PEPSY-QNS

Performance **E**valuation and **P**rediction **SY**stem
for **Q**ueueing **N**etwork**S**

G. Bolch, M. Kirschnick

Oktober 1992 TR-I4-21-92

PEPSY-QNS

Performance **E**valuation and **P**rediction **SY**stem for **Q**ueueing **N**etwork**S**

Ein Programmsystem zur Leistungsbewertung mit Hilfe
von Warteschlangennetzwerken unter Verwendung
analytischer Methoden und Simulation

G. Bolch, M. Kirschnick

Universität Erlangen Nürnberg
Institut für Mathematische Maschinen und Datenverarbeitung (IV)
Martensstraße 1
8520 Erlangen

bolch@immd4.informatik.uni-erlangen.de
kirschnick@immd4.informatik.uni-erlangen.de

Erste Ausgabe:	Oktober 1992
1. überarbeitete Fassung:	Januar 1993
2. überarbeitete Fassung:	Juni 1993

Übersicht

Im vorliegenden Bericht wird das am Lehrstuhl für Betriebssysteme (IMMD IV) der Universität Erlangen-Nürnberg in den letzten Jahren entwickelte Leistungsbewertungstool PEPSY-QNS (**P**erformance **E**valuation and **P**rediction **SY**stem for **Q**ueueing **N**etwork**S**) vorgestellt. Er stellt eine Fortsetzung des internen Berichts vom Oktober 1990 mit dem Titel “PEPSY - **P**erformance **E**valuation and **P**rediction **SY**stem” dar. Ursprünglich wurde PEPSY entwickelt, um verschiedene Methoden zur Analyse von Warteschlangennetzen unter einer einheitlichen und komfortablen Benutzerschnittstelle anwenden und vergleichen zu können. Inzwischen ist der Katalog der verfügbaren Verfahren sehr umfangreich, so daß der Schwerpunkt der Weiterentwicklung des Programmsystems bei der Anwendung der Warteschlangenanalyse in unterschiedlichen Bereichen liegen wird.

Nach einer kurzen Einführung in die Warteschlangenmodelle und deren Anwendbarkeit zur Modellierung und Leistungsbewertung von Rechensystemen wird der modulare Aufbau von PEPSY beschrieben. Die komfortable und sehr leicht erlernbare Benutzerschnittstelle wird übersichtlich dargestellt und an Beispielen verdeutlicht. Dabei wird auch kurz die graphische X-Windows Bedienungsoberfläche beschrieben. Eine Liste der verfügbaren Analyseverfahren und ein umfangreiches Literaturverzeichnis bilden den Abschluß.

Inhalt

1. Einleitung

1.1. Allgemeines

In einem Rechensystem wird im allgemeinen nicht nur ein Auftrag bearbeitet, sondern es befinden sich mehrere gleichzeitig laufende Progamme darin, welche um die Zuteilung der vorhandenen Betriebsmittel konkurrieren. Moderne Fertigungssysteme gliedern sich in viele Fertigungszellen, an denen eine Vielzahl von unterschiedlichen Werkstücken, nach verschiedenen Strategien zur Bearbeitung ansteht. Das Problem der Bewertung der Leistungsfähigkeit solcher Gesamtsysteme und ihrer einzelnen Funktionsgruppen wird dadurch so komplex, daß sich ein eigener Zweig der Informatik entwickelt hat, der sich mit der Ermittlung der Leistung von solchen Systemen befaßt.

Bei existierenden Systemen kann die Leistungsbestimmung durch Meßmonitore (Hardware- und Softwaremonitor) erfolgen. Diese Methode hat den Nachteil, sehr aufwendig zu sein. Ausserdem ist sie in der Entwurfs- und Entwicklungsphase nicht anwendbar.

Eine Alternative zur Messung stellt die Modellbildung dar. Durch Abstraktion werden die Vorgänge im Gesamtsystem soweit vereinfacht, daß die Analyse mit geringem Aufwand durchgeführt werden kann, die relevanten Merkmale des Systems aber erhalten bleiben.

Zur Modellbildung von Rechen- oder Fertigungssystemen eignen sich besonders Warteschlangennetzwerke. Die vorliegende Arbeit beschäftigt sich ausschließlich mit diesen.

1.2. Warteschlangennetzwerke

Warteschlangennetze bestehen aus mehreren unabhängigen Knoten, zwischen denen zu bearbeitende Aufträge weitergereicht werden. Die Aufträge können in Klassen unterteilt sein, die von den Bedienungssystemen unterschiedlich behandelt werden und die in ihrem Übergangsverhalten differieren. Durch diese Mechanismen versucht man, die in einem Rechner vorkommenden verschiedenen Auftragstypen zu berücksichtigen.

Die Bedienungssysteme (Knoten, Wartesysteme) sind aus einer Warteschlange und einer oder mehreren Bedienungsstationen zusammengesetzt. Ein eintreffender Auftrag, der keine freie Bedieneinheit vorfindet, wird in die Warteschlange eingereiht. Diese führt die wartenden Aufträge nach einer vorher festgelegten Abarbeitungsstrategie den freiwerdenden Bedienungsstationen zu. Ist ein Auftrag in einem Knoten abgearbeitet, so wird er mit einer bestimmten Wahrscheinlichkeit an einen anderen Knoten (eventuell auch erneut an sich selbst) weitergereicht oder verläßt das System.

Zur einheitlichen Beschreibung von Wartesystemen benutzt man die sogenannte Kendall'sche Notation:

A/B/m-Warteschlangendisziplin

Dabei bezeichnet A die Verteilung der Zwischenankunftszeiten, B die Verteilung der Bedienzeiten des Wartesystems und m die Anzahl der identischen Bedienungsstationen im Bedienungssystem $(m \geq 1)$.

Die im folgenden vorkommenden Verteilungen sind die Exponentialverteilung (Symbol M) und allgemeine Verteilungen (Symbol G). Als Warteschlangendisziplinen können First-Come-

First-Served (FCFS), Last-Come-First-Served (LCFS), Processor-Sharing (PS), Infinite-Server (IS) sowie verdrängende und nichtverdrängende Prioritätsdisziplinen (FCFS-PRE bzw. FCFS-NONPRE) vorkommen. Ein Sonderfall sind auch asymmetrische Bedienungssysteme, bei denen die Bedienungsstationen eines Bedienungssystemes unterschiedliche mittlere Bedienzeiten aufweisen (FCFS-ASYM)

Dabei werden die Typen M/M/m-FCFS, M/G/1-PS, M/G/∞–IS und M/G/1-LCFS-PRE auch als BCMP-Knoten oder Produktformknoten bezeichnet. Für Warteschlangensysteme, die nur solche Knoten enthalten (bei M/M/m-FCFS-Knoten müssen zusätzlich alle Auftragsklassen die gleiche Bedienrate besitzen), kann eine exakte Lösung in der sogenannten Produktform angegeben werden [BCMP75].

Die Angabe des Übergangsverhaltens der Aufträge jeder Klasse kann durch Übergangswahrscheinlichkeiten oder durch relative Besuchshäufigkeiten erfolgen. Dabei entsprechen die relativen Besuchshäufigkeiten einer Lösung des durch die Übergangsmatrix festgelegten Gleichungssystems.

Man unterscheidet offene, geschlossene und gemischte Warteschlangennetze. In offenen Netzen können in jedem Knoten Aufträge von außen eintreten bzw. das Netz verlassen. Bei geschlossenen Warteschlangennetzwerken zirkuliert eine feste Zahl von Aufträgen im Netz. Gemischte Netze enthalten sowohl geschlossene als auch offene Auftragsklassen

Zur Berechnung der Leistungsgrößen (wie Durchsatz, Auslastung und Verweilzeit) der so spezifizierten Warteschlangennetze sind verschiedene exakte und approximative Analyseverfahren entwickelt worden. Die wichtigsten dieser Methoden sind in [Bolch89] beschrieben; dieses Buch eignet sich auch zur Einführung in das Themengebiet “Warteschlangennetzwerke”.

1.3. Ziel von PEPSY-QNS

Um die Analyse von Warteschlangenmodellen mit verschiedenen Methoden unter einer einheitlichen und komfortablen Schnittstelle zu ermöglichen, ist am Lehrstuhl für Betriebssysteme des Instituts für mathematische Maschinen und Datenverarbeitung der Universität Erlangen-Nürnberg (IMMD IV) das Programmpaket PEPSY-QNS (***P**erformance **E**valuation and **P**rediction **SY**stem for **Q**ueueing **N**etwork**S***) entwickelt worden.
Der Aufbau und die Eigenschaften dieses Systems und der zugehörigen Programmteile werden in den nächsten Kapiteln genauer beschrieben.

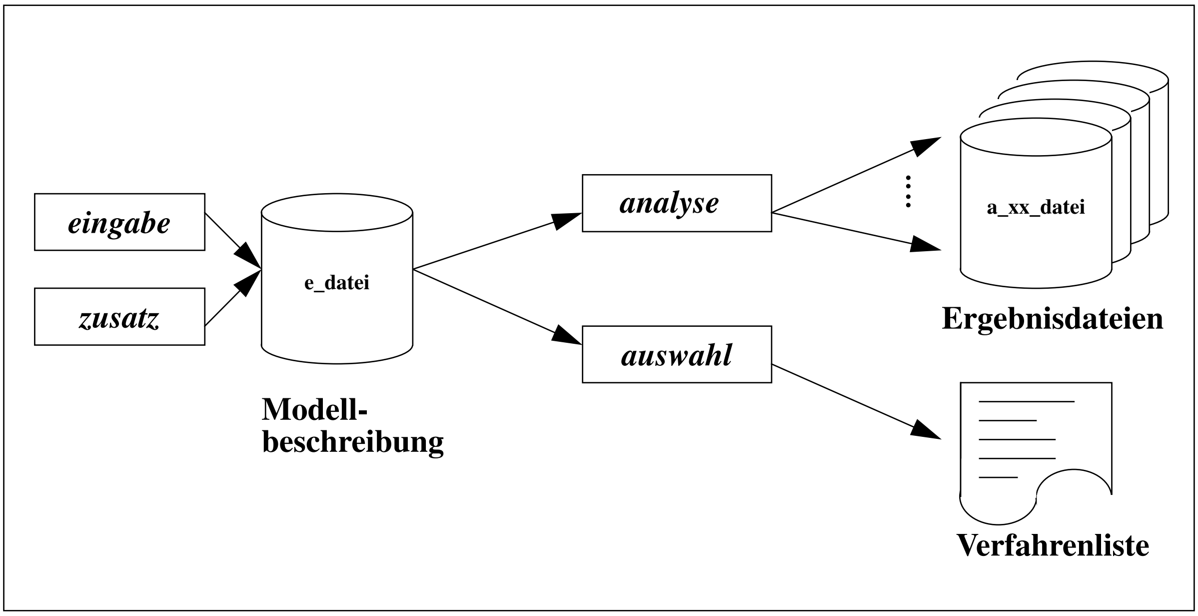
2. Das Programmpaket PEPSY-QNS

2.1. Aufbau von PEPSY-QNS

Dieser Abschnitt soll das Zusammenwirken der einzelnen Programme von PEPSY-QNS verdeutlichen. Auf die hier angesprochenen Programme und Dateien wird im Abschnitt 2.2 genauer eingegangen.

Die global gültige Eingabedatei für ein Warteschlangennetz wird mit dem Programm *eingabe* erstellt. Modifikationen sind durch das Programm *zusatz* möglich. Mit dem Aufruf von *auswahl* werden die auf das angegebene Netz anwendbaren Analyseverfahren angezeigt. Anschließend kann mit *analyse* die Berechnung der Leistungsgrößen gestartet werden. Dabei bleiben dem Benutzer der genaue Ablauf der Berechnung verborgen, so daß eine für alle Analyseverfahren gleiche Benutzersicht entsteht. Für jede Analyse wird pro Verfahren eine einheitliche Ausgabedatei abgelegt.

Abbildung 1: Programmaufbau des PEPSY-QNS



2.2. Benutzerschnittstelle

In diesem Abschnitt werden die für den Anwender relevanten Teile des Programmsystems PEPSY-QNS erläutert.

2.2.1. Dateien

Die vom PEPSY verwendeten Netzbeschreibungs-, Ergebnis- und Steuerdateien haben ein festgelegtes Format einzuhalten. Diese Dateien werden im folgenden kurz beschrieben.

Eingabedateien

Die PEPSY-Eingabedateien stellen die global gültige Beschreibung von Warteschlangennetzen im Programmpaket PEPSY-QNS dar. Sie haben Dateinamen der Form *e_name* und sind aufgeteilt in einen *Standardblock* und eventuell weitere *Zusatzblöcke* mit Informationen für spezielle Verfahren.

Die im Standardblock vorhandene Beschreibung enthält alles, um damit die meisten Analyseverfahren betreiben zu können. So enthält dieser die Anzahl der Knoten (Bedienungssysteme) und Auftragsklassen, die Bedienzeitverteilungen der einzelnen Knoten, das Übergangsverhalten der Aufträge (jobs) und die Beschreibung der Klassen.

Diese Dateien werden üblicherweise mit dem Programm *eingabe* erstellt, können aber auch durch andere Programme erzeugt oder mit einem üblichen ASCII-Editor eingegeben werden.

Die Eingabedatei darf - wie alle PEPSY-Dateien - beliebige Leerzeilen oder mit einen '#' beginnende Kommentarzeilen enthalten.

Im folgenden sind zwei Eingabedateien als Beispiel für die Syntax aufgeführt. In der Ersten werden Übergangswahrscheinlichkeiten verwendet. Die Angabe des Referenzknotens 'Aussen' in geschlossenen Klassen dient dabei der eindeutigen Berechnung der Besuchshäufigkeiten aus den Übergangswahrscheinlichkeiten.

```
# Eingabedatei Version  3
#
#
# Dateiname  beispiel
#


ANZAHL KNOTEN:  4
ANZAHL KLASSEN: 2

KNOTENBESCHREIBUNG

  Knoten |       Name           |      Typ
---------+---------------------+---------------------
     1   |       Knoten 1       |     M/M/1-FCFS
     2   |       Knoten 2       |     M/G/1-PS
     3   |       Knoten 3       |     M/G/1-PS
     4   |       Knoten 4       |     M/M/1-FCFS

KLASSENBESCHREIBUNG

  Klasse  | Ankunftsrate   Anzahl Auftraege
----------+---------------------------------
     1    |     -             7
     2    |    0.3            -

KLASSENSPEZIFISCHE KNOTENGROESSEN

KLASSE  1

     Knoten          |  Bedienrate   quad._Var.-Koeff.
--------------------+-----------------------------------
```

```
     Knoten 1        |  1        1
     Knoten 2        |  2        -1
     Knoten 3        |  1.5      -1
     Knoten 4        |  1        1


UEBERGANGSWAHRSCHEINLICHKEITEN

von/zu     | aussen    Knoten 1  Knoten 2  Knoten 3  Knoten 4
-----------+---------------------------------------------------
aussen     | 0.000000  1.000000  0.000000  0.000000  0.000000
Knoten1    | 0.000000  0.000000  0.333000  0.334000  0.333000
Knoten 2   | 1.000000  0.000000  0.000000  0.000000  0.000000
Knoten 3   | 1.000000  0.000000  0.000000  0.000000  0.000000
Knoten 4   | 1.000000  0.000000  0.000000  0.000000  0.000000

KLASSE  2

     Knoten          |  Bedienrate   quad._Var.-Koeff.
---------------------+----------------------------------
     Knoten 1        |  1        1
     Knoten 2        |  2        -1
     Knoten 3        |  2        -1
     Knoten 4        |  1        1


UEBERGANGSWAHRSCHEINLICHKEITEN

von/zu     | aussen    Knoten 1  Knoten 2  Knoten 3  Knoten 4
-----------+---------------------------------------------------
aussen     | 0.000000  1.000000  0.000000  0.000000  0.000000
Knoten 1   | 0.000000  0.000000  0.333000  0.500000  0.167000
Knoten 2   | 1.000000  0.000000  0.000000  0.000000  0.000000
Knoten 3   | 1.000000  0.000000  0.000000  0.000000  0.000000
Knoten 4   | 1.000000  0.000000  0.000000  0.000000  0.000000
```

Nun eine Eingabedatei mit Besuchshäufigkeiten.

```
# Eingabedatei Version  3
#
#
# Dateiname  e_besh
#


ANZAHL KNOTEN:  3
ANZAHL KLASSEN: 2
```

KNOTENBESCHREIBUNG

Knoten	Name	Typ
1	Knoten 1	M/M/1-FCFS
2	Knoten 2	M/G/1-PS
3	Knoten 3	M/G/0-IS

KLASSENBESCHREIBUNG

Klasse	Ankunftsrate	Anzahl Auftraege
1	-	2
2	0.5	-

KLASSENSPEZIFISCHE KNOTENGROESSEN

KLASSE 1

Knoten	Bedienrate	quad._Var.-Koeff.	Besuchshaeuf.
Knoten 1	1	1	1
Knoten 2	2	1	0.5
Knoten 3	3	1	0.4

KLASSE 2

Knoten	Bedienrate	quad._Var.-Koeff.	Besuchshaeuf.
Knoten 1	1	1	1
Knoten 2	3	1	0.6
Knoten 3	4	1	0.5

Zusatzblöcke

Zusatzblöcke dienen der Spezifizierung von Informationen, die in den Eingabeprogrammen *eingabe* und *xpepsy* nicht abgefragt werden. Sie werden an eine PEPSY-Eingabedatei angehängt. Ein Zusatzblock beginnt mit seinem Namen, der auch zum Auffinden desselben dient, da eine Eingabedatei durchaus mehrere Erweiterungen besitzen kann. Nach dem Namen folgen die eigentlichen Daten, wobei Leerzeilen und Kommentarzeilen ebenfallls zulässig sind.

