssystem krever relativt avansert programmering, og har dermed ikke blitt implementert i simuleringen som følge av mangel på tid.

Gjennom testing av forskjellige prioriteringer, ga en høyere prioritering for reallokeringer den beste løsningen i simuleringen. Høyere prioritering av reallokeringer sikrer en jevn flyt av

ordrer som er klare til plukking, men har den ulempen at ved de tidspunktene mange nye ordrer slippes inn i systemet, så vil plukkraten synke som følge av at arbeidsoppgavene relatert til reallokeringer vil prioriteres.

Den tilpassede logikken og kompleksiteten i simuleringen har forårsaket at simuleringen kjører treigt. Mot slutten av arbeidet med simuleringen har det tatt nesten et døgn å fullføre en hel simulering. Dette har begrenset valideringsarbeidet, da det har tatt lang tid å se resultater fra justeringer i modellen. Den trege hastigheten på simuleringen er et resultat av at studentene aldri har forsøkt å bygge en så avansert 3D-modell med en såpass høy grad av tilpasset logikk før. Det har blitt gjort forsøk på å øke hastigheten på modellen, blant annet gjennom samtaler med eksperter innenfor 3D-simulering og FlexSim, men en betydelig økning av hastigheten på simuleringen krever større endringer som det ikke har vært tilstrekkelig med tid til å forsøke å implementere.

8.0 Konklusjon

Gjennom analyser av ASKO SL Kjøl sitt automatlager er det konkludert med at COM-ene i OPM-området operer med 67% ytelse. Dette kommer i hovedsak av at COM må vente på å få levert brett fra ASRS. Aktivitetsbildet for ASRS påvirkes av prioriteringslogikken FEFO og av styringsparameteren for reallokeringer. Gjennom simulering av lageret er det vært mulig å analysere hvordan reallokeringsparameteren påvirker ytelsen og hvordan driftsbildet ser ut uten FEFO.

F1: I hvilken grad påvirker reallokeringer ytelsen til ASRS?

Analyser av produksjonsdata viser at reallokeringer tar kapasitet fra ASRS. Reallokeringer tar i snitt opp 4.1% av all tid brukt på produksjon. Ved lav pågang reallokeres det mer fordi ASRS har mindre ordre å jobbe på, mens ved høy pågang reallokeres det lite. Dette da ASRS i hovedsak leverer brett ut til COM.

F2: I hvilken grad påvirker FEFO-logikken antallet reallokeringer og distansen på reallokeringer?

FEFO-logikken sørger for at det gjennomføres flere reallokeringer, og hovedsakelig over større avstander enn én. FEFO alene står for 29% av alle reallokeringer som gjennomføres. Ved alle avstander lengre enn én er det FEFO som står for mer enn halvparten av reallokeringene. Tilsammen så står reallokeringer på grunn av FEFO for 1.2% av all tid brukt på produksjon.

F3: Hvilke kapasitetsbegrensninger pålegges av FEFO?

FEFO påvirker handlingsmønsteret til ASRS og det sørger for at det kommer mindre Output fra ASRS. Uten FEFO vil ordre bli plukket fortere, varebeholdningen vil gå ned og ASRS vil få økt tilgjengelighet. Det kreves færre reallokeringer og ASRS kan derfor jobbe mer effektivt med Output. De reallokeringene som gjennomføres tar mindre tid og det er færre reallokeringer som må bruke den alternative ruten.

F4: Hvordan påvirker endringer av styringsparameteren for lengre reallokeringer handlingsmønsteret for ASRS?

Settes styringsparameteren for lengre reallokeringer lavt, vil det føre til færre reallokeringer mellom ASRS, og flere reallokerte kolli vil ta den alternative ruten. Dette fører til at ASRS-kranene får økt tilgjengelighet, og dermed plukker raskere til COM så lenge det er tilgjengelig ordrer for plukking. Reallokering av ordrer tar derimot lengre tid som følge av den lengre reallokeringsruten, som kan skape et økt tidsrom mellom plukking av hver ordre. Den økte belastningen på inngående samlebånd vil kunne skape kø for inngående vareflyt i tidsrom med mye Input. For mye kø vil medføre at enkelte ordrer må vente lenger enn nødvendig på både innkommende kolli, og reallokerte kolli som tar den alternative ruten.

For høyere styringsparametere for lengre reallokeringer vil tilgjengeligheten til ASRS-kranene synke, og belastningen på inngående samlebånd vil bli noe redusert. Det tar kortere tid å reallokere ferdig ordrer, og plukkingen i COM blir jevnere som følge av den reduserte tilgjengeligheten til ASRS-kranene.

Lavere styringsparametere for lengre reallokeringer vil være fordelaktig når økt tilgjengelighet for ASRS-kraner er viktig. Antallet kolli som tar den alternative reallokeringsruten vil øke eksponentielt jo lavere styringsparameteren er. Et høyere styringsparametere for lengre reallokeringer vil være fordelaktig når Inputen i OPM-området er høy, eller hvis reallokeringer av ordrer må skje så raskt som mulig.

8.1 Videre arbeid

FEFO har en betydelig påvirkning av ytelsen til systemet, men som følge av verdiskapningen denne logikken har for kunden vil ikke være fordelaktig å fjerne denne. En mulighet for å redusere påvirkningen til FEFO kan være å gjøre logikken mindre streng. Dette vil føre til færre reallokeringer, men kan påvirke den verdiskapende faktoren for kunden.

På tross av at ASRS-kranene som regel ikke leverer nok kolli for å maksimere utnyttelsen av COM, har ASRS-kranene en del idle time, til og med ved høy pågang. Dette kan indikere at det også er andre årsaker som bidrar til den reduserte utnyttelsen av COM. En ujevn tilstrømming av ordrer vil skape ujevn produksjon, dersom det ikke implementeres tiltak for å jevne ut produksjonen. Som følge den høye kompleksiteten til det helautomatiske lageret, vil det være nyttig å undersøke andre mulige begrensende årsaker nøyere.

Det er mange muligheter for å gjøre 3D-simuleringen mer realistisk. Noen muligheter er allerede nevnt i *7.2.3 Utfordringer ved simulering.* Kollisjoner mellom Transfer Cars kan implementeres via FlexSims kollisjonsverktøy. Vedlikehold av komponenter kan implementeres via *Mean Time Before Failure* og *Mean Time To Repair* funksjonene. Logikken til modellen kan videreutvikles og forbedres for å gjøre simuleringen mer robust.

Referanser

ASKO, 2018. *ASKO Sentrallager KJØL AS.* [Internett]
Available at: https://asko.no/kontakt-oss/vare-asko-selskap/asko-sentrallager-kjol-as/

ASKO, 2020. *ASKO Noge AS.* [Internett]
Available at: https://asko.no/om-oss/

FlexSim, 2019a. *Key Concepts About Dashboards and Charts.* [Internett]
Available at:
https://docs.flexsim.com/en/19.0/GettingData/StandardDataGathering/KeyConceptsDashboards/
[Funnet 5 May 2021].

FlexSim, 2019b. *The Experimenter and Optimizer.* [Internett]
Available at: https://docs.flexsim.com/en/19.2/Reference/Tools/ExperimenterOptimizer/
[Funnet 05 May 2021].

FlexSim, 2019c. *Writing Logic in FlexSim.* [Internett]
Available at: https://docs.flexsim.com/en/19.2/Reference/CodingInFlexSim/WritingLogic/
[Funnet 05 May 2021].

