Auteursrechterlijke overeenkomst

Opdat de Universiteit Hasselt uw eindverhandeling wereldwijd kan reproduceren, vertalen en distribueren is uw akkoord voor deze overeenkomst noodzakelijk. Gelieve de tijd te nemen om deze overeenkomst door te nemen, de gevraagde informatie in te vullen (en de overeenkomst te ondertekenen en af te geven).

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling met

Titel: Simulatie-optimalisatie van beslissingen in een transferlijn
Richting: 3de jaar handelsingenieur - major operationeel management en logistiek Jaar: 2008

in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Ik ga akkoord,

GIOVANELLI, Bruno

Datum: 5.11.2008

Simulatie-optimalisatie van beslissingen in een transferlijn

Bruno GIOVANELLI

promotor :
Prof. dr. Gerrit JANSSENS

Eindverhandeling voorgedragen tot het bekomen van de graad handelsingenieur major operationeel management en logistiek

universiteit hasselt

WOORD VOORAF

Deze eindverhandeling is het afsluitende onderdeel van mijn opleiding tot Handelsingenieur in Operationeel Management en Logistiek aan de Universiteit Hasselt te Diepenbeek. De interesse in het operationele gebeuren is gegroeid tijdens de laatste twee jaren van mijn opleiding en dit heeft de keuze van mijn onderwerp sterk beïnvloed. In het kader van dit opleidingsonderdeel heb ik via simulaties onderzoek gedaan naar de optimale buffer allocatie in transferlijnen, rekening houdende met onbetrouwbare machines. Het resultaat is echter veel meer dan dit rapport alleen. Ik heb veel geleerd over het zoeken en samenbrengen van relevante literatuur en om daar conclusies uit te trekken.

Een eindverhandeling wordt niet alleen gemaakt en elke eindverhandeling heeft wel eens knelpunten en moeilijke momenten. Daarom zou ik van deze gelegenheid gebruik willen maken om mijn oprechte dank te betuigen aan iedereen die rechtstreeks en onrechtstreeks heeft meegewerkt om deze eindverhandeling tot een goed einde te brengen.

In de eerste plaats zou ik graag mijn promotor Prof. Dr. Janssens willen bedanken voor zijn begeleiding, advies en informatie die hij ter beschikking stelde. Zijn kritische standpunten hebben mij meerdere malen een dienst bewezen.

Verder zou ik graag ook mijn ouders willen bedanken omdat zij mij de kans hebben gegeven om mij de afgelopen jaren te concentreren op mijn studies. Verder zou ik hun eveneens graag willen bedanken voor hun hulp, geduld, morele en financiële steun die zij mij gegeven hebben tijdens mijn opleiding en de verwezenlijking van deze eindverhandeling. Zij waren eveneens een bron van energie en inspiratie om deze studies te voltooien.

Ten slotte zou ik graag deze eindverhandeling willen opdragen aan mijn overleden voetbalkameraad Wim Coenen die tijdens zijn opleiding Toegepaste Economische Wetenschappen aan de Universiteit Hasselt te Diepenbeek bezweek aan kanker. Requiem In Pace!

Bruno Giovanelli

SAMENVATTING

Deze eindverhandeling stelt zich tot doel meer inzicht te verwerven aangaande enerzijds de *modellering* van transferlijnen met onbetrouwbare machines en anderzijds de bepaling van de *optimale* buffer allocatie in dergelijke transferlijnen. Falingen van machines in transferlijnen kunnen een groot financieel verlies betekenen voor het bedrijf. Aangezien de machines serieel zijn geconfigureerd in dit type productiesysteem kan namelijk de gehele transferlijn stilvallen wanneer één machine faalt, indien geen tussenliggende voorraden werden voorzien. De resulterende bewerkingstijd is variabel omdat enerzijds falingen zich zelden voordoen, doch substantiële vertragingen veroorzaken wanneer falingen zich voordoen.

Aan de hand van simulatie-optimalisatie zal getracht worden om de optimale buffer allocatie te identificeren bij onbetrouwbare transferlijnen. In het tweede hoofdstuk wordt daarom een overzicht gegeven van een aantal **simulatie-optimalisatie** technieken. In dit overzicht worden deze methoden opgesplitst in twee deelgroepen. De eerste deelgroep betreft methoden voor **locale** optimalisatie. Deze deelgroep wordt verder ingedeeld naargelang de beslissingsruimte *discreet* of *continu* is.

In een **discrete** ruimte nemen de beslissingsvariabelen een discrete set van waarden aan. Voor de discrete beslissingsruimte worden zeven technieken bondig beschreven. Deze methoden zijn *Ranking And Selection*, *Multiple Comparison*, *Ordinal Optimization*, *Random Search*, *Nelder-Mead Simplex/Complex Search*, *Single Factor Method* en *Hooke-Jeeves Pattern Search*. De eerste twee technieken functioneren in een *eindige* beslissingsruimte, terwijl de overige technieken in een *oneindige* beslissingsruimte kunnen functioneren.

In een **continue** ruimte nemen de beslissingsvariabelen een reële set van waarden aan. Voor de continue beslissingsruimte worden zes technieken beschreven. Deze methoden zijn *Response Surface Methodology*, *Finite Difference Estimates*, *Perturbation Analysis*, *Frequency Domain Analysis*, *Likelihood Ratio Estimates* en *Stochastic Approximation*.

De tweede deelgroep betreft methoden voor **globale** optimalisatie. Deze technieken werden ontwikkeld voor problemen met multimodale responsoppervlakken, dit wil zeggen dat verscheidene locale optima aanwezig zijn. Voor de globale optimalisatie worden zeven technieken bondig aangehaald. Deze methoden zijn *Genetic Algorithms*, *Evolutionary Programming*, *Evolution Strategies*, *Tabu Search*, *Simulated Annealing*, *Bayesian/Sampling Algorithms* en *Gradient Surface Method*.

Het derde hoofdstuk wordt gewijd aan **buffer allocatie** bij transferlijnen. Dit hoofdstuk is opgedeeld in twee delen. Het eerste deel handelt over **transferlijnen**. In dit deel worden eerst enkele **definities** met betrekking tot transferlijnen gegeven. Deze definities helpen om *transferlijnen*, type van *falingen* en *machinetoestanden* te beschrijven. Vervolgens worden in dit deel een aantal **maatstaven** besproken om de prestaties van transferlijnen te bepalen. Hier komt de *doorzet*, *efficiëntie* en *beschikbaarheid* van transferlijnen aan bod. Ten slotte wordt dit deel afgerond met de **modellering** van transferlijnen. Hierbij wordt vooral aandacht besteed aan de *afhankelijkheid* tussen de machines bij transferlijnen en hoe deze afhankelijkheid doorbroken kan worden. De afhankelijkheid kan doorbroken worden enerzijds door *Redundante Machines* en anderzijds door *Intermediaire (tussenliggende) Buffercapaciteit*.

Het tweede deel van het derde hoofdstuk handelt over de **optimalisatie** van buffer allocatie in transferlijnen. Ten eerste wordt de productie *Line Specific Output Curve* kort aangehaald. Vervolgens worden enkele conventionele resultaten met betrekking tot transferlijnen besproken. Voor **homogene** transferlijnen wordt de *omkeerbaarheid* van een bufferprofiel aangehaald, alsook de *symmetrische voorraad allocatie* en de *geïnverteerde voorraad bowl*. Voor **heterogene** transferlijnen wordt de belangrijkheid van de *bottleneck-operatie* aangehaald. Ten slotte worden eveneens richtlijnen gegeven betreffende de bufferallocatie voor transferlijnen met **onbetrouwbare** machines.

Het vierde hoofdstuk is gewijd aan de **praktijkstudie**. Ten eerste wordt het gebruikte **simulatiemodel** in detail besproken. Zo wordt de logica van het model verklaard en de wijze waarop deze logica vertaald wordt naar de simulatiesoftware *Arena*. Eveneens wordt aandacht besteed aan elke module van het simulatiemodel en aan de input van deze modules.

Vervolgens wordt de methodiek van één specifieke optimalisatietechniek, de **Single Factor Method**, geïllustreerd aan de hand van dit simulatiemodel. Hierin wordt de *doelfunctie* nader gespecificeerd en

de *startsituatie* beschreven, namelijk een transferlijn zonder *Intermediaire Buffercapaciteit*. Het verdere verloop van de methode wordt eveneens beschreven tot het bereiken van de *optimale* oplossing op basis van de doelfunctie. De optimale oplossing die gevonden werd, lijkt aan te sluiten bij de *geïnverteerde voorraad bowl*.

Ten slotte wordt dit hoofdstuk afgerond met het eigenlijke **onderzoek**. De *Single Factor Method* wordt hiervoor toegepast zoals deze beschreven werd in het vorige deel. Hier wordt niet dieper ingegaan op de methodiek van de *Single Factor Method*. De resultaten van dit onderzoek worden overzichtelijk weergegeven en besproken. Ten eerste wordt een transferlijn met **aflopende** gemiddelde bewerkingstijden onderzocht. De optimale buffer allocatie blijkt te *dalen* naar het einde van de transferlijn toe. Vervolgens wordt een transferlijn met **oplopende** gemiddelde bewerkingstijden onderzocht. De optimale allocatie blijkt in dit geval te *stijgen* naar het einde van de transferlijn toe. Daarna wordt een transferlijn met een **bottleneck** onderzocht. De optimale allocatie is *evenredig* verdeeld over de buffers heen in dit geval. Ten slotte wordt een transferlijn onderzocht met een **onbetrouwbare bottleneck**. In dit geval is de optimale allocatie eveneens *evenredig* verdeeld over de buffers heen.

Het vijfde en laatste hoofdstuk bevat de **conclusie** van dit rapport. Hierin worden de belangrijkste conclusies omtrent de optimale buffer allocatie in transferlijnen meegedeeld. Eveneens wordt hierin gecontroleerd of de conclusies van deze eindverhandeling aansluiten bij de traditionele resultaten betreffende transferlijnen.

INHOUDSOPGAVE

HOOFDSTUK I: PROBLEEMSTELLING

In dit hoofdstuk zal aandacht besteed worden aan het praktijkprobleem. Vervolgens zal de centrale onderzoeksvraag geformuleerd worden. Daarna zullen de topics aan bod komen. Dan zullen enkele definities aangehaald worden. Ten slotte zal kort de onderzoeksmethode uitgelegd worden.

1.1 Praktijkprobleem

Het **praktijkprobleem** van deze eindverhandeling kan opgesplitst worden in twee delen, enerzijds het probleem omtrent de *modellering van transferlijnen* en anderzijds de *combinatorische aard van het buffer allocatie probleem*.

De verregaande automatisering van de productieprocessen heeft gezorgd dat productiesystemen steeds belangrijker zijn geworden. Falingen van machines kunnen een groot financieel verlies impliceren voor de onderneming. Dit is zeker het geval bij transferlijnen. Een gehele transferlijn zonder tussenliggende voorraden zal namelijk stilvallen indien één machine in faling gaat, aangezien de machines in dit type productiesysteem serieel zijn opgesteld. **Modellering** van deze productiesystemen is een onderwerp waar reeds veel onderzoek naar werd verricht. Het is echter gebleken dat de analyse van transferlijnen niet eenvoudig is. Enkel voor korte transferlijnen met twee machines konden exacte formules voor de berekening van prestatiemaatstaven ontwikkeld worden. Om de efficiëntie van langere transferlijnen ongeveer te kennen, zijn weliswaar benaderende modellen ter beschikking.

Lutz e.a. (1998) illustreren de **combinatorische** aard van het buffer allocatie probleem en identificeren het aantal mogelijke buffer allocatie combinaties. Veronderstellende dat enkel geheeltallige waarden van een hoeveelheid voorraad toegewezen kunnen worden aan een buffer en dat het totale aantal van voorraadplaatsen beperkt is, kunnen voorraadplaatsen op verschillende manieren toegewezen worden aan de verscheidene buffers. Eveneens veronderstellende dat het totale aantal van voorraadplaatsen aan één buffer kan worden toegewezen, worden formules gegeven om het *aantal van verschillende manieren* te berekenen om u plaatsen aan n buffers toe te wijzen en om het *totale aantal van*

bufferprofielen te berekenen met u totale voorraadplaatsen of minder en n buffers. Op deze formules wordt niet dieper ingegaan.

Indien een transferlijn drie buffers heeft en het totale aantal van voorraadplaatsen eveneens gelijk is aan drie, dan zijn in dit geval 20 bufferprofielen mogelijk. Al deze profielen dienen nagegaan te worden om het optimale bufferprofiel te bepalen. Indien het totale aantal van voorraadplaatsen verhoogd wordt tot 25, dan moeten 3.276 profielen nagegaan worden om de optimale bufferprofiel te kunnen bepalen. Indien een transferlijn vijf buffers heeft en het totale aantal voorraadplaatsen gelijk is aan 25, dan zouden meer dan 142.000 profielen nagegaan moeten worden om de optimale bufferprofiel te kunnen bepalen.

Door de combinatorische aard wordt het buffer allocatie probleem zeer complex indien het aantal buffers en het totale aantal van voorraadplaatsen stijgen. Vanuit een computationeel standpunt is het niet praktisch om de complete verzameling van alle bufferprofielen te doorzoeken om de optimale bufferprofiel te identificeren.

1.2 Centrale Onderzoeksvraag

In deze eindverhandeling wordt het probleem omtrent buffer allocatie onderzocht. Dit probleem wordt onderzocht voor transferlijnen met onbetrouwbare machines. Deze onbetrouwbare machines zullen de optimale buffer allocatie beïnvloeden. Hiermee wordt de buffer allocatie bedoeld die de hoogste winst genereert. Als uitgangspunt voor deze eindverhandeling wordt daarom de volgende **centrale onderzoeksvraag** gesteld:

Op welke wijze dienen buffers toegewezen te worden om de winst te maximaliseren in transferlijnen met onbetrouwbare machines?

De winst wordt uitgedrukt in functie van enerzijds de doorzet en anderzijds de voorraadkosten. Op deze manier zal een hogere doorzet afgewogen worden tegen de extra voorraadkosten die hiervoor nodig zijn.

1.3 Deelvragen

Ter ondersteuning van deze centrale onderzoeksvraag worden een aantal deelvragen geformuleerd. Op deze manier kan het praktijkprobleem op een structurele wijze onderzocht worden. Deze **deelvragen** maken eveneens een overzichtelijke weergave mogelijk.

Een eerste deelvraag heeft betrekking op de simulatie-optimalisatie methoden. *Welke simulatie-optimalisatie technieken zijn ter beschikking en welke voorwaarden zijn hieraan verbonden?* Een overzicht van de simulatie-optimalisatie technieken wordt gegeven in het tweede hoofdstuk.

Een tweede deelvraag betreft de eigenschappen van transferlijnen. *Welke belangrijke kenmerken kunnen een rol spelen bij de modellering van transferlijnen?* Deze kenmerken worden gedefinieerd in het derde hoofdstuk, meer bepaald in de gedeelten 3.1.1.1 tot en met 3.1.1.3.

Een volgende deelvraag is gerelateerd aan de falingen van machines in transferlijnen. *Aan welke falingen kunnen machines in transferlijnen onderhevig zijn?* Deze falingen worden verklaard in het gedeelte 3.1.1.4 van het derde hoofdstuk.

Een vierde deelvraag houdt verband met de machinetoestanden. *In welke toestanden kunnen machines zich bevinden in transferlijnen?* Een onderverdeling van deze machinetoestanden wordt gegeven in het gedeelte 3.1.1.5 van het derde hoofdstuk en eveneens uitgelegd.

Een vijfde deelvraag heeft te maken met de prestatie van transferlijnen. *Op welke manier kunnen de prestaties van transferlijnen bepaald worden?* In subsectie 3.1.2 van het derde hoofdstuk worden enkele prestatiemaatstaven voor transferlijnen aangehaald.

Een voorlaatste deelvraag heeft betrekking op de modellering van transferlijnen. *Hoe kan de afhankelijkheid tussen de machines van transferlijnen doorbroken worden?* Een antwoord op deze vraag wordt verstrekt in subsectie 3.1.3 van het derde hoofdstuk.

Een laatste deelvraag is gerelateerd aan de bestaande literatuur omtrent buffer allocatie in transferlijnen. *Wat zijn de belangrijkste richtlijnen omtrent de optimalisatie van buffer allocatie?* Sectie 3.2 van het derde hoofdstuk is gewijd aan een overzicht van deze richtlijnen.

1.4 Definitie Transferlijnen

Transferlijnen worden gedefinieerd als productiesystemen die bestaan uit een aantal machines die in *serie* opgesteld zijn. In transferlijnen worden de werkstukken *automatisch* getransfereerd van machine tot machine. De goederenstroom komt binnen in het systeem via de eerste machine. De goederen worden bewerkt op deze machine en via de eerste buffer doorgegeven aan de volgende machine. De transfers van producten van de ene machine naar de volgende zijn meestal *gesynchroniseerd* om gelijktijdig plaats te vinden. Dit proces herhaalt zich totdat de goederen op de laatste machine verwerkt zijn. Deze goederen verlaten het systeem dan als output.

1.5 Onderzoeksmethode

In de eerste fase zal een **literatuurstudie** uitgevoerd worden om vertrouwd te raken met enerzijds de *simulatie-optimalisatie technieken* en anderzijds het onderzoek dat verricht werd omtrent *buffer allocatie in transferlijnen*. Na de bespreking hiervan volgt een **praktijkstudie**. Ten eerste zal een *simulatiemodel* gebouwd worden door gebruik te maken van de simulatiesoftware *Arena*. Vervolgens zal aan de hand van simulatie getracht worden om een *optimale buffer allocatie* te vinden voor dit simulatiemodel. Ten slotte wordt met behulp van dit simulatiemodel *onderzoek* verricht naar buffer allocatie in transferlijnen.

1.5.1 Literatuurstudie

De literatuurstudie die werd uitgevoerd, is zowel gebaseerd op *primaire* bronnen als op *secundaire* bronnen. Aangezien Prof. Dr. Janssens actief onderzoek uitvoert naar transferlijnen, was het mogelijk

om bij hem de belangrijkste bronnen hieromtrent te bekomen. Voor het gedeelte over de optimalisatie technieken voor simulaties diende een extra zoekinspanning gedaan te worden. De gebruikte informatie werd vooral uit *vaktijdschriften* en *boeken* gehaald.