Zur Zeit existieren folgende Zusatzblöcke:

- SIM für die beiden Simulationsverfahren *simulation* und *sim2*, in dem im wesentlichen die Verteilungsannahmen für die Bedienungszeiten von ./G/m-Bedienungssystemen angegeben werden.
- HIERARCHIE zur Einteilung eines Warteschlangennetzes in Unternetze für das Verfahren

epascat.

- HIER2 — für die hierarchische Modellierung mit dem Verfahren *hm*.
- MEM — zur Eingabe der Variationskoeffizienten der Ankunftsprozesse bei offenen Klassen. Dieser Zusatzblock wird von mehreren Analysemethoden benötigt.
- ASYM — In diesem Zusatzblock werden Mehrbedienerstationen mit asymmetrischen Bedienungsraten spezifiziert.
- GRAPHIK — XPEPSY speichert das Layout des Warteschlangennetzes ebenfalls in einem Zusatzblock ab.

Augabedateien

Für jeden Analysevorgang wird eine Analysedatei mit den berechneten Leistungsgrößen erzeugt. Sie wird *a_XX_name* genannt, wobei *name* derselbe ist wie der der Eingabedatei, so daß sich Lösungen zu einer Modellbeschreibung relativ einfach identifizieren lassen. Die angewendete Analysemethode wird mit *XX* - einer kurzen Buchstaben-Zahlenkombination - kodiert. Diese, bei der Vielzahl möglicher Verfahren, unübersichtliche Dateiliste läßt sich auch mit PEPSY-spezifischen Kommentaren ausgeben (Abschnitt 2.4.10 pls(1)).
Jede Ergebnisdatei besteht aus einem kurzen Kopf mit dem Namen des Modells und der Eingabedatei, sowie das Verfahren, mit dem die Ergebnisse berechnet worden sind. Dann folgen die wichtigsten Leistungsgrößen aller Knoten nach Klassen getrennt; anschließend die Gesamtleistungsgrößen der Bedienungssysteme über alle Klassen und die charakteristischen Netzgrößen für alle Klassen getrennt. Zur Erläuterung wird noch eine kurze Legende der Abkürzungen angehängt.

```
LEISTUNGSGROESSEN FUER NETZ:  besh

Die Netzbeschreibung ist in Datei 'e_besh' abgelegt.

Das gemischte Netz wurde mit dem Verfahren 'priomva2m'  analysiert.


Auftragsklasse 1

priomva2m | lambda    e    1/mue    rho     mvz     maa     mwz    mwsl
----------+----------------------------------------------------------
Knoten  1 | 0.488  1.000   1.000   0.488   3.659   1.784   2.659   1.296
Knoten  2 | 0.244  0.500   0.500   0.122   0.620   0.151   0.120   0.029
Knoten  3 | 0.195  0.400   0.333   0.000   0.333   0.065   0.000   0.000


Auftragsklasse 2

priomva2m|  lambda    e    1/mue     rho     mvz     maa     mwz    mwsl
---------+----------------------------------------------------------
Knoten 1 |  0.500  1.000   1.000   0.500   5.568   2.784   4.568   2.284
Knoten 2 |  0.300  0.600   0.333   0.100   0.426   0.128   0.093   0.028
Knoten 3 |  0.250  0.500   0.250   0.000   0.250   0.062   0.000   0.000
```

```
Gesamtleistungsgroessen:

priomva2m  |  lambda     rho     maa
-----------+-------------------------
Knoten  1  |   0.988   0.988   4.568
Knoten  2  |   0.544   0.222   0.279
Knoten  3  |   0.445   0.000   0.128


charakteristische Netzgroessen

priomva2m  |  lambda     mvz     maa
-----------+-------------------------
Klasse  1  |   0.488   4.102   2.000
Klasse  2  |   0.500   5.949   2.974



Legende

e   : Besuchshaeufigkeit                mue   : Bedienrate
rho : Auslastung                        lambda: Ankunftsrate
mvz : mittlere Verweilzeit der Auftraege
maa : mittlere Anzahl der Auftraege
mwz : mittlere Wartezeit der Auftraege
mwsl: mittlere Warteschlangenlaenge
```

Steuerdateien

PEPSY benötigt globale Steuerdateien, in welchen für alle Verfahren der Ablauf der Analyse und die Grenzen der Methode eingetragen sind.

Die Dateien *grenzen.txt* und *grenzenall.txt* im Verzeichnis $PEPSYHOME/infos enthalten die Beschränkungen der Verfahren, nämlich der Netztyp, die maximalen für Knoten- und Klassenanzahlen, die für das jeweilige Analyseverfahren erlaubten Knotentypen, ob ein Verfahren mit Übergangswahrscheinlichkeiten oder Besuchshäufigkeiten arbeiten kann, ob ein - und wenn ja und welcher - Zusatzblock benötigt wird, ob die Bedienraten bei M/M/m-FCFS-Knoten in verschiedenen Klassen gleich sein müssen, welche Variationskoeffizienten der Bedienzeitverteilungen erlaubt sind und wieviele Bedieneinheiten pro Bedienungssystem höchstens vorhanden sein dürfen.
Dabei enthält *grenzen.txt* nur ausgewählte Verfahren, die zum Beispiel besonders schnell sind oder gute Ergebnisse liefern, während *grenzenall.txt* alle in PEPSY vorhandenen Analyseprogramme beinhaltet. Außerdem sind in diesen Dateien die in PEPSY verfügbaren Knoten- und Netztypen als Kommentarzeilen aufgelistet.

Diese Datensammlung wird in erster Linie verwendet, um zu prüfen, ob ein Verfahren auf eine Eingabedatei anwendbar ist.

In der Steuerdatei *code.txt* sind Informationen über den Ablauf der Analyse eingetragen. So sind für jedes Verfahren die Verfahrenskodierung, eventuelle Steuerflags, Pre- und Postfilterinformationen enthalten. Die Verfahrenskodierung dient der Benennung der Ausgabedateien, die von der Form *a_XX_name* sind. Dabei ist *XX* die eindeutige Verfahren-Iinfixkodierung. Steuerflags haben die Aufgabe, Analyseprogramme in ihrem Ablauf zu beeinflussen. So können z.B. in einem Programm mehrere Varianten vorhanden sein, die durch eine entsprechende Option angewählt werden. In PEPSY erscheinen diese Varianten als getrennte Analysemethoden.

Eine Auflistung dieser Steuerdateien ist aufgrund ihrer Größe an dieser Stelle nicht möglich.

2.3. Verfügbare Analyseverfahren

Dieser Abschnitt soll einen kurzen Überblick über die in PEPSY-QNS verfügbaren Analyseprogramme geben. Eine Validierung und ein Vergleich aller zur Verfügung stehenden Verfahren ist von [Brag91] durchgeführt worden. In [Sacko92] findet sich eine kurze Beschreibung aller Analysealgorithmen, on-line können Informationen und Kurzbeschreibungen mit dem pls(1) Befehl (s. Abschnitt 2.4.10) abgerufen werden.

Tabelle 1: Liste der in PEPSY-QNS verfügbaren Analyseverfahren

Verfahren	Netztyp	Infix	Knotentypen	Bemerkungen	Literatur
scat	c	a	BCMP	app., b	Bard79, Schweitzer79, Neuse81, Chandy82, Feldman83
linearizer	c	b	BCMP	app., b	
marie	c	c	1-5,7,9	app., k	Marie79, Marie80, *Scheller84*
openpfn	o	p	1,3,4,9	b	Jackson53, Jackson63, BCMP75, *Schmuck87*
sopenpfn	o	p1	BCMP	b, k,	
mvamix	c,o,m	m	1,3,4,9	b	Akyildiz83, *Schmuck87*
bounds	c,o,	h	1,3,4,9	app., k	Lazowska84, *Schmuck87*
kuehn	o	k	1,5,7	app., k	Kühn79, Lepp88
simulation	c,o,m	s	1-19	app.	Schmidt84a, Schmidt84b, Götz88
mmva	c	s2	BCMP	b	Reiser80, *Hahn88*
ammva	c	n2	BCMP	app.	
mem	c,o	l	1,5,7	app.	Lavenberg80, Kouvatsos88b, *Flemming88*
evlinp	c	i	1,3,4,9	app., b, m	Feldmann83, Koschinski85, *Hebeler89*
appscat	c	o	1-6,9,12,13	app., m	Bard79, Schweitzer79, Neuse81, Agrawal84, *Förster89*
priomva2c	c	f	BCMP,10,11	app. ,b	Chandy83, *Niesser89*
priomva2m	c,o,m	g1	1,3,4,9-11	app., b	
faltung	c	e	BCMP	b	Muntz74, Wong75, *Niesser89*
recal	c	d	BCMP	b	Conway86, *Niesser89*
epascat	c	a1	BCMP	app., b	Lazowska84, *Hüsing89*
hm	c	na3	1-19	app.	Lazowska84, *Übler90*
cmva	c	na2	BCMP	b	Lazowska84, Reiser80, *Übler90*
num	c	na1e	1-8,10-19	app.	Sauer81, *UÜbler90*
multisum	c	j1	1-11,14-19	app.	Hütter89, Bolch89, *Hahn90*
monosum	c	j2	1-11,14-19	app., k	

Verfahren	**Netztyp**	**Infix**	**Knotentypen**	**Bemerkungen**	**Literatur**
sum	c	j	1-9	app., k	Bolch89
epf	c	t	2,4,6	app., k	Shum77, *Hütter90*
ermem	o	t1	1,3-5,7,9	app.	Kouvatsos88b, *Hütter90*
pujolle	o	t2	1-9	app.	Pujolle86, *Hütter90*
chylla	o	t3	1-9	app.	Chylla86, *Hütter90*
gelenbe	o	t4	1,3-5,7,9	app.	Gelenbe87, *Hütter90*
whitt	o	t5	1-9	app., k	Whitt83, *Hütter90*
sim2	c,o,m	s2	alle	app.	*Kirschnick90*
diffprox	o	x	1,5,7	app.	Kobayashi74, Reiser74
abol	c	oe	1 bis 6, 9, 12, 13, 20, 21	app.	*Brag91*
ascat	c	od		app.	
amonosum	c	j3	1-11, 14-21	app.	
asymmva	c	n3	BCMP, 20, 21	app.	
bardschweitzer	c	oa	1, 3, 4, 9	app.	
bol_aky	c	o1	1 bis 6,9, 12,13	app.	*Förster89*
chan_neu	c	o0		app.	
rtp	c	o8		app.	
sum_chylla	c	jc	1,2 bis 8	app.	*Brag91*
sum_mem	c	jm		app.	
sum_pujolle	c	jp		app.	
sum_whitt	c	jw		app.	

Anmerkungen:

- Erscheint der Name des Verfahrens kursiv, so handelt es sich um ein exaktes Lösungsverfahren.
- Als Netztypen werden verwendet: c - für geschlossene, o - für offene und m - für gemischte Netzwerke.
- Die zu den angegebenen Nummern gehörigen Knotentypen sind folgender Tabelle zu entnehmen:

Nr.	Typ	Nr.	Typ	Nr.	Typ
1	M/M/1-FCFS	9	M/G/1-LCFS-PRE	17	M/M/m-FCFS-NONPRE
2	M/M/m-FCFS	10	M/M/1-FCFS-PRE	18	M/G/m-FCFS-PRE
3	M/G/1-PS	11	M/M/1-FCFS-NONPRE	19	M/G/m-FCFS-NONPRE
4	M/G/∞-IS	12	M/G/m-PS	20	M/M/m-FCFS-ASYM
5	M/G/1-FCFS	13	G/G/m-PS	21	M/G/m-FCFS-ASYM
6	M/G/m-FCFS	14	M/G/1-FCFS-PRE	BCMP: 1,2,3,4,9	
7	G/G/1-FCFS	15	M/G/1-FCFS-NONPRE		
8	G/G/m-FCFS	16	M/M/m-FCFS-PRE		

- In der Spalte Bemerkungen gelten folgende Abkürzungen:
  app. = approximatives Analyseverfahren,
  k = Analysemethode für Netzwerke mit maximal einer Auftragsklasse,
  b = die Bedienraten der M/M/m-Knoten müssen ins allen Klassen identisch sein
- Bei den Literaturreferenzen sind Studien- oder Diplomarbeiten, die die Implementierung des Verfahren beschreiben, kursiv hervorgehoben.

2.4. Die Programme des PEPSY-QNS

Im folgenden sollen die wichtigsten Programme des Programmsystems PEPSY-QNS kurz vorgestellt werden. Für eine genaue Beschreibung der Aufrufsyntax und der Parameter wird auf die On-Line-Manuals verwiesen, die teilweise auch im Anhang abgedruckt sind.

2.4.1. eingabe

Das Programm *eingabe* dient der interaktiven Eingabe des Warteschlangennetzwerkes und liefert eine Eingabedatei für die Analyse mithilfe des Programms *analyse*.

Die Eingabe der Werte erfolgt durch Bildschirmmasken, die mit der *curses*-Bibliothek erstellt worden sind. Daher müssen die Environment-Variablen TERM, LINES und COLUMNS richtig gesetzt sein.
Einzugebende Zahlenwerte dürfen auch als Formeln angegeben werden, die die Grundrechenarten, Klammerung und Potenzierung enthalten.

Um die erzeugte Modellbeschreibung als globale PEPSY-Eingabedatei zu kennzeichnen, wird vor den Dateinamen automatisch ein ‘e_’ vorangestellt. Bei allen Aufrufen, die die Angabe einer solchen Modellbeschreibung benötigen, kann dieser Prefix weggelassen werden.

2.4.2. zusatz

Zusätzlich zur Standard-Beschreibung des Netzwerkes ist für einige Analysemethoden eine genauere oder erweiterte Spezifikation des Warteschlangennetzes notwendig. Dies wird bei dem Programm *auswahl* angezeigt. Die notwendigen Informationen werden durch das Programm *zusatz* interaktiv in einem Maskensystem abgefragt und der Modellbeschreibung als Zusatzblock angefügt. Beim Aufruf des Programmes sind das Verfahren, das den Zusatzblock benötigt, sowie der Name des Modells anzugeben.

2.4.3. auswahl

Nach der Eingabe des Warteschlangenmodells kann mit dem Kommando *auswahl* eine Liste der auf die Modellbeschreibung anwendbaren Analyseverfahren erstellt werden.
Die Ausgabeliste besteht aus zwei Spalten. Auf der linken Seite werden die direkt anwendbaren Verfahren aufgeführt, und auf der rechten Seite die Verfahren, die noch einen Zusatzblock für die erweiterte Netzbeschreibung benötigen.

Außer den anwendbaren Verfahren können auch die nicht anwendbaren Methoden mit einem kurzen Hinweis auf den Hinderungsgrund ausgegeben werden (Option -f).

Im allgemeinen werden nicht alle in PEPSY-QNS verfügbaren Verfahren untersucht, sondern nur eine Untermenge von schnellen oder genauen Analyseverfahren verwendet. Die Suche nach anwendbaren Verfahren kann durch die Option ‘-a’ auf alle existierdenden ausgedehnt werden.

