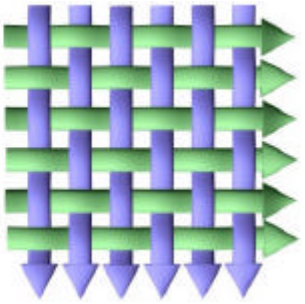
Sonderforschungsbereich 559
Modellierung großer Netze in der Logistik



Technical Report 03018

ISSN 1612-1376

Anwendung simulativer Aggregierung bei der Analyse eines Güterverkehrszentrums

Teilprojekt M1:
Carsten Tepper
Universität Dortmund
Informatik IV
August-Schmidt-Str. 12
44221 Dortmund

Dortmund, 20. Oktober 2003

1. Einleitung

Dieser Bericht dokumentiert die Anwendung des simulativen Aggregierungsinstrumentarium des Teilprojekts M1 an einem ausgewählten Anwendungsbeispiel, dass aus einer Kooperation mit dem Anwendungsteilprojekt A4 „Netze und Güterverkehrszentren" entstanden ist.

Eine weitverbreitete Analysetechnik stellt die Simulation dar. Die Simulation von grossen, detaillierten Systemen der Logistik kann sehr zeitaufwendig sein. Eine Effizienzsteigerung der Simulation (Verkürzung der Dauer eines Simulationslaufes und Reduzierung des benötigten Speicherplatzes) kann durch Aggregierung erzielt werden.

Bei der Aggregierung werden detaillierte Teilmodelle durch eine Voranalyse in einfache Ersatzdarstellungen überführt. Die Ersatzdarstellung, das sogenannte Aggregat, ist in der Regel nur dann exakt, wenn das Aggregat und die Umgebung, in die es eingesetzt wird, die Produktform-Eigenschaft (PF) erfüllen [BCMP75]. An allen anderen Fällen stellt das Aggregat eine Approximation des Teilmodells dar und es entsteht ein Aggregierungsfehler.

Die Voranalyse kann durch simulative, analytisch-algebraische und numerische Verfahren durchgeführt werden. Die Anwendung von nicht-simulativen Techniken zur Aggregierung wird in [AFK03] betrachtet.

In diesem Bericht wird Simulation zur Bestimmung der Ersatzdarstellung verwendet und das Aggregat ist vom Typ flussäquivalenter Bediener. Die Anwendung von Simulation hat den Vorteil gegenüber analytisch-algebraischen und numerischen Verfahren, dass das simulative Verfahren auf jedes Teilmodell anwendbar ist. Bei nicht simulativen Verfahren ist die Klasse der Teilmodelle beschränkt, da Teilmodelle auf andere Formalismen wie Warteschlangennetze [Fis00] und Petri-Netze [FKTW03, FiT03] abgebildet werden.

Als Modellwelt wird ProC/B [BBF02] verwendet. Die Modellwelt ProC/B, die auf der Prozessketten-Modellwelt nach Kuhn [Kuhn95] basiert, wurde um Elemente erweitert, die eine computergestützte Analyse von ProC/B-Modellen ermöglicht.

Die Aggregierung lässt sich in folgende 5 Phasen einteilen:

- Auswahl/Bestimmung der zu aggregierenden Subsysteme/Teilmodelle
- Wahl der Struktur des Aggregats
- Konstruktion und Parametrisierung des Aggregats
- Integration des Aggregats in das Gesamtmodell
- Analyse des aggregierten Simulationsmodells oder weitere Aggregierung (Multilevel-Aggregierung) durch Wiederholung der ersten vier Phasen

Damit ein komplexes Modell aggregiert werden kann, muss dieses in kleinere Teilmodelle zerlegt werden (Dekomposition).

Diese Zerlegung wird oft durch die hierarchische Struktur des komplexen Modells vorgegeben. Ein Teilmodell besitzt eine eindeutig definierte Schnittstelle zur Umgebung. Diese Schnittstellen werden in ProC/B Dienste genannt. Dienste werden von dem Teilmodell angeboten, und können von der Umgebung benutzt werden. Die interne Struktur des Teilmodells ist für die Umgebung nicht sichtbar. Die Umgebung kennt nur die Schnittstelle und merkt den Zeitverbrauch, den eine Dienstausführung benötigt.

Im Hinblick auf Aggregierung sollten die Teilmodelle disjunkt sein. Disjunkte Teilmodelle erfüllen die NCD-Eigenschaft (engl. NCD = nearly completely decomposable) [Cou77], d.h. die Interaktionen des Teilmodells mit seiner Umgebung sind möglichst gering. ProC/B-Modelle erfüllen diese Eigenschaft nur bedingt.

Weiter wird gefordert, dass keine Dienstaufrufe im Teilmodell terminieren bzw. im Teilmodell erzeugt werden. Wünschenswert wäre die Wiederverwendbarkeit von Teilmodellen, so dass ein Teilmodell mehrmals in einem Modell verwendet werden kann bzw. das Teilmodell in verschiedenen Modellen eingesetzt werden kann.

Bei der Aggregierung werden die disjunkten Teilmodelle durch eine Voranalyse zuerst separat betrachtet. Man versucht durch diese Voranalyse eine möglichst verhaltensäquivalente Ersatzdarstellung (Aggregat) des Teilmodells zu erhalten.

Ein Aggregat und ein Teilmodell besitzen dieselbe Schnittstelle und sind für eine Umgebung nicht zu unterscheiden. Das Aggregat besitzt eine einfachere interne Struktur als das Teilmodell und reduziert somit die Simulationsdauer, da die einfachere Struktur die Analyse durch weniger Ereignisse pro Dienstaufruf erleichtert.

Das Verhalten des Teilmodells und des Aggregats sowie der Zeitverbrauch zwischen Anfrage eines Dienstaufrufes und der Terminierung sollten annähernd gleich sein. Dieses kann nicht immer gewährleistet werden, so dass ein Aggregierungsfehler entsteht.

Der Bericht hat folgenden Aufbau:

In Kapitel 2 wird das Instrumentarium zur simulativen Aggregierung vorgestellt und anhand eines einfachen Beispieles eines M/M/1-Servers wird ein Konsistenz-Check des Instrumentariums durchgeführt.

Der Konsistenz-Check ist möglich, da die Umgebung PF-Eigenschaft besitzt und daher das Aggregat exakt sein sollte. Abweichungen können somit nur durch das Aggregierungsinstrumentarium entstehen sein.

Kapitel 3 wendet sich einem komplexeren, grösseren Modell zu. Es wird als Aggregattyp ein flussäquivalenter Bediener verwendet. Das Aggregat wird unter unterschiedlichen Lastsituationen betrachtet.