FlexSim, 2021a. *FlexSim.com.* [Internett]
Available at: https://www.flexsim.com/flexsim/
[Funnet 4 May 2021].

FlexSim, 2021b. *Types of 3D Objects.* [Internett]
Available at: https://docs.flexsim.com/en/21.1/Using3DObjects/TypesOfObjects/
[Funnet 5 May 2021].

Malinovskaya, A., 2021. *White paper: Material Handling Simulation.* [Internett]
Available at: https://www.anylogic.com/blog/white-paper-material-handling-simulation/

Manzini, R., 2012. *Warehousing in the global supply chain : advanced models, tools and applications for storage systems.* London ; New York: Springer.

Ten Hompel, M. & Schmidt, T., 2007. *Warehouse management : automation and organisation of warehouse and order picking systems.* Berlin ; New York: Springer.

van den Berg, J. P. & Zijm, W. H. M., 1999. Models for warehouse management: Classification and examples. *International Journal of Production Economics,* 59(1-3), p. 520.

WITRON, 2020a. *WITRON Order Picking Machinery.* [Internett]
Available at: https://www.witron.de/en/opm-order-picking-machinery.html

WITRON, 2020b. *WITRON Company.* [Internett]
Available at: https://www.witron.de/en/company.html

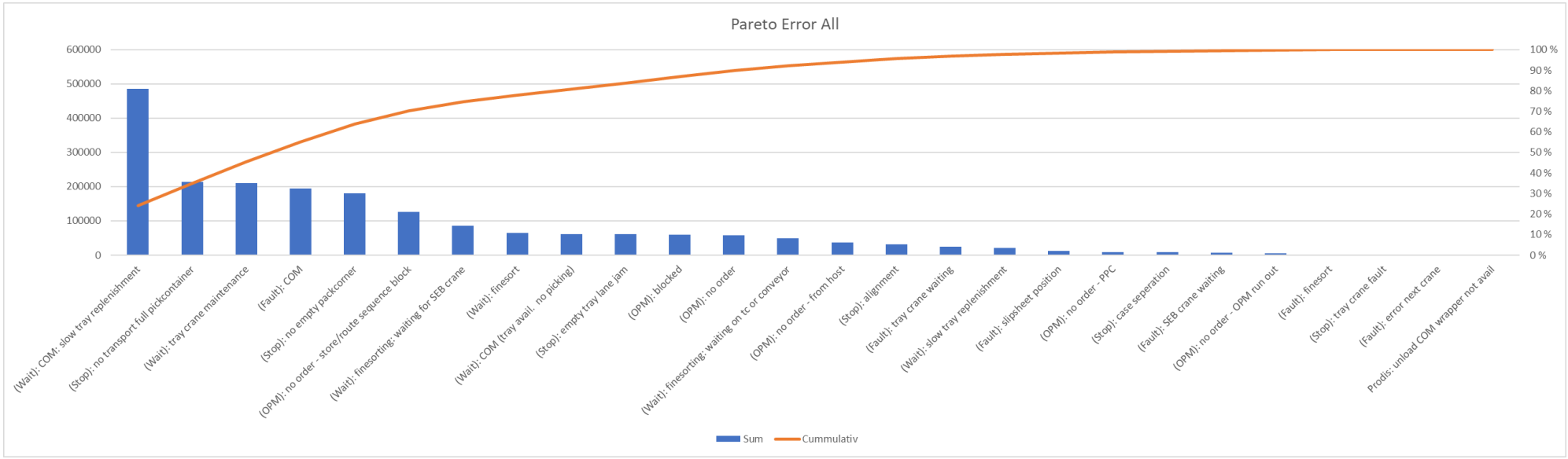
Bilder

Figurer

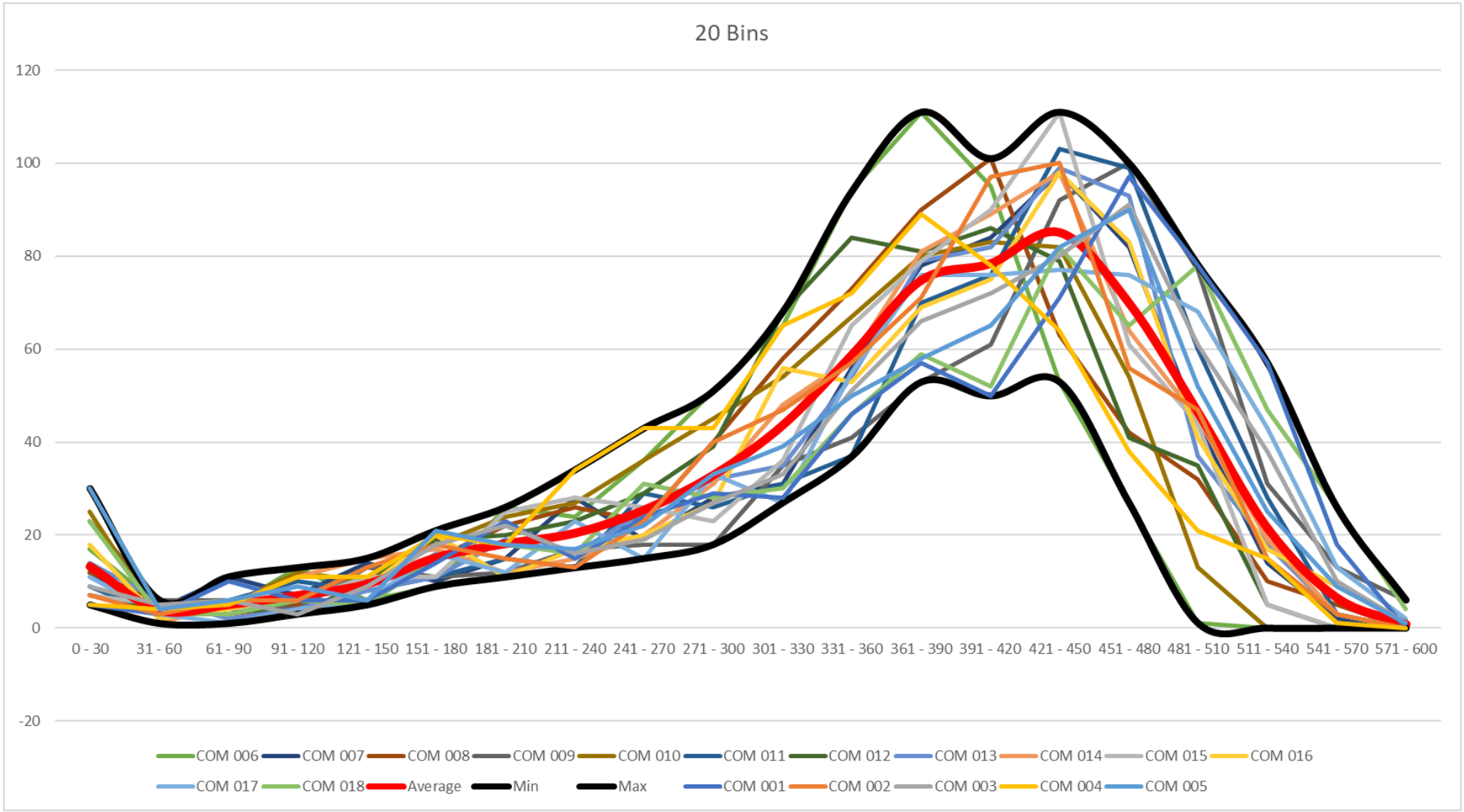
Tabeller

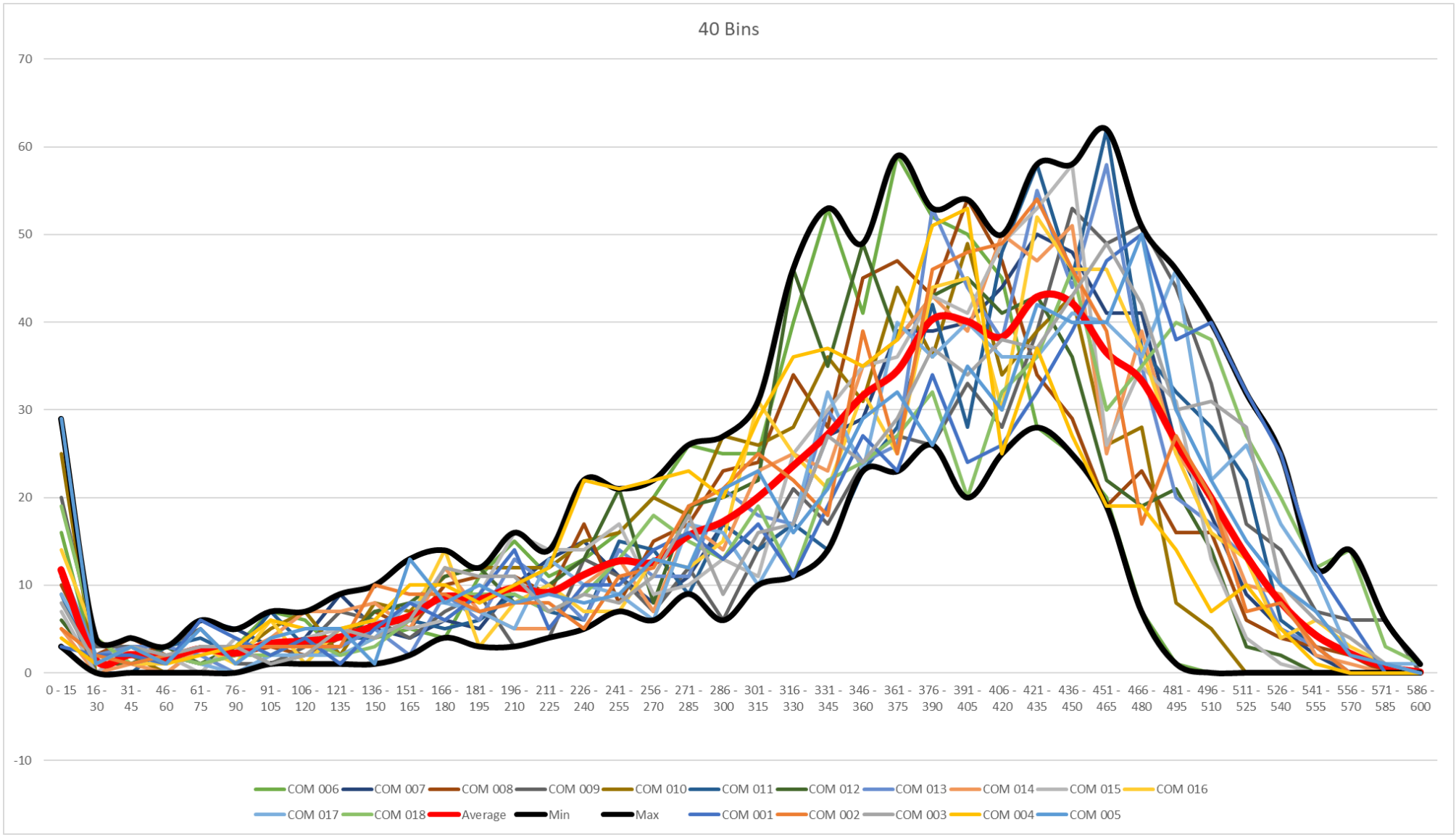
Vedlegg

Vedlegg 1: Full Paretoanalyse