2.4.4. pinfo

Pinfo dient zur Ausgabe von Informationen zu den implementierten Analyseverfahren. So können eine kurze Verfahrensbeschreibung, Literatur- und Implementierungshinweise sowie Referenzen auf verwandte oder ähnliche Verfahren angezeigt werden. Auch können die internen Daten über die Grenzen der Anwendbarkeit abgerufen werden; so werden u.a. die erlaubten Netz- und Knotentypen ausgegeben.

2.4.5. analyse

Analyse stellt das zentrale Programm des PEPSY-Systems dar, mit ihm wird die Leistungsgrössenberechnung angestoßen. Als Parameter werden unter anderen das Verfahren und der Name der Modellbeschreibung erwartet.

Beim Aufruf der Analyse werden als erstes die Modifikationszeiten des Analyseverfahrens, der Modellbeschreibung und ggfs. auch der vorhandenen Ergebnisdateien miteinander verglichen. So wird sichergestellt, daß die Analyseergebnisse auch wirklich die gewünschten sind. Sollten schon gültige Ergebnisse vorliegen, so werden diese auf dem Bildschirm angezeigt.
Ist eine Analyse notwendig, so wird die Eingabedatei von einem Eingabefilter in ein internes Modellbeschreibungsformat umgesetzt, das Berechnungsverfahren wird angestoßen und die Ergebnisse durch einen Postfilter in ein “lesbares” Format gebracht. Diese werden abschließend sowohl auf dem Bildschirm angezeigt als auch in einer Ergebnisdatei abgelegt.

Um eine versehentliche Ausführung einer Analyse mit einem nicht anwendbaren Verfahren zu verhindern, ruft *analyse* die Auswahl implizit auf. Dies kann mit der Option ‘-f’ umgangen werden. Ebenso wie *auswahl* verfügt *analyse* über die Option ‘-a’.
Es ist ebenfalls möglich, den einzelnen an dem Analysevorgang beteiligten Programmen weitere Parameter mitzugeben. Für weitere Hinweise sei an dieser Stelle auf das On-Line-Manual verwiesen.

2.4.6. pdiff

Das Kommando *pdiff* dient dem Vergleich der Ergebnisse zweier Analyseverfahren. Zu diesem Zweck werden die beiden Verfahren und der Name der Eingabedatei angegeben. *pdiff* ruft zunächst für jedes Verfahren *analyse* auf, um sicherzustellen, daß eventuell Modifikationen an der Netzbeschreibung erkannt werden und so gegebenenfalls korrekte Leistungsgrößen ermittelt werden. Dann werden die Ergebnisdaten mit dem UNIX-Standardprogramm *diff*(1) verglichen und ausgegeben.
Ein grafischer Vergleich von Analyseergebnissen ist mit dem Programm *XGad* möglich.

2.4.7. transform

Die Eingabe einer Modellbeschreibung verlangt entweder Übergangswahrscheinlichkeiten oder Besuchshäufigkeiten. Wurden Besuchshäufigkeiten angegeben, sind einzelne Analysemethoden nicht anwendbar. Mit *transform* ist es möglich, eine Netzbeschreibung mit Übergangswahrscheinlichkeiten zu erzeugen. Da die Umformung nicht unbedingt eindeutig ist, sind verschiedene Netzstrukturierungen erlaubt: Serien- und Parallelschaltung der Bedienungssysteme sowie ein Central-Server-Modell stehen zur Verfügung.

2.4.8. pcheck

Durch die getrennte Eingabe von Standardbeschreibung und Zusatzblöcken mit den Programmen *eingabe* bzw. *zusatz* sind Inkonsistenzen in der Modellbeschreibung möglich. Mit *pcheck* kann eine e_-Datei auf Konsistenz geprüft werden, so daß alle Verfahren - insbesondere diejenigen, die die Zusatzinformationen nicht verwenden! - eine identische Modellbeschreibung vorfinden und somit deren Ergebnisse auch vergleichbar sind.

2.4.9. pedit

Das Kommando *pedit* ist ein Aufruf des Editors vi(1). Dem angegebenen Modellnamen wird ein ‘e_’ vorangestellt, so daß die zugehörige Modellbeschreibungsdatei ediert wird.

2.4.10. pls

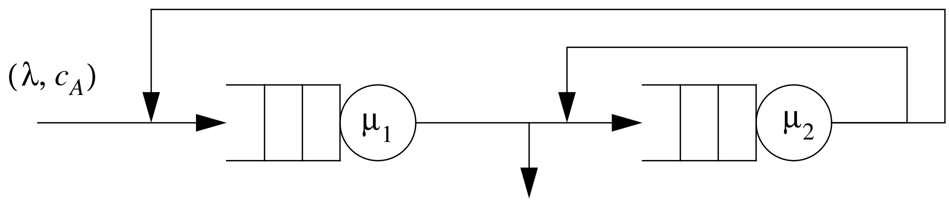
Die Infixkodierung des Verfahrens im Dateinamen der Resultatdateien ist bei der Vielzahl der Analysemethoden nicht sehr leserlich. Mit dem Programm *pls* steht eine Möglichkeit für eine kommentierte Ausgabe des Dateiverzeichnisinhalts bereit. Optionen beim Aufruf werden an *ls*(1) durchgereicht.

2.4.11. pversion

Für administrative Zwecke und zur Versionskontrolle ist dem Programmpaket dieses Skript beigefügt. Sollten Fehler im Programmsystems gefunden werden, so kann mithilfe von *pversion* der Fehlermeldung auch eine Liste der installierten PEPSY-Module und deren Aktualisierungsstand beigefügt werden.

2.5. Ein Beispiel

Folgendes Warteschlangennetz soll mit PEPSY-QNS untersucht werden (aus [Bolch89]):



Der Ankunftsprozeß von außen hat die Ankunftsrate $\lambda = 0.5$ und den quadrierten Variationskoeffizienten $c_A^2 = 0.94$. Die Parameter der Bedienprozesse lauten:

$$\mu_1 = 1.1 \qquad \mu_2 = 1.2 \qquad c_{B1}^2 = 0.5 \qquad c_{B2}^2 = 0.8$$

Gegeben sind die Übergangswahrscheinlichkeiten $p_{10} = p_{12} = p_{21} = p_{22} = 0.5$ und $p_{01} = 1$.

Um die Eingabe des Netzes zu starten wird *eingabe* ohne Parameter aufgerufen.

Die Eingabefelder werden mit den entsprechenden Werten gefüllt und abschließend der Dateiname e_bolch89 gewählt. Nach der Beendigung des Programms ist eine Eingabedatei mit folgendem Inhalt erzeugt worden:

```
# Eingabedatei Version  1
#
#
# Dateiname e_bolch89
#


ANZAHL KNOTEN:  2
ANZAHL KLASSEN: 1

KNOTENBESCHREIBUNG

  Knoten |      Name           |     Typ
---------+---------------------+---------------------
     1   |      Knoten  1      |     M/G/1-FCFS
     2   |      Knoten  2      |     M/G/1-FCFS

KLASSENBESCHREIBUNG

  Klasse  | Ankunftsrate   Anzahl Auftraege
----------+-----------------------------------
     1    |   0.5             -

KLASSENSPEZIFISCHE KNOTENGROESSEN

KLASSE  1

     Knoten         |  Bedienrate   quad._Var.-Koeff.
--------------------+-----------------------------------
     Knoten  1      |  1.1        0.5
     Knoten  2      |  1.2        0.8


UEBERGANGSWAHRSCHEINLICHKEITEN

von/zu     | aussen     Knoten  1  Knoten  2
-----------+-------------------------------
aussen     | 0.000000  1.000000  0.000000
Knoten  1  | 0.500000  0.000000  0.500000
Knoten  2  | 0.000000  0.500000  0.500000
```

Dann wird mit dem Komanndo “*auswahl -a bolch89*” diese Liste von anwendbaren Verfahren ausgegeben:

```
d1:
    Anwendbar                  |  Mit Zusatzblock anwendbar
```

```
---------------------------------------------------------
                          | kuehn
                          | mem
                          | simulation
                          | sim2
                          | ermem
                          | pujolle
                          | chylla
                          | gelenbe
                          | whitt
                          | diffprox
```

Da eine Analyse mit der Diffusionsapproximation vorgenommen werden soll, ist noch ein Zusatzblock notwendig. Dieser wird mit "*zusatz diffprox bolch89*" interaktiv eingegeben.

Eine erneute Verfahrenauswahl zeigt, daß jetzt alle gelisteten Verfahren, bis auf die zwei Simulationen, anwendbar sind. Ein Blick in die Eingabedatei zeigt folgenden Zusatzblock, der am Ende angehängt worden ist:

```
MEM

VARIATIONSKOEFFIZIENT DES ANKUNFTSPROZESSES

Knoten     | Klassen
-----------+----------------------------
Knoten  1  |      0.94
Knoten  2  |         1
```

Dabei handelt es sich um den sogenannten MEM-Zusatzblock.

Nun sind alle Voraussetzungen für einen Aufruf der eigentlichen Leistungsgrößenberchnung erfüllt. Er lautet: "*analyse diffprox bolch89*" und liefert diese Ausgaben auf dem Monitor und in der Datei *a_x_bolch89:*

```
LEISTUNGSGROESSEN FUER NETZ: bolch89


Die Netzbeschreibung ist in Datei 'e_d' abgelegt.

Das offene Netz wurde mit dem Verfahren 'diffprox'  analysiert.


Auftragsklasse 1

diffprox |  lambda      e    1/mue    rho    mvz     maa     mwz    mwsl
---------+----------------------------------------------------------------
Knoten 1 |  1.000   2.000   0.909   0.909   7.147   7.147   6.238   6.238
Knoten 2 |  1.000   2.000   0.833   0.833   4.151   4.151   3.318   3.318


charakteristische Netzgroessen

diffprox   |  lambda     mvz     maa
-----------+-------------------------
```

```
             |   0.500  22.595  11.298


    quad. Variationskoeffizienten Ankunftsprozess
    Knoten 1  0.920000
    Knoten 2  0.825000
    approximierte Auslastungen
    Knoten 1  0.872795
    Knoten 2  0.799243


    Legende

    e    : Besuchshaeufigkeit                  mue    : Bedienrate
    rho  : Auslastung                          lambda: Ankunftsrate
    mvz  : mittlere Verweilzeit der Auftraege
    maa  : mittlere Anzahl der Auftraege
    mwz  : mittlere Wartezeit der Auftraege
    mwsl: mittlere Warteschlangenlaenge
```

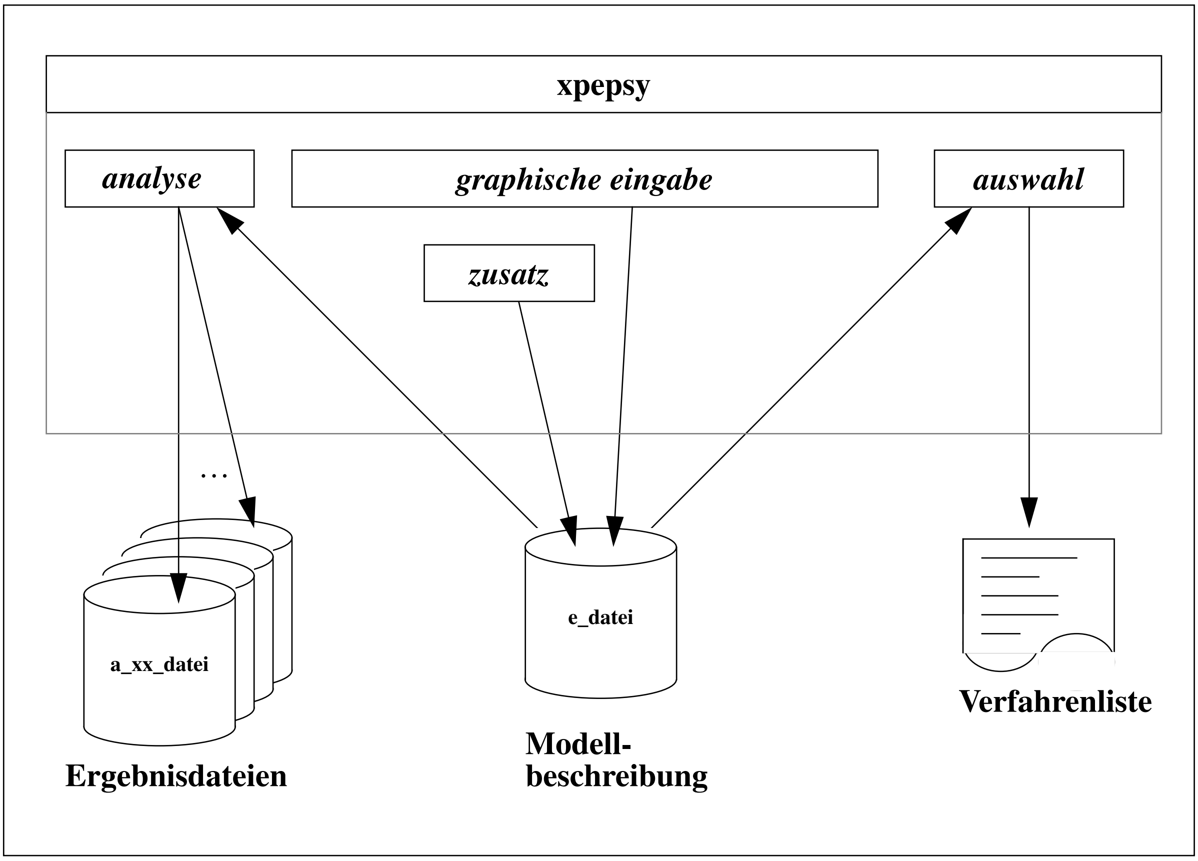
Wenn anschließend noch ein Vergleich der Ergebnisse mit denen des Chylla-Verfahrens gewünscht wird, so kann man mit “*pdiff chylla diffprox bolch89*” diese Ausgabe erhalten:

```
5c5
< Das offene Netz wurde mit dem Verfahren ''chylla''  analysiert.
---
> Das offene Netz wurde mit dem Verfahren 'diffprox'  analysiert.
10c10
< 'chylla'  |  lambda   e    1/mue    rho    mvz    maa   mwz   mwsl
---
> diffprox  |  lambda   e    1/mue    rho    mvz    maa   mwz   mwsl
12,13c12,13
< Knoten  1 |  1.000  2.000  0.909  0.909  8.009  8.009  7.100  7.100
< Knoten  2 |  1.000  2.000  0.833  0.833  4.896  4.896  4.062  4.062
---
> Knoten  1 |  1.000  2.000  0.909  0.909  7.147  7.147  6.238  6.238
> Knoten  2 |  1.000  2.000  0.833  0.833  4.151  4.151  3.318  3.318
20c20
< 'chylla'  |  lambda    mvz    maa
---
> diffprox  |  lambda    mvz    maa
22c22
<           |  0.500  25.810  12.905
---
>           |  0.500  22.595  11.298
25a26,31
> quad. Variationskoeffizienten Ankunftsprozess
> Knoten 1  0.920000
> Knoten 2  0.825000
> approximierte Auslastungen
> Knoten 1  0.872795
> Knoten 2  0.799243
```

3. Die graphische Schnittstelle XPEPSY

XPEPSY stellt eine komfortable Schnittstelle für die grafische, interaktive Eingabe und Analyse von Warteschlangennetzwerken dar. Dabei bedient sich das Programm der Standard-PEPSY-Moduln und Dateien, so daß eine 100%-ige Kompatibilität zum PEPSY-QNS gewährleistet ist.