Kapitel 4 fasst die Ergebnisse noch einmal zusammen.

2. Instrumentarium zur Aggregierung im ProC/B Editor

In diesem Kapitel wird exemplarisch eine einfache Bedienstation vom Typ M/M/1-Server durch einen flussäquivalenten Bediener ersetzt. Die Spezifikation des Aggregats kann leicht validiert werden und ermöglicht somit einen Konsistenz-Check des eingesetzten Software-Instrumentariums, da das Aggregat bzgl. der mittleren Durchlaufzeit exakt sein sollte.

Ein M/M/1-Server besteht aus einer einzelnen Bedienstation und einer vorgeschalteten Queue mit unendlicher Kapazität. Der Ankunftsprozess ist ein Poisson-Prozess, d.h. die Zwischenankunftszeit Z ist negativ-exponentiell verteilt. Die Bedienzeit B des Servers ist ebenfalls eine negativ-exponentielle Verteilungsfunktion.

$$Z(t) = P(A \le t) = 1 - e^{-lt}, E[Z] = \frac{1}{l}$$

$$B(t) = P(B \le t) = 1 - e^{-mt}, E[B] = \frac{1}{m}$$

Der Parameter λ wird Ankunftsrate genannt. Mit λ wird die mittlere Anzahl ankommender Anforderungen pro Zeiteinheit angegeben. Analog wird der Parameter μ als Bedienrate bezeichnet.

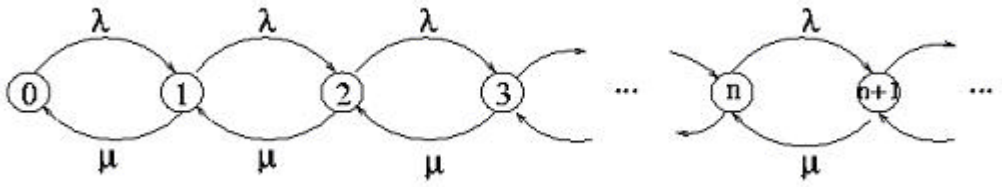


Abbildung 1: Zustandsübergangsdiagramm M/M/1-Server mit Bedienrate μ und poissonscher Ankunftsrate λ

Abbildung 1 zeigt das Zustandsübergangsdiagramm des M/M/1-Servers.
Die Durchlaufzeit, Population und Auslastung lassen sich analytisch bestimmen.
Die Auslastung ρ beträgt λ/μ und die Population ist ρ / (1 - ρ). Mit Hilfe des Gesetzes von Little ist die Durchlaufzeit T = Population / Durchsatz.

$$T = \frac{\frac{r}{(1-r)}}{l} = \frac{r}{(1-r)\cdot l} = \frac{\frac{l}{m}}{(1-\frac{l}{m})\cdot l} = \frac{1}{m-l}$$

In unserem Konsistenz-Check beträgt λ = 1.0 und μ = 10.0. Die Auslastung des Servers beträgt ρ = λ/μ =1.0/10.0 = 0.1 und ist kleiner 1.
Der Server wird unter stabilen Bedingung betrachtet und in diesem Fall ist der Durchsatz gleich der Ankunftszeit λ.

Abbildung 2 zeigt das ProC/B-Modell des M/M/1-Servers. Der M/M/1-Server wurde in einer Funktionseinheit (FE) modelliert, da im Instrumentarium zur Aggregierung FEs durch Aggregate ersetzt werden. Desweiteren können Durchlaufzeit, Population und Durchsatz nur an einer FE gemessen werden.

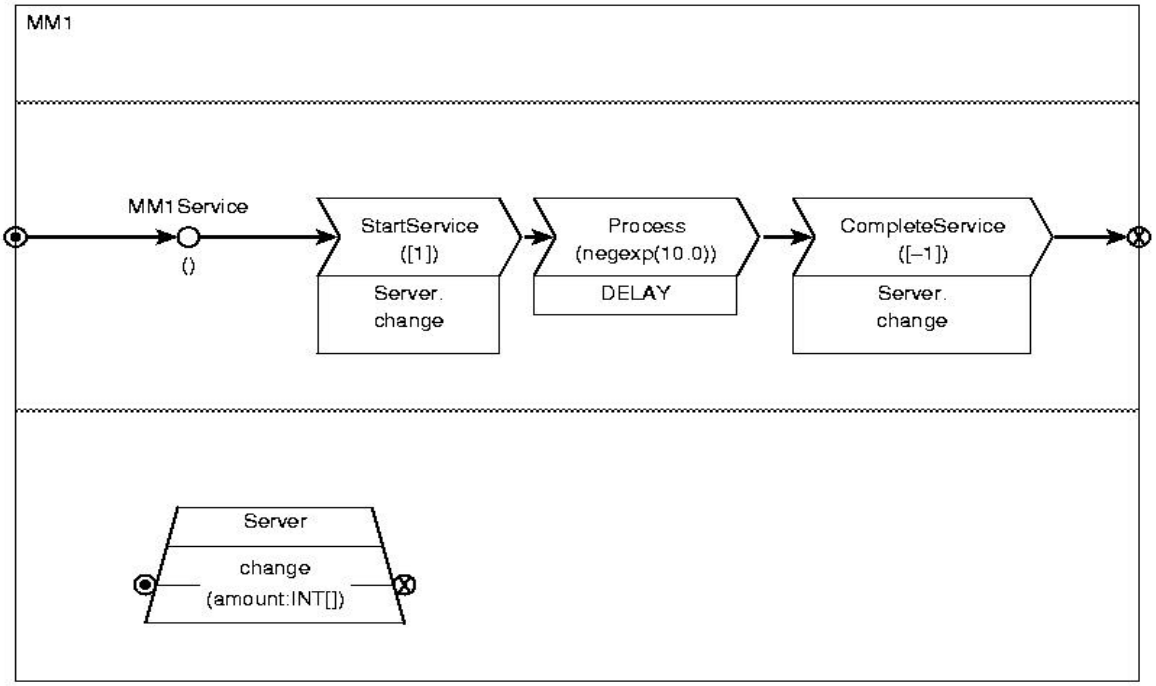


Abbildung 2: ProC/B Modell M/M/1-Server

Die Prozesskette der FE ‚MM1' besteht aus drei Prozesskettenelementen. Das erste und das dritte Element belegen bzw. geben die Bedienstation frei. Ob die Bedienstation belegt bzw. frei ist, wird über einen Counter festgestellt. Ein Counter ist eine passive Ressource. Der Delay-Baustein stellt die Bearbeitungszeit der Bedienstation dar.