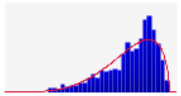
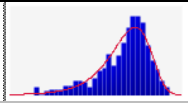
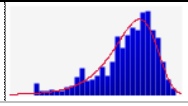
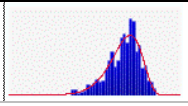
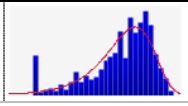
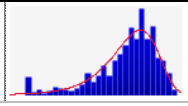


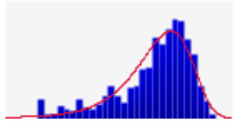
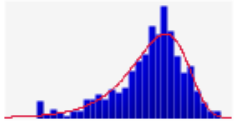
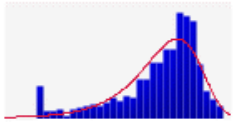
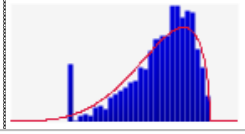
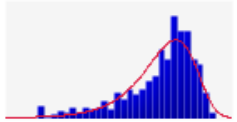
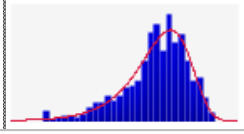
Vedlegg 2: Histogram plukk i COM , 20/40 Bins

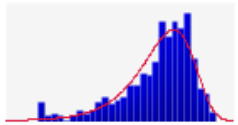
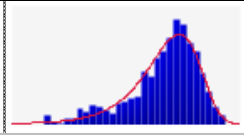
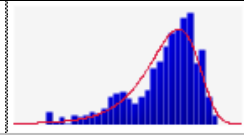
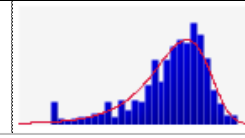
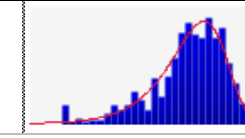
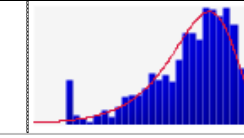



40 Bins
70
60
50
40
30
20
10
0
-10
0 - 15
16 - 30
31 - 45
46 - 60
61 - 75
76 - 90
91 - 105
106 - 120
121 - 135
136 - 150
151 - 165
166 - 180
181 - 195
196 - 210
211 - 225
226 - 240
241 - 255
256 - 270
271 - 285
286 - 300
301 - 315
316 - 330
331 - 345
346 - 360
361 - 375
376 - 390
391 - 405
406 - 420
421 - 435
436 - 450
451 - 465
466 - 480
481 - 495
496 - 510
511 - 525
526 - 540
541 - 555
556 - 570
571 - 585
586 - 600
COM 006
COM 007
COM 008
COM 009
COM 010
COM 011
COM 012
COM 013
COM 014
COM 015
COM 016
COM 017
COM 018
Average
Min
Max
COM 001
COM 002
COM 003
COM 004
COM 005

