Simulation betrieblicher Prozesse – ZF

Wiederholung FAP-Modul: Warteschlangentheorie

Die folgenden KPIs sind relevant:

- Durchschnittliche Anzahl Einheiten im System N
- Durchschnittliche Anzahl Einheiten im Wartezustand N_q
- Durchschnittliche Wartezeit im System W
- Durchschnittliche Wartezeit im Wartezustand W_q

Kendall-Notation von Warteschlangen $A/B/c/N^{max}/K/P$

- A Verteilung der Zwischenankunftszeiten
- B Verteilung der Bedienzeiten
- c Anzahl paralleler Server
- N^{max} Systemkapazität
- K Grösse der ankommenden Population
- P Warteschlangendisziplin (z.B. FIFO, LIFO)

M/M/1-Warteschlange

- exponentialverteilte Zwischenankunftszeiten (erstes 'M')
- exponentialverteilte Bearbeitungszeiten (zweites 'M')
- genau 1 Bearbeitungsstation (Server).
- Zwischenankunftszeiten und Bearbeitungszeiten sind exponentialverteilt mit Erwartungswerten $1/\lambda$ und $1/\mu$ bei $\lambda < \mu$
- Warteschlange befindet sich im **stationären Zustand**, d.h. Verteilungen/Parameter λ und μ ändern sich nicht über Zeit
- Durchschnittliche Anzahl Einheiten im System $N = \frac{\lambda}{\mu-\lambda}$
- Durchschnittliche Wartezeit im Wartezustand $W_q = \frac{\lambda}{\mu(\mu-\lambda)}$
- Durchschnittliche Wartezeit im System $W = \frac{1}{\mu-\lambda}$
- Durchschnittliche Anzahl Einheiten am Warten $N_q = \frac{\lambda^2}{\mu(\mu-\lambda)}$

Wieso Simulationen?

Warteschlangen sind eines der zentralsten Elemente einer **diskreten Ereignissimulation**.
Warum reicht es nicht aus, Warteschlangen analytisch zu beschreiben, wie im FAP-Modul dargestellt?

- Funktioniert nur mit sehr einfachen M/M/1- oder M/M/s-Warteschlangen
- Es wird immer angenommen, dass sich das System im Gleichgewicht (stationärer Zustand) befindet
  - Alle statistischen Verteilungen sind bekannt und statisch
  - Übergangsverhalten ist verborgen
  - Nur langfristige Aussagen möglich (typischerweise als Durchschnitte)

Überblick: G/G/1-Warteschlange

- beliebig verteilte Zwischenankunftszeiten (erstes 'G')
- beliebig verteilte Bearbeitungszeiten (zweites 'G')
- genau 1 Bearbeitungsstation (Server).
- Typischerweise könnte man Verteilungen wie abgeschnittene Normalverteilungen $N(mu, \sigma, 0, max)$ verwenden.
- Für allgemeine Verteilungen gibt es nur (komplizierte) Näherungen zur Berechnung der KPIs

System

- Eine Menge von Objekten, die miteinander in Beziehung stehen.

Objekt

- besitzt Attribute mit gewissen Werten,
- ist permanent oder temporär,
- ist stationär oder mobil.

Systemzustand

- Beschrieben durch momentanen vorhandenen Objekte, deren Attributwerte, und Beziehung der Objekte untereinander.
- Für eine Untersuchung ist es notwendig, sich auf die dafür relevanten Teile eines Systems zu beschränken.

Beispiel-System: Warteschlange am Bankschalter

Objekte (Beispiele):

- Objekt Bankkaufmann: permanent und stationär
- Objekt Schalter: permanent und stationär
- Objekte Kunden: temporär und mobil.

Beziehungen (Beispiele):

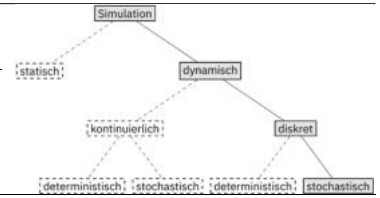
- Kunde ist momentan am Schalter, er steht in einer anderen Beziehung zum Schalter als die wartenden Kunden.

Attribute (Beispiele):

- Attribut jedes Kunden: die individuelle Ankunftszeit in der Bank
- Attribute des Bankkaufmanns: die Anzahl bedienter Kunden und die durchschnittliche Bediendauer.

Frage: was wäre ein permanentes und mobiles Objekt? → ein Kellner im Restaurant

Klassifikation von Systemen



Ereignisdiskrete Simulation von Systemen

- englisch: Discrete Event Simulation (DES)
- Berechnet den Systemzustand über die Zeit hinweg.
- Berücksichtigt diskrete (und dadurch abzählbare) Anzahl von Ereignissen.
- Systemzustand wird nur dann geändert, wenn diskretes Ereignis eintritt.

Ablauf der ereignisdiskreten Simulation

- Eine Simulationsuhr arbeitet die Ereignisse chronologisch sortiert nacheinander ab.
- Dabei werden ständig neue Ereignisse erzeugt.
- Einer Ereignisliste plant neue Ereignisse ihrem Zeitpunkt nach richtig sortiert ein.
- Die Simulationsuhr wird dabei stets auf das jeweils nächste Ereignis vorgestellt.
- Endet, wenn die Ereignisliste leer oder eine eingestellte Endzeit erreicht ist.

Untersuchung von Systemen

1. Experiment mit dem echten System
   - teuer und riskant umzusetzen (je nach System)
   - schwierig, die wichtigen Wirkfaktoren aus Vielzahl von Kausalbeziehungen zwischen Ursache/Effekt zu identifizieren.
2. Experimentsituation herstellen.
   - Versuche in einem abgegrenzten Rahmen
   - Wiederholung Experiment mit verschiedenen Rahmenbedingungen, um wichtigsten Systemfaktoren herauszufinden
   - Vereinfachung der Realität, sozusagen ein Modell
   - Kann oft auch virtuell stattfinden

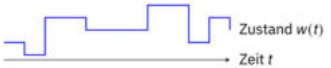
Modellversuch

	Analytische Berechnung	Diskrete Ereignissimulation	Reales Experiment
Aufwand für Ausführung	Klein	Mittel	Gross
Aufwand für Modellierung	Gross	Mittel	Gross
Aufwand für Auswertung	Klein	Mittel	Gross
Betrachter Zeithorizont	∞	Gross	Klein