Abbildung 3 zeigt die Umgebung, in der die FE ‚MM1' eingebettet ist. Die Quelle stellt den Ankunftsprozess dar.
Im nun folgenden wird beschrieben, wie aus der FE ‚MM1' durch Verwendung des Instrumentariums zur Aggregierung ein Aggregat erzeugt wird.

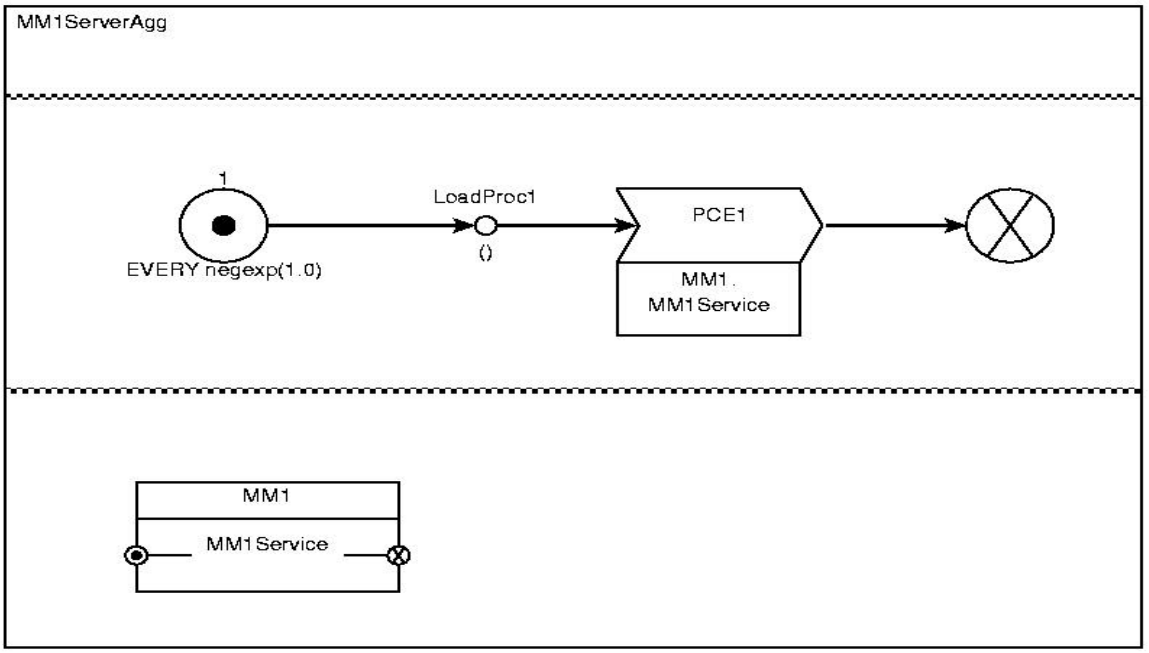


Abbildung 3: Umgebung des M/M/1-Server

Als Aggregattyp wählen wir einen flussäquivalenten Bediener [King90].
Dieser wird beschrieben durch einen Wert pro Populationsvektor und Dienst. Der Wert beschreibt die zustandsabhängige Bedienrate des flussäquivalenten Bedieners. Die zustandsabhängige Bedienrate *sk(n)* ist für einen Dienst *k* mit Population *n*. Die mittlere Bedienzeit wird aus dem Durchsatz des Submodells bei gegebener fester Population ermittelt.

Das Software-Instrumentarium erlaubt zwei Aggregatbeschreibungen. Die erste Variante ist die Beschreibung des Aggregats durch ProC/B-Elemente. Die zweite Variante nutzt die HIT-Aggregatbeschreibung (siehe Anhang A). Diese HIT-Aggregatbeschreibung kann durch das ProC/B-Element (siehe Abbildung 4) verwendet werden. Dieses ProC/B-Element liest die HIT-Aggregatbeschreibung ein und verwendet diese bei der Simulation durch HIT.

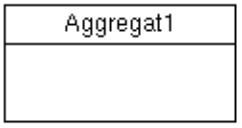


Abbildung 4: ProC/B-Element HIT-Aggregatbeschreibung

Im nun Folgenden werden beide Varianten beim Konsistenz-Check verwendet.
Für die erste Variante muss zuerst ein ProC/B-Modell für das Aggregat modelliert werden. Abbildung 5 zeigt das ProC/B-Modell eines flussäquivalenten Bedieners.
Das Aggregat besteht aus einem Server, der je nach Population unterschiedliche Bedienraten besitzt. Diese Bedienraten (Speed-Werte) werden von dem Instrumentarium zur Aggregierung berechnet und an das Schlüsselwort ″@@SPEED-VECTOR1″ wird durch den ermittelten Vektor von Speed-Werten

ersetzt. Der verwendete Server ist eine Bedienstation mit unendlich vielen Bedienern (SD-Server mit Bedienstrategie IS (Infinite Server)).