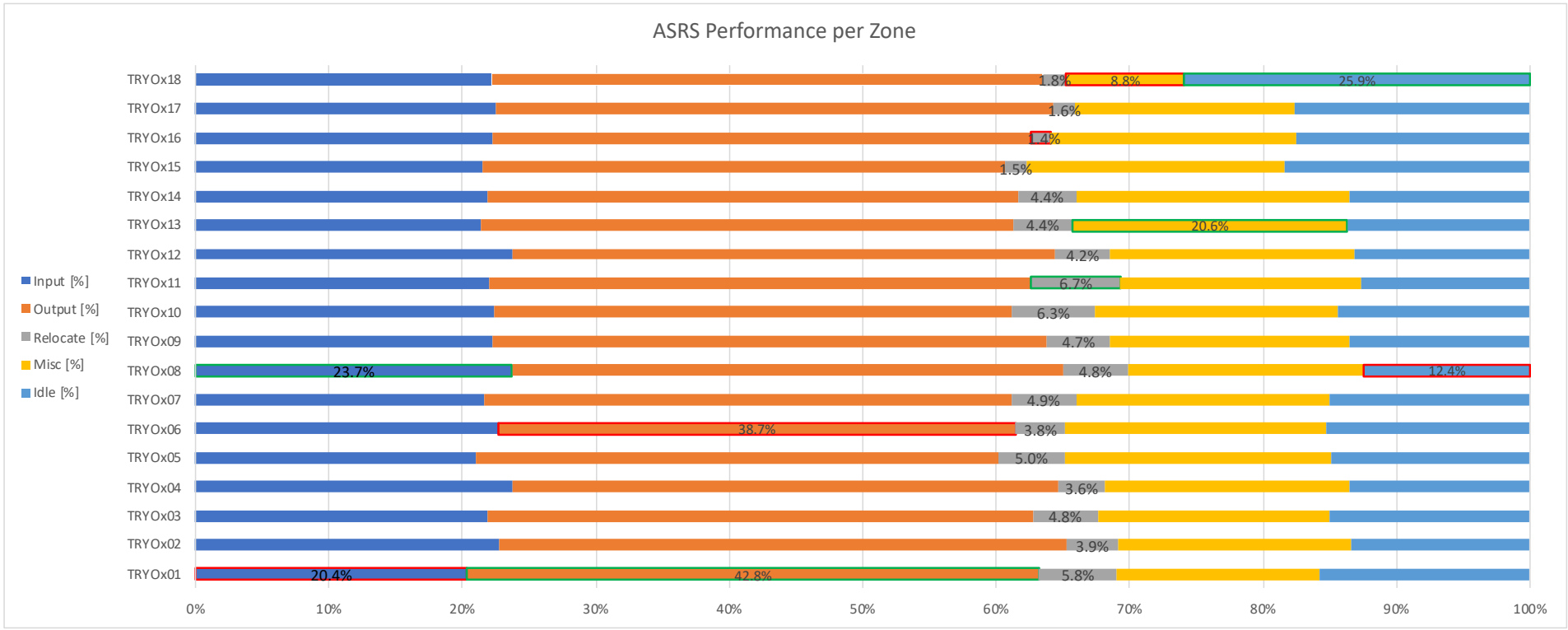
Vedlegg 3: @Risk *Fit to distribution* analyse

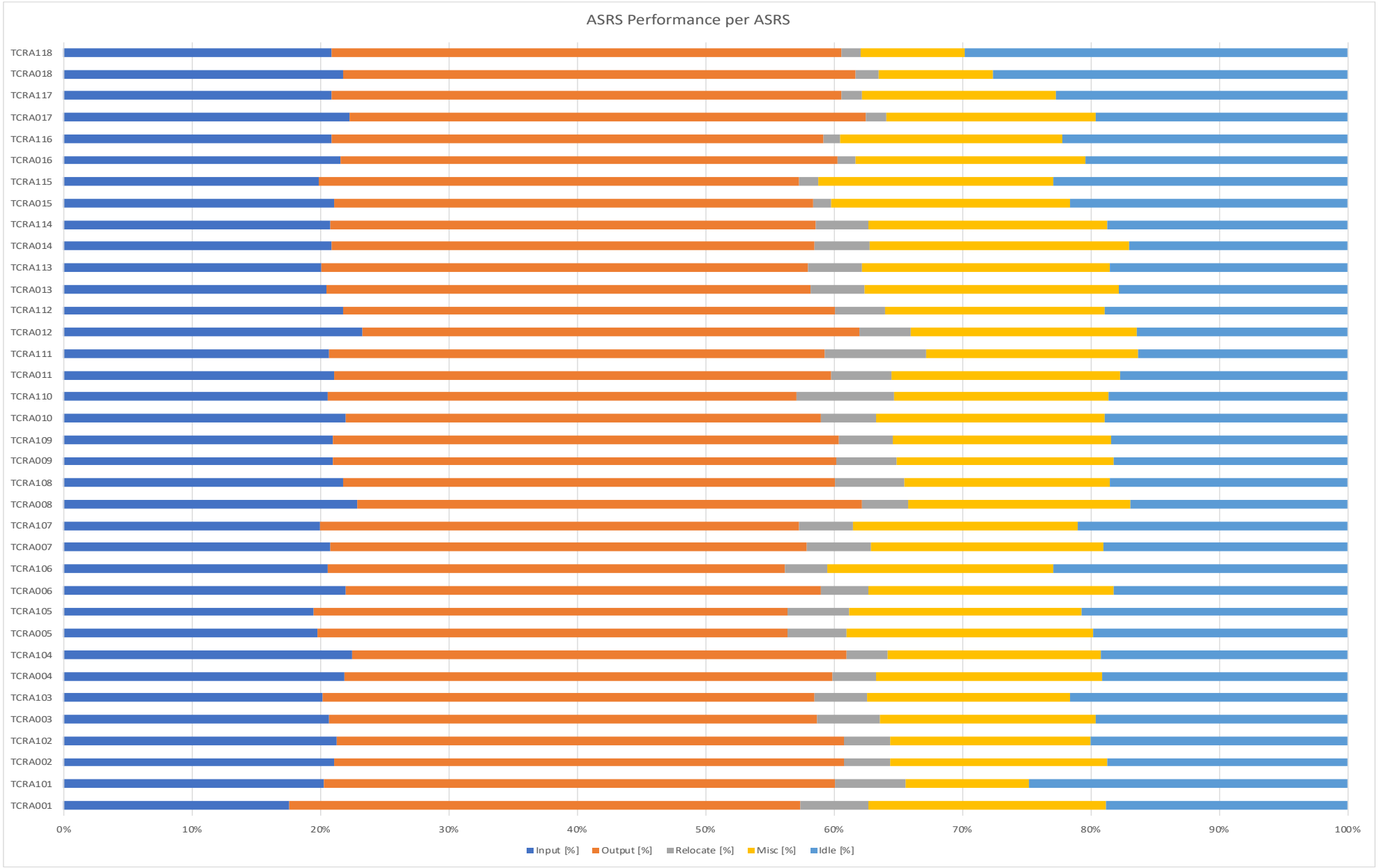
Name	COM 001	COM 002	COM 003	COM 004	COM 005	COM 006
Range	OP13_COM Pick Statistics!H2:H638	OP13_COM Pick Statistics!I2:I638	OP13_COM Pick Statistics!J2:J638	OP13_COM Pick Statistics!K2:K638	OP13_COM Pick Statistics!L2:L638	OP13_COM Pick Statistics!M2:M638
Best Fit (Ranked by AIC)	RiskPert(-142.08,472.41,568.69)	RiskExtValueMin(412.6945,80.3291)	RiskExtValueMin(430.5396,86.3532)	RiskKumaraswamy(7.4647,6.8022,-348.96,608.73)	RiskExtValueMin(416.8195,96.1750)	RiskExtValueMin(369.7869,71.9035)
Function	386.0416667	366.3272851	380.6951802	338.7770615	361.3057834	328.2830734
AIC	7720.8161	7624.8142	7726.2895	7669.2034	7877.2068	7502.3046
Minimum	-142.0754	−∞	−∞	-348.9561	−∞	−∞
Maximum	568.6872	+∞	+∞	608.7306	+∞	+∞
Mean	386.0434	366.327274	380.695182	338.7802	361.305816	328.283124
Mode	472.4121	412.694495	430.539594	379.6098	416.819531	369.786931
Median	406.031	383.252838	398.890035	351.5579	381.570152	343.433382
Std. Deviation	117.3868	103.026087	110.752264	103.3499	123.349228	92.219779
Graph						