Vorgehensweise in Simulationsstudien

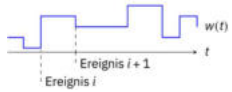
1. **Problemformulierung (interpretativ):** Definieren und schriftliches Festhalten einer eingegrenzten und klar formulierten Problemstellung und Zielen der Untersuchung.
2. **Modellentwurf (analytisch):** Systemkomponenten einschliesslich ihrer Eigenschaften und deren Zusammenspiel entsprechend der Problemformulierung für einen Blick auf das Wesentliche abstrahieren und ein Modell konzipieren.
3. **Datensammlung (entwickelnd):** Nötige Daten für das Modell identifizieren, spezifizieren, sammeln.
4. **Modellbau (entwickelnd):** Modellentwurf in eine funktionsfähige Simulation überführen.
5. **Verifikation (analytisch):** Überprüfen, ob die Simulation korrekt funktioniert.
6. **Validierung (analytisch):** Überprüfen, ob Simulation dem abgebildeten System entspricht.
7. **Analyse (analytisch):** Simulationsergebnisse interpretieren und Empfehlungen bezüglich Problemstellung entwickeln
8. **Dokumentation (interpretativ):** Unterstützende und vorliegende Informationen zusammenstellen.
9. **Implementation (entwickelnd):** Entscheidungen aus der Simulation umsetzen.

Ereignisdiskretisierung



- Grundidee: Zustände ändern sich ausschliesslich zu diskreten Zeitpunkten
- Es gibt nur abzählbar viele Zustandsänderungen.

Ereignis



- Ereignis: Die sprunghafte Zustandsänderung eines Objekts zu einem gewissen Zeitpunkt.
- Ereignis bildet unendlich kurzen Zeitraum ab.
- Im Rechner verursacht das Ereignis aber Rechenzeit, um neuen Zustand zu berechnen.
- Dualität Ereignisdauer versus Rechendauer

Aktivität

- Zeitraum zwischen zwei Ereignissen eines Objekts
- Aktivität bildet eine reale Zeitdauer ab.
- Im Rechner verursacht Aktivität keine Rechenzeit, weil Objekt zwischen zwei Ereignissen denselben Zustand beibehält.
- Dualität Aktivitätsdauer versus Rechendauer
- Recheneffizienz der ereignisdiskreten Simulation im Vergleich zu einer kontinuierlichen Simulation, bei der die Zustände sich stetig ändern.

Nebenläufigkeit bzw. Gleichzeitigkeit

- Nebenläufige Aktivitäten: Aktivitäten, die zur gleichen (realen) Zeit stattfinden.
- Gleichzeitige Ereignisse: Ereignisse, die zum gleichen (realen) Zeitpunkt stattfinden.
- Ereignisse können in der Simulation nicht gleichzeitig stattfinden Warum? Rechner arbeitet i. A. stets sequenziell.
- Gleichzeitige Ereignisse müssen künstlich in eine Reihenfolge gebracht und nacheinander abgearbeitet werden
- Die Reihenfolge ist undefiniert ▶ Wiederholbarkeit von Simulationen ist dann problematisch ▶ Reihenfolge festlegen!

Simulationsverlauf

- Ereignisdiskretisierung: Eine Zustandsänderung kann nur durch ein Ereignis ausgelöst werden. Zustandsverlauf ergibt sich durch sukzessives Abarbeiten aller Ereignisse.
- Wie sortiert man diese nach der Zeit? Einfache und performante Möglichkeit: ereignisorientierte Simulation.

Ereignisliste

Bringt momentan geplante Ereignisse in eine zeitliche Reihenfolge. Man könnte sie daher auch "ToDo-Liste" nennen.

Simulationslauf

1. Start-Ereignisse erzeugen
2. Falls die Ereignisliste leer oder die Simulation zu Ende ist, stoppen
3. Aus der Ereignisliste das nächste Ereignis e nehmen
4. Aktuelle Zeit auf den Eintrittszeitpunkt von e setzen
5. Für Ereignis e die passende Ereignisroutine ausführen
   - Systemzustand anpassen
   - neue Ereignisse planen (jetzt oder später)
   - Statistiken fortschreiben
6. Zu Schritt 2 springen

Ereignisroutine

- Gilt stets nur für einen bestimmten Ereignistyp.
- Entscheidet, was geschieht und welche neuen Ereignisse entstehen.
- Grundablauf, der sich für jedes Ereignis wiederholt:
  1. Neuen Zustand des Modells berechnen
  2. Eventuelle neue Ereignisse erzeugen
  3. Statistiken fortschreiben

Beispiel: Bankschalter mit Warteschlange
- G/G/1-Modell: Stochastische Zwischenankunftszeiten und Bedienzeiten sowie FIFO-Warteschlange
- Systemzustand: Warteschlange hat eine Länge Q, Schalter hat den Zustand S: f frei und a aktiv
- Ereignistypen: Ankunft eines Kunden sowie Bedienungsende

Ereignisroutine Ankunft eines Kunden
1. Erzeuge ein neues Ankunftsereignis (für den nachfolgenden Kunden, mit zufälliger Zwischenankunftszeit)
2. Wenn $S = f$ ist:
   1. Setze S auf a
   2. Erzeuge ein Bedienungsende-Ereignis (für diesen Kunden, mit zufälliger Bedienzeit)

   Sonst: Setze Q auf $Q + 1$

Ereignisroutine Bedienungsende

Wenn $Q > 0$ ist:
1. Setze Q auf $Q - 1$
2. Erzeuge ein Bedienungsende-Ereignis (für den ersten Kunden in der Warteschlange, mit zufälliger Bedienzeit)

Sonst: Setze S auf f

Technik für die Ereignisliste
- Ereignisliste muss stets das jeweils nächste künftige Ereignis liefern (Sortierung!)
- Ziel: kurze Rechenzeiten
- Flaschenhals Ereignisliste ▶ Ereignisliste schlank halten ▶ Datenstruktur " Prioritätsliste" nutzen
- Gegeben n Listeneinträge.
- Datenstruktur Lineare Liste mit Anhängen: $O(n)$ Zeit
  - nächstes Ereignis einplanen: hinten anhängen
  - nächstes Ereignis ermitteln: alle Listeneinträge durchsuchen!
- Datenstruktur Lineare Liste mit Einfügen: $O(n)$ Zeit
  - nächstes Ereignis einplanen: für den richtigen Einfügepunkt alle Listeneinträge durchsuchen!
  - nächstes Ereignis ermitteln: erster Listeneintrag

  ▶ Prioritätslisten-Datenstruktur: $O(log\ n)$ Zeit ist schnell durch eine Baum-Datenstruktur (Min-Heap, Splay-Baum)

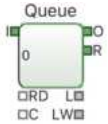
Modellbau
- Erstellung von Simulationsmodellen → Überführung eines Modellentwurfs in ein Modell
- Offenbart die Umsetzbarkeit der Überlegungen
- Zeigt, ob das gewählte Abstraktionsniveau passt

Netzwerkorientierte Struktur
- Netzwerk als Repräsentation des Systems
- Simulations-Ablauf durch Ablaufbausteine bestimmt
- Konnektoren verbinden diese zu einem Netzwerk

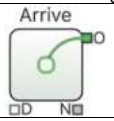
Entität
- Objekt, das über die Zeit hinweg durch Ablaufbausteine weitergegeben wird
- Ablaufbausteine verarbeiten Entitäten
- Deren Verbindung im Netzwerk bestimmt den Weg und den zeitlichen Verlauf der Entitäten.
- Quisim Entity: Eine Entität wird optisch durch eine Kugel repräsentiert.