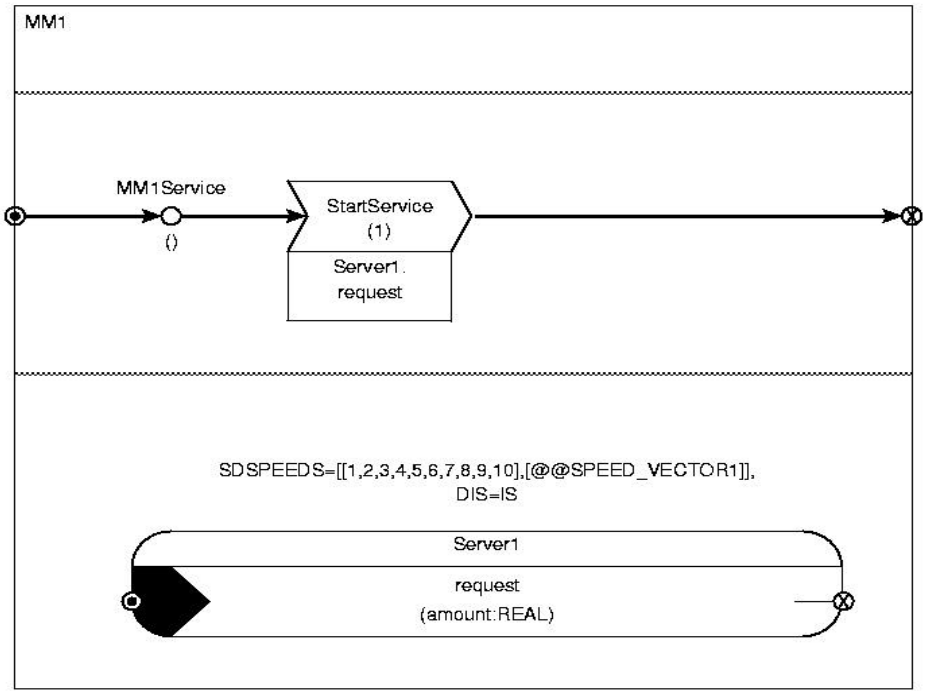


Abbildung 5: ProC/B Aggregatbeschreibung des M/M/1-Servers

Nachdem das Aggregat als ProC/B-Modell vorliegt, muss die Aggregierungstechnik gewählt werden. Wir verwenden simulative Aggregierung, d.h. die FE ‚MM1' wird ausgeschnitten und die virtuelle Senke und die virtuelle Quelle der FE werden kurzgeschlossen.
In unserem Beispiel werden jetzt 10 Simulation durchgeführt, da die Speed-Werte für die Population [1,2,3,4,5,6,7,8,9,10] berechnet werden sollen.
Zum Zeitpunkt t = 0 (Simulationsstart) wird pro Kunde jeweils ein Prozess erzeugt, d.h. im Falle Population 5 werden zum Simulationsstart 5 Prozesse erzeugt. Diese Prozesse verbleiben in der FE, da die virtuelle Senke und Quelle kurzgeschlossen sind.
Hierdurch wird gewährleistet, dass sich immer eine konstante Kundenzahl im System befindet und keine neuen Kunden von der Umgebung erzeugt werden.
Die Simulation läuft 50000 Zeiteinheiten (Modellzeit) und sie wird nur dann früher abgebrochen, wenn ein 95%-Konfidenzintervall (+/- 5.0) erreicht worden ist.

Im ProC/B-Editor wird die FE ‚MM1' selektiert und über den Menüpunkt „*Bearbeiten → Aggregieren von ...*" wird die Aggregierung gestartet. In einem dann erscheinenden Dialog (siehe Abbildung 6) wird der Aggregattyp, die Aggregierungstechnik, zum Beispiel „Simulative Aggregierung", sowie die maximale Population festgelegt.

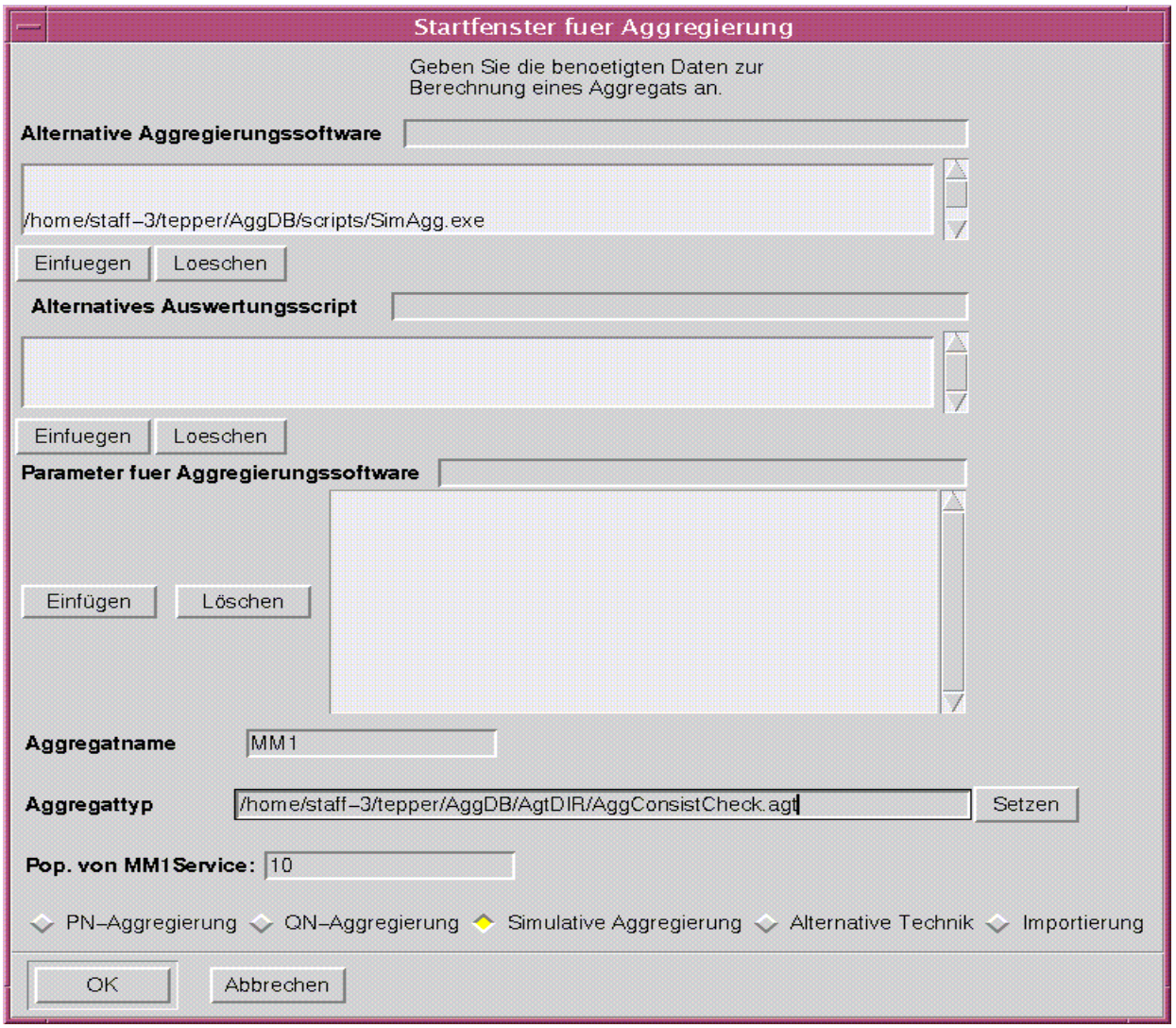


Abbildung 6: Einstellungsdialog des Instrumentariums zur Aggregierung

Wenn alle nötigen Einstellungen getroffen worden sind, startet man die simulative Aggregierung. In unserem Beispiel ist eine maximale Kundenanzahl (Population) von 10 eingestellt worden und da die FE ‚MM1' nur einen Dienst besitzt, werden nacheinander 10 Simulationsläufe gestartet.
Man erhält durch die Simulationsläufe die Durchsätze der FE ‚MM1' in Abhängigkeit von der Kundenanzahl (Population). Die Speed-Werte des flussäquivalenten Bedieners können aus den Durchsätzen ermittelt werden, in dem man die Durchsätze durch die Population teilt. Die ermittelten Durchsätze waren für alle Population 10.0. Tabelle 1 enthält die ermittelten Speed-Werte.