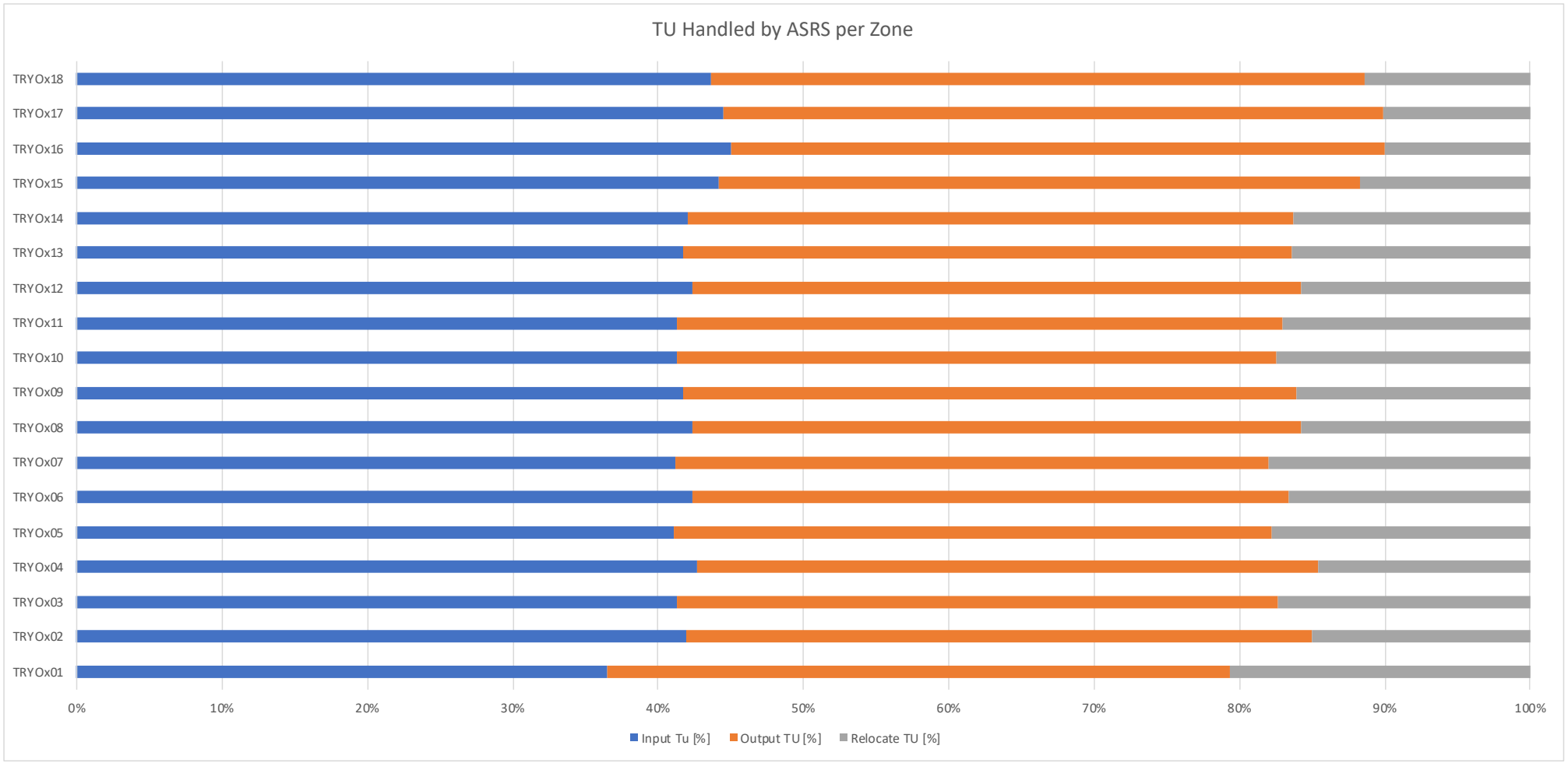
COM 007	COM 008	COM 009	COM 010	COM 011	COM 012
OP13_COM Pick Statistics!N2:N638	OP13_COM Pick Statistics!O2:O638	OP13_COM Pick Statistics!P2:P638	OP13_COM Pick Statistics!Q2:Q638	OP13_COM Pick Statistics!R2:R638	OP13_COM Pick Statistics!S2:S638
RiskExtValueMin(413.3796,82.5579)	RiskExtValueMin(397.5716,85.1488)	RiskExtValueMin(436.7676,90.6258)	RiskPert(-192.83,410.89,505.56)	RiskExtValueMin(428.6752,78.7311)	RiskExtValueMin(394.7062,78.7041)
365.7258869	348.4223788	384.4569686	326.0483333	383.2303758	349.2769606
7686.3226	7670.4152	7813.6604	7694.5773	7631.3758	7577.4843
−∞	−∞	−∞	-192.8284	−∞	−∞
+∞	+∞	+∞	505.5556	+∞	+∞
365.725889	348.422373	384.45701	326.0487	383.230343	349.277003
413.379612	397.571604	436.767636	410.8912	428.675185	394.706243
383.121069	366.363462	403.552113	345.6833	399.819208	365.860173
105.884641	109.207601	116.232092	115.3516	100.976597	100.94193
					