Quisim Blockstruktur
- **Value** – Blöcke zum Festlegen von Werten, Funktionen und Wahrscheinlichkeitsverteilungen
- **Entity** – Blöcke, die Entitäten verarbeiten
- **Report** – Blöcke zur Anzeige von Simulationsdaten und Bereitstellung von Statistiken

Blockanschlüsse
- Simulationsblöcke haben verschiedene Eingangs- und Ausgangsanschlüsse. Eingangsanschlüsse befinden sich immer auf der linken Seite eines Blocks, Ausgangsanschlüsse auf der rechten Seite.
- Datenanschlüsse zum Festlegen von Parametern, zum Senden und Empfangen von Daten (silberne Anschlüsse)
- Entitätsanschlüsse für die Übertragung von Entitäten (grüne Anschlüsse)
- Beispiel: Anschluss I ist ein Eingangsanschluss für Entitäten, die Anschlüsse O und R sind Ausgangsanschlüsse für Entitäten. Die Anschlüsse C und RD ermöglichen das Festlegen von Warteschlangenparametern. Die Anschlüsse L und LW liefern Daten über den aktuellen Status der Warteschlange.



Arrive-Block
Erstellt eine neue Entität am Anschluss O. Datenanschlüsse:
- Eingangsanschluss D: definiert eine Zwischenankunftszeit in Millisekunden
- Ausgangsanschluss N: zählt die Anzahl der erstellten Entitäten



Number-Block
Stellt einen bestimmten Wert bereit, der im Eigenschaftenfeld eingegeben werden kann. Datenanschlüsse:
- Eingangsanschluss V: die Zahl von aussen festlegen
- Ausgangsanschluss N: die Zahl, die an den verbundenen Block gesendet werden soll

Leave-Block
Entfernt Entitäten. Datenanschlüsse
- Ausgangsanschluss N: Anzahl der Entitäten, die das System hier verlassen haben

Schritt zur Stochastik

Was tun, wenn Zwischenankunftszeiten nicht konstant? Ersetzen Number-Block durch stochastischen Value-Blöcke:
- Exp
- Gamma
- Unif
- Normal
- Lognorm

Bei allen stochastischen Value-Blöcke in Quisim gilt:
- Eingangsanschluss S: ein Seed für den Zufallszahlengenerator
- Ausgangsanschluss V: eine Zufallszahl.
- Eingangsanschlüsse min und max: definieren das Intervall [min, max] → ACHTUNG: Ausnahme bei Uniform [min, max)

Exp-Block
Erzeugt exponentiell verteilte Zufallszahlen im Bereich [min, max]. Datenanschlüsse
- Eingangsanschluss μ: Mittelwert (und Standardabweichung) der exponentiellen Verteilung

Gamma-Block
Erzeugt gamma-verteilte Zufallszahlen im Bereich [min, max]. Datenanschlüsse
- Eingangsanschlüsse k und θ: Form und Skala der Gamma-Verteilung

Unif-Block
Erzeugt gleichverteilte Zufallszahlen im Bereich [min, max). Datenanschlüsse
- Eingangsanschlüsse min und max: definieren das Intervall [min, max)

Normal-Block
Erzeugt normalverteilte Zufallszahlen mit Mittelwert μ und Varianz σ^2. Datenanschlüsse
- Eingangsanschlüsse μ und σ: Mittelwert und Standardabweichung der Normalverteilung

Lognorm-Block
Erzeugt log-normalverteilte Zufallszahlen mit Mittelwert μ und Varianz σ^2. Datenanschlüsse
- Eingangsanschlüsse μ und σ: Mittelwert und Standardabweichung der Log-Normalverteilung

Visualisierung von Wahrscheinlichkeitsverteilungen

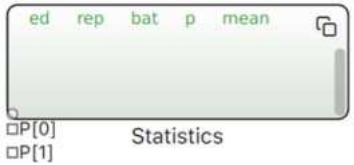
Wie können wir die Werte eine Zufallsvariable innerhalb von Quisim visualisieren?
- Chart-Block: Zeigt die Liste der Zufallszahlen als Diagramm an
- Report-Block: Zeigt die Zufallszahlen als Liste an
- Statistics-Block: Zeigt Statistiken über die Zufallszahl an

Chart-Block
Erzeugt ein Diagramm aus den bereitgestellten Werten. Datenanschlüsse
- Eingangsanschlüsse $P, \ldots, P[n]$: protokolliere einen oder mehrere anzuzeigende Werte

Report-Block
Wertetabelle mit: *ed* (Ausgabe), *rep* (Wiederholung), *time* (Zeit), *p* (Eingangs-Nr.), *v* (Wert). Datenanschlüsse
- Eingangsanschlüsse $P, \ldots, P[n]$: protokolliere einen oder mehrere anzuzeigende Werte

Statistics-Block
Erzeugt eine Statistikrechnung aus den bereitgestellten Werten. Datenanschlüsse
- Eingangsanschlüsse $P, \ldots, P[n]$: protokolliere einen oder mehrere anzuzeigende Werte.
- 'active' Daten-Anschluss (untere linke Ecke): Daten nur dann aufnehmen, wenn true.

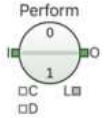
Der Statistics-Block hat eine Reihe von anpassbaren Parametern.
- numParameters: die Anzahl der Eingangsanschlüsse.
- timeAverage: wenn false, wird das arithmetische Mittel berechnet; wenn true, wird das zeitgewichtete Mittel berechnet.
- confidence, warmupDuration, batchDuration: später im Kurs behandelt.