Population	1	2	3	4	5	6	7	8	9	10
Speed-Wert	10.0	5.0	3.33	2.5	2.0	1.67	1.43	1.25	1.11	1.0

Tabelle 1: Populationsabhängige Speed-Werte für flussäquivalenten Bediener

Nachdem die Speed-Werte ermittelt worden sind, wird die FE ‚MM1' durch die ProC/B Aggregatbeschreibung ersetzt. Das Schlüsselwort/Platzhalter „@@SPEED-VECTOR1" wird durch den Vektor der Speed-Werte ersetzt.

Es existiert nach dem Aggregierungsvorgang ein ProC/B-Modell mit und ohne Aggregat. Für beide Modelle werden nun die Durchlaufzeiten, Populationen und Durchsätze simulativ ermittelt.
In Tabelle 2 sind die Werte der Modelle mit Aggregat und ohne Aggregat aufgelistet. Beide Modelle liefern dieselben Werte bezüglich Durchlaufzeit, Population und Durchsatz.

Die zweite Variante, d.h. das ProC/B-Element der HIT-Aggregatbeschreibung, erhält man, in dem man im Dialog aus Abbildung 6 das Feld ‚Aggregattyp' leer lässt. Das Software-Instrumentarium lädt dann die HIT-Aggregatbeschreibung (siehe Beispiel Aggregatbeschreibung Anhang A) in das ProC/B-Element der HIT-Aggregatbeschreibung.

Fazit: Der Konsistenz-Check war erfolgreich. Das eingesetzte Instrumentarium zur Aggregierung funktioniert und kann nun an einem grösseren Modell angewendet werden.

	Durchlaufzeit	Population	Durchsatz
ohne Aggregat	0.1	0.1	1.0
mit Aggregat	0.1	0.1	1.0

Tabelle 2: Abschätzung der Aggregatgüte

3. Einsatz von Aggregierung bei der Analyse eines Güterverkehrszentrum

Dieses Kapitel stellt zunächst das zugrundeliegende Modell eines Güterverkehrszentrums (GVZ) vor, das aus einer Kooperation mit dem TP A4 entstanden ist. Dieses Modell wird in [DiV03] detailliert beschrieben, so dass hier nur ein Überblick gegeben wird.

Das GVZ-Modell beinhaltet eine Stückgutumschlaghalle (SUH), in dieser wird der Güterumschlag zwischen LKWs realisiert. In einer räumlichen davon entfernten Umschlaghalle für den kombinierten Verkehr Schiene/Strasse (KV-Umschlag). Der Gütertransport zwischen SUH und KV-Umschlag erfolgt per LKW.

Das Modell wurde mit dem Prozessketteneditor [AEF03, AFT01, BBF02] erstellt. Der Prozessketteneditor erlaubt die graphische Modellierung einer Prozessketten-Modellwelt, die auf der Prozesskettenwelt von Kuhn [Kuhn95] basiert. So ist eine Prozessketten-Modellwelt entstanden, die eine qualitative und quantitative rechnergestützte Analyse von Prozesskettenmodellen erlaubt.

Die Abbildung 7 zeigt die hierarchische Struktur des GVZ-Modells.

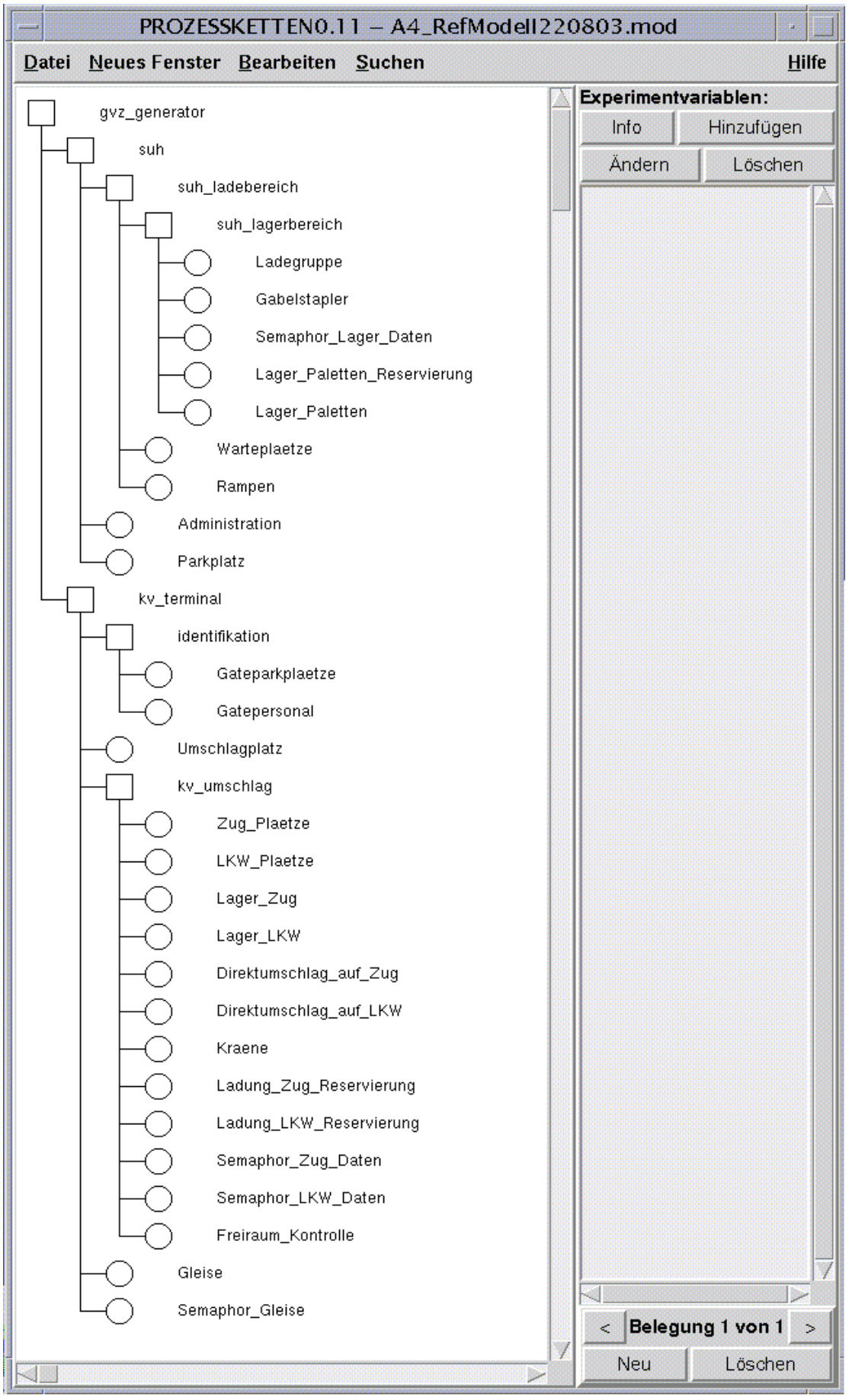


Abbildung 7: Hierarchische Struktur des GVZ-Modells

Die oberste Hierarchieebene (‚gvz_generator') des GVZ-Modells beschreibt die Belastung des Modells durch Spezifikation der Ankünfte der Verkehrsträger LKW und Zug sowie das Grobverhalten der Verkehrsträger innerhalb des GVZs. LKWs werden unterschieden, in LKWs die beladen bzw. entladen werden.