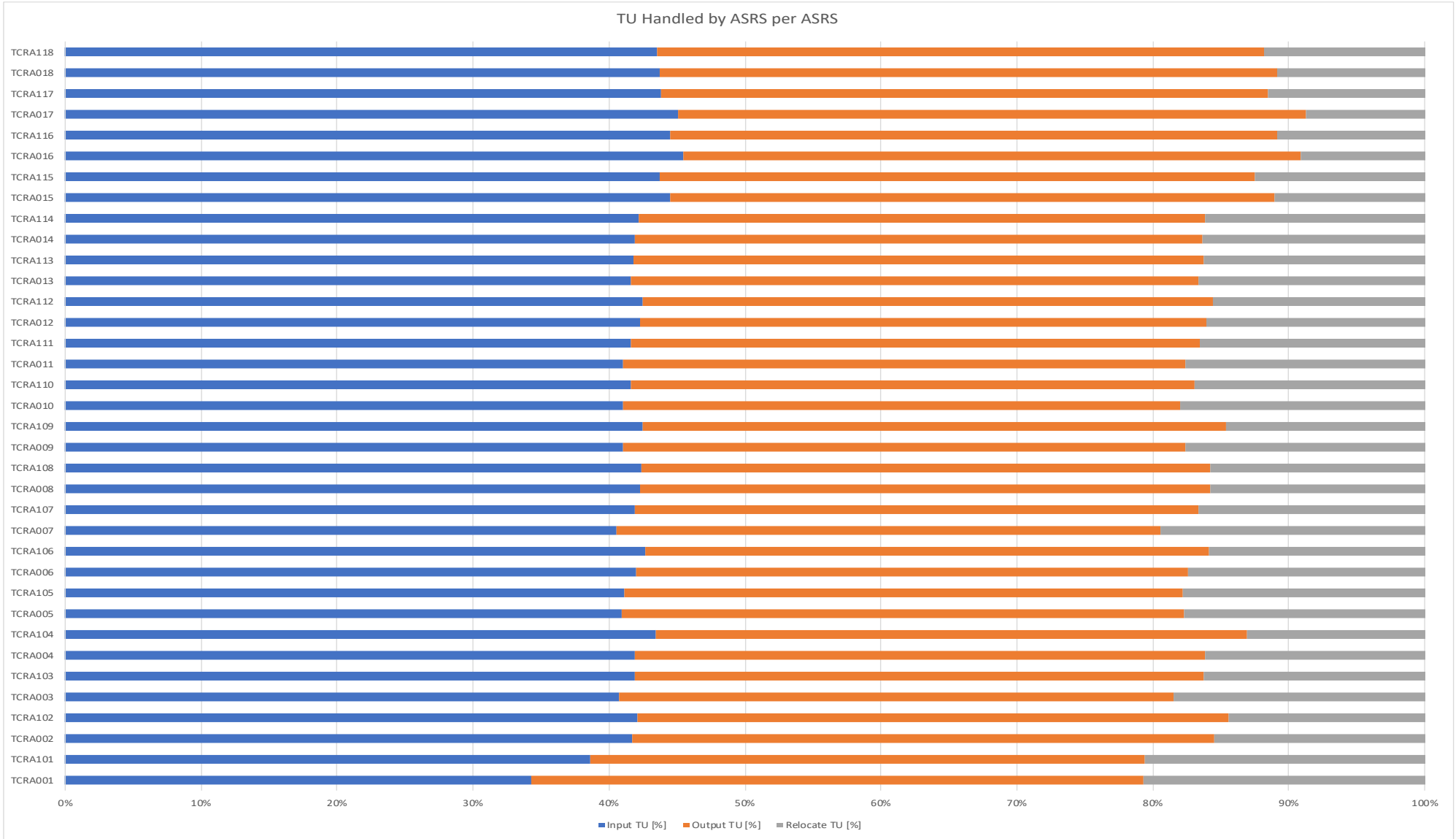
COM 013	COM 014	COM 015	COM 016	COM 017	COM 018
OP13_COM Pick Statistics!T2:T638	OP13_COM Pick Statistics!U2:U638	OP13_COM Pick Statistics!V2:V638	OP13_COM Pick Statistics!W2:W638	OP13_COM Pick Statistics!X2:X638	OP13_COM Pick Statistics!Y2:Y638
RiskExtValueMin(417.3908,79.1280)	RiskExtValueMin(416.0243,80.4849)	RiskExtValueMin(408.1355,76.1659)	RiskExtValueMin(417.3668,85.9802)	RiskExtValueMin(434.7268,84.5935)	RiskExtValueMin(438.6448,96.4813)
371.7168789	369.5671549	364.1713494	367.7376817	385.8981067	382.9542823
7623.6244	7624.697	7579.0079	7715.8244	7689.5425	7871.12
−∞	−∞	−∞	−∞	−∞	−∞
+∞	+∞	+∞	+∞	+∞	+∞
371.716849	369.567135	364.17138	367.73766	385.898139	382.954242
417.390791	416.024272	408.135525	417.36678	434.726804	438.644777
388.389343	386.525521	380.219742	385.853925	403.722211	403.283123
101.485649	103.225878	97.686549	110.273894	108.495317	123.742113
					

Vedlegg 4: ASRS Performance / TU Handled by ASRS




ASRS Performance per ASRS
TCRA118
TCRA018
TCRA117
TCRA017
TCRA116
TCRA016
TCRA115
TCRA015
TCRA114
TCRA014
TCRA113
TCRA013
TCRA112
TCRA012
TCRA111
TCRA011
TCRA110
TCRA010
TCRA109
TCRA009
TCRA108
TCRA008
TCRA107
TCRA007
TCRA106
TCRA006
TCRA105
TCRA005
TCRA104
TCRA004
TCRA103
TCRA003
TCRA102
TCRA002
TCRA101
TCRA001
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Input [%]
Output [%]
Relocate [%]
Misc [%]
Idle [%]


TU Handled by ASRS per Zone
TRYOx18
TRYOx17
TRYOx16
TRYOx15
TRYOx14
TRYOx13
TRYOx12
TRYOx11
TRYOx10
TRYOx09
TRYOx08
TRYOx07
TRYOx06
TRYOx05
TRYOx04
TRYOx03
TRYOx02
TRYOx01
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Input Tu [%]
Output TU [%]
Relocate TU [%]


TU Handled by ASRS per ASRS
TCRA118
TCRA018
TCRA117
TCRA017
TCRA116
TCRA016
TCRA115
TCRA015
TCRA114
TCRA014
TCRA113
TCRA013
TCRA112
TCRA012
TCRA111
TCRA011
TCRA110
TCRA010
TCRA109
TCRA009
TCRA108
TCRA008
TCRA107
TCRA007
TCRA106
TCRA006
TCRA105
TCRA005
TCRA104
TCRA004
TCRA103
TCRA003
TCRA102
TCRA002
TCRA101
TCRA001
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Input TU [%]
Output TU [%]
Relocate TU [%]

Vedlegg 5: findSlotsForBatch() script

findSlotsForBatch

```
/**Custom Code*/
Token token = param(1);

//control that SKUs in batch are same
int sku = token.SKUs[1];
Variant sameSku = 1;
for (int i=1; i<=token.SKUs.length; i++) {
    if (token.SKUs[i] != sku) {
        //batch contains different skus
        sameSku = 0;
    }
    sku = token.SKUs[i];
}

if (sameSku == 1) {
    if (token.items.length == 4) {
        Array skusInZone;
        for (int i=1; i<=18; i++) {
            int rackZone = i;
            string query1 = "WHERE zone = $2 AND slot.slotItems.length >= 1 AND slot.slotItems[1].item.SKU = $1 LIMIT 5";
            Array slots = Storage.system.querySlots(query1, 0, sku, rackZone);
            skusInZone.push(slots.length);
        }

        //lowVal contains the lowest amount of slots found in any of the zones
        int lowVal = 1000;
        for (int i=1; i<=skusInZone.length; i++) {
            if (skusInZone[i] < lowVal) {lowVal = skusInZone[i];}
        }
        //lowestValues contains the indexes of the zones with lowVal
        Array lowestValues;
        for (int i=1; i<=skusInZone.length; i++) {
            if (skusInZone[i] == lowVal) {lowestValues.push(i);}
        }
        //generate random number which is the index of one of the indexes in lowestValues
        int randNum = duniform(1, lowestValues.length, 0);
        //finds a random storage slot in one of the zones with least amount of slots containting same SKU
        string query2 = "WHERE zone = $1 AND slot.slotItems.length = 0 ORDER BY rand()";
        Storage.Slot foundSlot = Storage.system.findSlot(query2, 0, lowestValues[randNum]);

        for (int i=1; i<=token.items.length; i++) {
            Storage.Item itm = Storage.Item(token.items[i]);
            itm.assignedSlot = foundSlot;
        }
        return foundSlot;
    } else {
```