Perform-Block

Modelliert Aktivität mit bestimmten Verzögerung. Verzögert jede am Eingangsanschluss I ankommende Entität um die angegebene Dauer. Entität wird so schnell wie möglich an Ausgangsanschluss O gesendet. Die Aktivität hat eine bestimmte Kapazität von Entitäten parallel. Sie ist unbegrenzt, wenn sie auf inf (Infinity, unendlich) gesetzt ist. Datenanschlüsse:

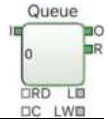
- Eingangsanschluss C: Kapazität (wie viele Entitäten parallel erlaubt sind)
- Eingangsanschluss D: Verzögerung (Bearbeitungszeit) in Millisekunden
- Ausgangsanschluss L: die Anzahl derzeit in diesem Block befindlicher Entitäten



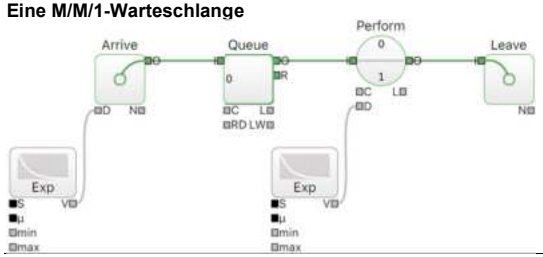
Queue-Block

Lagert Entitäten, die am Eingangsanschluss I ankommen. Die nächste Entität wird so schnell wie möglich an den Ausgang gesendet. Wenn der Ausgang blockiert ist und die Renege-Duration RD (max. Wartezeit) nach Entitäts-Ankunft überschritten ist, wird die Entität stattdessen an den Ausgangsanschluss R (outRenege) gesendet. Datenanschlüsse

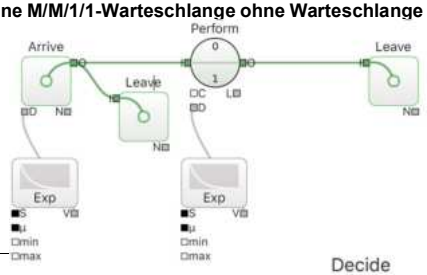
- Eingangsanschluss C: Kapazität (wie viele Entitäten die Warteschlange halten kann)
- Eingangsanschluss RD: max. Wartezeit (in Millisekunden), die bei Entitäts-Ankunft gesetzt wird
- Ausgangsanschluss L: Anzahl aktuell gelagerter Entitäten
- Ausgangsanschluss LW: die Wartezeit der letzten ausgegangenen Entität



Eine M/M/1-Warteschlange



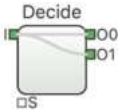
Eine M/M/1/1-Warteschlange ohne Warteschlange



Decide-Block

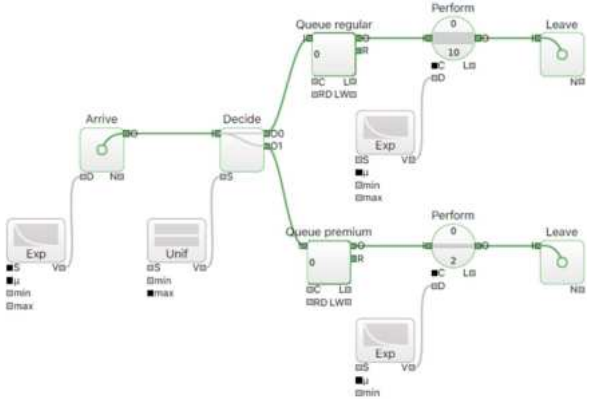
Eingehende Entität wird an Ausgang Os gesendet, wobei s durch Wert Selection-Parameters S bestimmt wird. Wenn 0 ≤ S < 1, dann wird die Entität an O0 gesendet; Wenn 1 ≤ S < 2, dann wird Entität an O1 gesendet (Abrundungslogik). Weitere Einstellungen können im Properties-Tab definieren. Datenanschlüsse

- Eingangsanschluss S: selektiert Ausgangsnummer (−1 selektiert keinen)



Priority Bearbeitung

- Betrachten Callcenter mit zwei verschiedenen Arten von Kunden: Reguläre und Premium-Kunden. Kunden werden zu verschiedenen Teilen des Callcenters geleitet.
- Callcenter hat 10 Mitarbeiter für reguläre Kunden und 2 Mitarbeiter für Premium-Kunden.
- Verhältnis zwischen Anrufen von regulären Kunden und Premium-Kunden beträgt 10:1.
- Alle Kunden haben eine exponentiell verteilte Gesprächsdauer mit einem Durchschnitt von 500 Sekunden.
- Alle Kunden haben eine exponentiell verteilte Zwischenanrufzeit mit einem Durchschnitt von 75 Sekunden.
- Anteil reguläre und Premium-Kunden:
- Verwenden Gleichverteilung im Intervall [0, 1.1). Bedeutet 10:1 Chance, dass solche gleichverteilte Zufallszahl u im Bereich [0, 1.0) liegt (und sie dem Ausgang O0 des Decide-Blocks zugeordnet wird).



Steering-Block

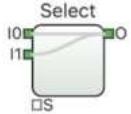
Einstellungen für Simulation wie Dauer, Anzahl Editionen und Replikationen, globaler Seed und Simulationsgeschwindigkeit. Anzeige aktuellen virtuellen Zeit und der aktuellen Replikation und Edition. Datenanschlüsse

- Eingangsanschlüsse D, E, R: Simulationsdauer D eines einzelnen Laufs, Anzahl der Editionen E und Anzahl der Replikationen R (wir haben $E \cdot R$ Läufe). Nur eine neue Edition setzt die Zufallsgeneratoren zurück.
- Eingangsanschluss S: Seed des globalen Zufallszahlengenerators.
- Eingangsanschluss V: Animationsgeschwindigkeit (relativ zu Millisekunden).
- Ausgangsanschlüsse T, F, EN, RN: hält die aktuelle Simulationszeit, ob Simulation fertig ist (true/false), Edition und Replikation.

Select-Block

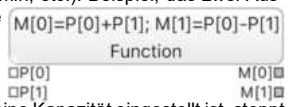
Eine eingehende Entität wird aus der im Selection Parameter S eingestellten Eingangs-Nummer entnommen und an den Ausgangsanschluss O gesendet. Die Abrundungslogik für S ist dieselbe wie beim Decide-Block. Weitere Eingangsanschlüsse können im Eigenschaften-Tab definiert werden. Datenanschlüsse

- Eingangsanschluss S: Selektion der Eingangsnummer (−1 wählt keinen Eingang).