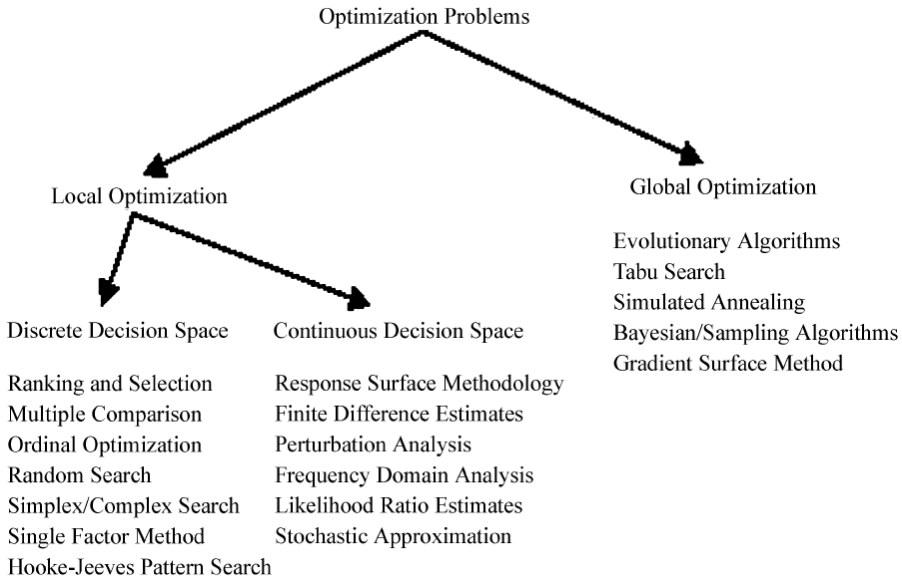
1.5.2 Zoekstrategie

Bij het zoeken naar relevante literatuur werd gebruik gemaakt van enkele belangrijke bibliografische databanken en zoekmachines die door de Universiteit Hasselt ter beschikking gesteld worden, zoals onder andere *EBSCOhost* en *ScienceDirect*. Verder werd ook de catalogi van bibliotheken, zoals *Anet*, geraadpleegd.

Aan de hand van trefwoorden zoals *simulatie-optimalisatie*, *simulation optimization*, *optimalisatie*, *optimization*, *transferlijnen*, *transfer lines*, *onbetrouwbare machines*, *unreliable machines*, *voorraad*, *buffer*, *inventory* en dergelijke kon de relevante literatuur geïdentificeerd worden. Verder werden verwijzingen in gevonden publicaties geraadpleegd om bijkomende literatuur te ontdekken.

HOOFDSTUK II: SIMULATIE-OPTIMALISATIE TECHNIEKEN

In dit hoofdstuk zal een overzicht gegeven worden van verscheidene simulatie-optimalisatie technieken. Tekin en Sabuncuoglu (2004) splitsen deze technieken op in twee soorten. Deze twee soorten zijn technieken om een locaal optimum te identificeren en technieken om een globaal optimum te identificeren. Locale optimalisatie technieken worden verder ingedeeld in twee groepen naargelang de beslissingsruimte discreet of continu is. Figuur 2.1 geeft deze indeling weer.



Figuur 2.1: Overzicht van simulatie-optimalisatie technieken

Deze indeling van Tekin en Sabuncuoglu (2004) zal gebruikt worden om een overzicht te geven van de verscheidene simulatie-optimalisatie technieken. Voor de praktijkstudie dient opgemerkt te worden dat gebruik gemaakt zal worden van een locale optimalisatie techniek met een discrete beslissingsruimte, namelijk de *Single Factor Method*. Dit werd besloten in overleg met de promotor. De overige methoden om te optimaliseren zullen desalniettemin eveneens vermeld en uitgelegd worden om een volledig beeld te schetsen.

2.1 Locale Optimalisatie

In deze sectie zullen de technieken besproken worden die gebruikt worden voor locale optimalisatie. Deze methoden worden in twee groepen ingedeeld, namelijk methoden die gehanteerd worden voor enerzijds een discrete beslissingsruimte en anderzijds een continue beslissingsruimte. In een discrete ruimte nemen de beslissingsvariabelen een discrete set van waarden aan, terwijl in een continue ruimte de beslissingsvariabelen een reële set van waarden aannemen.

2.1.1 Discrete Beslissingsruimte

De discrete beslissingsruimte kan verder ingedeeld worden in enerzijds een eindige beslissingsruimte en anderzijds een oneindige beslissingsruimte. De volgende technieken voor een discrete beslissingsruimte zullen besproken worden. Deze methoden zijn *Ranking And Selection*, *Multiple Comparison*, *Ordinal Optimization*, *Random Search*, *Nelder-Mead Simplex/Complex Search*, *Single Factor Method* en *Hooke-Jeeves Pattern Search*. De eerste twee technieken functioneren in een eindige beslissingsruimte, terwijl de overige technieken in een oneindige beslissingsruimte kunnen functioneren.

2.1.1.1 Ranking And Selection

Het probleem in simulatiestudies om het beste ‘systeem’ te vinden is waarschijnlijk het gebruikelijkste. Het doel is om het systeem te kiezen met de beste verwachte prestatiemaatstaf uit een eindig aantal gesimuleerde systemen. Om dergelijke problemen op te lossen zijn, volgens Tekin en Sabuncuoglu (2004), twee methodieken mogelijk met betrekking tot **Ranking And Selection**. Deze methoden zijn de *Indifference Zone* methode en de *Subset Selection* methode.

De **Indifference Zone** methode tracht om een systeem te kiezen waarvan de prestatiemaatstaf tenminste een bepaalde constante beter is dan de andere alternatieve systemen met een bepaalde

minimale *Probability Of Correct Selection*. Deze *Probability Of Correct Selection* krijgt gewoonlijk de notatie P^*, terwijl de bepaalde constante de notatie δ krijgt. Deze laatste wordt ook wel de *indifferentiezone parameter* genoemd. Deze dient beschouwd te worden als het minimale praktische verschil dat waard is om gedetecteerd te worden. Beide parameters worden door de gebruiker op voorhand bepaald. Indien systemen bestaan met een gemiddelde prestatiemaatstaf op ten hoogste een afstand δ verwijderd van de prestatiemaatstaf van het beste systeem, dan wordt verondersteld dat de gebruiker indifferent is aangaande de keuze van één van deze systemen. (Batur en Kim, 2006)

Sullivan en Wilson (1989) vermelden dat de *Subset Selection* methode verschilt met de *Indifference Zone* techniek. De **Subset Selection** methode resulteert namelijk in een deelverzameling van de volledige set van alternatieven. Deze subset heeft een willekeurige grootte m en bevat, met een kans van tenminste P^*, de populatie met een gemiddelde prestatiemaatstaf op ten hoogste een afstand δ verwijderd van de prestatiemaatstaf van het beste systeem. P^* is eveneens hier de *Probability Of Correct Selection* en m is uiteraard kleiner dan het maximale aantal mogelijke alternatieven. Het voordeel van deze techniek, in vergelijking met de vorige, is dat het de gebruiker toelaat om een grote set van alternatieven te screenen zodat de geselecteerde subset diepgaander onderzocht kan worden in een follow-up studie.

2.1.1.2 Multiple Comparison

Het idee van **Multiple Comparison** is om een aantal replicaties van het simulatiemodel uit te voeren per set van controle variabelen. Vervolgens kunnen conclusies getrokken worden van de prestatiemaatstaf door betrouwbaarheidsintervallen op te stellen of door hypothese testen te doen. Hochberg en Tamhane (1987) halen twee soorten procedures aan om *Multiple Comparison* toe te passen. Deze twee soorten zijn de *Protected Least Significant Difference* test en de *Bonferroni* procedure. Deze hebben geleid tot de ontwikkeling van twee uitgesproken klassen van *Multiple Comparison*, namelijk de *Single-Step* procedure en de *Multi-Step* procedure.

De **Protected Least Significant Difference** test sluit aan bij de *Multi-step* procedure en bestaat uit het uitvoeren van meerdere t-testen met significantieniveau α. De voorwaarde hiervoor is wel dat de voorafgaande F-test significant is op het significantieniveau α. Deze methodiek kan gezien worden als een procedure bestaande uit twee fasen. In de eerste stap wordt de hypothese H_0 getest door een α-niveau F-test. H_0 is de veronderstelling dat elk gemiddelde van de prestatiemaatstaf gelijk is aan elkaar. Dit gemiddelde wordt berekend voor elke set van beslissingsvariabelen op basis van het aantal replicaties. Indien de F-test niet significant is, dan wordt de procedure beëindigd zonder gedetailleerde inferenties te maken van gepaarde verschillen. Indien de test wel significant is, wordt elk gepaarde verschil getest door een α-niveau t-test.

De *Protected Least Significant Difference* test beheerst de *Familywise Error Rate* slechts in zwakke zin, daar deze enkel gelijk is aan α onder de hypothese H_0. Onder andere configuraties kan deze groter zijn dan α. De **Familywise Error Rate** is de kans om eender welke fout te maken in de gegeven familie van inferenties. Deze *Familywise Error Rate* is een alternatieve *Type I Error Rate*. De $\binom{k}{2}$ testen van gepaarde nulhypotheses worden als een familie beschouwd waarbij k het aantal mogelijke sets van beslissingsvariabelen is.

De **Bonferroni** procedure sluit aan bij de *Single-step* procedure en beheerst de *Familywise Error Rate* in sterke zin, oftewel onder alle mogelijke configuraties. Deze procedure bevat één enkele stap en bestaat uit het uitvoeren van meerdere t-testen met significantieniveau $\alpha^* = \alpha / \binom{k}{2}$ zonder een voorafgaande F-test. De **Per-Family Error Rate** van deze procedure is onder de hypothese H_0 gelijk aan α. Deze fout wordt gedefinieerd als het verwachte aantal van fouten in de familie. Onder andere configuraties is de *Per-Family Error Rate* kleiner dan α, daar het aantal ware hypotheses kleiner is dan $\binom{k}{2}$. Aangezien de *Familywise Error Rate* kleiner of gelijk is aan de *Per-Family Error Rate*, wordt de *Familywise Error Rate* sterk beheerst op het significantieniveau α.

De *Bonferroni* procedure kan eveneens omgezet worden om betrouwbaarheidsintervallen te verkrijgen voor alle gepaarde verschillen tussen de groepsgemiddelden. Voor elk paar van gemiddelde prestatiemaatstaven kan met andere woorden een $(1-\alpha)\cdot 100\%$ simultaan betrouwbaarheidsinterval

geconstrueerd worden. De set van beslissingsvariabelen die correspondeert met de betrouwbaarheidsinterval waarvan de verschillen met alle andere paren strikt negatief zijn, zal als minimum gekozen worden. In het geval van een maximalisatie probleem dienen de verschillen met alle andere paren strikt positief te zijn.

2.1.1.3 Ordinal Optimization

Ordinal Optimization is een algoritmische methode om goede ontwerpen te identificeren over een grote ontwerpruimte, verklaren Sullivan en Jacobson (2000). Deze identificatie van goede ontwerpen uit een grote set van mogelijke ontwerpen gebeurt op basis van ordinale rangordes. Met een ontwerp wordt hier een set van waarden bedoeld voor de beslissingsvariabelen.

Deze techniek relaxeert de vereiste om het optimale ontwerp te verkrijgen. Deze relaxatie wordt ook wel **doel tempering** genoemd. Hierdoor kan dit algoritme heel slechte ontwerpen efficiënt 'uitroeien' en een set van designs onderscheiden waartussen een goed ontwerp bestaat. Deze set van ontwerpen wordt de *geselecteerde* subset genoemd en het goede ontwerp wordt de subset genoemd dat *goed genoeg* is. De kwaliteit van de geselecteerde subset met betrekking tot de subset dat goed genoeg is, wordt verkregen door gebruik te maken van een **alignment probability**. Deze laatste meet de kans dat deze twee subsets een bepaalde integere waardegetal van overlappende ontwerpen inhouden. De aard van de relaxatie van de optimalisatievereiste op deze manier heeft geleid tot de toepasselijke term **soft optimization** om de uitvoering van *Ordinal Optimization* te beschrijven.

2.1.1.4 Random Search

Smith (1973) beschrijft in het kort hoe de methodiek van **Random Search** te werk gaat. Eerst wordt een algemene zoekruimte gedefinieerd. Dit gebeurt aan de hand van het ingeven van boven- en ondergrenzen van de controle variabelen. Deze zoekruimte stelt het gebied voor tot waarbinnen het zoeken naar een optimum beperkt is. Vervolgens selecteert deze techniek willekeurige punten in de zoekruimte. Elk van deze punten bepaalt de waarden van de controle variabelen die gebruikt worden

om een geobserveerde respons te verkrijgen. De procedure stopt wanneer een bepaald aantal computer runs werden gedaan, daar de zoekruimte vele mogelijke combinaties van punten bevat. Het punt met de beste respons wordt geselecteerd als optimum van het probleem.

Veel onderzoek werd niet gedaan naar *Random Search* om simulatie-optimalisatie problemen op te lossen. Het grootste nadeel van deze techniek is de trage convergentie naar een optimum, omdat voorgaande informatie verloren gaat bij elke iteratie. Desalniettemin kan *Random Search* gebruikt worden in een oneindige beslissingsruimte.

2.1.1.5 Nelder-Mead Simplex/Complex Search

De **Nelder-Mead Simplex/Complex Search** incorporeert verrichtingen om de simplex te wijzigen. De simplex is een verzameling van punten en de wijzingen gebeuren op basis van het locale gedrag van de doelfunctie. De *Complex Search* is de beperkte versie van de *Simplex Search*, halen Tekin en Sabuncuoglu (2004) aan. Deze zoekprocedure is gelijkaardig aan de *Simplex Search*, behalve dat een extra inspanning wordt gemaakt om te voorkomen dat de simplex het toegelaten gebied verlaat.

Barton en Ivey (1996) verklaren dat de **Nelder-Mead Simplex Search** een n-dimensionale simplex creëert door $n+1$ extreme punten te kiezen voor een responsfunctie met n controle variabelen. Vervolgens wordt de responsfunctie geëvalueerd in elk extreem punt van de simplex. In elke iteratie wordt een nieuw punt toegevoegd aan de simplex door dat punt te projecteren op het middelpunt van de resterende punten en het slechtste punt te schrappen van de set. Deze stap wordt ook wel *reflectie* genoemd. Bovendien worden simplex reflecties *uitgebreid* indien de gereflecteerde waarde uitermate goed is. Een slechte gereflecteerde waarde resulteert daarentegen in een *samentrekking* van de simplex. Daarbovenop wordt de grootte van de simplex *gereduceerd* indien de waarde van de responsfunctie in het samengetrokken punt nog slechter is. Deze procedure wordt voortgezet totdat de grootte van de simplex voldoende klein is. Dit wil zeggen dat alle punten in de simplex zich vrijwel in dezelfde plaats bevinden.

Zoals eerder werd vermeld, verloopt de **Complex Search** op een gelijkaardige manier als de *Simplex Search*. De Complex Search verschilt echter van de Simplex Search op een aantal aspecten. De *Complex Search* kan namelijk werken met beperkingen, vermelden Castro e.a. (2003). Eveneens wordt in deze techniek een n-dimensionale simplex gecreëerd, maar door $k = 2 \cdot n$ punten te kiezen. De *reflectie* verloopt analoog als die van de vorige methode, behalve dat een extra inspanning wordt gemaakt zodat het punt het toegelaten gebied niet verlaat. Indien het gereflecteerde punt het slechtste punt is of zich in het gebied bevindt dat niet toegelaten is, dan zal het stapsgewijs verplaatst worden in de richting van het middelpunt van de resterende punten. Dit proces wordt voortgezet totdat een aanvaardbaar punt gevonden wordt. Deze techniek wordt beëindigd wanneer dezelfde waarden voor de controle variabelen bekomen worden in consecutieve iteraties.

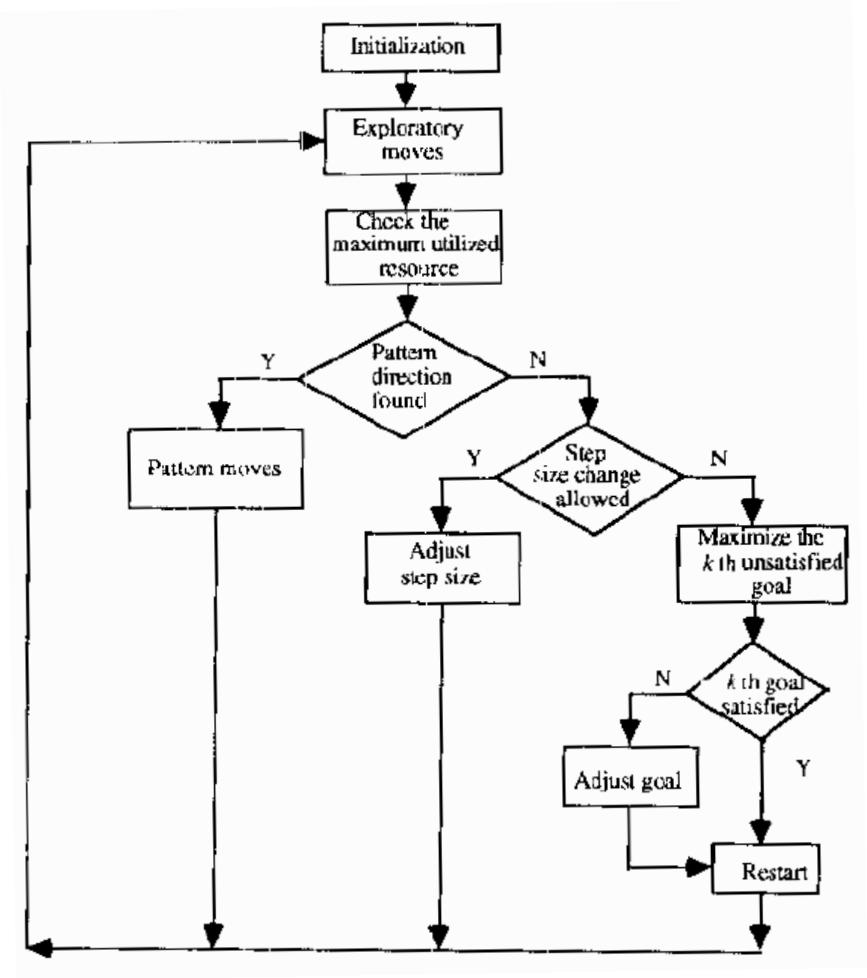
2.1.1.6 Single Factor Method

De **Single Factor Method** is een directe zoekmethode die in een oneindige beslissingsruimte kan werken. Smith (1973) verklaart dat deze methode één beslissingsvariabele verandert terwijl alle andere controle variabelen ongewijzigd blijven. Deze verandering gebeurt op een gecoördineerde manier. De *Single Factor Method* bepaalt namelijk of een verandering van een controle variabele, in positieve of negatieve zin, een verbetering levert in de geobserveerde respons. Indien dit het geval is, blijft deze variabele in dezelfde zin veranderd worden tot geen verdere verbetering vastgesteld wordt.