Abbildung 2: Die graphische Benutzungsschnittstelle XPEPSY



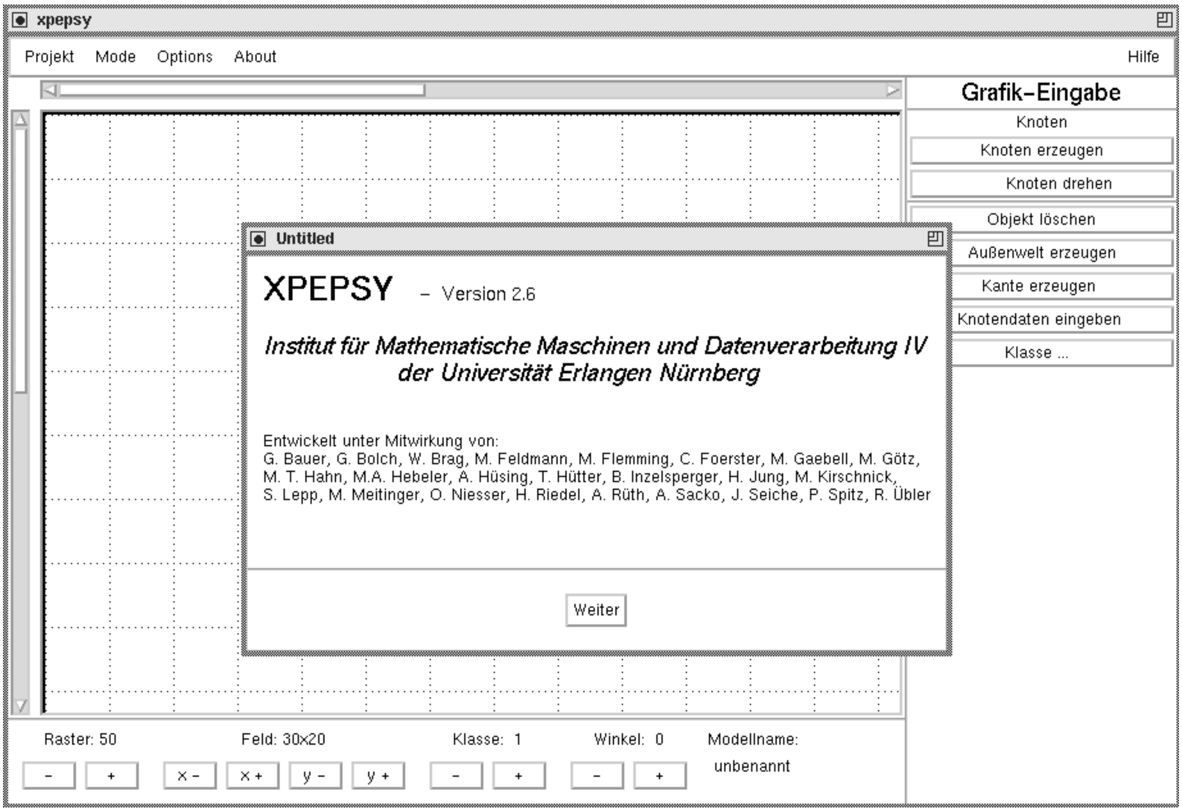
3.1. Ein Beispiel für die Arbeitsweise mit XPEPSY

Die Arbeitsweise mit XPEPSY soll im folgenden demonstriert werden.

Nach dem Start des Programms, durch Eingabe des Befehls "*xpepsy*" ohne Parameter, erscheint das Hauptfenster mit der Menue-Leiste und dem Begrüßungstext (Abbildung 3). Grundfunktionen zum Laden, Speichern und Löschen eines Warteschlangenmodells stehen im Menue *Projekt* zur Verfügung. Dort befindet sich auch der Menuepunkt zum Verlassen des Programmsystems *Quit*.
Mit dem *Options*-Menue sind zur Zeit zwei Einstellungen bezüglich des Aussehens der Wartesysteme möglich: die Namen der Knoten und/oder der Knotentyp können auf der Zeichenfläche eingeblendet werden oder nicht.
Über den *Hilfe*-Knopf soll eine On-Line Hilfe verfügbar sein. Dies ist in der aktuellen Version noch nicht implementiert, so daß leider nur ein Hinweis auf die fehlende Unterstützung gege-

ben wird.
Im *About*-Menu sind Informationen über das Programm selbst abrufbar. Dort kann auch die Versionsnummer ausgegeben werden.
Es gibt im wesentlichen zwei Grundmodi des Programms: ein Edier- und ein Analysemodus. Zwischen diesen kann über das *Mode*-Menue umgeschaltet werden. Im folgenden wird erst die Eingabe eines Warteschlangennetzwerkes und die Arbeitsweise mit dem grafischen Elementen im Ediermodus beschrieben.

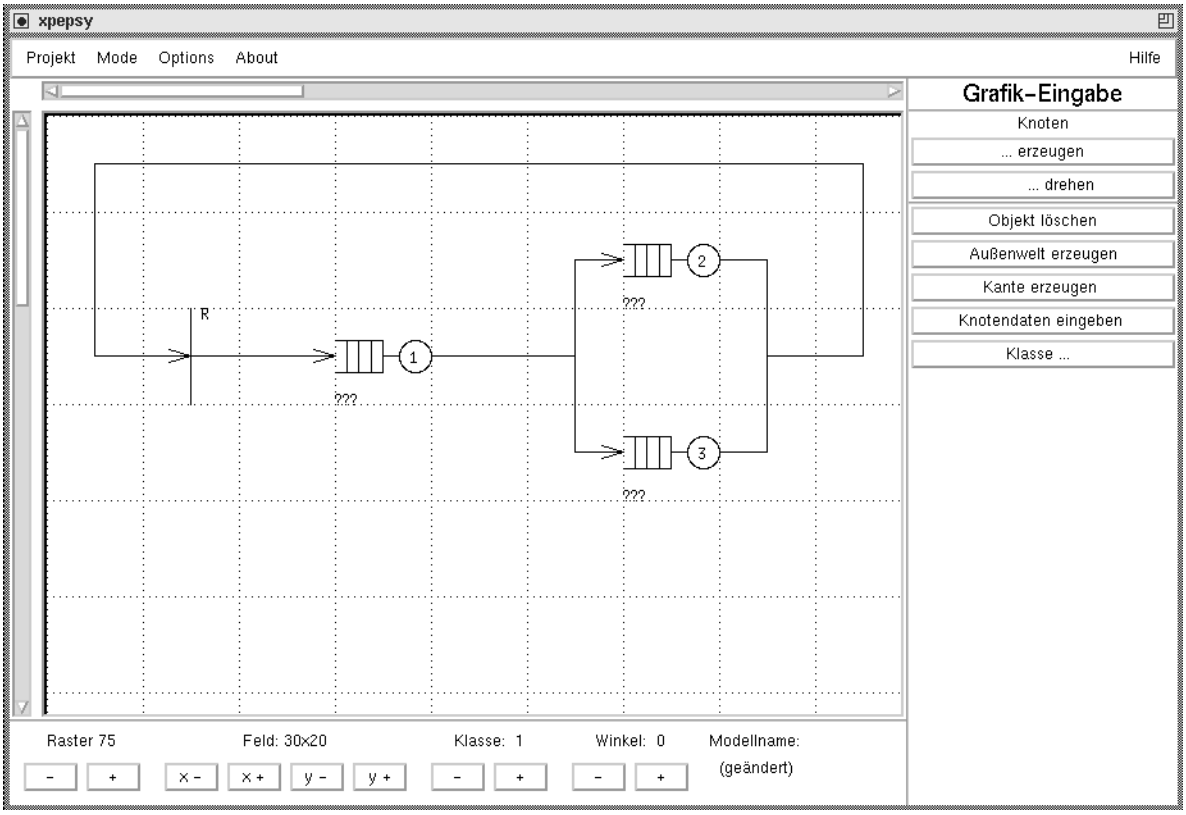
Abbildung 3: Das Hauptfenster des XPEPSY mit dem Begrüßungstext



Über die Menueleiste wird der Eingabemodus eingestellt. Dann kann mit der Eingabe des Netzwerkes begonnen werden. Als erstes platziert man üblicherweise eine Außenwelt, die für geschlossene Klassen als Referenzknoten dient und daher unbedingt notwendig ist. Dies geschieht durch Aktivieren des Knopfes "*Aussenwelt erzeugen*" und anschließendem anklicken der gewünschten Position im Raster des Zeichenfeldes. Ebenso verfährt man anschließend mit den Bedienungssystemen.
Sind alle Knoten platziert, so werden die Übergänge zwischen den Wartesystemen durch Kanten ermöglicht. Eine Kante wird erzeugt, indem im Grafikmenue rechts "*Kante erzeugen*" aktiviert wird. Dann sind Kanten einzugeben, indem der Quellknoten angeklickt wird, dann Stützstellen, die zur verbesserten grafischen Darstellung dienen, und abschließend der Zielknoten angeklickt wird. Dabei sollten Kanten rechtwinkelig verlaufen, d.h. entweder waagerecht oder senkrecht; dies sei für die einfachere Handhabung bei Verschiebe- und Korrekturoperationen empfohlen. Es ist möglich, Kanten übereinander zu legen. Wird beim Erzeugen einer Kante ein Stützpunkt auf eine bestehenden Kante platziert, so wird angenommen, daß ab diesem Punkt beide Kanten gleich verlaufen und die neue Kante automatisch vervollständigt.
An dieser Stelle ist eine kurze Erklärung zum Handling von Grafikobjekten in XPEPSY not-

wendig. Alle Gegenstände, sei es die Außenwelt, ein Knoten oder eine Kantenstützstelle, sind im Ediermodus durch ein Anklicken mit der linken Maustaste, anschließendem Verschieben bei gedrückter Taste und Loslassen der Taste an der gewünschten Position verschiebbar. Übereinanderliegende Kanten oder Kantenstützstellen werden gemeinsam behandelt, es sei denn, eine Kante ist direkt selektiert. Dies geschieht durch die Aktivierung des Quellknotens (mit der linken Maustaste) und des Zielknotens direkt hintereinander, die aktivierten Elemente erscheinen verstärkt gezeichnet. Eine Selektion wird aufgehoben, wenn ein leeres Quadrat im Rasterfeld angeklickt wird.
Mit dem Knopf "*Objekte löschen*" lassen sich sowohl Knoten als auch Kanten und Kantenstützstellen wieder entfernen. Nach der Aktivierung dieses Menuepunktes erscheint der Mauszeiger (Cursor) über der Zeichenfläche mit dem Aussehen eines Totenkopfes, um die bevorstehende Löschoperation anzuzeigen. Ebenso funktioniert der Menuepunkt "*Knoten drehen*", als Cursor erscheinen zwei kreisförmig gebogene Pfeile. Die Drehoperationen können beendet werden, indem auf ein beliebiges freies Feld der Zeichenfläche geklickt wird, oder ein neuer Menuepunkt angewählt wird. Die Löschoperation bezieht sich im Gegensatz dazu nur auf ein Objekt ist anschließend sofort deaktiviert. Eine typische Situation ist in der folgenden Abbildung dargestellt.

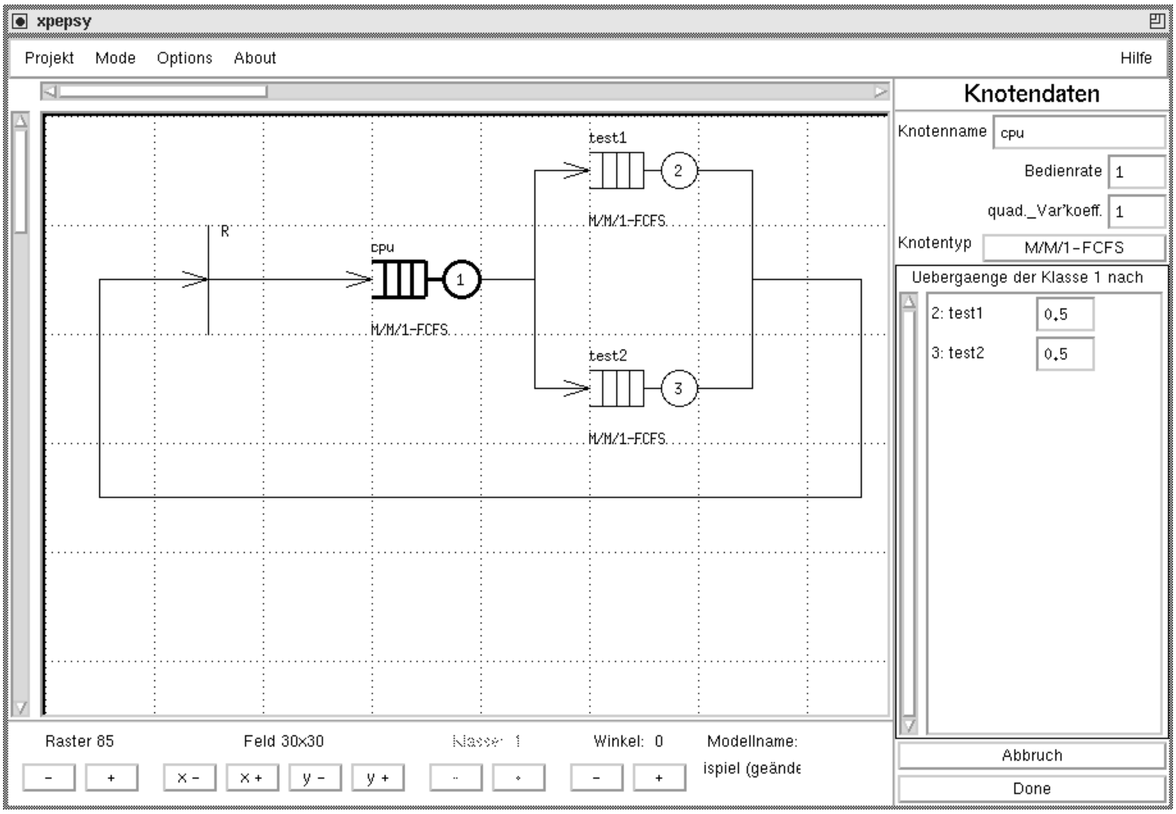
Abbildung 4: Die Eingabe der Netztopologie



Nach der Eingabe der Netztopologie werden die Parameter der Knoten in beliebiger Reihenfolge spezifiziert. Dies geschieht durch den Menueknopf "*Knotendaten eingeben*" und der Aktivierung des gewünschten Wartesystems. Dabei wechselt die Menueleiste auf der rechten Seite, und eine Eingabemaske für die Knotenparameter erscheint (siehe Abbildung 5).

Sind alle Kanten spezifiziert, so lassen sich über "*Klasse...*" weitere Auftragsklassen hinzufügen, in die man mit den “+/-”-Buttons am unteren Rand wechseln kann. Über diesen Menuepunkt lassen sich auch Klassen entfernen.

Abbildung 5: Eingabe von Parametern eines Wartesystems



Bevor ein Wechsel in den Analysemodus geschehen kann, ist das Speichern des Warteschlangennetzes notwendig (siehe Abbildung 6). Dies liegt in der Koppelung von XPEPSY zu PEPSY-QNS über die Dateischnittstelle. Ist das Modell nicht abgespeichert, wird man beim Aufruf des Menuepunktes “*Mode/Analyse*” darauf hingewiesen.

Der sichtbare Unterschied im Analysemodus ist das geänderte Menue auf der rechten Seite, sowie das Fehlen der Rasterlinien auf der Zeichenfläche. (siehe Abbildung 7)
Im unteren Teil wird eine Liste der anwendbaren Analyseverfahren angezeigt. Das, durch einfaches Anklicken des Verfahrens in der Liste, zuletzt ausgewählte Verfahren wird im oberen Drittel über den “*Analyse!*”- und “*Verfahren Info*”-Buttons angezeigt. Der Analyseaufruf geschieht durch Anklicken von “*Analyse!*”. Mit “*Verfahren Info*” kann der *pinfo*(1)-Befehl in XPEPSY erreicht werden. Dann erscheint ein Fenster mit der Ausgabe der Informationen zum aktuellen Verfahren.
Ebenso werden die Analyseergebnisse in einem eigenen Fenster angezeigt, so daß auch mehrere Analysemethoden verglichen werden können. Siehe hierzu die Abbildung 8.