Function-Block

Ruft den im Eigenschaften-Tab eingegebenen Funktionscode auf. Es verwendet das Eingabeparameter-Array P und schreibt die Ergebnisse in das Ausgabe-Array M. Mehrere Funktionsanweisungen müssen durch ein Semikolon getrennt werden. Alle JavaScript Math-Funktionen können verwendet werden (Math.max, Math.min, etc.). Beispiel, das zwei Ausgaben liefert (eine Addition und eine Subtraktion der zwei Parameter): Datenanschlüsse

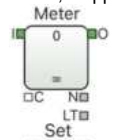


- Eingangsanschlüsse M[0], M[1], . . . : Funktions-Eingaben
- Ausgangsanschlüsse P[0], P[1], . . . : Funktions-Ausgaben

Meter-Block

Zählt die Anzahl der durch den Eingang I zum Ausgang O gegangenen Entitäten. Wenn eine Kapazität eingestellt ist, stoppt der Zähler das Weiterleiten von Entitäten, wenn diese Zahl erreicht ist. Datenanschlüsse

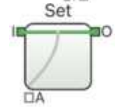
- Eingangsanschluss C: die Kapazität des Blocks
- Ausgangsanschluss N: Anzahl der durchgegangenen Entitäten
- Ausgangsanschluss LT: Zeitpunkt, zu dem die letzte Entität durchging



Set-Block

Setzt ein Attribut für jede durchlaufende Entität auf einen bestimmten Wert. Sie können den Attributnamen selbst wählen, jedoch nicht die schreibgeschützten Attribute ‘number’ und ‘innerEntitiesLength’ setzen.

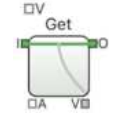
- Eingangsanschluss A: Name des Attributs
- Eingangsanschluss V: neuer Wert des Attributs



Get-Block

Ruft den Wert eines Attributs einer Entität ab, für die eine Durchleitung geprüft wird. Datenanschlüsse

- Eingangsanschluss A: Name des Attributs
- Ausgangsanschluss V: aktueller Wert des Attributs



Attribute von Entitäten

Zusätzlich zu selbst definierten Attributen gibt es Standard-Attribute und durch Blöcke gesetzte.

- Default attributes
  - ‘number’: unique number of an entity (read-only)
  - ‘color’: color of an entity
  - ‘symbol’: symbol of an entity
  - ‘innerEntitiesLength’: count of inner entities (read)
- Additional attributes set on passing a block
  - Arrive: ‘arrivalTime’
  - Provide: ‘provisionTime’
  - Queue: ‘lastEnqueuetime’, ‘lastWait’, ‘lastRenegeduration’
  - Perform: ‘lastDuration’

Function-Block Programmierung: Math siehe auch https://wiki.selfhtml.org/wiki/JavaScript/Objekte/Math

Beispiel: M[0] = Math.sqrt(P[0]) setzt M[0] auf die Quadratwurzel der Zahl in P[0].

Math.sqrt(a)	Quadratwurzel einer Zahl, also $\sqrt{a}$
Math.pow(a, b)	Potenz: a^b
Math.exp(a)	Exp(a)
Math.log(a)	Natürlicher Logarithmus einer Zahl: $\log a$
Math.abs(a)	absoluter Betrag einer Zahl: $\lvert a \rvert$
Math.round(a)	Zahl kaufmännisch runden
Math.floor(a)	Zahl abrunden: ⌊a⌋
Math.ceil(a)	Zahl aufrunden: ⌈a⌉
Math.max(a,b)	Grösste Zahl einsetzen (von beliebig vielen Zahlen), hier max{a, b}
Math.min(a,b,c)	Kleinste Zahl einsetzen, hier min{a, b, c}

Function-Block Programmierung: Entscheidungen

Beispiel: M[0]=(P[0]>=1)?1:0 setzt M[0] auf 1, wenn P[0]≥ 1 gilt, ansonsten auf 0.

Ternärer Operator a ? b : c → Wenn die logische Formel (oder bool-Variable) a wahr ist (true), dann wird b eingesetzt. Wenn a falsch ist (false), dann wird c eingesetzt. Logische Formeln:

- x < y ist erfüllt (true), wenn x echt grösser y ist;
- x <= y ist erfüllt (true), wenn x grösser gleich y ist;
- x > y ist erfüllt (true), wenn x echt kleiner y ist;
- x >= y ist erfüllt (true), wenn x kleiner gleich y ist;
- x == y ist erfüllt (true), wenn x gleich y ist;
- x != y ist erfüllt (true), wenn x ungleich y ist;
- (u) && (v) ist erfüllt (true), wenn u, v beide wahr (true) sind, sonst nicht;
- (u) || (v) ist erfüllt (true), wenn von u, v mindestens eins wahr (true) ist, sonst nicht.

Function-Block Programmierung: Arrays siehe auch https://wiki.selfhtml.org/wiki/JavaScript/Objekte/Array

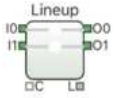
- Arrays werden erstellt mittels [a,b,. . .] mit beliebig vielen (oder keinen) Elementen.
- Zugriff auf Elemente des Arrays A per Index: A[a] mit a ∈ {0, 1, 2, . . . }.
- Beispiel: M[0] = [10,15,15,30,40][P[0]] setzt M[0] auf die Zahl 10, wenn P[0]= 0 ist oder auf 15, wenn P[0]= 1 ist, usw.

Array-Methoden, sei das Array A = [10,15,15,30,40] gegeben.

- A.indexOf(15) ergibt 1, den (ersten) Index bzw. die Position der Zahl 15 (sonst -1).
- A.lastIndexOf(15) ergibt 2, den letzten Index mit der Zahl 15 (sonst -1).
- A.some(n => n<20) ergibt true weil es mind. ein Element gibt, das die Bedingung erfüllt.
- A.every(n => n<20) ergibt false weil nicht jedes Element die Bedingung erfüllt.
- A.find(n => n<20) ergibt 30, das erste Element, das die Bedingung erfüllt.
- A.findIndex(n => n<20) ergibt 3, den Index des ersten Elements, das die Bed. erfüllt.

Lineup-Block

Der Block wartet, bis die benötigten Entitäten an verschiedenen Eingängen angekommen sind (damit sich die Entitäten aufstellen), und sendet dann die Entitäten gleichzeitig aus, jede an den entsprechenden Ausgang. Es muss 1 ≤ neededEntities ≤ numConnectors sein. Standardmässig ist neededEntities = numConnectors. Datenanschlüsse



- Eingangsanschluss C: Anzahl der aufzustellenden Entitäten (neededEntities, Standard: 2)
- Ausgangsanschluss L: Anzahl der aktuell aufgestellten Entitäten.