Nadat de zoektocht in de richting van de eerste beslissingsvariabele beëindigd is door gebrek aan verbetering, wordt een gelijkaardige zoektocht verricht in de richting van de tweede beslissingsvariabele. Deze zoektocht begint aan het vorige punt dat de grootste respons leverde. Deze procedure wordt gelijkaardig voortgezet in de richting van alle andere controle variabelen totdat elke controle variabele gewijzigd werd. Vervolgens wordt deze cyclus herhaald door te herbeginnen bij de eerste beslissingsvariabele. Dit proces wordt beëindigd wanneer het gespecificeerde aantal van beschikbare computer runs werd opgebruikt. Wanneer dit gebeurt, wordt het punt geïdentificeerd als het optimum. Dit punt is het punt dat de grootste respons leverde.

2.1.1.7 Hooke-Jeeves Pattern Search

De **Hooke-Jeeves Pattern Search** methode is een directe zoekprocedure die efficiënt niet-lineaire doel programmeringsproblemen kan oplossen, lichten Chen en Tsai (1996) toe. Een directe zoekprocedure streeft op directe wijze naar een optimum door gebruik te maken van simulatieresponses. Deze zoekmethode wordt uitgelegd aan de hand van een minimalisatie probleem en maakt gebruik van twee verschillende series van zetten, namelijk *verkennende* en *patroon* zetten. Verkennende zetten onderzoeken het locale gedrag van een functie en streven ernaar om de richting van elk afhellend dal te situeren dat aanwezig zou kunnen zijn. De patroon zetten maken gebruik van de informatie die in de verkenning gegenereerd werd om vlot doorheen de dalen te stappen. Figuur 2.2 toont een overzicht van dit algoritme.



Figuur 2.2: Schema van Hooke-Jeeves Pattern Search

De **verkenning** vertrekkende van een initieel punt loopt verder in elke richting van elke beslissingsvariabele door een gespecificeerde stapgrootte. Indien de doelfunctie waarde verbetert, wordt deze stap als een succes beschouwd. Indien de doelfunctie waarde niet verbetert, wordt deze stap niet uitgevoerd en vervangen door een stap in tegengestelde richting. Indien de doelfunctie waarde nog steeds niet verbetert, wordt de beslissingsvariabele in kwestie niet gewijzigd. Wanneer alle controle variabelen onderzocht zijn, wordt de verkennende zet afgerond. Het resulterende punt van deze zet wordt een *basispunt* genoemd. Vervolgens wordt een **patroon** zet bepaald via onderstaande formule:

$$X_{k+1} = X_k + (X_k - X_{k-1})$$

X_k vertegenwoordigt het huidige basispunt, X_{k-1} representeert het vorige basispunt en X_{k+1} is het volgende basispunt. Een patroon zet bestaat uit één enkele stap vanuit het huidige basispunt naar een nieuw basispunt. Deze stap wordt bepaald volgens een lijnsegment dat geconstrueerd wordt tussen het vorige basispunt en het huidige basispunt. Het nieuwe patroonpunt wordt dus met andere woorden bepaald door een extrapolatie te maken van dat lijnsegment. De procedure eindigt wanneer de doelfunctie waarde onverbeterd blijft voor elke verandering van elke beslissingsvariabele.

2.1.2 Continue Beslissingsruimte

Om een volledige bespreking van de simulatie-optimalisatie technieken te geven, zullen eveneens de technieken voor een continue beslissingsruimte besproken worden. Deze methoden zijn *Response Surface Methodology*, *Finite Difference Estimates*, *Perturbation Analysis*, *Frequency Domain Analysis*, *Likelihood Ratio Estimates* en *Stochastic Approximation*.

2.1.2.1 Response Surface Methodology

Response Surface Methodology is een iteratieve zoekmethode met als doel optimale waarden te vinden voor de beslissingsvariabelen. Deze methode past een reeks regressiemodellen aan de

responses van een simulatiemodel die geëvalueerd wordt in verschillende punten. De resulterende regressiefunctie wordt vervolgens geoptimaliseerd. Het is niet zeker dat de gevonden oplossing van deze regressiefunctie optimaal is. Het kan namelijk zijn dat deze oplossing slechts een locaal optimum is en eveneens is het mogelijk dat deze gevonden oplossing een zadelpunt is. Bovendien is deze oplossing slechts een schatting van het werkelijke optimum. Het basisalgoritme bestaat uit twee fasen. (Kleijnen en van Groenendaal, 1988)

In de eerste fase wordt bepaald in welke richting en in welke mate de variabelen veranderd moeten worden om het optimum te bereiken. Deze bepaling gebeurt aan de hand van de steilste verandering. Deze laatste wordt nagegaan door een **eerste-orde** regressiefunctie te schatten in een gebied dat kleiner is dan de beslissingsruimte. De volgende uitdrukking geeft een eerste-orde regressiefunctie weer met twee controle variabelen x_1 en x_2:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 \cdot x_1 + \hat{\beta}_2 \cdot x_2$$

Deze eerste fase wordt herhaald totdat de berekende helling van de regressiefunctie ongeveer nul wordt. Dit betekent dat de eerste-orde regressiefunctie geen goede schatting meer biedt voor de responses van het simulatiemodel. In de buurt van het optimum wordt gebruik gemaakt van een **tweede-orde** regressiefunctie. Dit is de tweede fase van de methode. Deze tweede-orde regressiefunctie wordt geoptimaliseerd door gebruik te maken van analytische wiskunde zoals bijvoorbeeld numerieke methoden. Onderstaande uitdrukking is een tweede-orde regressiefunctie met twee controle variabelen x_1 en x_2:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 \cdot x_1 + \hat{\beta}_2 \cdot x_2 + \hat{\beta}_{12} \cdot x_1 \cdot x_2 + \hat{\beta}_{11} \cdot x_1^2 + \hat{\beta}_{22} \cdot x_2^2$$

Response Surface Methodology biedt een algemene methodologie om problemen te optimaliseren via simulaties. Het grootste voordeel is dat deze methode gebruik maakt van welbekende statistische instrumenten. Vergeleken met vele gradiënt technieken is *Response Surface Methodology* een relatief efficiënte techniek van simulatie-optimalisatie. Met deze efficiëntie wordt het aantal simulatie-experimenten bedoeld.

2.1.2.2 Gradiëntgebaseerde Methoden

Tekin en Sabuncuoglu (2004) onderscheiden vier verschillende technieken om de gradiënten van de responses te schatten. Deze technieken zijn *Finite Difference Estimates*, *Perturbation Analysis*, *Frequency Domain Analysis* en *Likelihood Ratio Estimates*. Voor de bespreking van deze methoden wordt het volgende probleem gebruikt met de bijbehorende notatie. Het onderzoeksprobleem is de minimalisatie van de verwachte waarde van de doelfunctie gegeven een set van beperkingen. Dit wordt als volgt weergegeven:

$$\min_{X \in \Theta} H(X)$$

$H(X) = E[L(X, \varepsilon)]$ is de prestatiemaatstaf van het probleem. $L(X, \varepsilon)$ is de prestatie van de steekproef waarbij ε de stochastische effecten van het systeem en X de vector met p controleerbare factoren voorstellen. Θ stelt de set van beperkingen op X voor.

2.1.2.2.1 Finite Difference Estimates

Volgens Tekin en Sabuncuoglu (2004) wordt **Finite Difference Estimates** eveneens toegepast samen met andere technieken zoals *Hooke-Jeeves Pattern Search* en *Stochastic Approximation*. *Finite Difference Estimates* is gebaseerd op het bepalen van partiële afgeleiden van de doelfunctie $H(X)$. Dit wordt in de volgende expressie getoond:

$$\frac{\partial H(X)}{\partial X_i} = \frac{H(X_1, \ldots, X_i + \Delta X_i, \ldots, X_p) - H(X_1, \ldots, X_i, \ldots, X_p)}{\Delta X_i}$$

Om de gradiënt in elke punt te kunnen schatten, zijn tenminste $p+1$ evaluaties nodig van het simulatiemodel. Eveneens zijn meerdere observaties voor elke partiële afgeleide nodig om een betrouwbaardere schatting te bekomen.

2.1.2.2.2 Perturbation Analysis

Perturbation Analysis is een methode om alle gradiënten van een doelfunctie uit één enkele simulatierun te schatten. Tekin en Sabuncuoglu (2004) geven twee classificaties van *Perturbation Analysis*. Deze classificaties zijn *Finite Perturbation Analysis* en *Infinitesimal Perturbation Analysis*.

Finite Perturbation Analysis is ontwikkeld voor discrete controle variabelen. Deze techniek is een heuristiek die het verschil in een prestatiemaatstaf benaderd wanneer een discrete controle variabele wordt veranderd met één eenheid. **Infinitesimal Perturbation Analysis** wordt gebruikt om afgeleiden van continue controle variabele te verkrijgen. Deze methode schat alle partiële afgeleiden uit één enkele simulatierun door, gedurende een simulatierun, gerelateerde statistieken bij te houden van bepaalde gebeurtenissen. Deze statistieken worden bijgehouden door het volgende te berekenen:

$$\frac{\partial L}{\partial X} = \sum_{i,k} \frac{\partial L}{\partial T_k} \cdot \frac{\partial T_k}{\partial T_i} \cdot \frac{\partial T_i}{\partial X}$$

$\partial T_i / \partial X$ geeft weer hoe de verandering van de waarde van een controle variabele de timing van gebeurtenissen T_i verandert. $\partial T_k / \partial T_i$ drukt uit hoe wijzigingen in de timing van gebeurtenissen T_i de timing van gebeurtenissen T_k veranderen. $\partial L / \partial T_k$ geeft weer hoe de veranderingen in de timing van gebeurtenissen T_k de prestatie van het systeem wijzigen.

Een aantal moeilijkheden treden op om *Perturbation Analysis* toe te passen. Ten eerste moet het model een aantal beperkende voorwaarden vervullen. Vervolgens moet de maker van het model volledige kennis over het simulatiemodel bezitten. Desalniettemin krijgt *Perturbation Analysis* veel aandacht in simulatie-optimalisatie vanwege de efficiëntie van deze techniek.

2.1.2.2.3 Frequency Domain Analysis

De intuïtieve idee achter **Frequency Domain Analysis** is om de waarde van een controle variabele volgens een sinusoïdale functie te laten oscilleren tijdens simulatie, vermelden Tekin en Sabuncuoglu (2004). De grootte van de variatie in de prestatiemaatstaf geeft een indicatie van de relatieve gevoeligheid van de prestatiemaatstaf aan de controle variabele. Hiervoor wordt een **vector** van controle variabelen gemoduleerd. Deze modulatie gebeurt op de volgende wijze:

$$X(t) = X_0 + \alpha \cdot \sin(\omega \cdot t)$$

X_0 is de vector van controle variabelen in kwestie, α is de vector van oscillatie amplitudes en ω is de vector van oscillatie frequenties. In het algemeen is de tijdsvariabele t niet de simulatietijd, maar een variabele van het model die bepaalde statistieken bijhoudt tijdens elke simulatierun. Vervolgens dienen α, ω en t bepaald te worden. Hierna kan een kwadratisch respons metamodel verkregen worden door H te benaderen in X_0. Deze benadering gebeurt via een tweede-orde **Taylor series** expansie. Deze laatste wordt hieronder getoond met ∇_i als de partiële afgeleide naar X_i:

$$\nabla_i \hat{H}(X_0) = \lim_{T \to \infty} \lim_{\omega_i \to 0} \frac{2}{\alpha_i \cdot T} \cdot \sum_{i=1}^{T} \hat{H}[X(t)] \cdot \sin(\omega_i \cdot t)$$

Alhoewel *Frequency Domain Analysis* wordt gezien als een methode met veel potentieel om de efficiëntie van simulatie-optimalisatie te verbeteren, heeft deze techniek echter enkele nadelen. Ten eerste heeft deze methode een zorgvuldige indexering van simulatie observaties nodig die gepaard gaat met sinusoïdale variatie van de inputvariabelen. Deze variatie verloopt volgens een tijdsindex. Bovendien heeft deze techniek één bijkomende beperking. Het onderzoeksgebied moet namelijk klein zijn om grote variaties te vermijden.

2.1.2.2.4 Likelihood Ratio Estimates

Tekin en Sabuncuoglu (2004) verklaren dat **Likelihood Ratio Estimates** de onderliggende waarschijnlijkheidsmaatstaf van het systeem differentiëren. Deze maatstaf is in dit geval de verwachte waarde van de prestatie van de steekproef. Deze methode veronderstelt dat $L(Y)$ de prestatiemaatstaf is met Y als willekeurige vector. Deze vector heeft een *joint* cumulatieve verdelingsfunctie $F(X,\cdot)$ en een dichtheidsfunctie $f(X,\cdot)$. De afhankelijkheid van de prestatiemaatstaf op X uit zich dus enkel via de willekeurige vector Y. Door de onderstaande vergelijking te differentiëren naar X, kan de afgeleide van de prestatiemaatstaf en de prestatiemaatstaf zelf geschat worden. Deze vergelijking is de volgende:

$$E[L(Y)] = \int L(y) dF(X,y)$$

Een pluspunt van deze techniek is dat de afgeleide van de prestatiemaatstaf en de prestatiemaatstaf zelf in één enkele simulatierun geschat kunnen worden. Het nadeel is dat de variatie van de schatter toeneemt naarmate de lengte van de simulatierun toeneemt.

2.1.2.3 Stochastic Approximation

Om **Stochastic Approximation** uit te leggen, zal dezelfde notatie gehanteerd worden als die in de gradiëntgebaseerde methoden. Tekin en Sabuncuoglu (2004) bespreken de basisveronderstelling van deze techniek en de techniek zelf. De basisveronderstelling onderliggend aan deze methode is dat het originele probleem, gegeven door de uitdrukking $\min_{X \in \Theta} H(X)$, opgelost kan worden door $\nabla H(X)$ gelijk te stellen aan nul. Om op deze manier het optimum te bepalen, wordt gebruik gemaakt van een recursieve formule. De volgende vector van controle variabelen wordt met andere woorden bepaald op basis van de vorige vector. Deze formule wordt als volgt weergegeven:

$$X_{n+1} = \prod_{\Theta}\left(X_n - a_n \cdot \hat{\nabla} H_n\right)$$

a_n is een reeks van stapgroottes die gebaseerd zijn op werkelijke waarden zodat enerzijds $\sum a_n < \infty$ en anderzijds $\sum a_n^2 < \infty$. X_n is de geschatte waarde van het optimum aan het begin van iteratie n. $\hat{\nabla} H_n$ is een schatting van de gradiënt $\nabla H(X_n)$ uit iteratie n en $\prod_\Theta$ is een projectie op Θ. Als n naar oneindig gaat, benadert X_n een waarde zodat de theoretische regressiefunctie van het stochastische respons geminimaliseerd wordt.

De moeite met *Stochastic Approximation* is dat een groot aantal iteraties van de recursieve formule nodig zijn om het optimum van het probleem te bepalen. Bovendien is de keuze van de observatielengte voor elke iteratie een belangrijk probleem. De reden hiervoor is om conditionele bias te vermijden die het gevolg is van gekende informatie aan het begin van de iteratie.

2.2 Globale Optimalisatie

In deze sectie worden volledigheidshalve eveneens de technieken besproken die gebruikt worden voor globale optimalisatie. Deze strategieën werden ontwikkeld voor problemen met multimodale responsoppervlakken. Deze methoden zijn *Genetic Algorithms*, *Evolutionary Programming*, *Evolution Strategies*, *Tabu Search*, *Simulated Annealing*, *Bayesian/Sampling Algorithms* en *Gradient Surface Method*.

2.2.1 Evolutionary Algorithms

Evolutionary Algorithms zijn heuristische zoekmethoden die concepten uit de evolutietheorie gebruiken. De idee achter deze algoritmen is dat de slechte oplossingen uitsterven, terwijl de goede oplossingen evolueren om het optimum te bereiken. In tegenstelling tot de traditionele methoden, waarbij één enkele oplossing gebruikt wordt, maken deze algoritmen gebruik van een populatie van oplossingen. *Evolutionary Algorithms* hebben de laatste jaren meer interesse gekregen om te gebruiken in simulatie-optimalisatie. Enerzijds omdat deze algoritmen geen restrictieve assumpties vereisen en

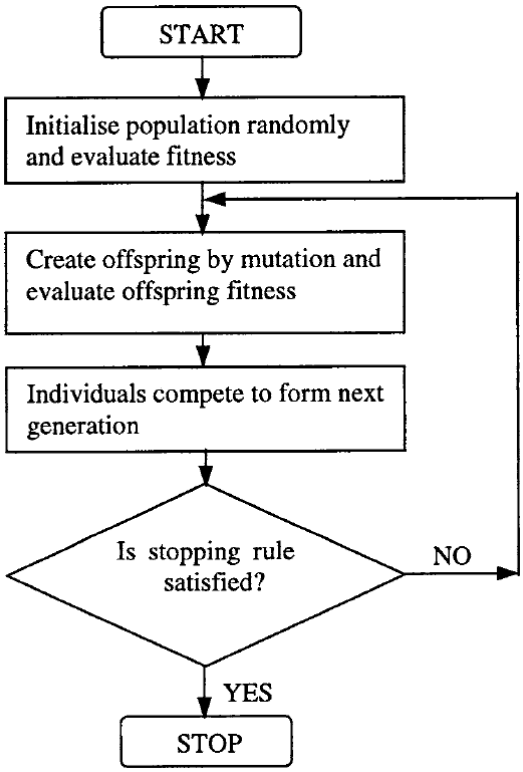
anderzijds omdat op voorhand geen kennis vereist is over de vorm van het responsoppervlak. De meest populaire algoritmen zijn *Genetic Algorithms*, *Evolutionary Programming* en *Evolution Strategies*. (Tekin en Sabuncuoglu, 2004)

2.2.1.1 Genetic Algorithms

Genetic Algorithms zijn zoekalgoritmen die gebaseerd zijn op de mechanismen van natuurlijke selectie en natuurlijke genetica, verklaart Goldberg (1989). Deze algoritmen maken efficiënt gebruik van historische informatie om nieuwe zoekpunten te identificeren met een verwachte verbeterde prestatie. *Genetic Algorithms* bestaan uit drie operatoren, namelijk reproductie, cross-over en mutatie. *Reproductie* is een proces waarbij individuele zoekpunten overgenomen worden rekening houdende met hun doelfunctie waarden. Na de reproductie volgt de *cross-over*. Hier worden de geselecteerde zoekpunten gepaard, vervolgens simultaan opgesplitst en onderling gerecombineerd met elkaar. *Mutatie* is de occasionele willekeurige verandering van één bepaalde waarde van het zoekpunt om bescherming te bieden tegen onherstelbaar verlies aan genetische informatie.