Abbildung 6: Der "Speichern"-Dialog

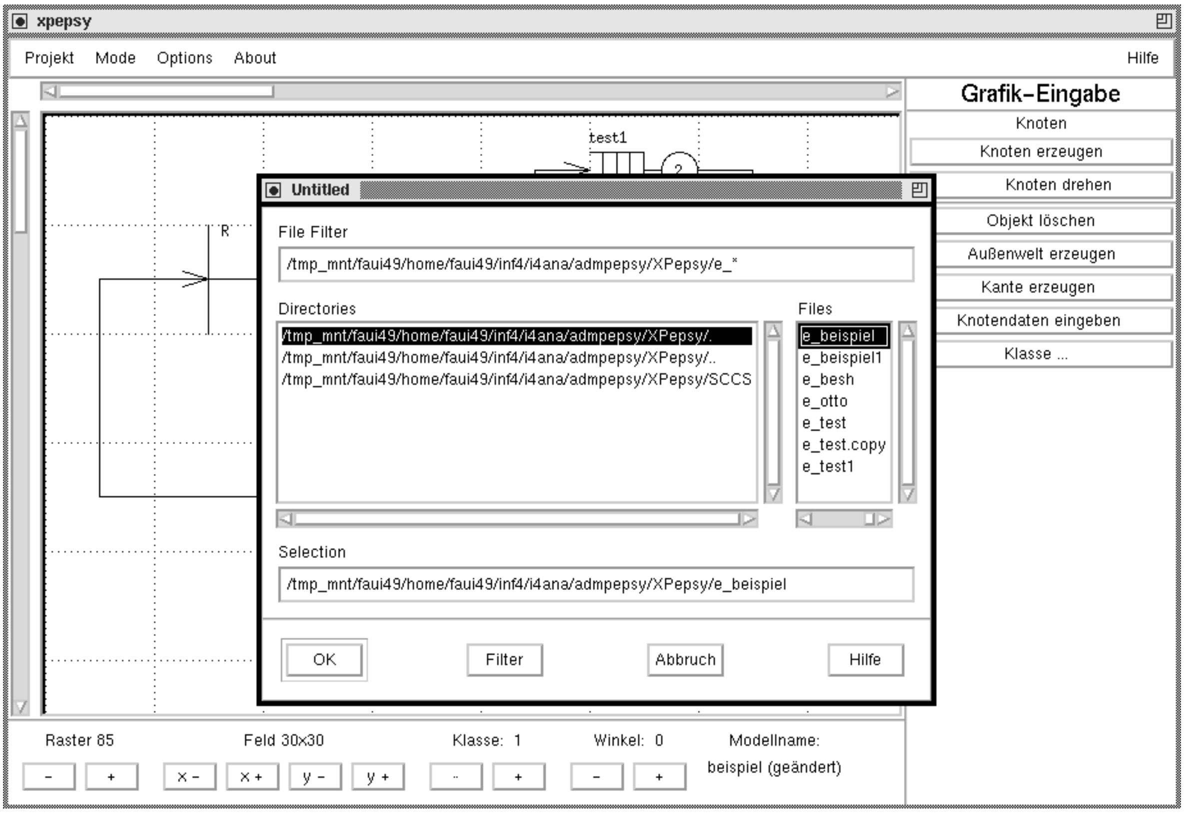


Abbildung 7: Der Analysemodus mit geladenem Warteschlangennetz

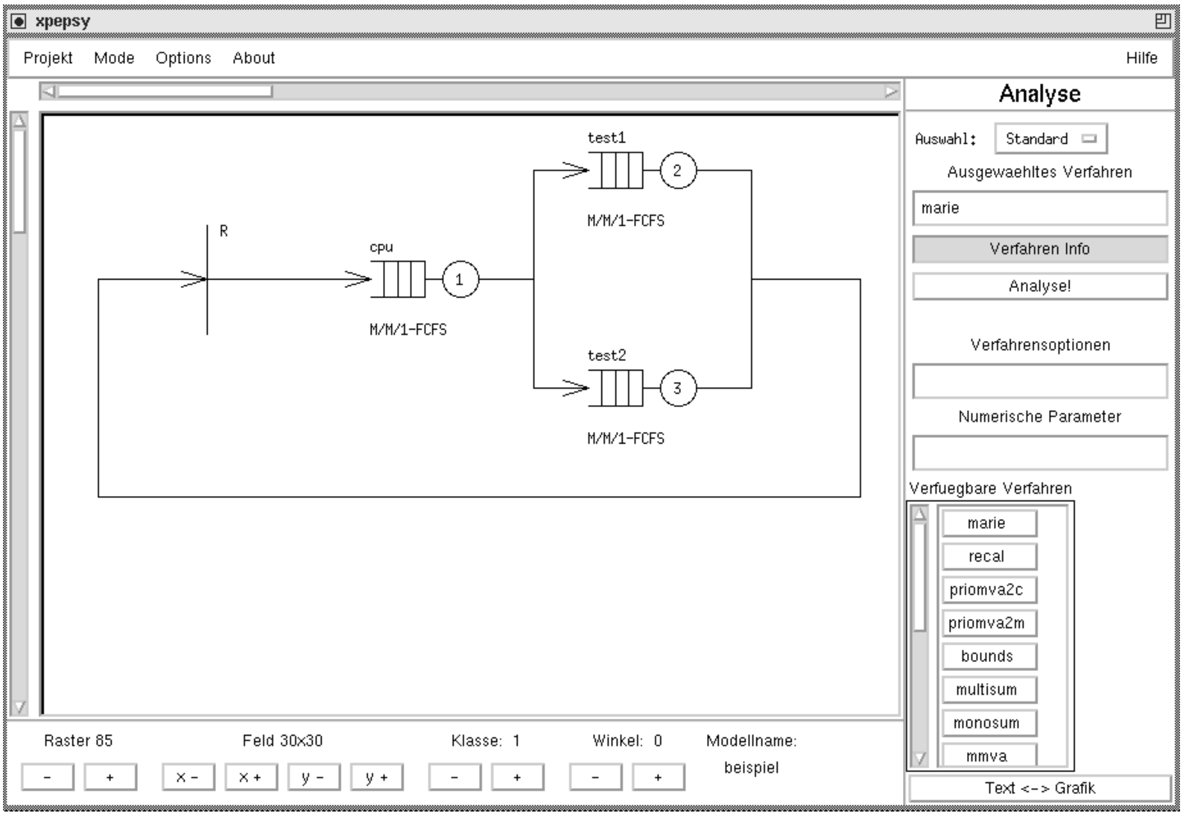
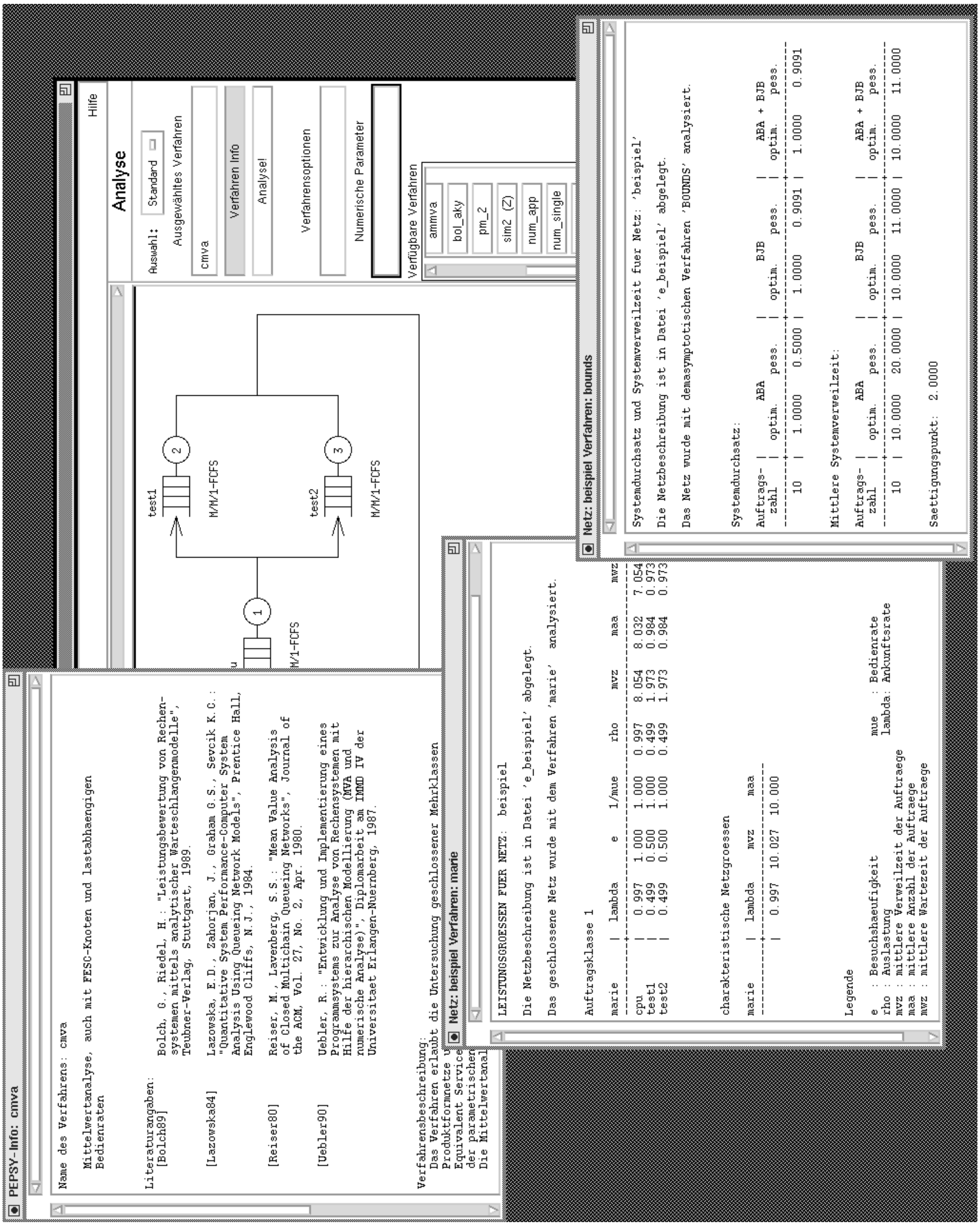


Abbildung 8: Arbeitsumgebung XPEPSY



4. Zusätzliche Programme

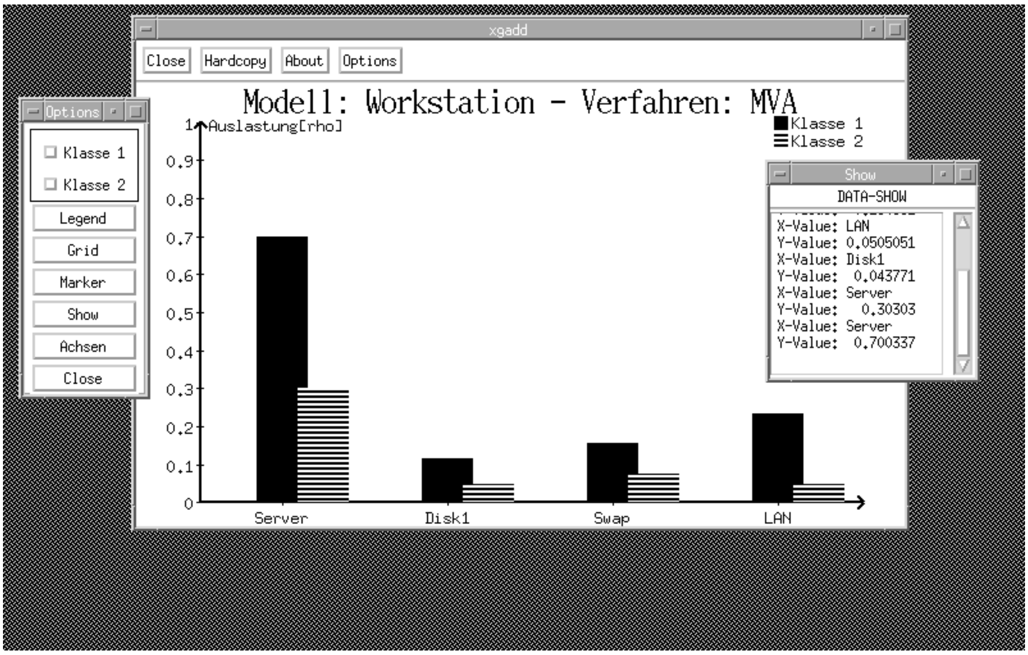
4.1. XGad

XGad ist als universelles Programm für die Visualisierung von Analyseergebnissen beliebiger Programme unter X-Windows entwickelt worden [Meitinger92].

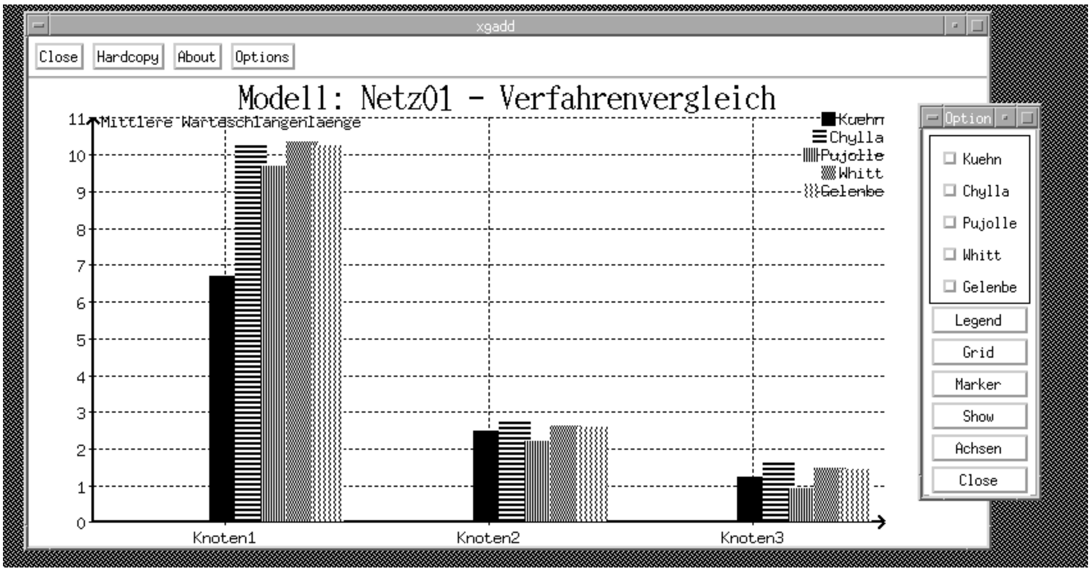
Es ist möglich, beliebige Balken- und Liniengraphiken aus einfachen Beschreibungendateien zu erzeugen, die eine anschauliche Dartstellung der Ergebnisse ermöglichen.
Die automatische Anbindung des Graphiktools an PEPSY-QNS wird mittels eines Filterprogramms vorgenommen, das aus Ergebnisdateien die relevanten Daten ausliest, daraus eine Graphikbeschreibung erzeugt und anschließend XGad aufruft.

Die den vielen Parametriesierungsmöglichkeiten von XGad lassen dem Benutzer einen enorm großen Freiraum für die Gestaltung der Graphiken. Bei der automatisierten Visualisierung wurden die vier wichtigsten Fälle berücksichtigt:

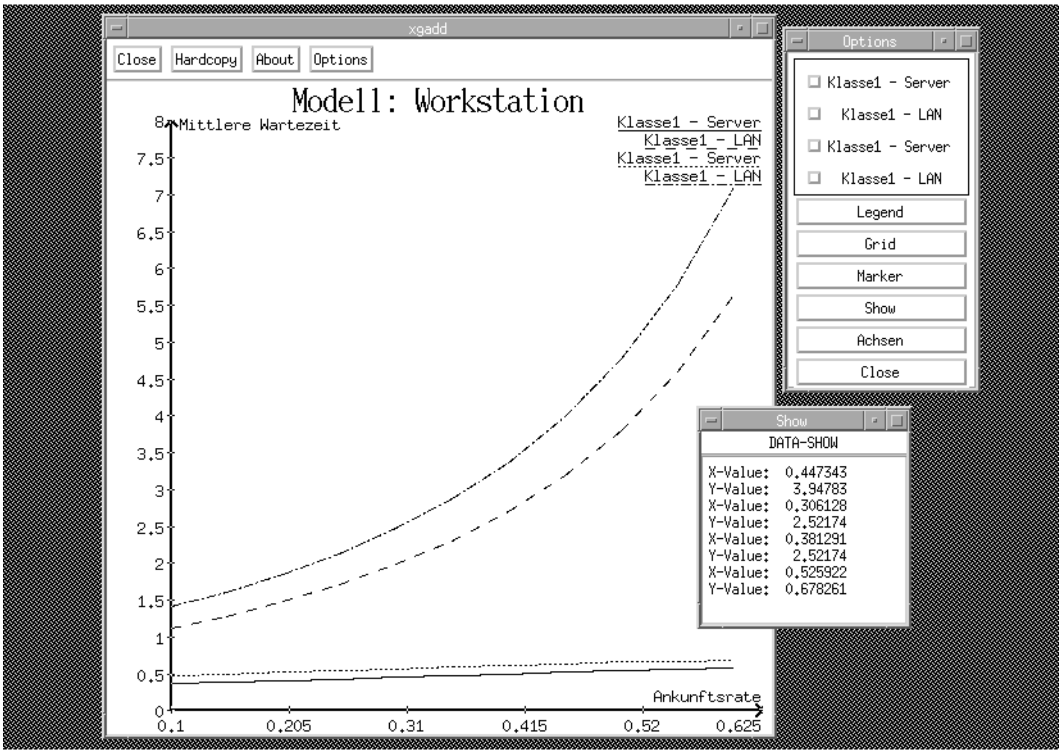
- *Ergebnisdarstellung*
  Hierbei werden die Analyseergebnisse eines Verfahrens für ein Modell dargestellt.