Pack-Block

Verpackt Entitäten, die am **Eingangsanschluss I** am Einlass ankommen, in eine **neu** erstellte **Entität**, die vom **Eingangsanschluss P** kommt. Diese Entität wird so schnell wie möglich an den Ausgangsanschluss O gesendet. Datenanschlüsse

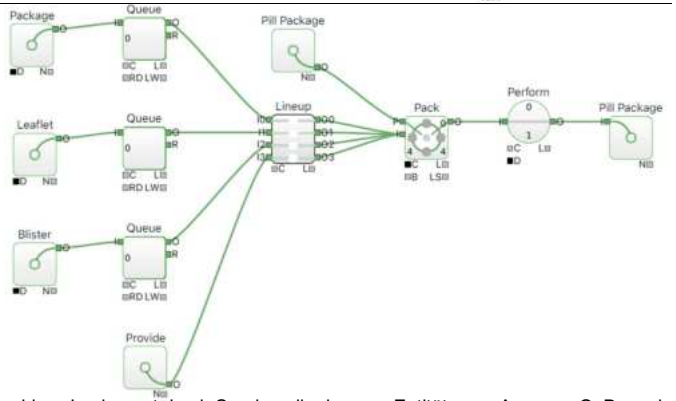
- Eingangsanschlüsse B und C: untere und obere Grenze der zu verpackenden Entitäten (Standard: B = C = ∞).
- Ausgangsanschluss L: Anzahl der im entstehenden Paket enthaltenen Entitäten.
- Ausgangsanschluss LS: Zahl enthaltener Entitäten im zuletzt versandten Paket.

Provide-Block

Stellt jederzeit eine neue Entität am Ausgang O bereit. Datenanschlüsse

- Ausgangsanschluss N: kumulative Anzahl bereitgestellter Entitäten.

Ein Verpackungsbeispiel

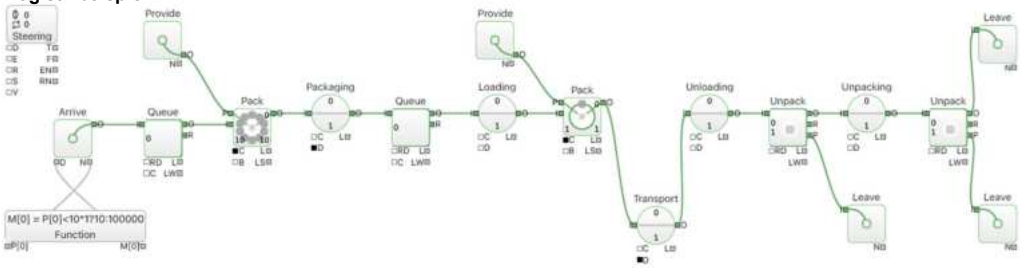


Unpack-Block

Entpackt Paketentität, die am Eingangsanschluss I ankommt durch Senden aller inneren Entitäten an Ausgang O. Danach wird Paketentität an Ausgangsanschluss P gesendet. Wie bei Queue kann Abbruchdauer (Renege duration) mit Eingangsanschluss RD eingestellt werden. Wenn ausgepackte Entität nicht innerhalb Abbruchdauer an nächsten Block gesendet werden kann, wird sie an Anschluss R gesendet. Datenanschlüsse

- Ausgangsanschluss L: die Anzahl der wartenden inneren Entitäten
- Ausgangsanschluss LW: die Wartezeit der zuletzt geschickten inneren Entität

Logistikbeispiel



Eingabemodellierung

Wie findet man die richtigen Eingabeparameter?

- Planung ist nützlich (vorläufige Messungen durchführen, Videos aufzeichnen, besondere Situationen erkennen)
- Daten während der Erhebung bereits auswerten und Plan ändern, wenn Daten unpraktisch oder überflüssig sind
- Homogene Messreihen erkennen und vereinigen (t-Test zur Überprüfung der Homogenität)
- Datenzensur beachten (künstlich verlängerte Zeiten; nicht aufgezeichnete Zeiten, wenn es zu lange gedauert hat)
- Abhängigkeit zwischen Variablen erkennen (durch Streudiagramme oder Korrelationstests); dazu gehört auch, eine Zeitabhängigkeit wie zur Tageszeit oder dem Wochentag zu erkennen
- Autokorrelation in scheinbar unabhängigen Beobachtungen erkennen
- Unterschied zwischen Eingabe- und Ausgabedaten erkennen (Eingaben sind nicht steuerbar; Ausgaben hingegen sind steuerbar, und diese wollen wir oft verbessern)

Fehlende Daten ergänzen

- Spezifikationen einer Maschine lesen (Drucker liefert 20 Seiten pro Minute und Toner hält durchschnittlich 5000 Seiten)
- In der Produktion bieten MTM-/REFA-Zeitmessungen einen Anhaltspunkt
  - Bottom-up-Ansatz: Methods-Time Measurement (MTM) ist eine Bewegungsanalysesystem, das hauptsächlich in industriellen Umgebungen verwendet wird, um Schritte für die Durchführung einer manuellen Aufgabe zu analysieren und daraufhin dafür eine Standardzeit festzulegen.
  - Top-down-Ansatz: REFA-Methodik.
- Expertenbefragung Schätzung Zeitdauer, mit optimistischen und pessimistischen Wert oder Variabilitätsangabe
- Physikalische Einschränkungen berücksichtigen (Tippgeschwindigkeit auf Tastatur, Laufgeschwindigkeit von Personen)

Wie findet man die richtigen Eingabeparameter?

- Oftmals ist die Eingabe arbiträr, d.h. sie wird durch eine Wahrscheinlichkeitsverteilung bestimmt.
- Was tun, um eine passende Wahrscheinlichkeitsverteilung zu finden?
- Oft geht es in der Eingabemodellierung um die Abbildung stochastischer Prozesse. Ergründet man die Natur des Prozesses, kann man sie mit der Natur bekannter Verteilungen vergleichen und dadurch die passendste auswählen.

Wie findet man eine gute Wahrscheinlichkeitsverteilung

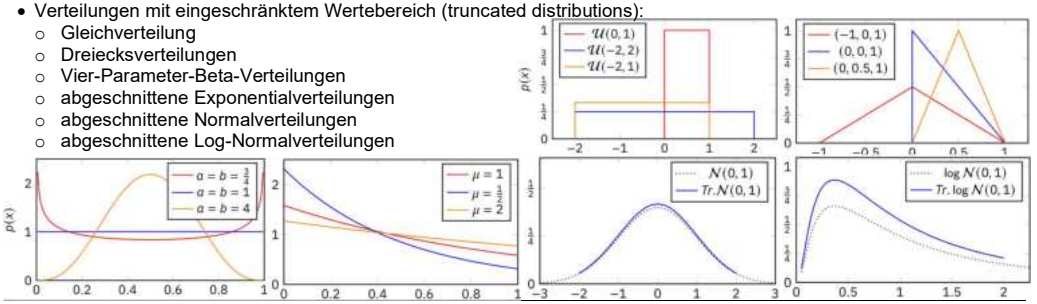
1. Daten sammeln:
   - durch ausreichende Anzahl Beobachtungen (100 bis 200) mit genügender Genauigkeit (zwei/drei signifikant Stellen)
   - Falls erforderlich, können auch Experteneinschätzungen verwendet werden
2. Eine Wahrscheinlichkeitsverteilung identifizieren:
   - mit einem Histogramm beginnen,
   - Kandidaten für Verteilungen auswählen (möglicherweise unter Verwendung Pearson-/CullenFrey-Diagrammen),
   - QQ-Diagramme zeichnen
3. Parameter auswählen: wenn möglich, diese aus den Daten ableiten
4. Schritte wiederholen, wenn Verteilung und Parameter nicht passen: t-Test oder Goodness-of-fit-Test (χ^2-/KS-Test)