2.2.1.2 Evolutionary Programming

Evolutionary Programming, verklaren Somasundaram e.a. (2004), is een stochastische optimalisatiestrategie die de nadruk legt op de manier waarop ouders verbonden zijn met hun nageslacht. Het is een sterke en algemene optimalisatiemethode die niet afhankelijk is van enerzijds de eerste en tweede afgeleide van de doelfunctie en anderzijds de beperkingen van het probleem. Het belangrijkste voordeel van deze techniek is dat het enkel de informatie van de doelfunctie gebruikt en dus niet afhankelijk is van de aard van de zoekruimte. Het algoritme bestaat uit drie processen. Deze processen zijn natuurlijke *selectie*, *mutatie* en *competitie*. Deze processen worden voortgezet totdat een gekozen maximumaantal van iteraties bereikt is. Het maximumaantal van iteraties dat gekozen dient te worden, is afhankelijk van het probleem. De basisstappen van dit algoritme worden weergegeven in figuur 2.3.



Figuur 2.3: Basisstappen van Evolutionary Programming

2.2.1.3 Evolution Strategies

Schwefel (1981) beschrijft hoe **Evolution Strategies** aan de hand van drie stappen te werk gaan. In de eerste stap, *initialisatie*, wordt een populatie bepaald die bestaat uit één ouder en één afstammeling. Deze worden elk geïdentificeerd aan de hand van hun genotype volgens een set van een aantal genen. In de tweede stap, *mutatie*, produceert de ouder van de generatie een afstammeling met een genotype die licht verschilt met de genotype van de ouder. De afwijkingen die plaatsvinden bij de reproductie verwijzen naar de individuele genen. Deze afwijkingen zijn eveneens willekeurig en onafhankelijk van elkaar. In de derde stap, *selectie*, wordt één individu geselecteerd als de ouder van de volgende generatie om nieuwe afstammelingen te produceren. Deze selectie gebeurt op basis van de overlevingswaarde van de individu's. Beide individu's hebben een verschillende capaciteit om te overleven in dezelfde omgeving door hun verschillende genotypen.

2.2.2 Tabu Search

Glover en Laguna (1999) vermelden dat **Tabu Search** gebaseerd is op concepten uit enerzijds de artificiële intelligentie en anderzijds de optimalisatie. De methode maakt gebruik van procedures die ontwikkeld zijn om de grenzen van toelaatbaarheid of locale optimaliteit te overschrijden. Door het systematisch opleggen en ontheffen van beperkingen kunnen 'verboden' gebieden onderzocht worden. *Tabu Search* is een metaheuristiek die een locale heuristische zoekprocedure begeleidt om de oplossingsruimte diepgaander te onderzoeken dan locale optimaliteit. De hoofdcomponenten van Tabu *Search* zijn het *adaptief* geheugen en het *responsief* verkennen. Deze componenten leiden tot een flexibeler zoekgedrag.

Het **adaptief** geheugen van *Tabu Search* is zowel expliciet als attributief en hangt af van vier elementen. Deze elementen zijn recentelijkheid, frequentie, kwaliteit en invloed. *Recentelijkheid* refereert naar het korte termijngeheugen om attributen van oplossingen bij te houden die veranderd zijn tijdens het recente verleden. *Frequentie* levert informatie op basis van het korte termijngeheugen die samengaat met de informatie van recentelijkheid. *Kwaliteit* refereert naar de mogelijkheid om te kunnen differentiëren tussen de verdiensten van verschillende oplossingen gevonden tijdens het zoeken. *Invloed* beschouwt niet enkel de impact van de gemaakte keuzes op kwaliteit tijdens het zoeken, maar eveneens op structuur. *Expliciet* geheugen houdt complete oplossingen bij en deze zijn meestal elite oplossingen die tijdens het zoeken gevonden worden. *Attributief* geheugen onthoudt informatie over de veranderingen in oplossingsattributen.

Het **responsief** verkennen van *Tabu Search* verondersteld dat een slechte strategische keuze meer informatie kan verstrekken dan een goede willekeurige keuze. In een systeem dat gebruik maakt van geheugen kan een slechte keuze gebaseerd op strategie nuttige aanwijzingen verschaffen over hoe de strategie positief kan veranderd worden. Responsief verkennen integreert de basisprincipes van het intelligent zoeken, namelijk het uitbuiten van de kenmerken van goede oplossingen terwijl nieuwe veelbelovende gebieden onderzocht worden.

2.2.3 Simulated Annealing

Simulated Annealing is gebaseerd op de analogie tussen enerzijds de simulatie van het ontharden van vaste stoffen en anderzijds het probleem om grote combinatorische optimalisatievraagstukken op te lossen, vermelden van Laarhoven en Aarts (1987). Om dit algoritme uit te leggen wordt gebruik gemaakt van een doelfunctie C en een controle variabele c. Eveneens wordt gebruik gemaakt van een partitie functie $Q(c)$. Deze is een normalisatieconstante die afhankelijk is van de controle variabele c. Dit algoritme wordt duidelijk gemaakt aan de hand van een minimalisatie probleem. Een maximalisatie probleem verloopt analoog.

Initieel wordt aan de controle variabele een hoge waarde gegeven en een sequentie van configuraties van de combinatorische optimalisatieproblemen gegenereerd. Hiervoor wordt een generatiemechanisme gedefinieerd zodat, gegeven een configuratie i, een configuratie j verkregen kan worden door willekeurig een element uit de omstreek van i te kiezen. Indien $\Delta C_{ij} \leq 0$ wordt de kans dat configuratie j de volgende configuratie wordt in de sequentie, gegeven door één. Als $\Delta C_{ij} > 0$ wordt de kans gegeven door $\exp(-\Delta C_{ij}/c)$. Deze kans is niet gelijk aan nul en betekent dus dat het mogelijk is om verder te doen met een configuratie met een hogere doelfunctie waarde. Dit proces wordt herhaald tot een evenwicht bereikt wordt. In dat geval benadert de kansverdeling van de configuraties de **Boltzmann** verdeling die gegeven wordt door:

$$\Pr(C = C_i) = \frac{1}{Q(c)} \cdot \exp\left(-\frac{C_i}{c}\right)$$

Vervolgens wordt de controle variabele verlaagd in stappen. Deze stappen worden op voorhand bepaald in een ‘afkoelingstabel’. Voor elke stap wordt een sequentie van configuraties gegenereerd zoals hierboven beschreven. Hierdoor zal in elke stap het systeem een evenwicht bereiken. Het algoritme wordt beëindigd voor een bepaalde kleine waarde van c waarvoor geen ‘afkoeling’ meer toegelaten wordt. De finale ‘bevroren’ configuratie wordt dan genomen als de optimale oplossing van het probleem.

Een minpunt om *Simulated Annealing* algoritmen te implementeren is dat alle parameters a priori bepaald dienen te worden. Het is met andere woorden noodzakelijk om op voorhand de begintemperatuur vast te stellen, evenals de eindtemperatuur en het aantal iteraties bij elke temperatuur.

2.2.4 Bayesian/Sampling Algorithms

Stuckman and Easom (1992) vergelijken verschillende **Bayesian/Sampling Algorithms** met andere globale zoekmethoden teneinde de prestaties van deze methoden vast te stellen. Zij lichten achtereenvolgens kort *Stuckman's Method* toe, *Mockus's Method*, *Perttunen's Method*, *Zilinskas's Method* en *Shaltenis's Method*. Eveneens lichten zij de andere globale zoekmethoden kort toe. Deze zijn *Torn's Method* (clusteringstechniek), *Monte Carlo Method* en *Simulated Annealing Method*. Ter illustratie van dit algoritme zal *Stuckman's Method* besproken worden voor het geval van maximalisatie. Het gebruik van deze techniek om te minimaliseren gebeurt op analoge wijze.

Stuckman's Method is een n-dimensionale globale optimalisatiemethode die gebaseerd is op *Kushner's* één-dimensionale zoekmethode. Het algoritme modelleert een onbekende doelfunctie, $f(X)$ met $X = \{x_1, x_2, \ldots, x_n\}$, als een *Brownian Motion Process*. Het berekent met andere woorden het punt van de volgende gissing, $f(X')$. Dit punt wordt zodanig berekend dat het de maximale kans heeft om het vorige punt te overtreffen met een bepaalde positieve constante ε_n. Dit wordt weergegeven in de volgende expressie:

$$\Pr\left[f(X') > f(X) + \varepsilon_n\right]$$

Deze methode zoekt naar het punt van maximale kans op een lijnsegment dat geconstrueerd is tussen twee punten waar de doelfunctie werd geëvalueerd. Nieuwe lijnsegmenten worden gevormd en het proces wordt voortgezet door al deze lijnsegmenten te onderzoeken. De methode begint door gissingen te maken in gebieden waar de gemiddelde prestatie, gemeten door simulatie, hoog is. In het begin is de

waarde van de constante hoog, maar vermindert zodat de optimalisatiezoektocht meer locaal wordt van aard. Door de waarde van deze constante dynamisch te bepalen, wordt de convergentie versneld.

2.2.5 Gradient Surface Method

De **Gradient Surface Method** verschilt van andere globale zoektechnieken. De reden hiervoor is dat deze methode gebruik maakt van traditionele zoektechnieken om een responsoppervlak globaal te onderzoeken. Deze methode combineert de voordelen van enerzijds *Response Surface Methodology* en anderzijds efficiënte technieken om afgeleiden te schatten zoals *Perturbation Analysis* en *Likelihood Ratio Estimates*. Dit alles gebeurt in conjunctie met algoritmen van *Stochastic Approximation*. (Tekin en Sabuncuoglu, 2004)

In de *Gradient Surface Method* wordt de schatting van de gradiënt verkregen door *Perturbation Analysis* of *Likelihood Ratio Estimates*. Daarna wordt de prestatie van het gradiëntoppervlak gepast aan deze schattingen van de gradiënten door gebruik te maken van *Least Squares Methods* zoals in *Response Surface Methodology*. Vervolgens worden nulpunten van dit gepaste gradiëntoppervlak genomen als de schattingen van de optimale oplossing.

Gradient Surface Method is een globaal zoekalgoritme. De reden hiervoor is dat deze methode bij elke iteratie gebruik maakt van de informatie van niet enkel de locale gradiënt, maar van alle gegevenspunten. Deze techniek heeft twee positieve eigenschappen. De schattingen van de gradiënt worden namelijk verkregen door één enkele simulatierun. Dit algoritme komt eveneens zeer snel in de buurt van de optimale oplossing door de globale oriëntatie van deze methode.

HOOFDSTUK III: BUFFER ALLOCATIE IN TRANSFERLIJNEN

In dit hoofdstuk zal het probleem omtrent het toewijzen van voorraden in transferlijnen besproken worden. Dit hoofdstuk is opgebouwd in twee secties. In de eerste sectie zal dieper ingegaan worden op transferlijnen. In de tweede sectie zullen enkele conventionele resultaten met betrekking tot buffer allocatie in transferlijnen toegelicht worden.

3.1 Transferlijnen

In deze sectie zullen transferlijnen nader toegelicht worden. Ten eerste zullen enkele termen verklaard worden betreffende transferlijnen. Vervolgens zullen een aantal mogelijke prestatiemaatstaven aan bod komen om de prestaties van transferlijnen te bepalen. Ten slotte zal uitleg gegeven worden over het modelleren van transferlijnen.

3.1.1 Definities

In deze subsectie zullen enkele uitdrukkingen uitgelegd worden. Ten eerste zullen een aantal verschillen aan bod komen zoals het verschil tussen discrete en continue, homogene en heterogene, betrouwbare en onbetrouwbare transferlijnen. Daarna zal het verschil tussen operatie-afhankelijke en tijdsafhankelijke falingen verklaard worden. Ten slotte zullen de voornaamste machinetoestanden besproken worden.

3.1.1.1 Discrete Versus Continue Transferlijnen

Wild (1972) licht het verschil tussen discrete en continue transferlijnen nader toe. Bij **discrete** transferlijnen worden complexe goederen geproduceerd. De goederenstroom bestaat in dit geval uit afzonderlijke eenheden. De eenheden zijn met andere woorden duidelijk van elkaar te onderscheiden.

Bij **continue** transferlijnen bestaat de goederenstroom daarentegen niet uit afzonderlijke eenheden. De individuele eenheden kunnen met andere woorden niet van elkaar onderscheiden worden. In dit geval gaat het bijvoorbeeld om de verwerking van grote hoeveelheden bulkgoederen, vloeistoffen of semi-vloeibare producten.

3.1.1.2 Homogene Versus Heterogene Transferlijnen

Dit verschil wordt soms in de literatuur aangehaald als het verschil tussen gebalanceerde en ongebalanceerde transferlijnen. Conway e.a. (1988) definiëren een **gebalanceerde** transferlijn als een transferlijn waarbij enkel de gemiddelde bewerkingstijd dezelfde is voor ieder werkstation. De verscheidene standaarddeviaties van de bewerkingstijden moeten met andere woorden niet dezelfde zijn voor alle werkstations. Bij **homogene** transferlijnen zijn dus de gemiddelde bewerkingstijden identiek voor alle machines.

Lutz e.a. (1998) omschrijven een **ongebalanceerde** transferlijn als een transferlijn waarbij een bottleneck aanwezig is met een gemiddelde bewerkingstijd die groter is dan deze van de andere werkstations. Het outputniveau wordt bij deze transferlijn gemaximaliseerd door de ijdele tijd van de bottleneck, veroorzaakt door uitputting of blokkade, te verminderen of te vermijden. Bij **heterogene** transferlijnen zijn de gemiddelde bewerkingstijden niet identiek voor alle machines.

Opgemerkt dient te worden dat het verschil tussen homogene en heterogene transferlijnen enkel van toepassing is op discrete transferlijnen. Bij continue transferlijnen kunnen namelijk geen afzonderlijke goederen onderscheiden worden en kunnen dus onmogelijk bewerkingstijden bepaald worden.

3.1.1.3 Betrouwbare Versus Onbetrouwbare Transferlijnen

Bij **betrouwbare** transferlijnen zijn de werkstations niet onderhevig aan falingen van de werktuigen. De machines bij deze transferlijnen vereisen met andere woorden geen herstellingen. Bewerkingen uitgevoerd door betrouwbare machines resulteren in minder variabele bewerkingstijden dan bewerkingen uitgevoerd door onbetrouwbare machines.

Conway e.a. (1988) definiëren **onbetrouwbare** transferlijnen als transferlijnen waarbij de machines stoppen met werken als gevolg van faling van de werktuigen of andere willekeurige oorzaken. Het effect hiervan is hetzelfde als het verlengen van de bewerkingstijd van het product in bewerking. De resulterende bewerkingstijd is zeer variabel omdat falingen zich zelden voordoen en substantiële vertragingen veroorzaken wanneer falingen zich wel voordoen.

3.1.1.4 Operatie-afhankelijke Versus Tijdsafhankelijke Falingen

Buzacott en Hanifin (1978) verklaren dat falingen veroorzaakt kunnen worden door zowel operatie-afhankelijke als tijdsafhankelijke factoren. **Operatie-afhankelijke** falingen kunnen niet voorkomen wanneer de machine, geblokkeerd, uitgeput of in herstelling is. Deze falingen kunnen dus met andere woorden slechts voorkomen indien de machine operationeel is. De oorzaak is bij deze vorm van falingen vooral te wijten aan onvolmaaktheden of slijtage van de machine. De meeste falingen zijn van deze soort.

Tijdsafhankelijke falingen kunnen wel voorkomen wanneer een werkstation zich in een niet actieve toestand bevindt. Deze falingen treden dus altijd op met hetzelfde ritme. Dit is ongeacht of het werkstation werkt of niet, of zelfs trager werkt. Deze vorm van falingen kan verscheidene oorzaken hebben. Het wisselen van ploeg in een ploegendienst is een mogelijke oorzaak van tijdsafhankelijke falingen.

Merk op dat falingen betrekking kunnen hebben op enerzijds één of enkele machines en anderzijds de hele transferlijn. Slijtage van werktuigen kan bijvoorbeeld resulteren in een faling van één machine, terwijl waarschijnlijk een stroompanne resulteert in een faling van de hele transferlijn.

3.1.1.5 Machinetoestanden

De machinetoestanden in transferlijnen zijn afhankelijk van twee factoren, namelijk enerzijds of de machine *operationeel* is of niet en anderzijds of de machine *actief* is of niet. De mogelijke toestanden van machines worden in tabel 3.1 weergegeven.

Tabel 3.1: Classificatie van machinetoestanden

Machine State	*Operational*	*Not Operational*
Active	Working	/
Not Active	Forced Down: - Starved - Blocked	Failed

Jafari en Shanthikumar (1989) vermelden dat een machine in een transferlijn ofwel operationeel ofwel niet operationeel is. Een machine bevindt zich in de toestand **Working** indien het werkstation *operationeel* en *actief* is. De machine is met andere woorden een product aan het verwerken.

Buzacott en Hanifin (1978) verklaren dat een machine zich in de toestand **Failed** bevindt indien het werkstation stopt met werken als gevolg van een faling. De machine is in herstelling en is dus zowel *niet operationeel* als *niet actief*.

Een werkstation bevindt zich in de toestand **Forced Down** indien de machine in kwestie nog steeds capabel is om te werken maar moet stoppen wegens een faling van een andere machine. De machine is dus met andere woorden *niet actief* maar wel *operationeel*. Twee mogelijke toestanden doen zich voor in een *Forced Down* situatie, namelijk de toestanden *Starved* en *Blocked*.

De machinetoestand **Starved** doet zich voor wanneer het werkstation niet kan werken omdat geen input beschikbaar is, delen Jafari en Shanthikumar (1989) mee. Alhoewel de machine *operationeel* is, is de machine *niet actief* omdat de vorige buffer leeg is. Bij gesynchroniseerde transferlijnen veroorzaakt een uitgeputte machine dat alle volgende machines uitgeput zijn. Werkstations die zich in deze toestand bevinden, zijn niet kwetsbaar voor pannes. De eerste machine wordt verondersteld nooit uitgeput te zijn.

Een machine bevindt zich in de toestand **Blocked** wanneer de machine niet verder kan werken omdat geen plaats beschikbaar is om de output te stockeren. De machine is in deze situatie *operationeel* maar *niet actief* omdat de volgende buffer vol is. Bij gesynchroniseerde transferlijnen veroorzaakt een geblokkeerd werkstation dat alle vorige machines geblokkeerd zijn. Machines die zich in deze toestand bevinden, zijn eveneens niet kwetsbaar voor pannes. De laatste machine wordt verondersteld nooit geblokkeerd te zijn.

3.1.2 Prestatie Transferlijnen

In deze subsectie worden drie verschillende manieren vermeld om de prestaties van transferlijnen uit te drukken. Een eerste manier betreft de doorzet van transferlijnen. Vervolgens kan ook de beschikbaarheid van transferlijnen gebruikt worden. Ten slotte kan eveneens de efficiëntie van transferlijnen de prestaties bepalen.