Der Prozess des reinen Umschlagens ist recht einfach und kann wie folgt beschrieben werden.

Im GVZ kommen LKWs unterschiedlicher Art an und wollen entweder Ware entladen und/oder beladen. Auf dem Parkplatz stellt der Fahrer den LKW ab und gibt seine Frachtpapiere an der dafür zuständigen Stelle ab. Dort wird seine Ent- bzw. Beladliste erstellt und mit dieser fährt der LKW die nächste freie Rampe an. Ist keine Rampe frei, muss der LKW so lange warten bis eine Rampe frei geworden ist. Hat der LKW eine Rampe angefahren, wird der LKW vom Entladepersonal unter Einsatz von Ladehilfsmittel ent- bzw. beladen. Nach Beendigung der Ent- bzw. Beladung setzt der LKW von der Rampe ab und verlässt das Betriebsgelände des GVZ.

Die hierarchische Struktur des GVZ-Modells besteht aus einer Toplevel FE ‚gvz_generator' und 6 weiteren Sub-FEs ‚suh', ‚suh_ladebereich', ‚suh_lagerbereich', ‚kv_terminal', ‚identifikation' und ‚kv_umschlag' (siehe Abbildung 7).

In der Toplevel FE ‚gvz_generator' werden zwischen 190 und 210 LKWs und ein Zug pro Tag (86400 Sekunden) erzeugt. Die LKWs werden durch eine Fallunterscheidung unterschieden in LKWs, die ihren Umschlag über das SUH und/oder KV-Terminal (KV-Umschlag) durchführen.

Zunächst wird für das Referenzmodell (Abbildung 7) simulativ die Population, der Durchsatz und die Durchlaufzeit ermittelt (siehe Tabelle 3). Die Simulation wird bis zu einer Modellzeit von 3 Monaten (7800000 Sekunden) durchgeführt. Die Modellzeit ist lang genug, so dass ein 95%-Konfidenzintervall (+/- 5.0) bei den zumessenden Kennzahlen (Population, Durchsatz und Durchlaufzeit) erreicht worden ist. Für die Berechnung wurde eine CPU-Zeit von 286,12 Sekunden benötigt. Die CPU-Zeit vom Referenzmodell wird mit der ermittelten CPU-Zeit der anderen Modelle mit Aggregat/Aggregaten verglichen. Wird die CPU-Zeit geringer, dann hat sich die Aggregierung positiv auf die Simulationsdauer ausgewirkt.

FE	Population	Durchsatz	Durchlaufzeit
kv_terminal	2,1652	0,0012	732,5224
suh	5,9633	0,0021	2831,0281

Tabelle 3: Population, Durchsatz und Durchlaufzeit des Referenzmodells

Es wird nun sukzessive eine FE nach der anderen durch ein Aggregat ersetzt, so dass am Ende nur noch ein Modell mit zwei Aggregaten für die beiden FEs ‚suh' und ‚kv_terminal' übrig bleibt. Als Aggregat wird immer das ProC/B-Element der HIT-Aggregatbeschreibung des flussäquivalenten Bedieners verwendet.

In folgenden Aggregierungsschritten werden die FEs aggregiert:

1. FE ‚identifikation'
2. FE ‚kv_umschlag'
3. FE ‚kv_terminal'
4. FE ‚suh_lagerbereich'
5. FE ‚suh_ladebereich'
6. FE ‚suh'

Nach jedem Schritt wird das entstandene Modell mit Aggregat/Aggregaten aus diesem Schritt mit einer Modellzeit von 3 Monaten simuliert. Verglichen werden das entstandene Modell dieses Schrittes mit dem Modell des vorherigen Schrittes bezüglich des Aggregierungsfehlers der Kennzahlen Population, Durchsatz und Durchlaufzeit. Zum anderen wird überprüft, ob sich die CPU-Zeit von Schritt zu Schritt verringert, d.h. ob sich der Simulationsaufwand verringert hat.

Tabelle 4 zeigt die ermittelten Populationen, Durchsätze und Durchlaufzeiten der entstandenen Modelle des jeweiligen Schrittes. In Klammern stehen die Aggregierungsfehler in Prozent des Modells dieses Schrittes im Vergleich zum Modell des vorherigen Schrittes.

FE	Schritt	Population	Durchsatz	Durchlaufzeit
kv_terminal	1	2,1575 (0,4%)	0,0012 (0%)	724,4647 (1,1%)
	2	1,9245 (10,8%)	0,0012 (0%)	674,2512 (6,9%)
	3	1,8976 (1,4%)	0,0012 (0%)	670,3420 (0,6%)
	4	1,8315 (3,5%)	0,0012 (0%)	619,7640 (7,5%)
	5	1,6854 (8,0%)	0,0012 (0%)	588,5426 (5,0%)
	6	1,6228 (3,7%)	0,0012 (0%)	582,2331 (1,1%)
suh	1	5,9702 (0,1%)	0,0021 (0%)	2857,2326 (0,9%)
	2	5,9823 (0,3%)	0,0021 (0%)	2859,9390 (1,0%)
	3	5,9959 (0,5%)	0,0021 (0%)	2858,4363 (0,96%)
	4	6,2484 (4,6%)	0,0021 (0%)	3007,0616 (5,9%)
	5	6,4431 (7,5%)	0,0021 (0%)	3087,7032 (8,3%)
	6	6,4713 (0,4%)	0,0021 (0%)	3093,2250 (0,2%)

Tabelle 4: Populationen, Durchsätze und Durchlaufzeiten der Modelle nach jedem Aggregierungsschritt

Durch Aggregierung der FE ‚identifiaktion' (Schritt 1) ist die CPU-Zeit, die für die Simulation des Modells aus Schritt 1, d.h. das Modell mit Aggregat für die FE ‚identifikation', annähernd gleich geblieben. Die benötigte CPU-Zeit beträgt 287,32 Sekunden, dieses ist ein Anstieg von 0,41%. Auch der Aggregierungs-fehler der Kennzahlen im Vergleich zum Referenzmodell ist gering und liegt maximal bei 1,1%.