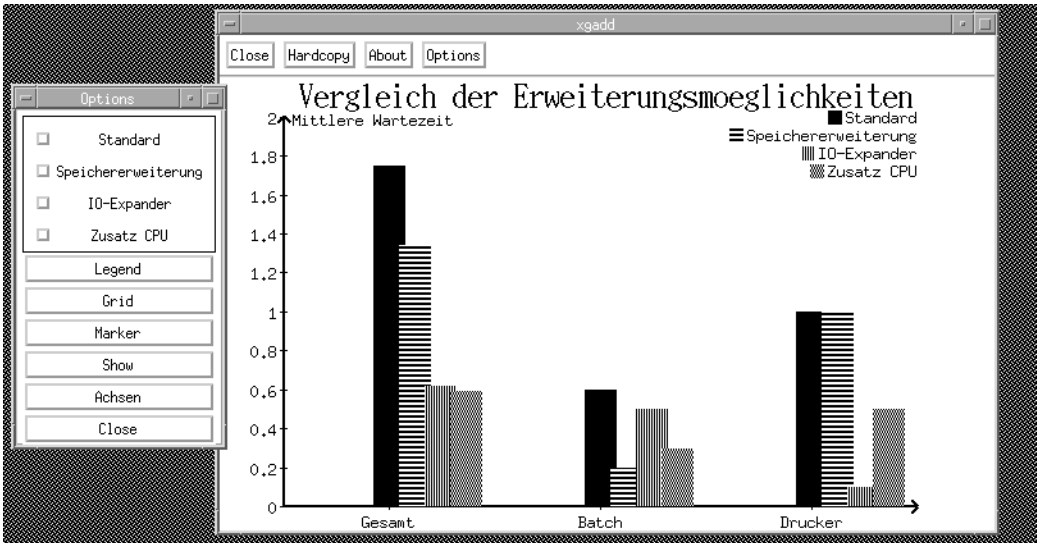
- *Verfahrenvergleich*
  Diese Art der Graphik dient dem Vergleich der Ergebnisse verschiedener Analyseverfahren bei Anwendung auf dasselbe Modell.



- *Parametervariation*
  Ein Warteschlangennetzwerk wird wiederholt mit demselben Verfahren untersucht und dabei ein Modellparameter geändert.



- *Modellvariation*
  Dies ist der allgemeinste Fall, mit dem Vergleich ausgewählter Analyseergebnisse verschiedener Modelle, die eventuell mit unterschiedlichen Analysemethoden gewonnen wurden, sind Beurteilungen von Modellvarianten möglich.
  Die Modelle können sich dabei z.B. im Detaillierungsgrad, in der Art der Modellumsetzung in ein Warteschlangennetzwerk oder in Topologieparametern unterscheiden.



In der vorligenden Erprobungsversion geschieht die Erzeugung der Graphiken noch durch den Aufruf eines Programms *(pepsy2xgad*). Eine direkte Anbindung an XPEPSY soll der nächste Schritt sein. Weitere in der Enwicklung befindliche Optionen des Filterprogramms sind die Erzeugung von Analyseergebnisdateien für Standard-Tabellenkalkulationen oder andere Grafikprogramme.

5. Anhang

5.1. Literaturverzeichnis

[Agrawal84] Agrawal, S.C., Buzen, J.P., Shun, A.W.: “Response Time Preservation: A General Technique for Developing Approximate Algorithms for Queueing Network”, ACM Sigmetrics Performance Evaluation Review, Vol. 12, No. 3, pp. 63-77, Aug. 1984.

[Akyildiz83] Akyildiz, I.F., Bolch, G.: “Erweiterung der Mittelwertanalyse zur Berechnung der Zustandswahrscheinlichkeiten für geschlossene und gemischte Netze”, 2.GI/NTG-Fachtagung “Messung, Modellierung und Bewertung von Rechensystemen”, Stuttgart 21-23.2.1983.

[Akyildiz88a] Akyildiz, I.F., Bolch, G.: “Mean Value Analysis Approximation for Multiple Server Queueing Networks”, Performance Evaluation, Vol. 8, No. 2, pp. 77-91, April 1988.

[Bard79] Bard, Y.: “Some Extensions to Multiclass Queueing Network Analysis”, 4th Int. Symp. on Modeling and Performance Evaluation of Computer Systems, Vol. 1 Wien, Feb. 1979.

[Bauer89] Bauer, G.: “Optimierung von Leistungsgrößen mit Hilfe der Lagrangemethode”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1988.

[BCMP75] Baskett, F., Chandy, K.M., Muntz, R.R., Palacios, G.F.: “Open, Closed and Mixed Networks of Queues with Different Classes of Customers”, Journal of ACM, vol. 25, no. 2, pp. 126-134, 1975.

[BFS87] Bolch, G., Fleischmann, G., Schreppel, R.: “Ein funktionales Konzept zur Analyse von Warteschlangennetzen und Optimierung von Leistungsgrößen”, Informatik-Fachberichte 154, Springer-Verlag, pp. 327-342, 1987.

[BoZe87] Bolch, G., Zeis, G.: “Softwaretools zur Leistungsbewertung von Rechensystemen”, Angewandte Informatik, No. 11, pp. 470-480, 1987.

[Bolch89] Bolch, G., Riedel, H.: “Leistungsbewertung von Rechensystemen mittels analytischer Warteschlangenmodelle”, Teubner-Verlag, Stuttgart, 1989.

[BoJu90] Bolch, G., Jung, H., Niesser O.: “PEPSY, Performance Evaluation and Prediction System”, ein Programmsystem zum Vergleich von analytischen Verfahren zur Berechnung der Leistungsgrößen von Warteschlangennetzen, Interner Bericht, Oktober 1990.

[Brag90] Brag, W.: “Validierung und Vergleich der in PEPSY implementierten Verfahren”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1990.

[Brag91] Brag, W.: “Analytische Methoden zur Leistungsbewertung von Warteschlangennetzen”, Diplomarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1991.

[Buzen71] Buzen, J.P.: “Queueing Network Models of Programming”, Ph.D.Thesis, Div. of Engineering and Applied Physics, Haward University, 1971.

[Chandy82] Chandy, K.M., Neuse, D.: “Linearizer: A Heuristic Algorithm for Queueing Network Modells of Computing Systems”, CACM, Vol. 25, Nr. 2, Feb. 1982.

[Chandy83] Chandy, K.M., Lakshmi M.S.: “An Approximation Technique for Queueing Networks with Preemptive Priority Queues”, Techn. Rep. Dept. Computer Sciences, Univ. of Texas at Austin, Fe. 1983.

[Chandy75a] Chandy, K.M., Herzog, U., Wool, L.: “Approximate Analysis of General Queueing Networks”, IBM Journal of Research and Development , Vol. 19, No. 1, pp. 36-42, Jan. 1975.

[Chylla86] Chylla, P.: “Zur Modellierung und approximativen Leistungsanalyse von Vielteilnehmer-Rechensystemen”, Dissertation an der Fakultät für Mathematik und Informatik der TU München. 1986.

[Conway86] Conway, A.E., Georganas N.D.: “RECAL A New Effizient Algorithm for the Exact Analysis of Multiple-Chain Closed Queueing Networks”, Journal of the ACM, Vol. 33, No. 4, pp. 768-791, Oct. 1986.

[Duda87] Duda, A.: “Approximate Performance Analysis of Parallel Systems”, Proc. 2nd Int. Workshop on Applied Mathematics and Performance/Reliability Models an Computer/Communication Systems, pp. 189-202, 1987.

[Feldmann83] Feldmann, M., “Implementierung von Verfahren zur Leistungsanalyse von Rechensystemen”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1983.

[Fleischmann86] Fleischmann, G.: “Little’s Gesetz für höhere Momente und Approximation der Verteilungsfunktion aus Momenten”, Diplomarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1986.

[Flemming88] Flemming, M.: “Implementierung der Maximum Entropie Methode”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1988.

[Förster89] Förster, C.: “Implementierung und Validierung von MVA-Approximationen”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1989.

[Gaebell92] Gaebell, M.: “Entwicklung und Validierung der Schließmethode zur Analyse offener Warteschlangennetze”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg. 1992.

[Gelenbe87] Gelenbe, E.: “Introducing to Queueing Networks”, J. Willey, 1987.

[Götz88] Götz, M.: “Erweiterung des Programmsystems PEPSY um eine Simulationskomponente”, Diplomarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1988.

[Hahn88] Hahn, T.: “Implementierung und Validierung der Mittelwertanalyse für höhere Momente und Verteilungen”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1988.

[Hahn90] Hahn, T.: “Erweiterung und Validierung der Summationsmethode”, Diplomarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1990.

[Hebeler89] Hebeler, M.A.: “Erweiterung des LINEARIZER durch nichtlineare Approximation”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1989.

[Hüsing89] Hüsing, A: “Implementation der hierarchischen Modellierung unter Verwendung des SCAT-Algorithmus”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1989.

[Hütter89] Hütter, T.: “Erweiterung der Summationsmethode zur Leistungsbewertung von Rechensystemen” Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1989.

[Hütter90] Hütter, T.: “Erweiterung des Programmsystems PEPSY um die EPF-, Maximun-Etropie-, und die Kühn-Methode”, Diplomarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1990.

[Inzelsperger89] Inzelsperger, B.: “Entwicklung analytischer Formeln für Leistungsgrößen aus Simulationsergebnissen bei geschlossenen Warteschlangennetzen”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1989.

[Jackson57] Jackson, J.R.: “Network of Waiting Lines”, Operation Research, Vol. 5, No. 4, pp. 518-521, 1957.

[Jackson63] Jackson, J.R.: “Jobshop-Like Queueing Systems”, Management Science, Vol. 10, No. 1, pp. 131-142, Oct.1963.

[Kirschnick90] Kirschnick, M.: “Entwurf und Implementierung eines Simulationsprogramms zur Analyse allgemeiner Warteschlangenmodelle”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1990.

[Kobayashi74] Kobayashi, H.: “Application of the Diffusion Approximation to Queueing Networks, Part I: Equilibrium Queue Distributions”, Journal of the ACM, Vol. 21, No. 2, pp. 316-328, Apil 1974.

[Koschinski85] Koschinski, B.: “Ein Approximationsalgorithmus zur Berechnung von Leistungsmaßen in Basis-Warteschlangen-Netzwerken”, Diplomarbeit, Institut für Informatik Abteilung II, Bonn 1985.

[Kouvatsos85] Kouvatsos, D.D.: “Maximum Entropy Methods for General Queueing Networks”, Proc. Int. Conf. on Modelling Techniques and Tools for Performance Analysis, pp. 589-608, 1985.

[Kouvatsos88a] Kouvatsos, D.D., Georgatsos, P.H.E., Tabet-Aouel, N.M: “A Universal Maximum Entropie Algorithm for General Multiple Class Open Networks with Mixed Service Disciplines”, Proc. 4th Int. Conf. on Modeling Techniques and Tools for Computer Performance Evaluation, Palma de Mallorca, Sept 1988, pp. 473-492.

[Kouvatsos88b] Kouvatsos, D.D., Almond, J.: “Maximum Entropy Two-Station Cyclic Queues with Multiple General Servers”, Acta Informatica 26, 241-267, Springer-Verlag, 1988.

[Kühn79] Kühn, P.J.: “Approximate Analysis of General Queueing Networks by Dekomposition”, IEEE Transactions on Communications, Vol. 27, No. 1, pp. 113-126, Jan. 1979.

[Lavenberg80] Lavenberg, S.S.: “Mean-Value Analysis of Closed Multichain Queueing Networks”, JACM 27, Feb. 1980, pp. 313-322.

[Lazowska84] Lazowska, E.D., Zahorjan, J., Graham G.S., Sevcik K.C.: “Quantitative System Performance-Computer System Analysis Using Queueing Network Models”, Prentice Hall, Englewood Cliffs, N.J., 1984.

[Lepp88] Lepp, S.: “Untersuchung neuer Näherungsverfahren für die Mittelwertanalyse durch Anwendung von G/G/m-Approximationsformeln”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1989.

[Marie79] Marie, R.A.: “An Approximate Analytic Method for General Queueing Networks”, to appear in IEEE, TSE, 1979.

[Marie80] Marie, R.A.: “Calculating Equilibrium Probabilities for $\lambda(n)/C_k/1/N$-Queues”, ACM Sigmetrics Performance Evaluation Review, Vol. 9, No. 2, Summer 1980, pp. 117-125.

[Meitinger92] Meitinger, M.: “Entwurf und Implementierung eines Programms zur graphischen Aufbereitung von Analyseergebnissen unter X-Windows”, Diplomarbeit am IMMD IV der Universität Erlangen-Nürnberg, 11/92, 1992.

[Muntz74] Muntz, R.R., Wong, J.W.: “Efficient Computional Procedures for Closed Queueing Network Models”, Proc. of the 7th Hawaii Int. Conf. on System Science, pp. 33-36, Jan. 1974.

[Neuse81] Neuse, D., Chandy, K.M.: “SCAT: A heuristic Algorithm for Queueing Network Models of Computing Systems”, ACM Sigmetrics Performance Evaluation Review, Vol. 10, No. 3, pp. 59-79, Fall 1981.

[Niesser89] Niesser, O.: “Implementierung neuer Verfahren für PEPSY (Prioritätsnetz, Faltung, RECAL)”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1989.

[Pujolle86] Pujolle, G., Wu, A.: “A Solution for Multiserver and Multiclass Open Queueing Networks”, INI-OR, Vol. 24, No. 3, 1986, pp. 221-230.

[Reiser74] Reiser, M., Kobayashi, H.: “Accuracy of the Diffusion Approximation for some Queueing Systems”, IBM Journal of Research and Development, Vol. 18, No. 2, pp. 110-124, March 1974.

[Reiser80] Reiser, M., Lavenberg, S.S.: “Mean Value Analysis of Closed Multichain Queueing Networks”, Journal of the ACM, Vol. 27, No. 2, Apr. 1980.

[Reiser81] Reiser, M.: “Mean-Value Analysis and Convolution Method for Queue-Dependent Servers in Closed Queueing Networks”, Performance Evaluation, Vol. 1, pp. 110-124, 1981.

[Rüth87] Rüth, A.: “Entwicklung, Implementierung und Validierung neuer Approximationsverfahren für die Mittelwertanalyse zur Leistungsbewertung von Rechenanlagen”, Diplomarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1987.

[Sauer81] Sauer, C.H., Chandy, K.M.: “Computer Systems Performance Modeling”, Prentice-Hall, 1981.

[Scheller84] Scheller, J.: “Implementierung von Verfahren zur Leistungsanalyse von Rechensystemen”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1984.

[Schmidt84a] Schmidt, B.: “Der Simulator GPSS-Fortran Version 3”, Springer-Verlag, Berlin-Heidelberg-New York 1984, Fachberichte Simulation; Band 2.

[Schmidt84b] Schmidt, B.: “Modellierung mit GPSS-Fortran Version 3”, Springer-Verlag, Berlin-Heidelberg-New York 1984, Fachberichte Simulation; Band 3.

[Schmuck87] Schmuck, G.: “Programmsystem PEPSY: Weitere Berechnungsverfahren, automatische Verfahrenauswahl und Optimierung”, Diplomarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1987.

[Schweitzer79] Schweitzer, P.: “Approximate Analysis of Multiclass Closed Networks of Queues”, Proc. Int. Conf. on Stochastic Control and Optimization, Amsterdamm, 1979.

[Seiche91] Seiche, J.: “Modellierung von Rechensystemen mit Hilfe von Warteschlangenmodellen”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1991.

[Shum76] Shum, A.W.: “Queueing Models for Computer Systems with General Service Time Distributions”, Ph. D. Thesis, Div. of Engineering and Applied Physics, Harvard Univ., Dec. 1976.

[Shum77] Shum, A.W., Buzen, J.P.: “The EPF Technique: A Method for Obtaining Approximate Solutions to Closed Queueing Networks with General Service Times”, Proc. 3rd Symp. on Measuring, Modeling and Evaluating Computer Systems, Bonn, North Holland, 1977, pp. 201-220.