3.1.2.1 Doorzet

Van Oudheusden en Janssens (1994) vermelden dat de **doorzet** d van transferlijnen als volgt kan bepaald worden, waarbij $q(\tau)$ de geproduceerde hoeveelheid op tijdstip τ weergeeft:

$$d = \lim_{t \to \infty} \frac{\int_0^t q(\tau) d\tau}{t}$$

3.1.2.2 Beschikbaarheid

Van Oudheusden en Janssens (1994) verklaren dat de **beschikbaarheid** A van transferlijnen de verhouding is van de *uptime* over de *total time*, oftewel de som van de *uptime* en de *downtime*. De *uptime* wordt gedefinieerd als de tijd dat transferlijnen operationeel zijn tussen twee falingen. De *downtime* wordt gedefinieerd als de periode dat transferlijnen geen output produceren, wanneer de transferlijnen met andere woorden niet actief zijn. De beschikbaarheid van transferlijnen is een maatstaf tussen nul en één en wordt in de onderstaande formule geïllustreerd:

$$A = \lim_{t\to\infty} \frac{uptime}{total\ time} = \lim_{t\to\infty} \frac{uptime}{uptime + downtime}$$

3.1.2.3 Efficiëntie

Buzacott en Hanifin (1978) omschrijven de **efficiëntie** E van transferlijnen. In deze expressie stelt $q(t)$ de gerealiseerde productie in tijdsperiode t voor en $Q(t)$ de fictieve productie met een perfecte transferlijn in tijdsperiode t. De efficiëntie van transferlijnen is een maatstaf tussen nul en één, uitgedrukt als volgt:

$$E = \lim_{t\to\infty} \frac{q(t)}{Q(t)}$$

3.1.3 Modellering Transferlijnen

Bij gesynchroniseerde transferlijnen, verklaart Prave (2000), kan de faling van één enkele machine resulteren tot het stilvallen van de hele transferlijn. Het gevolg hiervan is dat potentiële productie verloren gaat. De reden is dat de vorige machine van een gefaalde machine geblokkeerd wordt. De

geblokkeerde machine blokkeert op zijn beurt alle vorige machines. Tegelijkertijd zal de volgende machine van de gefaalde machine uitgeput raken waardoor alle volgende machines uitgeput raken.

Tussen de verschillende werkstations van transferlijnen is met andere woorden sprake van **afhankelijkheid**, haalt Gelaesen (2003) aan. Deze afhankelijkheid kan doorbroken worden op twee manieren, namelijk door gebruik te maken van *Redundante Machines* of *Intermediaire Buffercapaciteit*.

Een eerste mogelijkheid om de efficiëntie van transferlijnen te verhogen is de installatie van **Redundante Machines**. Dit houdt in dat bepaalde werkstations in transferlijnen dubbel en in parallel voorkomen. Deze kunnen op basis van twee mogelijke systemen geïmplementeerd worden. Bij het *Splitting System* is het mogelijk om de totale productie te spreiden over de werkstations. In dit geval zullen de werkstations elk de helft van de goederenstroom verwerken. Indien één van de werkstations uitvalt wegens een faling, verdubbelt het andere werkstation het productieritme en verwerkt de volledige goederenstroom. Bij het *Standby System* zal daarentegen slechts één werkstation produceren. Het andere werkstation wordt als reservetoestel aangewend. Indien het eerste toestel defect raakt wegens een faling, zal met behulp van een schakelaar de goederenstroom naar het reservetoestel omgeleid worden.

Een tweede mogelijkheid om de afhankelijkheid tussen de machines van transferlijnen te doorbreken is het aanleggen van **Intermediaire Buffercapaciteit**. Prave (2000) licht toe dat de werkstukken van buiten het systeem naar de eerste machine stromen en vervolgens naar de eerste buffervoorraad. Daarna stromen de werkstukken naar de tweede machine en vervolgens naar de tweede buffer. Uiteindelijk zullen de werkstukken naar de laatste machine stromen en vervolgens het systeem verlaten. In de gevallen waar het toegelaten is om buffers aan te leggen in transferlijnen, zijn de locaties en de capaciteiten van de buffervoorraden belangrijke beslissingsvariabelen. De verdeling van de totale buffercapaciteit beïnvloedt de efficiëntie van het systeem en de gemiddelde hoeveelheid goederen in bewerking.

Gelaesen (2003) merkt echter op dat buffercapaciteit zeer duur is aangezien enerzijds ruimte en installaties vereist zijn en anderzijds vaak extra materiaalbehandeling nodig is. Goederen in voorraad kunnen bovendien hun waarde verliezen of beschadigd raken. Door het aanhouden van buffers wordt

eveneens een groot deel van het kapitaal geblokkeerd. Het is dus belangrijk om de extra winst – als gevolg van de hogere efficiëntie – af te wegen tegen de kosten die voorraden met zich meebrengen.

In de literatuur over transferlijnen met eindige buffervoorraden worden vier klassen van modellen onderscheiden, vermeldt De Koster (1989). Een eerste klasse betreft systemen waar de bewerkingstijden willekeurige variabelen zijn en de producten discreet. Een tweede klasse veronderstelt deterministische bewerkingstijden, maar de machines zijn onbetrouwbaar en falen van tijd tot tijd. De producten zijn discreet en de herstellingstijden hebben geometrische verdelingen. Een derde klasse betreft continue goederenstromen. De machinesnelheden zijn deterministisch, maar machines kunnen falen. Een vierde klasse betreft discrete producten, falingen van machines en willekeurige bewerkingstijden.

3.2 Optimalisatie Buffer Allocatie

Zowel de grootte van de buffers als de locatie hiervan hebben een invloed op het outputniveau van transferlijnen, stellen Lutz e.a. (1998). Een bepaalde buffer allocatie wordt een **bufferprofiel** genoemd. Dit profiel representeert dus een unieke combinatie van voorraad allocaties.

In het algemeen zal het maximale outputniveau stijgen wanneer het totale aantal van voorraadplaatsen in transferlijnen stijgt, maar meestal zal deze stijging echter afnemen. Een plot van de maximale outputniveaus tegenover de totale voorraadniveaus wordt een productie **Line Specific Output Curve** genoemd. Deze curve karakteriseert de prestatie van transferlijnen relatief gezien ten opzichte van de totale buffervoorraad in transferlijnen. Het outputniveau is gevoeliger ten aanzien van buffer allocaties in het lage gedeelte van de curve en minder gevoelig in het vlakke gedeelte van de curve.

Met betrekking tot buffer allocaties bij **gebalanceerde** transferlijnen kunnen enkele fenomenen vastgesteld worden. Een eerste fenomeen betreft de *omkeerbaarheid* van een bufferprofiel. Dit wordt ook het *dualiteit theorema* genoemd. Een bepaald bufferprofiel zal dus hetzelfde outputniveau genereren als zijn spiegelbeeld. Vervolgens levert de *symmetrische voorraad allocatie* vaak de hoogste outputniveaus van transferlijnen. De symmetrische voorraad allocatie is een bufferprofiel waarbij de

buffers symmetrisch in transferlijnen geplaatst worden. Ten slotte genereert de *geïnverteerde voorraad bowl* eveneens vaak de hoogste outputniveaus van transferlijnen. De geïnverteerde voorraad bowl is een bufferprofiel waarbij de grootste buffers in het midden van transferlijnen worden geplaatst. Dit bufferprofiel is met andere woorden een speciaal geval van de symmetrische voorraad allocatie.

Bij **ongebalanceerde** transferlijnen dient extra aandacht gegeven te worden aan de *bottleneck-operatie*. Deze machine dient omringd te worden door buffers die groter zijn dan de buffers bij de niet-bottleneck machines. Een buffer voor de bottleneck is nodig om uitputting van de bottleneck te vermijden of de kans erop te verminderen en een buffer na de bottleneck is nodig om blokkade te vermijden of verminderen veroorzaakt door de bottleneck.

Conway e.a. (1988) geven eveneens enkele richtlijnen met betrekking tot *homogene* en *heterogene* transferlijnen. Bovendien worden ook richtlijnen gegeven voor *onbetrouwbare* transferlijnen. Deze worden vervolgens kort besproken.

Bij *symmetrische* transferlijnen is de beste buffer allocatie eveneens symmetrisch en dit is een manifestatie van het bowl fenomeen. Buffers in het midden van een transferlijn presteren significant beter dan buffers aan de uiteinden van een transferlijn geplaatst, maar ongeveer in het midden is bijna even goed als exact in het midden. Bij *ongebalanceerde* transferlijnen leveren buffers een mindere stijging op in capaciteit. De buffers worden in dit geval bij voorkeur geplaatst voor en na de bottleneck. Bij *onbetrouwbare* transferlijnen zou de buffercapaciteit in veelvouden moeten zijn van de gemiddelde hoeveelheid producten die een machine produceert tijdens de herstelling van een andere machine. De grootte van dit veelvoud is onder andere afhankelijk van de mate van variabiliteit van de herstellingstijd.

Sheskin (1976) ontwikkelt een benaderingsmethode om de efficiëntie te berekenen van transferlijnen met eindige voorraadbuffers. Bij deze transferlijnen wordt verondersteld dat elke machine onderhevig is aan willekeurige falingen en herstellingen. Het gaat hier dus om **onbetrouwbare** transferlijnen. Gebaseerd op numerieke resultaten worden richtlijnen gegeven voor buffer allocatie in transferlijnen waarbij de machines even betrouwbaar zijn of in stijgende of dalende volgorde van betrouwbaarheid.

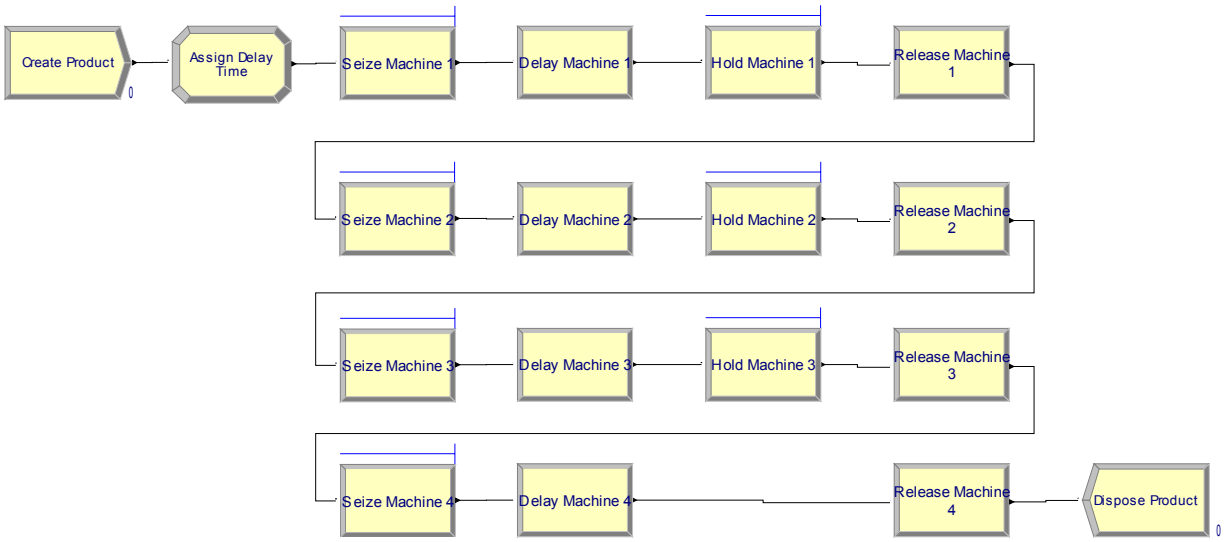
Indien alle machines *even* betrouwbaar zijn wordt het productieritme gemaximaliseerd door de buffers ongeveer zo gelijk mogelijk te maken in grootte. Wanneer de machines in *dalende* volgorde van betrouwbaarheid zijn, wordt de grootste buffer aan het einde van de lijn geplaatst voor de minst betrouwbare machine. Bij deze transferlijnen stijgen dus de buffergroottes naar het einde van de lijn toe. Wanneer de machines in *stijgende* volgorde van betrouwbaarheid zijn, wordt de grootste buffer aan het begin van de lijn geplaatst voor de minst betrouwbare machine. Bij deze transferlijnen dalen de buffergroottes naar het einde van de lijn toe.

HOOFDSTUK IV: SIMULATIE-OPTIMALISATIE BIJ TRANSFERLIJNEN

In dit hoofdstuk zal de praktijkstudie besproken worden. Ten eerste zal begonnen worden met de bespreking van het gebouwde model in *Arena*. Een grondig overzicht van het model zal gegeven worden met de nodige uitleg. Vervolgens zal de *Single Factor Method* op dit model toegepast worden. Elke stap zal in detail verklaard worden om inzicht te krijgen in de werking van de techniek. Ten slotte zullen de resultaten van het onderzoek weergegeven worden. Deze resultaten zullen uiteraard eveneens besproken worden.

4.1 Model

Voor de praktijkstudie zal gebruik gemaakt worden van een simulatiemodel. Dit model werd met behulp van de simulatiesoftware **Arena** gebouwd. Het model betreft een transferlijn met vier machines en dus eveneens vier buffers. De eerste buffer bevindt zich helemaal vooraan en de overige buffers zijn intermediaire buffers. Deze buffers bevinden zich met andere woorden tussen twee machines in. De capaciteit van de eerste buffer wordt verondersteld oneindig groot te zijn, terwijl deze van de intermediaire buffers eindig zijn. De **beslissingsvariabelen** van dit model zijn enerzijds de *capaciteiten* van de intermediaire buffers en anderzijds de *wachtrij-discipline*. De voorstelling van dit model in Arena is hieronder te zien in figuur 4.1 en zal vervolgens nader toegelicht worden.

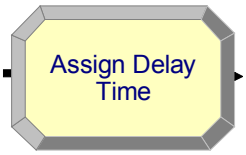


Figuur 4.1: Schematische voorstelling van het simulatiemodel

De aankomsten worden in *Arena* visueel weergegeven door de **Create** module. In deze module wordt het *aankomstenproces* gemodelleerd. Voor dit proces werd een exponentiële kansverdeling gekozen met als gemiddelde 14 minuten voor de verdeling van de tussenaankomsttijden. Dit wil zeggen dat gemiddeld om de 14 minuten een product zal toekomen in het model. In deze module kan bepaald worden hoeveel verschillende producten de transferlijn verwerkt door meerdere entiteiten in te geven. In dit model werd gekozen voor één entiteit. In *Arena* kan eveneens worden bepaald hoeveel producten per aankomst toegelaten worden in het model. In dit model werd dit bepaald op één. Deze module zal 1.000 aankomsten genereren en vervolgens stoppen. Op deze manier zal het model niet eeuwig blijven runnen. De *Create* module wordt in figuur 4.2 geïllustreerd.

Figuur 4.2: Voorstelling van de Create module in Arena

De volgende module is de **Assign** module. In deze module kunnen bepaalde eigenschappen aan entiteiten toegekend worden. In dit model werd deze module gebruikt om de *bewerkingstijden* bij elke machine op voorhand toe te kennen aan de producten. In figuur 4.3 wordt de *Assign* module weergegeven.



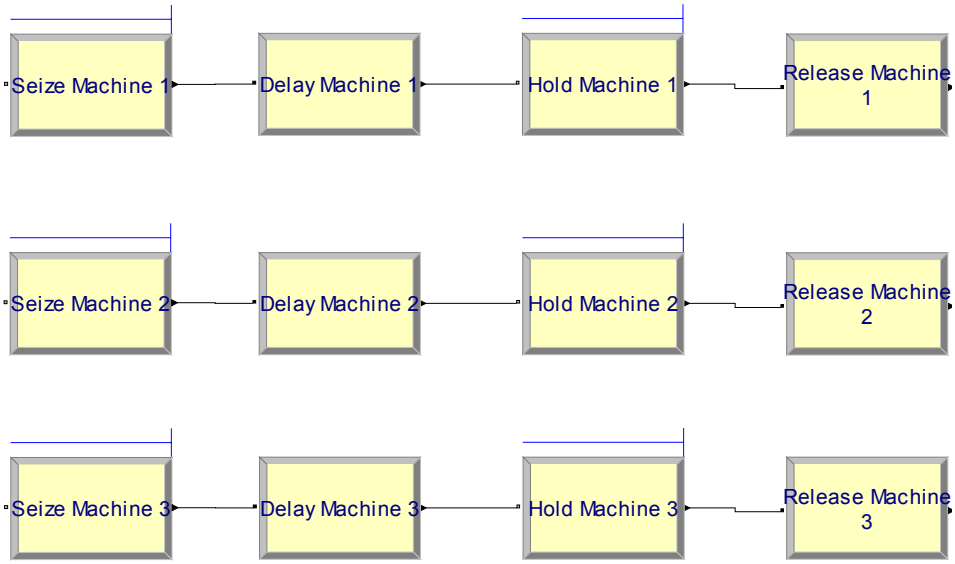
Figuur 4.3: Voorstelling van de Assign module in Arena

De **bewerkingstijden** zijn stochastisch en volgen met andere woorden een kansverdeling. Voor alle machines werd geopteerd voor een uniforme kansverdeling. Aangezien dit model vier machines telt, werden vier afzonderlijke attributen gedefinieerd. In onderstaande tabel is een overzicht te vinden van deze attributen en hun respectievelijke kansverdeling.

Tabel 4.1: Overzicht van de bewerkingstijden

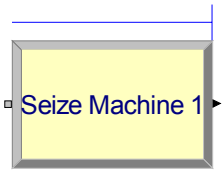
Attribuut	Kansverdeling	Gemiddelde Bewerkingstijd	Minimale Bewerkingstijd	Maximale Bewerkingstijd
Delay Time Machine 1	Uniform	10	6	14
Delay Time Machine 2	Uniform	15	12	18
Delay Time Machine 3	Uniform	10	3	17
Delay Time Machine 4	Uniform	15	10	20

De eerste drie machines werden gemodelleerd door gebruik te maken van vier modules. Deze modules zijn de *Seize*, *Delay*, *Hold* en *Release* module. In *Arena* kan een machine eenvoudig voorgesteld worden door een **Process** module, maar deze module werd niet gebruikt om meer flexibiliteit te verkrijgen. In figuur 4.4 wordt het model van de eerste drie machines weergegeven en vervolgens zal dieper ingegaan worden op de afzonderlijke modules. Bij de bespreking van deze modules zal enkel de modules van de eerste machine getoond worden, maar uiteraard is de werking hiervan van toepassing op de tweede en derde machine.