findSlotsForBatch

```
    } else {
        int slotLength = 4 - token.items.length;
        string query3 = "WHERE slot.slotItems.length > 0 AND slot.slotItems[1].item.SKU = $2 AND slot.slotItems.length <= $1 ORDER BY rand()";
        Storage.Slot foundSlot = Storage.system.findSlot(query3, 0, slotLength, sku);

        if (foundSlot) {
            for (int i=1; i<=token.items.length; i++) {
                Storage.Item itm = Storage.Item(token.items[i]);
                itm.assignedSlot = foundSlot;
            }
            return foundSlot;
        } else {
            Storage.Slot foundSlot = Storage.system.findSlot("WHERE slot.slotItems.length = 0 ORDER BY rand()", 0);

            if (foundSlot) {
                for (int i=1; i<=token.items.length; i++) {
                    Storage.Item itm = Storage.Item(token.items[i]);
                    itm.assignedSlot = foundSlot;
                }
                return foundSlot;
            }
        }

    }
} else {
    //should not be possible
    //for bug fixing purposes
    print("Batch not containing same SKUs");
    print(time());
    print("");
}

//storage slot not found
//find zone with lowest content
Array zoneContent = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0];
for (int i=1; i<=Group("Racks").length; i++) {
    zoneContent[Group("Racks")[i].zone] += Group("Racks")[i].stats.content.value;
}
int lowestContent = 100000;
for (int i=1; i<=zoneContent.length; i++) {
    if (zoneContent[i] <= lowestContent) {
        lowestContent = zoneContent[i];
    }
}
int lowestContentZone = zoneContent.indexOf(lowestContent);
string query4 = "WHERE zone IS $1 AND slot.slotItems.length = 0 ORDER BY rand()";
Storage.Slot foundSlot = Storage.system.findSlot(query4, 0, lowestContentZone);

if (foundSlot) {
    for (int i=1; i<=token.items.length; i++) {
        Storage.Item itm = Storage.Item(token.items[i]);
        itm.assignedSlot = foundSlot;
    }
    return foundSlot;
}
```

Vedlegg 6: findCOM() script

findCOM*

```
/**Custom Code*/
Token token = param(1);

//array with all SKUs in order, and quantity of each SKU
Array SKUs;
for (int n=1; n<=token.Rows.length; n++) {
    Array info;
    info.push(Table("OrderHistory")[token.Rows[n]][3]);
    info.push(Table("OrderHistory")[token.Rows[n]][4]);
    SKUs.push(info);
}

//array with one value per zone. Value is number of SKUs from order in given zone
Array SKUs_in_zone;

//itemms array contains all items in storage system with SKUs matching order
Array itemms;
for (int j=1; j<=SKUs.length; j++) {
    itemms.push(Storage.system.queryItems("WHERE SKU IS $1 ORDER BY RAND() LIMIT 500", 0, SKUs[j][1]));
}

//Loop goes through each COM and counts how many SKUs from the order are in each COMs zone
for (int i=1; i<=Group("COMstackers").length; i++) {
    //counter for SKUs_in_zone
    int nr = 0;
    //racks array contains the racks belonging to the current COM in the loop
    Array racks;
    //Push racks in same zone as COM to array
    for (int k=1; k<=Group("Racks").length; k++) {
        if (Group("COMstackers")[i].zone == Group("Racks")[k].zone) {
            racks.push(Group("Racks")[k]);
        }
    }
    //if SKU is in racks in same zone as COM, nr++
    for (int l=1; l<=itemms.length; l++) {
    int counter = 1;
        for (int k=1; k<=itemms[l].length; k++) {
            for (int m=1; m<=racks.length; m++) {
                if (itemms[l][k].as(Storage.Slot.Item).item.up == racks[m].as(Object)) {
                    if (counter <= SKUs[l][2]) {
                        nr++;
                        counter++;
                    } else {
                        //doesnt count multiple items with same SKU in same rack with break function
                        break;
                    }
                }
            }
        }
    }
    SKUs_in_zone.push(nr);
}
```

findCOM

```
53
54 //three highest values from SKUs_in_zone array
55 int maxVal = 0;
56 for (int o=1; o<=SKUs_in_zone.length; o++) {
57     if (SKUs_in_zone[o] > maxVal) {maxVal = SKUs_in_zone[o];}
58 }
59 int sndMaxVal = 0;
60 for (int o=1; o<=SKUs_in_zone.length; o++) {
61     if (SKUs_in_zone[o] == maxVal) {continue;}
62     if (SKUs_in_zone[o] > sndMaxVal) {sndMaxVal = SKUs_in_zone[o];}
63 }
64 int trdMaxVal = 0;
65 for (int o=1; o<=SKUs_in_zone.length; o++) {
66     if (SKUs_in_zone[o] == maxVal) {continue;}
67     if (SKUs_in_zone[o] == sndMaxVal) {continue;}
68     if (SKUs_in_zone[o] > trdMaxVal) {trdMaxVal = SKUs_in_zone[o];}
69 }
70
71 //find and prioritize COMs
72 //COM in zone with highest item count is priority 1 and so forth
73 //COMs are prioritized in order in allComs array
74 Array allComs;
75 Array comsPri1;
76 Array comsPri2;
77 Array comsPri3;
78 for (int i=1; i<=SKUs_in_zone.length; i++) {
79     if (SKUs_in_zone[i] == maxVal) {comsPri1.push(Group("COMstackers")[i]); continue;}
80     if (SKUs_in_zone[i] == sndMaxVal) {comsPri2.push(Group("COMstackers")[i]); continue;}
81     if (SKUs_in_zone[i] == trdMaxVal) {comsPri3.push(Group("COMstackers")[i]); continue;}
82 }
83 if (comsPri1 != []) {allComs.push(comsPri1);}
84 if (comsPri2 != []) {allComs.push(comsPri2);}
85 if (comsPri3 != []) {allComs.push(comsPri3);}
86
87 //Check availability of COMs
88 //these tokens contains orders which are currently in queue to be reallocated or picked
89 Array OrderTokens = gettokens(current, getactivity("ProcessFlow2", "Enter OrderPickingLimiter"));
90 Array ReallocTokens = gettokens(current, getactivity("ProcessFlow2", "Enter ReallocationsLimiter"));
91
92 Array avail;
93 for (int i=1; i<=allComs.length; i++) {
94     avail.push([]);
95     for (int j=1; j<=allComs[i].length; j++) {
96         int count = 0;
97         for (int k=1; k<=OrderTokens.length; k++) {
98             if (OrderTokens[k].bestCOM == allComs[i][j]) {count++;}
99         }
100        for (int k=1; k<=ReallocTokens.length; k++) {
101            if (ReallocTokens[k].bestCOM == allComs[i][j]) {count++;}
102        }
103        avail[i].push(count);
104    }
105 }
106
```

findCOM

```
106
107 //returns COM with highest priority which is most available
108 //one COM can have a maximum of 10% of current orders in queue
109 int maxTokensPerCOM = (OrderTokens.length + ReallocTokens.length)*0.1;
110 for (int i=1; i<=avail.length; i++) {
111     Array isAvail;
112     int low = 1000;
113     for (int j=1; j<=avail[i].length; j++) {
114         if (avail[i][j] <= low) {
115             low = avail[i][j];
116         }
117     }
118     for (int j=1; j<=avail[i].length; j++) {
119         if (avail[i][j] == low) {
120             isAvail.push(j);
121         }
122     }
123     if (isAvail.length == 0 || avail[i][isAvail[1]] > maxTokensPerCOM) {
124         continue;
125     } else {
126         int rand = duniform(1, isAvail.length, 1);
127         return allComs[i][isAvail[rand]];
128     }
129 }
130
131 //if no COMs are available, return COM in zone with highest item count
132 //with least amount of orders in queue
133 int lowVal = 1000;
134 for (int i=1; i<=avail[1].length; i++) {
135     if (avail[1][i] <= lowVal) {
136         lowVal = avail[1][i];
137     }
138 }
139 return comsPri1[avail[1].indexOf(lowVal)];
```