Alternativen zu häufig verwendeten Wahrscheinlichkeitsverteilungen

- Gemessene Werte
  - gemessene Werte direkt und in der gleichen Reihenfolge als Eingabe verwenden
  - dies vermeidet die Abweichung von künstlichen Werten
  - erfordert repräsentative und entsprechend lange oder mehrere Messreihen
- Empirische Verteilung
  - Wiederholt werte zufällig aus der Menge aller gemessenen Werte ziehen (mit Zurücklegen)
  - geeignet für den Fall dass keine gängige Verteilung passt
  - diese Verteilung entspricht der Messwert-Verteilung
  - erfordert viele Messungen, da sich sonst dieselben Werte zu häufig wiederholen

Wahrscheinlichkeitsverteilungen mit eingeschränktem Wertebereich

- Ein Simulationsmodell sollte Verteilungen mit uneingeschränktem Wertebereich vermeiden.
- Dies schliesst die am häufigsten verwendete Familie der Normalverteilungen aus.
- Verteilungen mit eingeschränktem Wertebereich (truncated distributions):
  - Gleichverteilung
  - Dreiecksverteilungen
  - Vier-Parameter-Beta-Verteilungen
  - abgeschnittene Exponentialverteilungen
  - abgeschnittene Normalverteilungen
  - abgeschnittene Log-Normalverteilungen



Die Bedeutung von Wahrscheinlichkeitsverteilungen

- **Binomialverteilung**: Anzahl Erfolge aus n Versuchen (z.B. Anzahl defekter Computerchips bei Produktion von n Stück)
- **Negative Binomialverteilung**: Anzahl Versuche, die für k Erfolge erforderlich sind (z.B. Anzahl Chips, die untersucht werden müssen, um k defekte Chips zu entdecken)
- **Poissonverteilung**: Für Anzahl unabhängiger Ereignisse in festen Zeitintervall (z.B. Anzahl Kunden in einer Stunde). Parameter α, wobei der Schätzer α dem Mittelwert entspricht
- **Normalverteilung**: Für Summe der Dauer von Teilprozessen (Hinweis: Wert kann immer noch negativ sein). Parameter μ oder sein Schätzer entspricht dem Mittelwert, der Schätzer von σ^2 entspricht der Varianz
- **Lognormalverteilung**: Für Produkt von Teilprozesseigenschaften (z.B. Gewinn in Prozent, der sich aus Teilgewinnen zusammensetzt). Die Parameter entsprechen dem Logarithmus von Mittelwert und Varianz
- **Exponentialverteilung**: Für die gedächtnislose verbleibende Dauer eines Prozesses oder für die Zeit zwischen Ereignissen (während Poisson die Anzahl im Intervall zählt). Parameter λ entspricht dem Kehrwert des Mittelwerts
- **Weibullverteilung**: Generalisierte Exponentialverteilung (z.B. für Zeit bis zum Auftreten eines Defekts)
- **Erlangverteilung**: Summe mehrerer Exponentialverteilungen • **Gammaverteilung**: Generalisierte Erlangverteilung

Korrelierte Zufallsvariablen

- Korrelierte Zufallsvariablen werden verwendet, wenn die zu simulierenden Ereignisse oder Prozesse Abhängigkeiten oder Korrelationen untereinander aufweisen.
- **Warteschlangen**: Kunden, die in Gruppen ankommen, haben korrelierte Ankunftszeiten.
- **Supply Chains**: Lieferzeiten zwischen verschiedenen Stufen (z.B. Produktion, Transport, Lager)
- **Manufacturing**: Abhängigkeiten zwischen verschiedenen Produktions-Schritten wie Ausfall Maschine
- **Kommunikations-Netzwerke**: Ankunftszeiten von Datenpaketen oder Übertragungszeiten zwischen Netzwerk-Knoten.
- Zur Erzeugung korrelierter Zufallszahlen gibt es verschiedene Techniken, zum Beispiel:
  - eine **multivariate Wahrscheinlichkeitsverteilung**
  - eine gegebene **Kovarianz-Matrix** Σ und die **Cholesky-Zerlegung** $\Sigma = L \cdot L^T$

Modellprüfung

Warum Modellprüfung?

- **Problem**: Entscheidungsträger, die Informationen aus Ergebnissen von Simulationsmodellen verwenden, sind besorgt, ob Modell und Ergebnisse 'korrekt' sind. Anliegen wird durch Verifikation/Validierung Simulationsmodells angesprochen.
- **Ziel**: Ein genaues und glaubwürdiges Simulationsmodell erstellen.
- **Hinweis**: Simulationsmodelle sind ungefähre Nachahmungen von realen Systemen. Daher sollte Modell im Masse verifiziert und validiert werden, wie es für den beabsichtigten Zweck oder die Anwendung des Modells erforderlich ist. Verifikation und Validierung sind iterative Prozesse, die während der gesamten Entwicklung eines Modells stattfinden.

Modellverifikation

Im Kontext der Computersimulation ist die Verifikation eines Modells

- Prozess der Bestätigung, dass es in Bezug auf das **konzeptionelle Modell korrekt implementiert** ist
- wird getestet, um **Fehler** in der **Implementierung** des Modells zu **finden** und zu **beheben**
- dies umfasst
  - Überprüfung durch Experten
  - Erstellung Flussdiagrammen, die jede logisch mögliche Aktion enthalten
  - Überprüfung der Modellausgabe auf Plausibilität unter verschiedenen Einstellungen der Eingabeparameter
  - Verwendung eines interaktiven Debuggers

Modellvalidierung

Validierung überprüft Genauigkeit der Darstellung des realen Systems durch Modell. Gibt viele Ansätze dafür. Ansätze reichen von subjektiven Überprüfungen bis hin zu objektiven statistischen Tests. Dreistufiger Ansatz zur Modellvalidierung:

1. Erstelle ein Modell mit hoher Augenscheinvalidität
2. Validiere die Modellannahmen
3. Vergleiche Eingabe-Ausgabe-Transformationen des Modells mit Eingabe-Ausgabe-Transformationen realen Systems

Augenscheinvalidität

- Modell, erscheint Personen, die mit realen **System vertraut** sind, als **vernünftige Nachahmung** des realen Systems.
- Augenscheinvalidität ist sehr **schwache Form** der Validierung, aber am **Anfang** des **Validierungsprozesses wertvoll**.
- **Glaubwürdigkeit** Modell für Benutzer und **Vertrauen** der Benutzer in Modell **steigt**.
- **Empfindlichkeit** gegenüber **Modelleingaben** kann zur Beurteilung der Augenscheinvalidität verwendet werden.