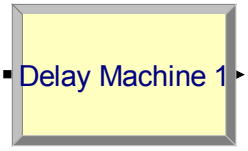
Figuur 4.4: Schematische voorstelling van de eerste drie machines van het model

De **Seize** module in *Arena* heeft als doel om de resource *toe te wijzen* aan een entiteit. In deze module moet bepaald worden om welke resource het gaat en hoeveel resources beschikbaar zijn. In dit model is de machine zelf de resource en is per machine één resource beschikbaar. Indien een resource reeds werd toegewezen aan een entiteit, kan de resource niet toegewezen worden aan een volgende entiteit vooraleer de resource werd vrijgegeven. De buffer van de eerste *Seize* module is oneindig groot, maar de buffers van de overige *Seize* modules hebben een eindige capaciteit. Bij al deze buffers dient de wachtrij-discipline ingegeven te worden en is met andere woorden een beslissingsvariabele. Figuur 4.5 toont de *Seize* module van *Arena*.



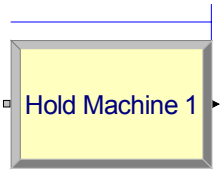
Figuur 4.5: Voorstelling van de Seize module in Arena

De **Delay** module in *Arena* simuleert de bewerkingstijd van de machine. In deze module dient de *bewerkingstijd* van het product bepaald te worden. Deze bewerkingstijden werden eigenlijk reeds bepaald in de *Assign* module. In deze module dient daarom enkel het desbetreffende attribuut ingegeven te worden als bewerkingstijd. In het geval van de eerste machine is dit het volgende attribuut, namelijk *Delay Time Machine 1*. Onderstaande figuur 4.6 illustreert de *Delay* module in *Arena*.



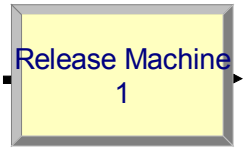
Figuur 4.6: Voorstelling van de Delay module in Arena

Het doel van de **Hold** module in *Arena* is in dit model uniek. Deze module zal eerst nagaan of de volgende buffer al dan niet volzet is. Indien de buffer vol is, zal deze module de entiteit tegenhouden totdat een plaats vrijkomt in de volgende buffer. Hierdoor zal de machine geblokkeerd worden aangezien de resource nog niet werd vrijgegeven. De *Hold* module werd voor de *Release* module geplaatst om te vermijden dat de resource wordt vrijgegeven, terwijl de volgende buffer volzet is. Om dit doel te verwezenlijken werd gekozen voor de optie *Scan For Condition*. Vervolgens wordt bepaald wanneer de entiteit naar de volgende module wordt verstuurd. In deze modules zullen dus de waarden van de beslissingsvariabelen bepaald worden, aangezien hier de capaciteiten van de volgende buffers bepaald zullen worden. In figuur 4.7 is de *Hold* module van *Arena* te zien.



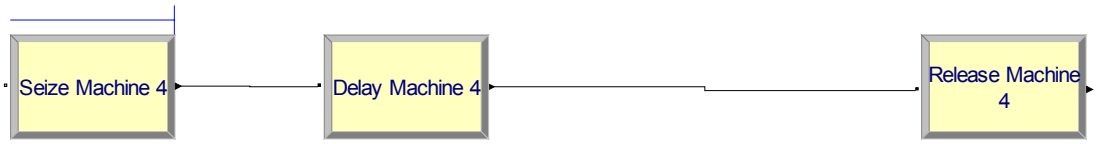
Figuur 4.7: Voorstelling van de Hold module in Arena

De **Release** module in *Arena* heeft als doel het tegenovergestelde van de *Seize* module. Daar waar de *Seize* module de resource toewijst aan de entiteit, zal de *Release* module de resource weer *vrijgeven*. Hier dient eveneens bepaald te worden om welke resource het gaat en de hoeveelheid hiervan. Indien het product naar de volgende machine kan, wordt de resource vrijgegeven zodat de huidige machine aan het volgende product kan beginnen. Figuur 4.8 geeft de *Release* module in *Arena* weer.



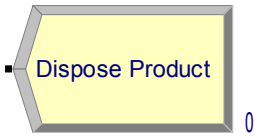
Figuur 4.8: Voorstelling van de Release module in Arena

De laatste machine werd gemodelleerd door gebruik te maken van drie modules en niet vier zoals bij de vorige drie machines. De *Hold* module werd niet gebruikt om deze machine te modelleren. De reden hiervoor is dat geen controle uitgevoerd dient te worden om na te gaan of plaats beschikbaar is voor het eindproduct te stockeren. De veronderstelling is namelijk dat hiervoor altijd plaats beschikbaar is. De overige modules vervullen dezelfde rol als deze bij de vorige drie machines. In figuur 4.9 wordt het model van de laatste machine getoond.



Figuur 4.9: Schematische voorstelling van de laatste machine van het model

De laatste module van het model betreft de **Dispose** module. Deze module heeft als doel het *verwijderen* van alle entiteiten die aankomen in deze module. Op deze manier zal het geheugen van het model niet overbelast raken, aangezien de entiteiten uit het model verwijderd worden nadat deze het volledige model doorlopen hebben. Hieronder wordt in figuur 4.10 de *Dispose* module van *Arena* geïllustreerd.



Figuur 4.10: Voorstelling van de Dispose module in Arena

4.2 Methodiek

In deze sectie zal de methodiek van de *Single Factor Method* besproken worden. Dit zal aan de hand van het bovenstaande simulatiemodel besproken worden. Voor een formele bespreking van de methode wordt naar het tweede hoofdstuk verwezen, namelijk 2.1.1.6. Opgemerkt dient te worden dat voor elke simulatierun vijf replicaties uitgevoerd zullen worden om minder variabiliteit te bekomen. Ten eerste zal dieper in gegaan worden op de doelfunctie. Elke parameter van deze doelfunctie zal bepaald

worden, alsook de waarden hiervan. Vervolgens zal de startsituatie weergegeven worden samen met de waarde van de doelfunctie. Ten slotte zal de methodiek toegepast worden totdat de optimale oplossing bekomen wordt. De oplossing zal als optimaal beschouwd worden wanneer de doelfunctie met minder dan één procent stijgt voor elke beslissingsvariabele.

4.2.1 Doelfunctie

De doelfunctie die gebruikt zal worden om de prestaties van het simulatiemodel te bepalen betreft een winstfunctie. Deze functie heeft vijf componenten, namelijk één voor de opbrengsten en vier voor de kosten. De **winstfunctie** wordt in de onderstaande formule weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{TT/E} \cdot \frac{P}{E} - B_T \cdot \frac{K_B/C}{T} - B_2 \cdot \frac{K_2/E}{T} - B_3 \cdot \frac{K_3/E}{T} - B_4 \cdot \frac{K_4/E}{T}$$

Op deze manier hangt de winst af van enerzijds de prestatie van de transferlijn en anderzijds de extra kosten die nodig zijn om deze prestatie te bekomen. De gebruikte **symbolen** en de waarden voor de parameters worden in de volgende tabel gedemonstreerd.

Tabel 4.2: Overzicht van de symbolen

Symbool	Verklaring	Uitdrukking
Π	Winst	€
T	Tijdseenheid	Dag
TT	Totale Tijd	Minuut
E	Eenheid	Eenheid
P	Prijs	€ 100
B_T	Totale Buffercapaciteit	Capaciteit
K_B	Kost Buffercapaciteit	€ 5
C	Capaciteit	Capaciteit
B_2	Bufferbezetting 2	Eenheid
K_2	Kost Bufferbezetting 2	€ 1
B_3	Bufferbezetting 3	Eenheid
K_3	Kost Bufferbezetting 3	€ 2
B_4	Bufferbezetting 4	Eenheid
K_4	Kost Bufferbezetting 4	€ 3

Opgemerkt dient te worden dat de kost van de bufferbezetting, oftewel de voorraadkost, stijgt naarmate stroomafwaarts wordt gegaan in de transferlijn. De reden hiervoor is dat het product reeds langer bewerkt werd naarmate stroomafwaarts wordt gegaan in de transferlijn. Het product wordt dan ook verondersteld een grotere voorraadkost te hebben.

4.2.2 Startsituatie

Als **startsituatie** wordt gekozen voor een transferlijn *zonder* tussenliggende buffers met als wachtrij-discipline *First In First Out*. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 1*. De waarden in de onderstaande tabel en alle volgende tabellen in deze sectie zijn gemiddelden over de vijf replicaties heen. Voor de betrouwbaarheidsintervallen wordt naar de bijlage verwezen, meer specifiek naar de kolom *Half Width*. Dit is het getal dat van het gemiddelde afgetrokken en bij het gemiddelde opgeteld moet worden om het 95% betrouwbaarheidsinterval te bekomen. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.3: Overzicht van de belangrijkste resultaten van Simulatierun 1

Output	Waarde
Totale Tijd	1.053,71
Bufferbezetting 2	0
Bufferbezetting 3	0
Bufferbezetting 4	0

De waarde van de doelfunctie wordt bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn, zonder gebruik te maken van tussenliggende buffers, bedraagt *€ 136,66* en wordt in de volgende uitdrukking weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{1.053,71} \cdot 100 - 0 \cdot 5 - 0 \cdot 1 - 0 \cdot 2 - 0 \cdot 3 = 136,66$$

4.2.3 Optimum

Het **optimum** van de doelfunctie wordt bekomen door voor elke controle variabele na te gaan of een daling of stijging van één eenheid resulteert in een stijging van de doelfunctie met minstens meer dan één procent. Ten eerste worden de afzonderlijke *buffers* aan deze test onderworpen. Vervolgens wordt eveneens nagegaan of de *wachtrij-discipline* een rol kan spelen om de winst te maximaliseren bij deze transferlijn.

In de eerste fase wordt begonnen door de **tweede** buffercapaciteit te laten stijgen met *één* eenheid. Deze buffercapaciteit laten dalen is geen optie, aangezien een negatieve buffercapaciteit niet toegelaten is. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 2*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.4: Overzicht van de belangrijkste resultaten van Simulatierun 2

Output	Waarde
Totale Tijd	1.051,72
Bufferbezetting 2	0,9854
Bufferbezetting 3	0
Bufferbezetting 4	0

De waarde van de doelfunctie wordt hier eveneens bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 130,93* en wordt in de onderstaande expressie weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{1.051{,}72} \cdot 100 - 1 \cdot 5 - 0{,}9854 \cdot 1 - 0 \cdot 2 - 0 \cdot 3 = 130{,}93$$

Aangezien de dagelijkse winst gedaald is, zal de buffercapaciteit van de **tweede** buffer terug op *nul* gezet worden. Vervolgens wordt de zoektocht verder gezet in de **derde** buffercapaciteit door deze te laten stijgen tot *één*. Hier is een daling van de buffercapaciteit eveneens geen optie om dezelfde reden als voor de tweede buffer. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 3*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.5: Overzicht van de belangrijkste resultaten van Simulatierun 3

Output	Waarde
Totale Tijd	816,30
Bufferbezetting 2	0
Bufferbezetting 3	0,5446
Bufferbezetting 4	0

De waarde van de doelfunctie wordt weer bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 170,32* en wordt in de onderstaande formule weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{816{,}30} \cdot 100 - 1 \cdot 5 - 0 \cdot 1 - 0{,}5446 \cdot 2 - 0 \cdot 3 = 170{,}32$$

Aangezien de dagelijkse winst gestegen is met tenminste één procent, namelijk met tenminste 1,37, zal de buffercapaciteit van de **derde** buffer verhoogd worden tot *twee* in de volgende simulatierun. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 4*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.6: Overzicht van de belangrijkste resultaten van Simulatierun 4

Output	Waarde
Totale Tijd	778,73
Bufferbezetting 2	0
Bufferbezetting 3	1,198
Bufferbezetting 4	0

De waarde van de doelfunctie wordt weer bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 172,52* en wordt in de onderstaande uitdrukking weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{778,73} \cdot 100 - 2 \cdot 5 - 0 \cdot 1 - 1,198 \cdot 2 - 0 \cdot 3 = 172,52$$

Aangezien de dagelijkse winst gestegen is met tenminste één procent, namelijk met tenminste 1,70, zal de buffercapaciteit van de **derde** buffer verhoogd worden tot *drie* in de volgende simulatierun. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 5*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.7: Overzicht van de belangrijkste resultaten van Simulatierun 5

Output	Waarde
Totale Tijd	767,65
Bufferbezetting 2	0
Bufferbezetting 3	1,9036
Bufferbezetting 4	0

De waarde van de doelfunctie wordt weer bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 168,78* en wordt in de onderstaande expressie weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{767,65} \cdot 100 - 3 \cdot 5 - 0 \cdot 1 - 1,9036 \cdot 2 - 0 \cdot 3 = 168,78$$

Aangezien de dagelijkse winst gedaald is, zal de buffercapaciteit van de **derde** buffer terug op *twee* gezet worden. Vervolgens wordt de zoektocht verder gezet in de **vierde** en laatste buffercapaciteit door

deze te laten stijgen tot *één*. Hier is een daling van de buffercapaciteit weer geen optie om dezelfde reden als voor de tweede en de derde buffer. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 6*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.8: Overzicht van de belangrijkste resultaten van Simulatierun 6

Output	Waarde
Totale Tijd	670,62
Bufferbezetting 2	0
Bufferbezetting 3	0,5213
Bufferbezetting 4	0,7358

De waarde van de doelfunctie wordt weer bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 196,48* en wordt in de onderstaande formule weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{670{,}62} \cdot 100 - 3 \cdot 5 - 0 \cdot 1 - 0{,}5213 \cdot 2 - 0{,}7358 \cdot 3 = 196{,}48$$

Aangezien de dagelijkse winst gestegen is met tenminste één procent, namelijk met tenminste 1,73, zal de buffercapaciteit van de **vierde** buffer verhoogd worden tot *twee* in de volgende simulatierun. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 7*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.9: Overzicht van de belangrijkste resultaten van Simulatierun 7

Output	Waarde
Totale Tijd	667,46
Bufferbezetting 2	0
Bufferbezetting 3	0,3801
Bufferbezetting 4	1,2587

De waarde van de doelfunctie wordt weer bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 191,21* en wordt in de onderstaande uitdrukking weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{667{,}46} \cdot 100 - 4 \cdot 5 - 0 \cdot 1 - 0{,}3801 \cdot 2 - 1{,}2587 \cdot 3 = 191{,}21$$

Aangezien de dagelijkse winst gedaald is, zal de buffercapaciteit van de **vierde** buffer terug op *één* gezet worden. Vervolgens wordt de zoektocht herbegonnen in de **tweede** buffercapaciteit door deze opnieuw te laten stijgen tot *één*. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 8*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.10: Overzicht van de belangrijkste resultaten van Simulatierun 8

Output	Waarde
Totale Tijd	663,31
Bufferbezetting 2	0,975
Bufferbezetting 3	0,562
Bufferbezetting 4	0,7451

De waarde van de doelfunctie wordt weer bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 192,76* en wordt in de onderstaande expressie weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{663{,}31} \cdot 100 - 4 \cdot 5 - 0{,}975 \cdot 1 - 0{,}562 \cdot 2 - 0{,}7451 \cdot 3 = 192{,}76$$

Aangezien de dagelijkse winst gedaald is, zal de buffercapaciteit van de **tweede** buffer terug op *nul* gezet worden. Vervolgens wordt de zoektocht verder gezet in de **derde** buffercapaciteit door deze te laten dalen tot één of te laten stijgen tot drie. Ten eerste wordt de derde buffercapaciteit verlaagd tot *één*. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 9*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.11: Overzicht van de belangrijkste resultaten van Simulatierun 9

Output	Waarde
Totale Tijd	686,21
Bufferbezetting 2	0
Bufferbezetting 3	0,2366
Bufferbezetting 4	0,7016

De waarde van de doelfunctie wordt weer bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 197,27* en wordt in de onderstaande formule weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{686{,}21} \cdot 100 - 2 \cdot 5 - 0 \cdot 1 - 0{,}2366 \cdot 2 - 0{,}7016 \cdot 3 = 197{,}27$$

Aangezien de dagelijkse winst niet gestegen is met tenminste één procent, namelijk met tenminste 1,96, zal de buffercapaciteit van de **derde** buffer verhoogd worden tot *drie*. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 10*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.12: Overzicht van de belangrijkste resultaten van Simulatierun 10

Output	Waarde
Totale Tijd	667,51
Bufferbezetting 2	0
Bufferbezetting 3	0,8229
Bufferbezetting 4	0,7483

De waarde van de doelfunctie wordt weer bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 191,84* en wordt in de onderstaande uitdrukking weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{667{,}51} \cdot 100 - 4 \cdot 5 - 0 \cdot 1 - 0{,}8229 \cdot 2 - 0{,}7483 \cdot 3 = 191{,}84$$

Aangezien de dagelijkse winst gedaald is, zal de buffercapaciteit van de **derde** buffer terug op *twee* gezet worden. Het heeft geen zin om de zoektocht opnieuw verder te zetten in de vierde buffercapaciteit, aangezien deze situatie reeds onderzocht werd. Opgemerkt dient te worden dat de huidige maximale winst € 196,48 bedraagt per dag. Vervolgens wordt onderzocht of de **wachtrij-discipline** een rol kan spelen om de winst te doen stijgen. Aangezien in de startsituatie de wachtrij-discipline *First In First Out* was, wordt deze nu ingesteld op *Last In First Out*. De resultaten van deze

simulatie zijn terug te vinden in de bijlage *Simulatierun 11*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.13: Overzicht van de belangrijkste resultaten van Simulatierun 11

Output	Waarde
Totale Tijd	668,32
Bufferbezetting 2	0
Bufferbezetting 3	0,5225
Bufferbezetting 4	0,7488

De waarde van de doelfunctie wordt weer bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 197,17* en wordt in de onderstaande expressie weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{668{,}32} \cdot 100 - 3 \cdot 5 - 0 \cdot 1 - 0{,}5225 \cdot 2 - 0{,}7488 \cdot 3 = 197{,}17$$

Aangezien de dagelijkse winst niet gestegen is met tenminste één procent, namelijk met tenminste 1,96, zal de **wachtrij-discipline** veranderd worden tot *Longest Processing Time*. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 12*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.14: Overzicht van de belangrijkste resultaten van Simulatierun 12

Output	Waarde
Totale Tijd	669,32
Bufferbezetting 2	0
Bufferbezetting 3	0,568
Bufferbezetting 4	0,7529

De waarde van de doelfunctie wordt weer bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 196,75* en wordt in de onderstaande formule weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{669{,}32} \cdot 100 - 3 \cdot 5 - 0 \cdot 1 - 0{,}568 \cdot 2 - 0{,}7529 \cdot 3 = 196{,}75$$

Aangezien de dagelijkse winst niet gestegen is met tenminste één procent, namelijk met tenminste 1,96, zal de **wachtrij-discipline** veranderd worden tot *Shortest Processing Time*. De resultaten van deze simulatie zijn terug te vinden in de bijlage *Simulatierun 13*. De belangrijkste resultaten worden hieronder opgesomd.