Vedlegg 7: findItemNearCOM() script

findItemNearCOM

```
/**Custom Code*/
Token token = param(1);
int com = token.bestCOM.zone;
Storage.Slot.Item foundItem;

//find lowest Expiry Date of SKU
Storage.Slot.Item slotItem = Storage.system.findItem("WHERE SKU IS $1 ORDER BY Item.item.ExpiryDate ASC", 0, token.SKU);

//if SKU doesnt exist in storage system
if (!slotItem) {
    return 0;
}
int expDate = slotItem.item.ExpiryDate;

//check if item is in same zone as COM
string queryStr = "WHERE SKU IS $1 AND Item.slot.storageObject.zone IS $2 AND Item.item.ExpiryDate IS $3 ORDER BY Item.slot.slotItems.length ASC, RAND()";
foundItem = Storage.system.findItem(queryStr, 0, token.SKU, com, expDate);
if (foundItem) {
    return foundItem;
}

//Check rest of zones, alternating to check closest zones first
int counter = com;
int limit = 3;
for (int i=1; i<=17; i++) {
    switch (limit)
    {
        case 2:
        {
            counter--;
            break;
        }
        case 1:
        {
            counter++;
            break;
        }
        case 3:
        {
            if (i%2 == 0) {
                if (counter + i <= 18) {
                    counter += i;
                } else {
                    limit = 2; //upper limit reached
                    counter--;
                }
            } else {
                if (counter - i >= 1) {
                    counter -= i;
                } else {
                    limit = 1; //lower limit reached
                    counter++;
                }
            }
        break;
        }
    }
    //return item in closest zone
    foundItem = Storage.system.findItem(queryStr, 0, token.SKU, counter, expDate);
    if (foundItem) {
        return foundItem;
    }
}

//no item found
return 0;
```

Vedlegg 8: checkInboundItems() script

checkInboundItems

```
/**Custom Code*/
//check if order delaying is allowed
if (!allowOrderDelaying) {
    return 0;
}

Token token = param(1);

int sku = token.SKU;
int now = time();
int depTime = token.DepartureTime;

Table table = Table.query("SELECT StartTime FROM Inbound WHERE SKU IS $1", sku);
table.cloneTo("QueryResult");
Table res = Table("QueryResult");

//check if sku is in inbound table
if (res.numRows == 0) {
    return 0;
}

//if inbound SKU comes in given timezone, delay order
for (int i=1; i<=res.numRows; i++) {
    if (res[i][1] < (depTime - startOrder) && res[i][1] > now) {
        return res[i][1];
    }
}

//dont delay order
return 0;
```

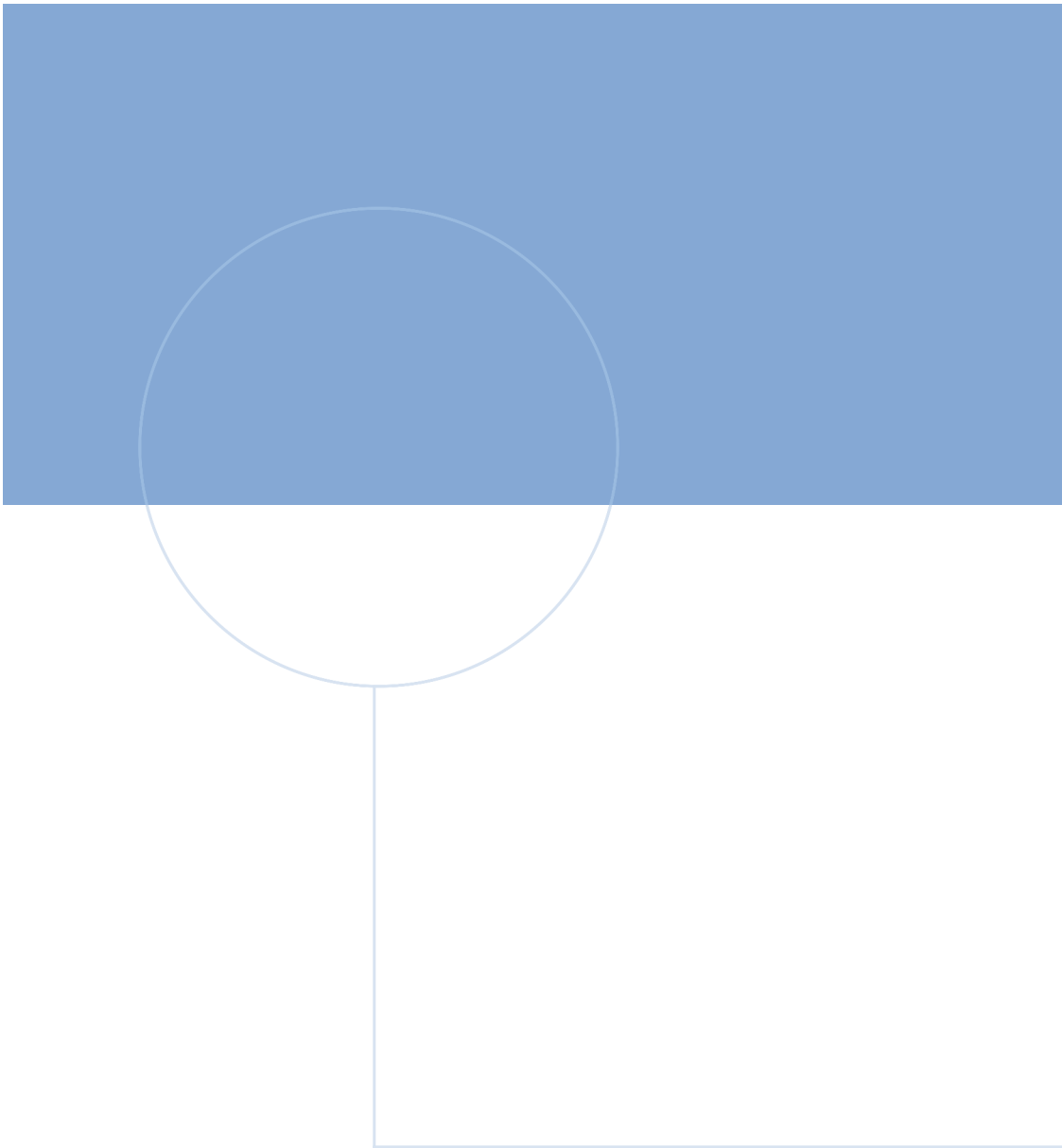
Vedlegg 9: ReallocReason() script

ReallocReason

```
/**Custom Code*/
Token token = param(1);

int zone = token.bestCOM.zone;
int itemZone = token.SlotItem.as(Storage.Slot).storageObject.zone;

if (zone == itemZone) {
    //item is in same zone as COM, no reallocation
    return 0;
} else {
    //Decide reallocate reason:
    Storage.Slot.Item checker = Storage.system.findItem("WHERE SKU IS $1 AND Item.slot.storageObject.zone IS $2", 0, token.SKU, zone);
    if (checker) {
        //reallocate reason is because of FEFO
        return "FEFO";
    } else {
        //reallocate reasin is because of item not existing in zone
        return "Closest Item";
    }
}
```



NTNU

Norwegian University of
Science and Technology