[Spitz88] Spitz, P.: “Entwicklung und Implementierung eines Programms zur Ermittlung optimaler Leistungsgrößen von Rechensystemen”, Studienarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1988.

[Strelen85] Strelen, J.C.: “Eine Verallgemeinerung der Mittelwertanalyse auf höhere Momente”, Informatik-Fachberichte 110, Springer-Verlag, pp. 156-169, 1985.

[Übler90] Übler, R.: “Entwicklung und Implementierung eines Programmsystems zur Analyse von Rechensystemen mit Hilfe der hierarchischen Modellierung (MVA und numerische Analyse)”, Diplomarbeit am IMMD IV der Universität Erlangen-Nürnberg, 1987.

[Whitt83] Whitt, W.: “The Queueing Network Analyzer”, The Bell System Technical Journal, Vol. 62, No. 9, Nov. 1983, pp. 2779-2815.

[Wong75] Wong, J.W.: “Queueing Network Models for Computer Systems”, Ph. D. Thesis, Scholl of Engineering and Applied Science, Univ. of California, Los Angeles, Oct. 1975.

[Zahorjan81] Zahorjan, J., Wong, E.: “The Solution of Separable Queueing Network Models Using Mean Value Analysis”, ACM Sigmetrics Performance Evaluation Review, Vol. 10, No. 3, pp. 80-85, Fall 1981.

5.2. Manualseiten

Auf den folgenden Seiten sind die On-Line-Manuale des PEPSY-QNS abgedruckt. (Stand 20.10.1992).

analyse(1)

NAME

analyse - Analyse eines Warteschlangennetzes

SYNOPSIS

analyse [-d <lvl>] [-v] [-t] [-b] [-f] [-O <Verfahrensoptionen>] [-N <Analyseparameter>] [-p<Prefilterflags>][-P<Postfilterflags>] [-m<num>] <verfahren> [<eingabedatei>]
oder
anallyse -V

Es können mehrere Modelle gleichzeitig angegeben werden und die Namen der Eingabedateien können wahlweise mit oder ohne Prefix 'e_' eingegeben werden! Für die Angabe mehrerer Verfahren siehe die Option -m.

BESCHREIBUNG

analyse ist das Steuerprogramm zur Untersuchung von Warteschlangennetzen im Rahmen des Programmpaketes *pepsy(1)*. Nach Erstellen einer Netzbeschreibung mit *eingabe(1)* und Aufruf von *auswahl(1)* steht fest, welche Verfahren auf dieses konkrete Netz angewendet werden können. Damit liegen die letzten beiden Parameter von analyse fest. In der Version 1.10 stehen folgende Optionen zur Verfügung:

-a Wird die Option -a verwendet, so wird die Datenbasis aller implementierten Verfahren untersucht. Im Normalfall wird nur eine Teilmenge der effizienten und guten Verfahren untersucht.

-b Durch Angabe der Option -b wird die Ausgabe auf dem Bildschirm unterdrückt,

-d ermöglicht Debug-Kontrollausgaben zur Überwachung des Programmablaufs und erwartet den numerischen Parameter 'lvl' , der die Ausführlichkeit der Meldungen beeinflußt.

-f Durch diese Option wird ein impliziter Aufruf von *auswahl(1)* unterdrückt, um Verfahrensgrenzen zu umgehen. Dies bedeutet das Abschalten einer Sicherheitsmaßnahme und ist daher nur mit Vorsicht anzuwenden!

-m Seit der neuen Version 1.3 ist es möglich sowohl mehrere Modelldateien gleichzeitig anzugeben als auch mehrere Verfahren auf eine oder mehrere Modelle anzuwenden. Nach der Option -m wird eine Zahl erwartet, die angibt, wieviele der nachfolgenden Parameter Namen von Verfahren sind. Werden mehrere Verfahren oder mehrere Modelle gleichzeitig angegeben, so wird die Bildschirmausgabe unterdrückt. Es ist nicht sinnvoll, durch Mehrfachangabe der Simulation mehrere Simulationen gleichzeitig zu starten, da diese parallel abgearbeitet werden.

-N Mit -N "numerische Parameter" können dem Analyseverfahren die numerischen Parameter numerische Parameter übergeben und mit der Option

-O "Parameter" die zus ätzlichen Aufrufparameter Parameter übergeben werden.

-p

-P Mit diesen Optionen können den Verfahren vor- bzw. nachgeschalteten Filtern Aufrufparameter mitgeteilt werden.

-t Durch Angabe von -t wird die Zeit für das eigentliche Analyseverfahren gemessen und ausgegeben.

-v Mit dieser Option können die Schritte bei dem Aufruf der Analyseverfahren mitverfolgt werden.

-V Mit der Option -V kann die aktuelle Version des Programmes ausgegeben werden (*pversion(1)*).

Analyse überprüft die Modifikationszeitpunkte der Netzbeschreibung und der Ergebnisdatei und ruft das Analyseverfahren nur dann auf, wenn die Netzdatei datei jünger ist als die Ausgabedatei oder wenn das Analyseverfahren in der Zwischenzeit modifiziert wurde. Ein Aufruf erfolgt auch, wenn die vorhandene Ausgabedatei leer ist (Länge=0), da solche Dateien unter Umst änden bei einem Abbruch des Analyseverfahrens mit einer Fehlermeldung entstehen. Um eine erneute Analyse zu vermeiden, können die entsprechenden Dateien mit *touch(1)* aufgefrischt werden.

BEISPIELE

Mittelwertanalyse mit Momentenberechnung nach Strelen

analyse -O”-s” mmva beispiel

Mittelwertanalyse mit Berechnung der Zustandswahrscheinlichkeiten ohne Ausgabe auf dem Bildschirm

analyse -O”-w” -b mmva beispiel

Grenzwertanalyse für den Bereich 5 bis 10 Aufträge

analyse -N”5 10” bounds beispiel

Produktform-Lösung für ein allgemeines offenes Netz

analyse -f openpfn beispiel

Mittelwertanalyse und Summationsmethode auf die Modelldatei e_test:

analyse -m 2 mmva multisum test

oder

analyse -m 2 mmva multisum e_test

Anwendung von Kühn’s und Chylla’s Verfahren auf alle Modelldateien im aktuellen Verzeichnis:

analyse -m 2 kuehn chylla e_*

DATEIEN

$PEPSYHOME/bin/analyse	Steuerprogramm
$PEPSYHOME/infos/analyse.*	Sprachunterstützungsdateien
$PEPSYHOME/infos/code.txt	Steuerdatei
$PEPSYHOME/bin2/verfahren/*	Analyseverfahren

WEITERE HINWEISE

pepsy(1), eingabe(1), auswahl(1), zusatz(1) *auswahl(1)*

NAME

auswahl - suche anwendbare Verfahren

SYNOPSIS

auswahl [-afq] [-m verfahren] datei [datei ...]

oder

auswahl -V

Die Modelldateien können mit oder ohne Prefix ‘e_’ angegeben werden.

BESCHREIBUNG

Auswahl dient im Rahmen von *pepsy(1)* dazu, für ein mittels *eingabe(1)* spezifiziertes Warteschlangennetz die anwendbaren Analyseverfahren auszugeben. Optionen:

-a Durch -a werden alle implementierten Verfahren untersucht, standardmäßig wird nur eine Teilmenge der Verfahren auf ihre Anwendbarkeit hin untersucht.

-d Mit -d num kann die Funktion des Programmes näher beobachtet werden.

-f Mit dieser Option zeigt auswahl an, warum ein Verfahren nicht anwendbar ist.

-q Zur Unterdrückung aller Ausgaben kann die Option -q angegeben werden. Das Ergebnis kann durch den Returnwert abgefragt werden. (Nützlich für Batchdateien.)

-m Mit -m kann explizit die Anwendbarkeit eines Verfahrens getestet werden.

Die Ausgabe ist in zwei Spalten unterteilt, wobei unter anwendbar all diejenigen Verfahren auftauchen, die sofort auf die angegebene Datei anwendbar sind. In der Spalte vielleicht anwendbar erscheinen all die Verfahren, die zwar anwendbar sind, jedoch durch *zusatz(1)* noch in ihrem Ablauf beeinflußt werden können.

Zur Bestimmung der anwendbaren Verfahren werden die nachfolgend aufgelisteten Eigenschaften nachgeprüft:

- Anzahl Knoten überprüfen jedes Verfahrens auf seine Obergrenze bezüglich der Anzahl der Knoten
- Anzahl Klassen überprüfen jedes Verfahrens auf seine Obergrenze bezüglich der Anzahl der Klassen
- Knotentypen überprüfen der von jedem Verfahren handhabbaren Knotentypen, z. B. multiple server
- Typ des Netzes überprüfen der von jedem Verfahren handhabbaren Netztypen
- MMm-FCFS überprüfen für jedes Verfahren, ob bei Typ-1-Knoten die Bedienzeit für alle Klassen gleich sein muß
- Variationskoeffizienten überprüfen, abhängig vom Knotentyp, der für jedes Verfahren handhabbaren Variationskoeffizienten der Bedienzeit
- Zusatzblock überprüfen, ob ein Zusatzblock angegeben werden kann
- Anzahl paralleler Bedieneinheiten überprüfen, wiewiele parallele Bedieneinheiten bei multiple server-Knoten zulässig sind

Alle unter “vielleicht anwendbar” angegebenen Verfahren sind auch ohne Aufruf von *zu-*

satz(1) anwendbar, die fehlenden Parameter werden dabei mit Exponentialverteilungen vorbesetzt.

RETURNS

Falls eines der angegebenen Verfahren auf irgendeine der angegebenen Modelldateien nicht anwendbar ist, liefert der Aufruf den Returnwert 1 ansonsten 0.

DATEIEN

$PEPSYHOME/bin/auswahl	Steuerprogramm
$PEPSYHOME/infos/auswahl.*	Sprachunterstützungsdateien
$PEPSYHOME/infos/grenzen.txt	Datenbasis für Verfahren
$PEPSYHOME/infos/grenzenall.txt	Datenbasis für alle Verfahren

WEITERE HINWEISE

pepsy(1), eingabe(1), zusatz(1), pepsy(4) *pcheck(1)*

NAME

pcheck - prüfe Eingabdatei auf Konsistenz

SYNOPSIS

pcheck <eingabedatei>

BESCHREIBUNG

Pcheck dient im Rahmen von *pepsy(1)* dazu, Eingabedateien auf Konsistenz zu prüfen. Da die Zusatzblöcke *zusatz(1)* voneinander unabhängig sind, wird nicht erzwungen, daß ihre Parameter untereinander konsistent und somit auch Ergebnisse verschiedener Bewertungen vergleichbar sind. Mithilfe von pcheck kann diese Prüfung nachgeholt werden.

Momentan wird überprüft:

- Der Variationskoeffizient des Ankunftsprozesse für offene Netze für die Simulation, das Kühn-Verfahren, die Diffusionsapproximation und die Maximum-Entropie-Methode
- Der Variationskoeffizient der Bedienzeit für die Simulation und alle anderen Verfahren.

DATEIEN

$PEPSYHOME/infos/pcheck.*	Sprachunterstützungsdateien
$PEPSYHOME/bin/pcheck	Steuerprogramm

WEITERE HINWEISE

pepsy(1), auswahl(1), zusatz(1)

pdiff(1)

NAME

pdiff - zeige die Unterschiede der Ergebnisse zweier Verfahren

SYNOPSIS

pdiff verfahren1 verfahren2 datei

BESCHREIBUNG

Pdiff erstellt eine Tabelle, in der die unterschiedlichen Zeilen der Ausgabedateien der beiden Analyseverfahren verfahren1 und verfahren2 aufgelistet sind.

Durch den impliziten Aufruf von *analyse(1)* ist sichergestellt, daß Modifikationen an der Netzbeschreibung nachgezogen werden.

DATEIEN

$PEPSYHOME/bin/pdiff	Steuerprogramm
e_datei	Netzbeschreibung
a_?1_datei	formatierte Ausgabe, Verfahren 1
a_?2_datei	formatierte Ausgabe, Verfahren 2

WEITERE HINWEISE

pepsy(1), eingabe(1), auswahl(1), zusatz(1), analyse(1), diff(1) *pepsy(1)*

NAME

pepsy - performance evaluation and prediction system

BESCHREIBUNG

Diese Manual-Seite beschreibt den Aufbau und die Verwendung der unter dem Namen PEPSY-QNS, dem 'Erlanger-Warteschlangen-Analyse-Baukasten' (ehemals: EWAB.-vBG), zusammengefaßten Programme und Analyseverfahren.

Ziel des ganzen Baukastens ist die Illusion einer einheitlichen, komfortablen Benutzerschnittstelle für eine vielzahl unterschiedlichster Analyseprogramme.

Eine typische Pepsy-Sitzung läuft wie folgt ab:

eingabe -> auswahl -> analyse

Voraussetzung ist die Definition der Environment Variablen PEPSYHOME, die die Wurzel des installierten Programmsystems im Dateibaum beschreibt. Weiterhin muß der Suchpfad um das Verzeichnis $PEPSYHOME/bin erweitert werden.

Mit dem Programm *eingabe(1)* wird interaktiv die allgemeine Beschreibung eines Warteschlangennetzes eingegeben und im global gültigen Format abgespeichert.

Mit *auswahl(1)* kann man sich alle Verfahren auflisten lassen, die auf die gerade erstellte Datei anwendbar sind. Dabei kann es vorkommen, daß für gewisse Analyseverfahren noch weitere Angaben nötig sind, die dann im Programm *zusatz(1)* eingegeben werden müssen (natürlich nur, wenn man das eingebene Netz mit diesem Verfahren analysieren möchte).

Nach vollständiger Spezifikation des Warteschlangennetzes, kann man es mit dem Programm *analyse(1)* bewerten lassen. Die Ergebnisse werden auf dem Bildschirm ausgegeben und gleichzeitig in einer Datei abgespeichert.

Mit *pdiff(1)* können die unterschiedlichen Ergebnisse zweier Analyseverfahren am Bildschirm angezeigt werden.

Grundlage aller Pepsy-Aktivitäten ist eine global gültige Eingabedatei, die mit *eingabe(1*) und *zusatz(1)* angelegt wird und über *auswahl(1)* und *analyse(1)* ausgewertet wird. Diese Datei wird über eine Vorverarbeitung *prefilter(1)*, anhängig vom gerade aktuellen Analyseverfahren, in ein Format überführt, das das Analyseverfahren versteht und weiterverarbeiten kann. Die Ausgaben des Analyseverfahrens werden mit *pepout(1)* oder *boundsout(1)* in ein endgültiges, leicht lesbares Format gebracht. In der Regel ist dieser ganze Vorgang im Programm *analyse(1)* verborgen und für den normalen Benutzer nicht weiter interessant.

Bis auf die Simulation werden alle Analysen online abgewickelt und man erhält nahezu unmittelbar die Ergebnisse am Bildschirm aufgezeigt.

Mit *transform(1)* können Besuchshäufigkeiten in Übergangswahrscheinlichkeiten überführt werden, die spezifizierten Netze somit simuliert werden. Diese Transformation ist nicht eindeutig, so daß die Ergebnisse nur mit Vorsicht weiterverwendet werden können.

Mit *pcheck(1)* kann überprüft werden, ob die spezifizierten Zusatzblöcke konsistent sind, damit Ergebnisse verschiedener Bewertungsläufe auch miteinander verglichen werden können.

DATEIEN

$PEPSYHOME/bin/*	Steuerprogramme
$PEPSYHOME/bin2/*	indirekt aufrufbare Programme
$PEPSYHOME/bin2/verfahren	Analyseverfahren
$PEPSYHOME/bin2/prefilter	Prefilter
$PEPSYHOME/bin2/postfilter	Postfilter
$PEPSYHOME/infos/code.txt	Codierung der Verfahren
$PEPSYHOME/infos/grenzen.txt	Grenzen der Verfahren

PROBLEME

Der Aufruf von *analyse(1)* kann in seltenen Fällen zu Endlosschleifen führen. Das Programm sollte in diesem Fall - nach einer Überprüfung mit der Option -v - abgebrochen werden.

WEITERE HINWEISE

eingabe(1), auswahl(1), analyse(1), zusatz(1), pdiff(1), pinfo(1), pls(1), pcheck(1), pinfo(1), transform(1), pepsy(4), pepsy(1M) *pls(1)*

NAME

pls - *ls(1)* mit Kommentaren zu PEPSY-Dateien

SYNOPSIS

pls [-d <dbglvl>] [<Datei>|<Directory> ...]

oder

pls -V

BESCHREIBUNG

Pls dient im Rahmen von *pepsy(1)* dazu, Dateiverzeichnisse auszugeben und dabei PEPSY-spezifische Dateien mit Kommentaren zu versehen.