Tabel 4.15: Overzicht van de belangrijkste resultaten van Simulatierun 13

Output	Waarde
Totale Tijd	666,53
Bufferbezetting 2	0
Bufferbezetting 3	0,5469
Bufferbezetting 4	0,7462

De waarde van de doelfunctie wordt weer bekomen door de bovenstaande waarden in te vullen in de winstfunctie. De dagelijkse **winst** van de transferlijn bedraagt nu *€ 197,71* en wordt in de onderstaande uitdrukking weergegeven:

$$\frac{\Pi}{T} = 24 \cdot 60 \cdot \frac{1}{666{,}53} \cdot 100 - 3 \cdot 5 - 0 \cdot 1 - 0{,}5469 \cdot 2 - 0{,}7462 \cdot 3 = 197{,}71$$

Aangezien de dagelijkse winst niet gestegen is met tenminste één procent, namelijk met tenminste 1,96, wordt besloten dat de **wachtrij-discipline** in deze transferlijn **geen** significante invloed heeft op de dagelijkse winst. De maximale **winst** bedraagt **€ 196,48** per dag en wordt bekomen door de tweede **buffercapaciteit** te zetten op **nul**, de derde op **twee** en de vierde op **één**. Deze optimale buffer allocatie blijkt aan te sluiten bij de **geïnverteerde voorraad bowl**.

4.3 Onderzoek

Voor het onderzoek zullen vier gevallen geanalyseerd worden met behulp van de *Single Factor Method*, zoals hierboven beschreven. Het simulatiemodel, het aantal replicaties, de doelfunctie en het stopcriterium blijven onveranderd. Ten eerste wordt de optimale buffer allocatie geïdentificeerd voor een transferlijn waarbij de gemiddelde bewerkingstijden aflopen naarmate stroomafwaarts wordt gegaan in

de transferlijn. Vervolgens wordt dezelfde situatie onderzocht met als verschil dat de gemiddelde bewerkingstijden oplopen. Daarna wordt de optimale buffer allocatie geïdentificeerd voor een transferlijn met een bottleneck. Ten slotte wordt dezelfde situatie onderzocht met als verschil dat de bottleneck onderhevig is aan falingen.

4.3.1 Aflopende Bewerkingstijden

Het simulatiemodel dat voor dit onderzoek gebruikt wordt, is dezelfde als hierboven besproken. Het enige verschil is dat de bewerkingstijden **aflopen** naarmate stroomafwaarts wordt gegaan in de transferlijn. Een overzicht van deze bewerkingstijden is te vinden in onderstaande tabel 4.16.

Tabel 4.16: Overzicht van de aflopende bewerkingstijden

Attribuut	Kansverdeling	Gemiddelde Bewerkingstijd	Minimale Bewerkingstijd	Maximale Bewerkingstijd
Delay Time Machine 1	Uniform	14	11	17
Delay Time Machine 2	Uniform	13	8	18
Delay Time Machine 3	Uniform	12	5	19
Delay Time Machine 4	Uniform	11	7	15

In tabel 4.17 is een overzicht te vinden van het uitgevoerde onderzoek. In deze tabel zijn enkel de beslissingsvariabelen en de waarden van de doelfunctie opgenomen. Dit werd gedaan om alles overzichtelijk te houden.

Tabel 4.17: Overzicht van het onderzoek bij aflopende bewerkingstijden

Run	Wachtrij-discipline	Buffercapaciteit 2	Buffercapaciteit 3	Buffercapaciteit 4	Winst
1	FIFO	0	0	0	175,68
2	FIFO	1	0	0	265,24
3	FIFO	2	0	0	265,20
4	FIFO	1	1	0	525,17
5	FIFO	1	2	0	543,22
6	FIFO	1	3	0	542,68
7	FIFO	1	2	1	550,28
8	FIFO	1	2	2	545,27
9	FIFO	0	2	1	232,65
10	FIFO	2	2	1	566,75
11	FIFO	3	2	1	561,77
12	FIFO	2	1	1	569,00
13	FIFO	2	3	1	561,85
14	FIFO	2	2	0	561,34
15	FIFO	2	2	2	561,75
16	LIFO	2	2	1	564,30
17	LPT	2	2	1	477,95
18	SPT	2	2	1	639,79

In deze transferlijn heeft de **wachtrij-discipline** een significante invloed op de maximale winst per dag. De optimale wachtrij-discipline is de **Shortest Processing Time**. De maximale **winst** bedraagt **€ 639,79** per dag en wordt bekomen door de tweede **buffercapaciteit** te zetten op **twee**, de derde op **twee** en de vierde op **één**. Deze optimale buffer allocatie blijkt te **dalen** naar het einde van de transferlijn toe.

4.3.2 Oplopende bewerkingstijden

Dit onderzoek lijkt op het vorige, maar in deze transferlijn **lopen** de bewerkingstijden **op** naarmate stroomafwaarts wordt gegaan in de transferlijn. Een overzicht van de oplopende bewerkingstijden is te vinden in onderstaande tabel 4.18.

Tabel 4.18: Overzicht van de oplopende bewerkingstijden

Attribuut	Kansverdeling	Gemiddelde Bewerkingstijd	Minimale Bewerkingstijd	Maximale Bewerkingstijd
Delay Time Machine 1	Uniform	11	7	15
Delay Time Machine 2	Uniform	12	5	19
Delay Time Machine 3	Uniform	13	8	18
Delay Time Machine 4	Uniform	14	11	17

In tabel 4.19 is een overzicht te vinden van het uitgevoerde onderzoek. In deze tabel zijn eveneens enkel de beslissingsvariabelen en de waarden van de doelfunctie opgenomen. Dit werd gedaan om alles overzichtelijk te houden.

Tabel 4.19: Overzicht van het onderzoek bij oplopende bewerkingstijden

Run	Wachtrij-discipline	Buffercapaciteit 2	Buffercapaciteit 3	Buffercapaciteit 4	Winst
1	FIFO	0	0	0	177,95
2	FIFO	1	0	0	175,63
3	FIFO	0	1	0	244,99
4	FIFO	0	2	0	240,52
5	FIFO	0	1	1	531,94
6	FIFO	0	1	2	566,04
7	FIFO	0	1	3	568,07
8	FIFO	1	1	2	590,98
9	FIFO	2	1	2	585,22
10	FIFO	1	0	2	301,52
11	FIFO	1	2	2	585,33
12	FIFO	1	1	1	569,88
13	FIFO	1	1	3	585,35
14	LIFO	1	1	2	584,89
15	LPT	1	1	2	591,79
16	SPT	1	1	2	587,75

In deze transferlijn wordt besloten dat de **wachtrij-discipline** hier **geen** significante invloed heeft op de maximale winst per dag. De maximale **winst** bedraagt **€ 590,98** per dag en wordt bekomen door de tweede **buffercapaciteit** te zetten op **één**, de derde op **één** en de vierde op **twee**. Deze optimale buffer allocatie blijkt te **stijgen** naar het einde van de transferlijn toe.

4.3.3 Bottleneck

Voor dit onderzoek is een **bottleneck** aanwezig in de transferlijn. Dit wil zeggen dat de gemiddelde bewerkingstijd hiervan groter is dan bij de overige machines. De overige machines hebben allemaal dezelfde gemiddelde bewerkingstijden. Een overzicht van deze bewerkingstijden is te vinden in onderstaande tabel 4.20.

Tabel 4.20: Overzicht van de bewerkingstijden met een bottleneck

Attribuut	Kansverdeling	Gemiddelde Bewerkingstijd	Minimale Bewerkingstijd	Maximale Bewerkingstijd
Delay Time Machine 1	Uniform	13	6	20
Delay Time Machine 2	Uniform	13	9	17
Delay Time Machine 3	Uniform	15	12	18
Delay Time Machine 4	Uniform	13	8	18

In tabel 4.21 is een overzicht te vinden van het uitgevoerde onderzoek. In deze tabel zijn weer enkel de beslissingsvariabelen en de waarden van de doelfunctie opgenomen. Dit werd gedaan om alles overzichtelijk te houden.

Tabel 4.21: Overzicht van het onderzoek met een bottleneck

Run	Wachtrij-discipline	Buffercapaciteit 2	Buffercapaciteit 3	Buffercapaciteit 4	Winst
1	FIFO	0	0	0	206,94
2	FIFO	1	0	0	264,75
3	FIFO	2	0	0	258,89
4	FIFO	1	1	0	316,22
5	FIFO	1	2	0	309,43
6	FIFO	1	1	1	562,92
7	FIFO	1	1	2	558,55
8	FIFO	0	1	1	438,04
9	FIFO	2	1	1	557,16
10	FIFO	1	0	1	353,16
11	FIFO	1	2	1	556,40
12	LIFO	1	1	1	568,53
13	LPT	1	1	1	528,89
14	SPT	1	1	1	562,37

In deze transferlijn wordt besloten dat de **wachtrij-discipline** hier **geen** significante invloed heeft op de maximale winst per dag. De maximale **winst** bedraagt **€ 562,92** per dag en wordt bekomen door de tweede **buffercapaciteit** te zetten op **één**, de derde op **één** en de vierde op **één**. Deze optimale buffer allocatie is **evenredig** verdeelt over de buffers heen.

4.3.4 Onbetrouwbare Bottleneck

Het onderzoek van deze subsectie sluit aan bij het onderzoek van de vorige subsectie. Het verschil is dat in dit onderzoek de **bottleneck onbetrouwbaar** is. In dit model kan de bottleneck falen. Zowel voor de *uptime* als de *downtime* wordt een exponentiële verdeling gekozen. De gemiddelde *uptime* bedraagt 420 minuten en de gemiddelde *downtime* bedraagt 10 minuten. Een overzicht van de bewerkingstijden van dit onderzoek is te vinden in onderstaande tabel 4.22.

Tabel 4.22: Overzicht van de bewerkingstijden met een onbetrouwbare bottleneck

Attribuut	Kansverdeling	Gemiddelde Bewerkingstijd	Minimale Bewerkingstijd	Maximale Bewerkingstijd
Delay Time Machine 1	Uniform	13	7	19
Delay Time Machine 2	Uniform	13	9	17
Delay Time Machine 3	Uniform	15	12	18
Delay Time Machine 4	Uniform	13	8	18

In tabel 4.23 is een overzicht te vinden van het uitgevoerde onderzoek. In deze tabel zijn weer enkel de beslissingsvariabelen en de waarden van de doelfunctie opgenomen. Dit werd gedaan om alles overzichtelijk te houden.

Tabel 4.23: Overzicht van het onderzoek met een onbetrouwbare bottleneck

Run	Wachtrij-discipline	Buffercapaciteit 2	Buffercapaciteit 3	Buffercapaciteit 4	Winst
1	FIFO	0	0	0	163,80
2	FIFO	1	0	0	180,18
3	FIFO	2	0	0	174,23
4	FIFO	1	1	0	201,07
5	FIFO	1	2	0	194,19
6	FIFO	1	1	1	335,16
7	FIFO	1	1	2	330,16
8	FIFO	0	1	1	320,08
9	FIFO	2	1	1	329,31
10	FIFO	1	0	1	235,78
11	FIFO	1	2	1	328,44
12	LIFO	1	1	1	332,11
13	LPT	1	1	1	330,52
14	SPT	1	1	1	332,04

In deze transferlijn wordt besloten dat de **wachtrij-discipline** hier **geen** significante invloed heeft op de maximale winst per dag. De maximale **winst** bedraagt **€ 335,16** per dag en wordt bekomen door de tweede **buffercapaciteit** te zetten op **één**, de derde op **één** en de vierde op **één**. Deze optimale buffer allocatie is **evenredig** verdeelt over de buffers heen.

HOOFDSTUK V: CONCLUSIE

In dit afsluitende hoofdstuk worden de **conclusies** van deze eindverhandeling geformuleerd. In het eerste hoofdstuk werd het praktijkprobleem aangehaald en de centrale onderzoeksvraag geformuleerd, alsook de deelvragen. De **centrale onderzoeksvraag** was de volgende:

Op welke wijze dienen buffers toegewezen te worden om de winst te maximaliseren in transferlijnen met onbetrouwbare machines?

Om hier een antwoord op te vinden werd in het tweede hoofdstuk de simulatie-optimalisatie technieken besproken. Uit deze technieken werd de *Single Factor Method* gekozen om toe te passen. In het derde hoofdstuk werd buffer allocatie in transferlijnen besproken. Enerzijds werd dieper ingegaan op transferlijnen en anderzijds werden conventionele resultaten omtrent optimale buffer allocaties in transferlijnen vermeld. In het vierde hoofdstuk werd simulatie-optimalisatie bij transferlijnen besproken.

Ten eerste werd een simulatiemodel van een transferlijn gebouwd met behulp van *Arena*. Vervolgens werd de methodiek van de *Single Factor Method* uitgelegd aan de hand van dit model. Ten slotte volgde het eigenlijke onderzoek van deze eindverhandeling.

Voor de *ongebalanceerde* transferlijn die gebruikt werd om de methodiek van de *Single Factor Method* uit te leggen, kan gesteld worden dat de optimale buffer allocatie blijkt aan te sluiten bij de *geïnverteerde voorraad bowl*. Voor de transferlijn met *aflopende* gemiddelde bewerkingstijden kan gesteld worden dat de optimale buffer allocatie blijkt te *dalen* naar het einde van de transferlijn toe. Voor de transferlijn met *oplopende* gemiddelde bewerkingstijden kan gesteld worden dat de optimale buffer allocatie blijkt te *stijgen* naar het einde van de transferlijn toe. Voor de transferlijn met een *bottleneck* kan gesteld worden dat de optimale buffer allocatie een *evenredige* verdeling van de capaciteit blijkt in te houden. Voor de transferlijn met een *onbetrouwbare* bottleneck kan gesteld worden dat de optimale buffer allocatie eveneens in dit geval een *evenredige* verdeling van de capaciteit blijkt in te houden.

Met betrekking tot de *wachtrij-discipline* werd slechts in één transferlijn een significante invloed hiervan vastgesteld. In de overige transferlijnen had de wachtrij-discipline geen significante invloed op de maximale winst per dag. De transferlijn waar de wachtrij-discipline een significante rol vervulde, was de transferlijn met *aflopende* gemiddelde bewerkingstijden. De *Shortest Processing Time* werd geïdentificeerd als de optimale wachtrij-discipline.

Ten slotte wordt een antwoord gegeven op de centrale onderzoeksvraag. Dit antwoord is van toepassing op een transferlijn met één onbetrouwbare machine, die eveneens een bottleneck is. Het **antwoord** op de centrale onderzoeksvraag is de volgende:

De buffers dienen evenredig verdeeld te worden om de winst te maximaliseren in transferlijnen met onbetrouwbare machines.

Opgemerkt dient te worden dat verder onderzoek strikt noodzakelijk is om dit resultaat te verifiëren, aangezien gebruik gemaakt werd van slechts één transferlijn. Deze transferlijn had slechts één onbetrouwbare machine, namelijk de bottleneck.

LIJST VAN GERAADPLEEGDE WERKEN

Barton, R.R. en Ivey, J.S.Jr. (1996) Nelder-Mead Simplex Modifications for Simulation Optimization, *Management Science*, 42:7, 954-973

Batur, D. en Kim, S.H. (2006) Fully sequential selection procedures with a parabolic boundary, *IIE Transactions*, 38:9, 749-764

Buzacott, J.A. en Hanifin, L.E. (1978) Models of Automatic Transfer Lines with Inventory Banks A Review and Comparison, *AIIE Transactions*, 10:2, 197-207

Castro, I.A., Silva R.S.F., Tirapegui, J., Borsato, D., Bona, E. (2003) Simultaneous optimization of response variables in protein mixture formulation: constrained simplex method approach, *International Journal of Food Science and Technology*, 38:2, 103-110

Chen, M.C. en Tsai, D.M. (1996) Simulation optimization through direct search for multi-objective manufacturing systems, *Production Planning & Control*, 7:6, 554-565

Conway, R., Maxwell, W., McClain, J.O., Thomas, L.J. (1988) THE ROLE OF WORK-IN-PROCESS INVENTORY IN SERIAL PRODUCTION LINES, *Operations Research*, 36:2, 229-241

De Koster, M.B.M. (1989) *Capacity oriented analysis and design of production systems*, Springer, Berlin

Gelaesen, S. (2003) *Modellering van een lijn- productiesysteem met onzekerheid in de werking van de machines*, Limburgs Universitair Centrum, Diepenbeek

Glover, F. en Laguna, M. (1999) *Tabu Search*, Kluwer, Norwell

Goldberg, D. (1989) *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading

Hochberg, Y. en Tamhane, A.C. (1987) *Multiple Comparison Procedures*, Wiley, New York

Jafari, M.A. en Shanthikumar, J.G. (1989) Determination of Optimal Buffer Storage Capacities and Optimal Allocation in Multistage Automatic Transfer Lines, *IIE Transactions*, 21:2, 130-135

Kleijnen, J.P.C. en van Groenendaal, W.J.H. (1988) Simulatie: technieken en toepassingen, Academic Service, Schoonhoven

Lutz, C.M., Davis, K.R., Sun, M. (1998) Determining buffer location and size in production lines using tabu search, *European Journal of Operational Research*, 160:2/3, 301-316

Prave, D. (2000) *TRĪS MAŠĪNU UN DIVU BUFERU SISTĒMAS PIEEJAMĪBA*, Universiteit Antwerpen, Antwerpen

Schwefel, H.P. (1981) *Numerical Optimization of Computer Models*, Wiley, Chichester

Sheskin, T.J. (1976) Allocation of Interstage Storage Along an Automatic Production Line, *AIIE Transactions*, 8:1, 146-152

Smith, D.E. (1973) An empirical investigation of optimum-seeking in computer simulation situation, *Operations Research*, 21:2, 475-497

Somasundaram, P., Kuppusamy, K., Devi, R.P.K. (2004) EVOLUTIONARY PROGRAMMING BASED ECONOMIC DISPATCH WITH LINE FLOW CONSTRAINTS, *International Journal of Computational Engineering Science*, 5:1, 81-89

Stuckman, B.E. en Easom, E.E. (1992) A Comparison of Bayesian/Sampling Global Optimization Techniques, *IEEE Transactions on Systems, Man and Cybernetics*, 22:5, 1024-1032

Sullivan, D.W. en Wilson, J.R. (1989) RESTRICTED SUBSET SELECTION PROCEDURES FOR SIMULATION, *Operations Research*, 37:1, 52-71

Sullivan, K.A. en Jacobson, S.H. (2000) Ordinal Hill Climbing Algorithms for Discrete Manufacturing Process Design Optimization Problems, *Discrete Event Dynamic Systems: Theory and Applications*, 10:4, 307-324