Dieses war so zu erwarten, da die FE eine recht einfache Struktur besitzt. Diese FE ist somit kein geeigneter Aggregierungskandidat.

In Schritt 2 ist die Struktur der zu aggregierenden FE komplexer und somit wird eine CPU-Zeit von 258,20 Sekunden für die Simulation des Modells dieses Schrittes benötigt. Der Zeitvorteil beträgt 10,1%. Beim Vergleich der Kennzahlen der Modelle von Schritt 1 und 2 (Tabelle 4) erkennt man, dass ein maximaler Aggregierungsfehler von 10,8% entsteht. Dieser Aggregierungsfehler kann zum einen aufgrund der Überschreitung der Population der Simulation des Modells von Schritt 2 entstehen, da das Aggregat für eine maximale Population von 10 für jeden Dienst berechnet worden ist. Der Aggregierungsfehler kann verringert werden, in dem eine grössere maximale Population verwendet wird. Ein anderes Problem stellt sicherlich das Abbruchkriterium 95%-Konfidenzintervall (+/- 5.0) dar. Wählt man ein 98%-Konfidenzintervall (+/-2.0) wird das Aggregat exakter sein, aber die Berechnung dauert in diesem Fall länger.

Bei Schritt 3 verringert sich die CPU-Zeit von 258,20 Sekunden von Schritt 2 auf 255,48 Sekunden, also ein Zeitvorteil von 1%. Der Zeitvorteil ist so gering, da im Schritt 2 schon beide FEs ‚identifikation' und ‚kv_umschlag' aggregiert worden sind. Der Aggregierungsfehler beträgt auch nur maximal 1,4%.

Beim Aggregierungsschritt 4 der FE ‚suh_lagerbereich' tritt das Problem der Verklemmung (Deadlock) bei der Berechnung des Aggregats auf. Die Verklemmung entsteht, wenn kein LKW zum Entladen und nur ein LKW zum Beladen die SUH anfährt, d.h. bei einer maximalen Population des Dienstes ‚Entladen' von 0 und einer maximalen Population des Dienstes ‚Beladen' von 1. Der zu beladene LKW fährt eine Rampe an und will nur beladen, d.h. er hat keine Güter zum Entladen. Das Lager ist aber leer, da er der erste LKW ist. Somit kann er nicht beladen werden und er wartet nun auf einen LKW der Entladen wird. Doch dieser LKW kommt nicht und somit ist eine Verklemmung eingetreten. Diese Verklemmung tritt im Referenzmodell (Abbildung 7) nicht auf, da dort auf jeden Fall irgendwann ein LKW zum Entladen die SUH anfährt.

Zur Vermeidung von Verklemmungen bei der Aggregierung ist eine vorgeschaltete funktionale Analyse des Modells, mit dem das Aggregat berechnet wird, nötig. Die Tool-unterstützte Erkennung von Verklemmungen wird in [AFK03] betrachtet.

Die CPU-Zeit dieses Modell beträgt 123,08 Sekunden und somit ein Zeitvorteil von 51,8%. Der Aggregierungsfehler beträgt maximal 7,5%.

Im Schritt 5 wird noch einmal ein Zeitvorteil von 22,7% erzielt (CPU-Zeit 95,15 Sekunden). Der Aggregierungsfehler wird um maximal 8,3% grösser.

Schritt 6 ist der letzte Schritt. Nach diesem Schritt liegt nur noch ein Modell mit der Toplevel FE ‚gvz_generator' und zwei Aggregaten für die FEs ‚suh' und ‚kv_terminal' vor. Die CPU-Zeit verringert sich nur noch unwesentlich auf 93,35 Sekunden. Dieses entspricht einem Zeitvorteil von 1,9% im Vergleich zum Modell von Schritt 5. Der Aggregierungsfehler beträgt maximal 3,7%.

Zum Abschluss wird das Modell aus Schritt 6, also dem letzten Schritt, mit dem Referenzmodell verglichen. Der maximale Aggregierungsfehler beträgt 25%. Das Referenzmodell benötigte eine CPU-Zeit von 286,12 Sekunden und das Modell aus Schritt 6 nur 93,35 Sekunden. Dieses ist ein Zeitvorteil von 67,4%. Der Aggregierungsfehler beträgt für die FE ‚suh' maximal 8,5% und für die FE

‚kv_terminal' maximal 25%. Der Aggregierungsfehler ist für die FE ‚suh' deutlich geringer, somit ist diese FE besser geeignet zur Aggregierung.

Tabelle 5 listet noch einmal die gebrauchten CPU-Zeiten der einzelnen Modelle aus jedem Schritt auf und gibt den Zeitvorteil in Prozent von Schritt zu Schritt an.

Schritt	CPU-Zeit (Sekunden)	Zeitvorteil
Referenzmodell	286,12	-
1	287,32	0% (sogar Anstieg um 0,41%)
2	258,20	10,1%
3	255,48	1%
4	123,08	51,8%
5	95,15	22,7%
6	93,35	1,9%

Tabelle 5: CPU-Zeiten der Modelle nach jedem Aggregierungsschritt und Zeitvorteil von Schritt zu Schritt

4. Zusammenfassung

Simulative Aggregierung stellt eine Möglichkeit dar, den Simulationsaufwand von komplexen Modellen zu verringern. Durch eine Voranalyse wird für ein Submodell ein Aggregat berechnet, welches die Simulation beschleunigt. Einher mit der Verringerung des Simulationsaufwandes entsteht ein Aggregierungsfehler. Dieser kann je nach Detaillierungsgrad des zu aggregierenden Submodells gross werden.

Es wurde das Software-Instrumentarium zur Aggregierung von ProC/B-Modellen vorgestellt, welches das Experimentieren deutlich vereinfacht. Mit Hilfe eines einfachen Beispiels eines M/M/1-Server wurde ein Konsistenz-Check des Software-Instrumentariums durchgeführt. Dieser Test war positiv ausgefallen, so dass das Instrumentarium an einem detaillierten, komplexen Modell angewendet werden konnte. Dieses Modell eines Güterverkehrszentrums ist in einer Kooperation der Teilprojekte M1 und A4 entstanden.