DATEIEN

$PEPSYHOME/bin/pls	Steuerprogramm
$PEPSYHOME/infos/pls.*	Sprachunterstützungsdateien

WEITERE HINWEISE

pepsy(1), auswahl(1), zusatz(1)

postfilter(1)

NAME

postfilter - Nachbereitung eines Analyselaufes

SYNOPSIS

boundsout [datei] pepout [datei]

BESCHREIBUNG

Pepout ist der Postfilter für Bewertungsläufe im Rahmen des Warteschlangen-Analysepaketes *pepsy(1)*. Das Programm liest von stdin und schreibt formatiert und kommentiert nach stdout und erwartet das folgende Eingabeformat:

Name des Analyseverfahrens
Netztyp Knoten Klassen
für alle Klassen
    für alle Knoten
        lambda e s rho mvz maa mwz mwsl
verfahrensspezifische Zusatzinformation

mit

lambda	Durchsatz
e	Besuchshaüfigkeit
s	mittlere Bedienzeit
rho	Auslastung
mvz	mittlere Verweilzeit
maa	mittlere Anzahl Aufträge
mwz	mittlere Wartezeit
mwsl	mittlere Warteschlangenlänge

Aus den Leistungsgrößen pro Klasse werden die Leistungsgrößen pro Knoten berechnet und ebenfalls mit ausgegeben. Die verfahrensspezifische Zusatzinformation wird lediglich kopiert und sonst in keiner Weise modifiziert. Der Aufruf von pepout geschieht normalerweise indirekt über analyse(1), wobei die Ausgabe auch in einer Datei abgespeichert wird.
Boundsout ist der Ausgabefilter für das Verfahren bounds und wird nur in diesem Zusammenhang verwendet.

DATEIEN

$PEPSYHOME/infos/pepout.*	
$PEPSYHOME/infos/boundsout.*	Sprachunterstützungsdateien
$PEPSYHOME/bin2/postfilter/pepout	allgemeiner Ausgabefilter
$PEPSYHOME/bin2/postfilter/boundsout	Ausgabefilter für bounds-Verfahren

WEITERE HINWEISE

pepsy(1), analyse(1)

prefilter(1)

NAME

prefilter - konstruiere aus Netzbeschreibung verfahrensspezifische Datei

SYNOPSIS

prefilter verfahren datei

BESCHREIBUNG

Prefilter ist der Eingabefilter für Analyseverfahren im Rahmen des Warteschlangen-Analysepaketes *pepsy(1)*. Das Programm wird in der Regel nur indirekt über analyse(1) aufgerufen und automatisch mit den richtigen Parametern versorgt. Aufgabe von prefilter ist es, eine global gültige Netzbeschreibung, gesteuert über den Verfahrensnamen, in einen Zahlenfriedhof umzuformen, der als Eingabe für das gewünschte Verfahren dient. Die transformierte Datei wird auf stdout geschrieben und kann mit einer pipe an das Analyseverfahren weitergeleitet werden. Wenn Übergangswahrscheinlichkeiten in der Netzbeschreibung angegeben sind, das Analyseverfahren aber auch mit Besuchshäufigkeiten arbeiten kann, werden diese in *prefilter(1)* schon berechnet. Neue Analyseverfahren müssen somit lediglich Besuchshäufigkeiten verkraften können.

DATEIEN

$PEPSYHOME/bin2/prefilter/*	Prefilterprogramme
$PEPSYHOME/infos/code.txt	Steuerdatei

WEITERE HINWEISE

pepsy(1), analyse(1), pepsy(4)

transform(1)

NAME

transform - konstruiere aus Besuchshäufigkeiten Übergangswahrscheinlichkeiten

SYNOPSIS

transform [modell] eingabedatei [ausgabedatei]

BESCHREIBUNG

Transform konstruiert aus Besuchshäufigkeiten Übergangswahrscheinlichkeiten. Da diese Abbildung nicht eindeutig ist, werden zwei Modelle angeboten.

Mit -m1 wird eine Serienschaltung aller Knoten im Netz berechnet, die die Quotienten der Besuchshäufigkeiten untereinander erhält. Mit -m2 wird statt der Serienschaltung eine Parallelschaltung errechnet. Mit -m3 oder -c wird das Netzwerk in ein Central-Server Netzwerk transformiert, wobei angenommen wird, daß der Knoten 1 die CPU modelliert. Die Transformation funktioniert sowohl für offene als auch für geschlossene Netze. Mit -m4 oder -t wird das Netzwerk in ein Central-Server Netzwerk mit Terminals transformiert, wobei angenommen wird, daß der Knoten 1 die Terminals modelliert und Knoten 2 die CPU modelliert. Besitzt das Netzwerk offene Klassen, so dürfen diese die Terminals nicht besuchen.

Ist keine Ausgabedatei angegeben, so wird das umgeformte Warteschlangennetz auf dem Bildschirm ausgegeben.

ANMERKUNG

Ist mindestens eine Besuchshäufigkeit >= 1, so werden in der Serienschaltung sogar die Werte der Besuchshäufigkeiten erhalten. Sind alle Besuchshäufigkeiten <= 1, so werden ihre Werte in einer Parallelschaltung erhalten.

DATEIEN

$PEPSYHOME/bin/transform Programm

WEITERE HINWEISE

pepsy(1), analyse(1), auswahl(1)

NAME

zusatz - ergänze eine Warteschlangennetz Spezifikation

SYNOPSIS

zusatz verfahren datei

BESCHREIBUNG

Ist bei *auswahl(1)* ein Analyseverfahren in der Spalte vielleicht anwendbar aufgetaucht, so kann mit zusatz das Warteschlangenetz noch näher beschrieben werden. Typische Beispiele sind etwa die Variationskoeffizienten der Ankunftsprozesze beim Kühn-Verfahren oder die Angabe der Simulationszeit.

Ist schon ein Zusatzblock für das Analyseverfahren verfahren vorhanden, so wird zusatz mit einer Fehlermeldung abgebrochen. Ein Ändern der Zusatzinformation setzt das Löschen der alten Information voraus.

DATEIEN

$PEPSYHOME/bin/zusatz	Steuerprogramm
$PEPSYHOME/infos/zusatz.*	Sprachunterstützungsdateien

WEITERE HINWEISE

pepsy(1), eingabe(1), auswahl(1)

5.3. Technische Daten

PEPSY-QNS ist in ‘C’ implementiert und auf allen UNIX-Systemen (z.B. SUN-OS, SINIX, ULTRIX, HP-UX,...) ablauffähig.
Zur Zeit existieren etwa 5,5 MB Source-Code, die durch eine Hierarchie von sogenannten “Makefiles” verwaltet werden. Auch sind einige Verwaltungsprogramme für die Übersetzung und Installation des Programmsystems vorhanden, mit denen PEPSY systemglobal für alle Benutzer oder lokal für einen Anwender installiert werden kann. Insgesamt werden ca. 20-30 MB Plattenkapazität benötigt.

An Universitäten und Schulen, also öffentlichen Einrichtungen der Lehre und Forschung wird PEPSY-QNS i.A. für nichtkommerzielle Zwecke kostenlos, für sonstige Interessenten gegen eine geringe Vergütung, abgegeben.
Ansprechpartner ist:

Dr. G. Bolch
Institut für Mathematische Maschinen und Datenverarbeitung IV
Universität Erlangen-Nürnberg
Martensstraße 1
8520 Erlangen

e-mail: bolch@immd4.informatik.uni-erlangen.de

XPEPSY wird als Aufsatz für PEPSY-QNS entwickelt und benötigt X-Windows (X11R4) sowie Motif Version 1.1. Da es momentan stark überarbeitet wird und noch nicht auf verschiedenen Rechnerplattformen getestet ist, gehört es nicht zum Standard-PEPSY, soll aber in absehbarer Zeit in das Programmpaket aufgenommen werden. Ebenso verhält es sich mit XGad.

5.4. Kommentierte Liste der Verfahren

Die folgende Liste enthält die in PEPSY-QNS verfügbaren Analyseverfahren, sortiert nach den Namenskürzeln die beim Arbeiten mit PEPSY angegeben werden müssen.

abol
Erweiterung des SCAT-Algorithmus auf M/M/m-ASYMM-Knoten.
[Brag91], [Bolch89].

ammva
Mittelwertanalyse, die auch unterschiedliche Bedienzeiten in verschiedenen Klassen bei M/M/m-FCFS-Knoten behandeln kann.
[Reiser80], [Lazowska84], [Rüth87], [Hahn88].

amonosum
Asymmetrische Summationsmethode für Einklassennetze.
[Brag91], [Bolch89].

appscat_MG1_1
SCAT-Algorithmus, erweitert für M/G/1-FCFS-Knoten wobei nur die lokale Auslastung der aktuellen Klasse in die Antwortzeitberechnung bei M/G/m-Knoten einfließt. Der implementierte Algorithmus ist auf geschlossene Produktformnetze mit mehreren Auftragsklassen anwendbar.
[Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].

appscat_MG1_2
SCAT-Algorithmus, erweitert für M/G/1-FCFS-Knoten wobei die Auslastung aller Klassen in die Antwortzeitberechnung bei M/G/m-Knoten einfließt. Der implementierte Algorithmus ist auf geschlossene Produktformnetze mit mehreren Auftragsklassen anwendbar.
[Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].

appscat_app
SCAT-Algorithmus mit Berechnung der Randwahrscheinlichkeiten.
Die Antwortzeitberechnung für M/G/m-FCFS Knoten ist um einen Korrekturfaktor erweitert. Algorithmus für geschlossene Produktformnetze mit mehreren Klassen.
[Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].

asymmva
Erweiterung des MVA-Algorithmus auf M/M/m-ASYMM-Knoten.
[Brag91].

bardschweitzer
Verfahren nach Bard und Schweitzer
[Bard79], [Schweitzer79], [Neuse81].

bol_aky
SCAT-Algorithmus mit Berechnung der Randwahrscheinlichkeiten nach Bolch/Akyildiz.
Für M/G/1-FCFS-Knoten wird analog dem Verfahren *appscat_MG1_1* vorgegangen, für M/G/m-FCFS-Knoten ist das Vorgehen wie beim Verfahren *pm_2*.
[Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].

bounds
Grenzwertanalyse.
[Lazowska84], [Schmuck87], [Bolch89].

chan_neu	SCAT-Algorithmus mit Berechnung der Randwahrscheinlichkeiten nach Chandy/Neuse. Für M/G/1-FCFS-Knoten wird analog dem Verfahren *appscat_MG1_1* vorgegangen, für M/G/m-FCFS-Knoten ist das Vorgehen wie beim Verfahren *pm_2*. [Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].
chylla	Dekompositionsverfahren nach Chylla. [Chylla86], [Hütter90].
cmva	Mittelwertanalyse, die auch FESC-Knoten mit lastabhängigen Bedienraten behandeln kann. [Bolch89], [Lazowska84], [Reiser80], [Übler90].
diffprox	Diffusionsapproximation nach Reiser und Kobayashi. [Kobayashi74], [Reiser74], [Bolch89].
epascat	Hierarchischer SCAT-Algorithmus. [Lazowska84], [Hütter89].
epf	Erweiterte Produktform-Methode. [Shum76], [Shum77], [Hütter90], [Bolch89].
ermem	Erweiterte Maximum-Entropie-Methode. [Kouvatsos88a], [Hütter90].
evlin1	Linearizer Algorithmus erster Stufe. [Feldmann83], [Koschinski85], [Hebeler89].
evlin2	Linearizer Algorithmus zweiter Stufe. [Feldmann83], [Koschinski85], [Hebeler89].
evlin3	Linearizer Algorithmus dritter Stufe. [Feldmann83], [Koschinski85], [Hebeler89].
faltung	Faltungsalgorithmus (Convolution). [Buzen71], [Muntz74], [Wong75], [Niesser89].
gelenbe	Dekompositionsverfahren nach Gelenbe. [Gelenbe87], [Hütter90].
hm	Hierarchische Analyse unter Verwendung der Verfahren *cmva* und *num*. [Lazowska84], [Übler90].
kuehn	Dekompositionsverfahren nach Kühn. [Kühn79], [Hütter90].
linearizer	Linearizer Algorithmus. [Bard79], [Bolch89].
marie	Methode von Marie. [Marie79], [Marie80], [Bolch89].
mem	Maximum Entropie Methode. [Kouvatsos88a], [Hütter90].

mmva	Mittelwertanalyse. [Reiser80], [Akyildiz83], [Strelen85], [Hahn88].
monosum	Summationsmethode für Netzwerke mit einer Auftragsklasse. [Bolch89], [Hahn90], [Hütter89].
multisum	Summationsmethode für Mehrklassennetzwerke. [Bolch89], [Hahn90], [Hütter89].
mvamix	Mittelwertanalyse für gemischte Netze. [Zahorjan81], [Akyildiz83], [Strelen85].
num	Numerische Analyse. [Sauer81], [Übler90], [Herzog90].
num_app	Numerische Analyse, bei der M/M/m-FCFS-Knoten durch M/M/m-RANDOM-Bedienungssysteme approximiert werden. [Sauer81], [Übler90], [Herzog90].
num_single	Numerische Analyse für Netze mit einer Auftragsklasse. [Sauer81], [Übler90], [Herzog90].
openpfn	Algorithmus für offene Produktformnetze. [BCMP75], [Bolch89].
pm_0	SCAT-Algorithmus ohne Berechnung der Wartewahrscheinlichkeit. [Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].
pm_1	SCAT-Algorithmus mit Berechnung der Randwahrscheinlichkeiten, wobei nur die lokale Auslastung der aktuellen Klasse in die Antwortzeitberechnung einfließt. [Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].
pm_2	SCAT-Algorithmus mit Berechnung der Randwahrscheinlichkeiten, wobei die Auslastung aller Klassen in die Antwortzeitberechnung einfließt. [Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].
pmapp_1	SCAT-Algorithmus mit approximativer Berechnung der Randwahrscheinlichkeiten, wobei nur die lokale Auslastung der aktuellen Klasse in die Antwortzeitberechnung einfließt. [Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].
pmapp_2	SCAT-Algorithmus mit approximativer Berechnung der Randwahrscheinlichkeiten, wobei die Auslastung aller Klassen in die Antwortzeitberechnung einfließt. [Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].
priomva2c	Mittelwertanalyse mit Approximation für die Berechnungen bei Prioritätsknoten. [Bolch89], [Niesser91].
priomva2m	Mittelwertanalyse mit Approximation für die Berechnungen bei Prioritätsknoten. [Bolch89], [Niesser91].

pujolle	Dekompositionsverfahren nach Pujolle. [Pujolle86], [Hütter90].
recal	REcursion by Chain-ALgorithm. [Conway86], [Niesser89].
rtp	Response-Time-Preservation [Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].
rtpapp	Response-Time-Preservation mit Approximation der Verweilzeit für geschlossene Netzwerke. [Bard79], [Schweitzer79], [Neuse81], [Agrawal84], [Förster89].
scat	Self-Correcting-Approximation-Technique. [Bard79], [Schweitzer79], [Neuse81], [Bolch89].
sim2	Discrete event simulation für Warteschlangennetzwerke. [Kirschnick90].
simulation	Simulationsmethode, implementiert mit Hilfe von GPSS-C. [Schmidt84a], [Schmidt84b], [Gietz88].
sopenpfn	Algorithmus für offene Produktformnetze mit einer Auftragsklasse. [Jackson57], [Jackson63], [BCMP75], [Schmuk87].
sum	Summationsmethode für Einklassenwarteschlangennetzwerke. [Bolch89], [Hahn90], [Hütter89].
sum_chylla	Auf der Basis der Summationsmethode erweiterte Dekompositionsmethode nach Chylla für geschlossene Netzwerke. [Brag91].
sum_ermem	Auf der Basis der Summationsmethode erweiterte Maximum-Entropie-Methode für geschlossene Netzwerke. [Brag91].
sum_pujolle	Auf der Basis der Summationsmethode erweiterte Dekompositionsmethode nach Pujolle für geschlossene Netzwerke. [Brag91].
sum_whitt	Auf der Basis der Summationsmethode erweiterte Dekompositionsmethode nach Whitt für geschlossene Netzwerke. [Brag91].
whitt	Dekompositionsverfahren nach Whitt. [Whitt83], [Hütter90].