Tekin, E. en Sabuncuoglu, I. (2004) Simulation optimization: A comprehensive review on theory and applications, *IIE Transactions*, 36:11, 1067-1081

van Laarhoven, P.J.M. en Aarts, E.H.L. (1987) *Simulated Annealing: Theory and Applications*, D. Reidel, Dordrecht

Van Oudheusden, D.L. en Janssens, G.K. (1994) Availability of three-machine two-buffer systems, *Belgian Journal of Operations Research, Statistics and Computer Science*, 34:2, 17-36

Wild, R. (1972) *Mass-production management: the design and operation of production flow-line systems*, Wiley, London

LIJST VAN FIGUREN

LIJST VAN TABELLEN

BIJLAGE: SIMULATIE-OUTPUT VAN ARENA

Simulatierun 1

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	1003.89	226,82	713.75	1162.38	0.00	2272.79

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	1053.71	226,87	763.60	1212.27	37.4252	2326.70

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	65.6409	14,07	47.6344	75.3557	0.00	146.00

Simulatierun 1

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	6.0844	0,08	6.0179	6.1899	0.02642330	13.6097
Hold Machine 2.Queue	2.3898	0,18	2.2381	2.5368	0.00032999	7.5742
Hold Machine 3.Queue	5.8768	0,06	5.8110	5.9471	0.00204075	16.7711
Seize Machine 1.Queue	992.90	226,87	702.68	1151.22	0.00	2258.98
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 4.Queue	0.00	0,00	0.00	0.00	0.00	0.00

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3715	0,01	0.3656	0.3784	0.00	1.0000
Hold Machine 2.Queue	0.06116978	0,00	0.05595285	0.06417987	0.00	1.0000
Hold Machine 3.Queue	0.2523	0,01	0.2477	0.2601	0.00	1.0000
Seize Machine 1.Queue	61.8521	14,07	43.8342	71.5612	0.00	142.00
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 4.Queue	0.00	0,00	0.00	0.00	0.00	0.00

Simulatierun 2

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	1001.90	226,64	712.12	1161.44	0.00	2269.00

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	1051.72	226,69	761.97	1211.33	37.4252	2322.92

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	65.5291	14,06	47.5392	75.3060	0.00	146.00

Simulatierun 2

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	6.0863	0,10	6.0388	6.2228	0.00021772	13.6909
Hold Machine 2.Queue	2.3958	0,17	2.2476	2.5403	0.00032999	7.5742
Hold Machine 3.Queue	5.8838	0,06	5.8289	5.9524	0.00204075	16.7711
Seize Machine 1.Queue	975.15	226,66	685.31	1134.39	0.00	2239.97
Seize Machine 2.Queue	15.8132	0,11	15.6781	15.9092	0.00	19.9979
Seize Machine 3.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 4.Queue	0.00	0,00	0.00	0.00	0.00	0.00

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3669	0,01	0.3596	0.3723	0.00	1.0000
Hold Machine 2.Queue	0.06154764	0,00	0.05650721	0.06459272	0.00	1.0000
Hold Machine 3.Queue	0.2527	0,01	0.2483	0.2604	0.00	1.0000
Seize Machine 1.Queue	60.7580	14,06	42.7561	70.5233	0.00	141.00
Seize Machine 2.Queue	0.9854	0,01	0.9775	0.9893	0.00	1.0000
Seize Machine 3.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 4.Queue	0.00	0,00	0.00	0.00	0.00	0.00

Simulatierun 3

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	766.48	213,50	504.04	899.28	0.00	1802.37

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	816.30	213,58	553.89	949.33	37.4252	1856.28

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	52.5173	13,76	35.5647	60.9640	0.00	121.00

Simulatierun 3

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	5.6229	0,12	5.5489	5.7831	0.00561873	13.6909
Hold Machine 2.Queue	2.4644	0,16	2.3059	2.6225	0.00072645	7.9308
Hold Machine 3.Queue	6.3948	0,06	6.3354	6.4585	0.00134314	16.7711
Seize Machine 1.Queue	747.12	213,32	484.70	879.46	0.00	1786.27
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	8.4659	0,75	7.6227	9.2111	0.00	19.9948
Seize Machine 4.Queue	0.00	0,00	0.00	0.00	0.00	0.00

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3485	0,01	0.3430	0.3572	0.00	1.0000
Hold Machine 2.Queue	0.02745295	0,00	0.02166354	0.03115145	0.00	1.0000
Hold Machine 3.Queue	0.3244	0,01	0.3161	0.3348	0.00	1.0000
Seize Machine 1.Queue	48.0675	13,74	31.1222	56.4771	0.00	117.00
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	0.5446	0,05	0.4910	0.5931	0.00	1.0000
Seize Machine 4.Queue	0.00	0,00	0.00	0.00	0.00	0.00

Simulatierun 4

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	728.91	218,05	459.92	884.86	0.00	1755.77

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	778.73	218,13	509.77	934.92	37.4252	1809.68

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	50.3412	14,12	32.8700	60.2187	0.00	118.00

Simulatierun 4

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	5.5258	0,13	5.4370	5.7032	0.00561873	13.5521
Hold Machine 2.Queue	2.3889	0,13	2.2903	2.5129	0.01352809	7.7428
Hold Machine 3.Queue	6.4485	0,09	6.3738	6.5532	0.00134314	16.7711
Seize Machine 1.Queue	699.54	218,28	429.35	853.89	0.00	1725.24
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	18.5368	1,84	16.4308	20.0670	0.00	39.7444
Seize Machine 4.Queue	0.00	0,00	0.00	0.00	0.00	0.00

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3440	0,01	0.3380	0.3534	0.00	1.0000
Hold Machine 2.Queue	0.02145656	0,01	0.01483892	0.02592579	0.00	1.0000
Hold Machine 3.Queue	0.3348	0,01	0.3266	0.3439	0.00	1.0000
Seize Machine 1.Queue	45.2228	14,13	27.6845	54.9998	0.00	113.00
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	1.1980	0,12	1.0646	1.2925	0.00	2.0000
Seize Machine 4.Queue	0.00	0,00	0.00	0.00	0.00	0.00

Simulatierun 5

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	717.83	223,05	446.18	884.01	0.00	1752.97

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	767.65	223,13	496.03	934.07	37.4252	1806.89

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.6967	14,43	32.0296	60.1747	0.00	118.00

Values Across All Replications

Simulatierun 5

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	5.4931	0,13	5.4011	5.6577	0.00561873	13.7794
Hold Machine 2.Queue	2.3529	0,09	2.2767	2.4460	0.01133112	7.5739
Hold Machine 3.Queue	6.4636	0,09	6.3920	6.5811	0.00134314	16.7711
Seize Machine 1.Queue	677.62	222,37	404.91	841.17	0.00	1706.22
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	29.4104	3,13	26.5066	31.9654	0.00	58.5309
Seize Machine 4.Queue	0.00	0,00	0.00	0.00	0.00	0.00

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3419	0,01	0.3357	0.3514	0.00	1.0000
Hold Machine 2.Queue	0.01882324	0,01	0.01182397	0.02353353	0.00	1.0000
Hold Machine 3.Queue	0.3385	0,01	0.3296	0.3481	0.00	1.0000
Seize Machine 1.Queue	43.8688	14,38	26.1456	54.1902	0.00	112.00
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	1.9036	0,20	1.7208	2.0593	0.00	3.0000
Seize Machine 4.Queue	0.00	0,00	0.00	0.00	0.00	0.00

Simulatierun 6

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	620.80	231,21	352.69	786.24	0.00	1535.76

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	670.62	231,31	402.54	836.29	37.4252	1589.68

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	44.0291	15,12	26.4372	54.6767	0.00	106.00

Simulatierun 6

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	5.3097	0,10	5.2443	5.4455	0.00561873	13.7794
Hold Machine 2.Queue	1.9223	1,34	0.00	2.5232	0.00	6.6832
Hold Machine 3.Queue	5.8998	0,36	5.4487	6.2322	0.00129280	16.7668
Seize Machine 1.Queue	593.59	232,28	322.68	755.97	0.00	1510.40
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	7.9386	3,21	3.5520	10.1916	0.00	39.7237
Seize Machine 4.Queue	11.2056	1,24	9.7178	12.4776	0.00	19.9990

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3329	0,01	0.3281	0.3405	0.00	1.0000
Hold Machine 2.Queue	0.00477530	0,00	0.00	0.00767738	0.00	1.0000
Hold Machine 3.Queue	0.1916	0,05	0.1274	0.2253	0.00	1.0000
Seize Machine 1.Queue	38.9710	15,19	21.1920	49.4252	0.00	102.00
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	0.5213	0,21	0.2334	0.6693	0.00	2.0000
Seize Machine 4.Queue	0.7358	0,08	0.6385	0.8158	0.00	1.0000

Simulatierun 7

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	617.64	230,48	352.61	782.34	0.00	1519.96

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	667.46	230,57	402.46	832.39	37.4252	1573.88

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	43.8575	15,09	26.4318	54.4779	0.00	105.00

Simulatierun 7

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	5.2932	0,11	5.2173	5.4409	0.00561873	13.5902
Hold Machine 2.Queue	1.9517	1,39	0.00	2.7800	0.00	7.6853
Hold Machine 3.Queue	5.9253	0,36	5.6572	6.3488	0.00412767	16.8586
Seize Machine 1.Queue	585.54	231,85	315.87	744.18	0.00	1494.60
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	5.7864	4,05	0.6802	8.6224	0.00	39.7444
Seize Machine 4.Queue	19.1539	4,00	13.7722	22.4269	0.00	39.7444

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3319	0,01	0.3281	0.3395	0.00	1.0000
Hold Machine 2.Queue	0.00357541	0,00	0.00	0.00590313	0.00	1.0000
Hold Machine 3.Queue	0.1351	0,06	0.04974648	0.1744	0.00	1.0000
Seize Machine 1.Queue	38.4737	15,18	20.7448	48.7048	0.00	100.00
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	0.3801	0,27	0.04469198	0.5663	0.00	2.0000
Seize Machine 4.Queue	1.2587	0,26	0.9049	1.4678	0.00	2.0000

Simulatierun 8

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	613.49	232,76	344.20	785.15	0.00	1532.08

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	663.31	232,86	394.05	835.20	37.4252	1586.00

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	43.5832	15,23	25.8989	54.6188	0.00	106.00

Simulatierun 8

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	5.2865	0,09	5.2106	5.4012	0.00021772	12.5705
Hold Machine 2.Queue	1.8957	1,17	0.2152	2.4307	0.00444048	7.1786
Hold Machine 3.Queue	5.9143	0,41	5.3879	6.2512	0.00129280	16.7668
Seize Machine 1.Queue	570.73	233,63	298.44	739.11	0.00	1490.99
Seize Machine 2.Queue	14.8356	0,17	14.6461	14.9831	0.00	19.9894
Seize Machine 3.Queue	8.5534	3,38	3.9746	11.0132	0.00	39.7237
Seize Machine 4.Queue	11.3389	1,19	9.9380	12.5843	0.00	19.9990

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3253	0,01	0.3204	0.3312	0.00	1.0000
Hold Machine 2.Queue	0.00544564	0,00	0.00002832	0.00861025	0.00	1.0000
Hold Machine 3.Queue	0.1974	0,05	0.1330	0.2318	0.00	1.0000
Seize Machine 1.Queue	37.4985	15,29	19.6148	48.3344	0.00	100.00
Seize Machine 2.Queue	0.9750	0,01	0.9652	0.9857	0.00	1.0000
Seize Machine 3.Queue	0.5620	0,22	0.2616	0.7238	0.00	2.0000
Seize Machine 4.Queue	0.7451	0,08	0.6540	0.8230	0.00	1.0000

Simulatierun 9

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	636.39	234,18	358.43	796.02	0.00	1565.77

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	686.21	234,27	408.28	846.07	37.4252	1619.69

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	44.9490	15,27	26.7818	55.2076	0.00	108.00

Simulatierun 9

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	5.3512	0,09	5.2979	5.4798	0.00561873	13.5521
Hold Machine 2.Queue	2.1575	0,28	1.8573	2.4097	0.00648337	7.6630
Hold Machine 3.Queue	5.8137	0,29	5.4418	6.0602	0.00129280	16.7668
Seize Machine 1.Queue	614.28	234,80	335.18	772.36	0.00	1540.41
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	3.6119	1,01	2.1641	4.0268	0.00	19.9929
Seize Machine 4.Queue	10.7086	0,92	9.5301	11.5754	0.00	19.9990

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3352	0,01	0.3301	0.3422	0.00	1.0000
Hold Machine 2.Queue	0.00793029	0,00	0.00267966	0.01168751	0.00	1.0000
Hold Machine 3.Queue	0.1678	0,04	0.1188	0.1932	0.00	1.0000
Seize Machine 1.Queue	40.2354	15,32	21.9869	50.3977	0.00	103.00
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	0.2366	0,07	0.1419	0.2643	0.00	1.0000
Seize Machine 4.Queue	0.7016	0,06	0.6250	0.7553	0.00	1.0000

Simulatierun 10

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	617.69	230,48	352.69	782.34	0.00	1519.96

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	667.51	230,57	402.54	832.39	37.4252	1573.88

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	43.8603	15,09	26.4372	54.4779	0.00	105.00

Simulatierun 10

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	5.2944	0,11	5.2175	5.4409	0.00561873	13.5902
Hold Machine 2.Queue	1.9206	1,37	0.00	2.6973	0.00	7.6853
Hold Machine 3.Queue	5.9987	0,41	5.4487	6.3180	0.00129280	16.7668
Seize Machine 1.Queue	585.57	231,91	315.84	744.18	0.00	1494.60
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	12.5253	6,85	3.5520	16.9162	0.00	58.5309
Seize Machine 4.Queue	11.3870	1,37	9.7178	12.7873	0.00	19.9990

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3319	0,01	0.3281	0.3395	0.00	1.0000
Hold Machine 2.Queue	0.00357560	0,00	0.00	0.00588897	0.00	1.0000
Hold Machine 3.Queue	0.2040	0,06	0.1274	0.2469	0.00	1.0000
Seize Machine 1.Queue	38.4753	15,18	20.7429	48.7047	0.00	100.00
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	0.8229	0,45	0.2334	1.1110	0.00	3.0000
Seize Machine 4.Queue	0.7483	0,09	0.6385	0.8369	0.00	1.0000

Simulatierun 11

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	618.50	234,92	350.24	781.21	0.00	14968.44

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	668.32	235,02	400.09	831.26	37.4252	15024.00

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	43.9423	15,41	26.2685	54.4569	0.00	104.00

Simulatierun 11

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	5.3094	0,11	5.2296	5.4247	0.00021772	13.2254
Hold Machine 2.Queue	1.9510	1,41	0.00	2.9030	0.00	7.7445
Hold Machine 3.Queue	5.9072	0,51	5.2013	6.2236	0.00204075	16.4916
Seize Machine 1.Queue	591.01	235,55	320.77	750.72	0.00	14959.78
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	7.9480	3,81	2.5303	9.6698	0.00	1279.28
Seize Machine 4.Queue	11.3878	0,96	10.1773	12.2563	0.00	19.9990

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3333	0,01	0.3285	0.3412	0.00	1.0000
Hold Machine 2.Queue	0.00492593	0,00	0.00	0.00784980	0.00	1.0000
Hold Machine 3.Queue	0.1978	0,04	0.1391	0.2210	0.00	1.0000
Seize Machine 1.Queue	38.8589	15,45	21.0610	49.1804	0.00	100.00
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	0.5225	0,25	0.1666	0.6336	0.00	2.0000
Seize Machine 4.Queue	0.7488	0,06	0.6702	0.8029	0.00	1.0000

Simulatierun 12

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	619.50	218,75	374.18	772.47	0.00	14731.20

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	669.32	218,85	424.03	822.52	37.4252	14773.54

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	43.9356	14,31	27.8170	53.7342	0.00	104.00

Simulatierun 12

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	5.3243	0,11	5.2444	5.4633	0.01118653	13.7687
Hold Machine 2.Queue	1.9528	0,93	0.7308	2.7109	0.04703905	7.4259
Hold Machine 3.Queue	6.0424	0,32	5.7247	6.4246	0.00344615	16.7668
Seize Machine 1.Queue	591.04	219,48	342.10	741.33	0.00	14690.14
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	8.6555	3,38	4.7845	11.7093	0.00	924.59
Seize Machine 4.Queue	11.4692	0,89	10.6469	12.2446	0.00	19.9990

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3332	0,01	0.3285	0.3423	0.00	1.0000
Hold Machine 2.Queue	0.00524181	0,00	0.00014426	0.00885129	0.00	1.0000
Hold Machine 3.Queue	0.2083	0,04	0.1695	0.2339	0.00	1.0000
Seize Machine 1.Queue	38.7971	14,36	22.4423	48.4302	0.00	101.00
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	0.5680	0,22	0.3148	0.7682	0.00	2.0000
Seize Machine 4.Queue	0.7529	0,06	0.7006	0.7999	0.00	1.0000

Simulatierun 13

Replications: 5 Time Units: Minutes

Entity

Time

VA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

NVA Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Wait Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	616.71	238,06	350.42	800.78	0.00	14406.98

Transfer Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	0.00	0,00	0.00	0.00	0.00	0.00

Other Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	49.8197	0,22	49.5965	50.0518	34.0530	66.8462

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	666.53	238,17	400.27	850.83	37.4252	14454.96

Other

Number In	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

Number Out	Average	Half Width	Minimum Average	Maximum Average
Product	1000.00	0,00	1000.00	1000.00

WIP	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Product	43.8058	15,57	26.2912	55.6272	0.00	106.00

Simulatierun 13

Replications: 5 Time Units: Minutes

Queue

Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	5.3168	0,10	5.2145	5.4030	0.00056344	13.5566
Hold Machine 2.Queue	2.3674	0,16	2.2556	2.5318	0.00801228	7.6630
Hold Machine 3.Queue	5.8857	0,31	5.5156	6.2137	0.00847340	16.7325
Seize Machine 1.Queue	588.91	238,91	319.18	768.17	0.00	14390.25
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	8.3243	3,74	4.3155	11.4318	0.00	1734.34
Seize Machine 4.Queue	11.3514	1,09	9.9493	12.3881	0.00	19.9990

Other

Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Hold Machine 1.Queue	0.3335	0,01	0.3297	0.3403	0.00	1.0000
Hold Machine 2.Queue	0.00498952	0,00	0.00149800	0.00836930	0.00	1.0000
Hold Machine 3.Queue	0.1955	0,04	0.1400	0.2302	0.00	1.0000
Seize Machine 1.Queue	38.7033	15,63	20.9654	50.2233	0.00	101.00
Seize Machine 2.Queue	0.00	0,00	0.00	0.00	0.00	0.00
Seize Machine 3.Queue	0.5469	0,24	0.2837	0.7474	0.00	2.0000
Seize Machine 4.Queue	0.7462	0,07	0.6541	0.8099	0.00	1.0000