Es wurden sukzessive die vorkommenden FEs durch Aggregate ersetzt. In 6 Aggregierungsschritten wurden alle FEs durch Aggregate ersetzt, so dass nur noch ein Modell mit einer Toplevel FE und 2 Aggregaten übrig geblieben ist. Nach jedem Aggregierungsschritt wurde für das entstandene Modell der Aggregierungsfehler der Kennzahlen Population, Durchsatz und Durchlaufzeit bezüglich des Modells des vorherigen Schrittes ermittelt. Des Weiteren wurden die CPU-Zeiten verglichen, ob sich der Simulationsaufwand von Schritt zu Schritt verringert hat.

Fazit: Simulative Aggregierung ermöglicht die Aggregierung jeder FE eines ProC/B-Modells im Gegensatz zu anderen Voranalysetechniken wie Warteschlangennetze und Petri-Netze, die erst das ProC/B-Modell in ein Warteschlangennetz bzw. Petri-Netz transformieren müssen. Eine Verringerung des

Simulationsaufwandes war feststellbar, aber einher geht eine Vergrösserung des Aggregierungsfehler. Der Aggregierungsfehler ist nicht vermeidbar, da das Aggregat immer eine Approximation der FE darstellt.

Anhang A

Die HIT-Aggregatbeschreibung startet mit 4 Zeilen Kommentar. In den Kommentarzeilen steht der Experimentname, Komponententypname und das Datum und die Uhrzeit der Berechnung des Aggregats. Als weitere Information wird die verwendete Voranalyse-Methode angegeben. Die Kommentare werden durch eine Leerzeile von der Komponenten-Beschreibung des Aggregats getrennt.

Die Tabelle der angebotenen Dienste wird durch das Schlüsselwort ‚%SPEEDS' eingeleitet. Dem Schlüsselwort ‚%SPEEDS' folgen D+1 Zeilen, die pro angebotenen Service D die maximale Population des Services angibt.

Zeile	Spalten 1-25	27-34	30-39	40-65
1	: NUMBER OF PROVIDES	<Integer>		
2	: <Provide name 1>		POPULATION	<Integer>
…	: …		...	...
D+1	: <Provide name D>		POPULATION	<Integer>

Zum Abschluss folgt die Liste der Speed-Werte.

Beispiel einer HIT-Aggregatbeschreibung (M/M/1-Server aus Kapitel 2):

```
 % Experiment Aggregatio
% Component Type F19MM1
% Used Method Simulative Aggregation
% Date 12-11-2003 Time 13:40

TYPE F19MM1 COMPONENT;
PROVIDE SERVICE
P24MM1Service;
END PROVIDE;

TYPE P24MM1Service SERVICE;
END TYPE P24MM1Service;

END TYPE F19MM1;

%SPEEDS
NUMBER OF PROVIDES              1
P24MM1Service                 POPULATION                          10
 1.003800E+001 4.989285E+000 3.341903E+000 2.475714E+000 1.974875E+000
 1.667143E+000 1.426952E+000 1.234531E+000 1.103926E+000 9.933333E-001
```

Literaturverzeichnis

[AEF03] Arns, M.; Eickhoff, M; Fischer, M; Tepper, C; Völker, M.: New Features in the ProC/B toolset. In: [Bau03c], S. 26-29, 2003.

[AFK03] Arns, M.; Fischer, M.; Kemper, P.: Anwendung nicht-simulativer Techniken zur Analyse eines dezentralen Güterverkehrszentrums. SFB 559-Bericht 03017, ISSN 1612-1376, 2003.

[AFT01] Arns, M.; Fischer, M.; Tatlitürk, H.; Tepper, C.; Völker, M.: Modeling and Analysis Framework of Logistic Process Chains. In. Proc. Of Joint Tool Session at PNPM/MMB/PAPM Conferences, pp. 56-61, Aachen, Germany, Sept. 2001.

[Bau03c] Bause, F.: Tools of the 2003 Illinois International Multiconference on Measurement, Modelling, and Evaluation of Computer-Communication Systems. Forschungsbericht Nr. 781 des Fachbereichs Informatik der Universität Dortmund, 2003.

[BBF02] Bause, F.; Beilner, H.; Fischer, M.; Kemper, P.; Völker, M.: The ProC/B Toolset for the Modelling and Analysis of Process Chains. In T. Field, P.G. Harrison, J. Bradley, U. Harder (eds.) Computer Performance Evaluation Modelling Techniques and Tools (Proc. Performance TOOLS 2002), Springer, LNCS 2324, pp. 51-70, 2002.

[BCMP75] Baskett, F.; Chandy, K.M.;Muntz R.R.; Palacios F.: Open, closed and mixed networks of queues with different classes of customers. Journal of the ACM, 22: 248-260, 1975.

[BSt89] Buchholz, P.; Stahl, H.: Aggregation Scenarios in the IMSE. IMSE Projekt Report R5.4-2 Version 2, 1989.

[BSt91] Buchholz, P.; Stahl, H.: On the Integration of New Aggregation Techniques in the IMSE. IMSE Projekt Report R5.4-7 Version 1, 1991.

[Cou77] Courtois, P. J. : Decomposability : Queueing and Computer System Application. Academic Press, 1977.

[DiV03] Dilling, C.; Völker, M.: Beispielmodellierung eines Güterverkehrszentrums im ProC/B-Paradigma. SFB 559-Bericht 03016, ISSN 1612-1376, 2003.

[Fis00] Die Abbildung von B1-Prozessketten zu separablen Warteschlangen-netzen. SFB Report 00009, 2000.

[FiT03] Fischer, M.; Tepper, C.: GSPNs to support aggregation in the ProC/B Toolset. In P. Kemper (ed.): Workshop on Stochastic Petri nets and related formalisms. Technical Report 780, Uni Dortmund, Fachbereich Informatik, 2003.

[FKTW03] Fischer, M.; Kemper, P.; Tepper, C.; Wu, Z.: Abbildung von ProC/B nach Petri-Netzen – Version 2. SFB 559-Bericht 03011, ISSN 1612-1376, 2003.

[Fin03] Finzel, J.: Toolunterstützung zur Aggregatbildung in Prozessketten-orientierten Modellwelten. Lehrstuhl IV, Universität Dortmund, 2002.

[Kan92] Kant, K.:Introduction to Computer System Performance Evaluation. McGraw-Hill, New York, 1992.

[King90] King, Peter J. B.: Computer and Communication Systems Performance Modelling. Prentice Hall International (UK), 1990.

[Kuhn95] Kuhn, A.; Prozessketten in der Logistik, Entwicklungstrends und Umsetzungsstrategien. Verlag Praxiswissen, Dortmund, 1995.

[Str89] Strüwer, M.:Simulative Aggregierung von Rechensystemmodellen: Konzepte, Techniken und Einsatz im Modellierungstool HIT. Lehrstuhl Informatik IV, Universität Dortmund, 1989.