Strukturelle Annahmen

- Annahmen darüber, wie System funktioniert und wie es physisch angeordnet ist.
- **Beispiel**: Anzahl Server in Fast-Food-Durchfahrspur und wie sie genutzt werden, wenn es mehr als einen gibt?
- **Beispiel**: Arbeiten Server parallel, wobei Kunde Transaktion durch Besuch eines einzelnen Servers abschliesst, oder nimmt Server Bestellungen entgegen und bearbeitet Zahlung, während andere Bestellung vorbereitet und serviert?

Datenannahmen

- Müssen **ausreichend geeignete Daten verfügbar** sein, um konzeptionelles **Modell** zu **erstellen** und Modell zu **validieren**. Das **Fehlen** geeigneter **Daten** ist oft Grund, warum **Validierungsversuche** eines Modells **scheitern**.
- **Daten** sollten **verifiziert** werden, um sicherzustellen, dass sie aus zuverlässigen Quelle stammen. Typischer Fehler ist die Annahme einer unangemessenen statistischen Verteilung der Daten.
- Angenommene **statistische Modell** sollte mit **Anpassungstests** und anderen Techniken **getestet werden**. Beispiele für Anpassungstests sind der Kolmogorov-Smirnov-Test und Chi-Quadrat-Test. Ausreisser in Daten überprüfen.

Validierung von Eingabe-Ausgabe-Transformationen

- Modell wird als Eingabe-Ausgabe-Transformation betrachtet.
- Validierungstest: Ausgaben betrachteten System mit Modellausgaben für gleiche Eingabebedingungen zu vergleichen.
- Um Test durchzuführen, müssen Daten verfügbar sein, die während Beobachtung des Systems aufgezeichnet wurden.
- Modellausgabe, die von primärem Interesse ist, sollte als Leistungsmass verwendet werden.
- **Beispiel.** Für Fast-Food-Drive-Through, bei dem Eingabe in Modell die Ankunftszeit Kunden ist und Leistungsmassnahme die durchschnittliche Wartezeit Kunden in Schlange, werden tatsächliche Ankunftszeit und die in Schlange verbrachte Zeit Kunden im Drive-Through aufgezeichnet. Modell wird mit aufgenommenen Ankunftszeiten ausgeführt und durchschnittliche Wartezeit im Modell wird mit tatsächlichen Wartezeit in Schlange unter Verwendung Tests verglichen.
- Simulationsziele:
  - Erhalten zuverlässigsten Aussagen über unbekannte Grössen
  - Erfordert korrekte statistische Ergebnisanalyse

Beispiel

- **Ziel**: Schätzung Mittelwerts θ eines Objektattributs
- **Methode**: Beobachte Zustände Objektattribute während Simulation und leite Schätzung $\hat{\theta}$ von θ ab
- **Problem**: Schätzung hängt vom spezifischen Simulationslauf ab. Weiterer Simulationslauf ergibt andere Schätzung
- **Frage:** Wie erhält man ausreichend gute Schätzung? **Antwort**: Statistische Analyse der Simulationsergebnisse

Statistische Hypothesen-Tests

- Stat. Hypotest. mit t-Test können als Grundlage verwenden, um Modell als gültig akzeptieren oder ungültig abzulehnen.
- Die zu testende Hypothese ist H_0 : Modell-KPI = System-KPI vs. H_1 : Modell-KPI ≠ System-KPI.
- Test wird für bestimmte Stichprobengrösse und Signifikanzgrad α durchgeführt.
- Um Test durchzuführen, werden Reihe von n statistisch unabhängigen Replikationen des Modells durchgeführt und ein Durchschnitts- oder Erwartungswert $\bar{y}$ für die interessierende Variable y erzeugt.
- Dann wird Teststatistik t_0 für gegebenen Werte n, $\bar{y}$ und den beobachteten Wert μ_y für System berechnet.
- Berechne $t_0 = \frac{\bar{y}-\mu_y}{\sigma_y/\sqrt{n}}$ mit σ_y als Standardabweichung der Stichprobe y.
- Berechnen den p-Wert als $p = 2 \cdot \left(1 - T(|t_0|, n-1)\right)$ mit Student-t-Verteilung T.
- Wenn $p > \alpha$, dann Hypothese H_0 ablehnen, d.h. das Modell muss angepasst werden.

Simulationsverlauf

- Zustand eines Objektattributes über die Simulationszeit t
- Als Folge von Zuständen $(t_i, w(t_i))$, für $i = 1, \ldots, n$ mit **Zeitpunkt** t_i und **Zustand** $w(t_i)$, der nach Zustandswechsel gilt.

Weshalb stationärer Zustand?

- Gewünscht: Kennzahlen im stationären Zustand ermitteln → Problem: Erst im Langzeitverhalten erkennbar
- Grund: Einschwingzeit zu Beginn unterscheidet sich von Langzeitverhalten→ Lösung: Einschwingzeit entfernen
- Wann stationärer Zustand? Wenn Wahrscheinlichkeit, dass Kennzahl bestimmten Wert annimmt, sich nicht mehr ändert.
- nicht von Startzustand und Startzeit abhängig
- gleichbleibende Erwartungswert und Varianz
- unvorhersehbare und starke Veränderung der Kennzahl dennoch möglich

Beispiel für stationären vs. transienten Zustand

- Betrachten M/M/1-Modell mit Startwarteschlangenlänge k.
- Angenommen, Zwischenankunftszeit $1/\lambda = 100$ und durchschnittliche Servicezeit $1/\mu = 99$.
- Stationär Zustand: erw. Wartezeit $W_q = \lambda/(\mu(\mu-\lambda)) \approx 98$ & erw. Warteschlangenlänge $\lambda^2/(\mu(\mu-\lambda)) \approx 98$
- Um Warteschlange mit bestimmten Anzahl von Entitäten vorzuladen, können wir **Meter-Block** verwenden.

Meter-Block

Zählt Anzahl der durch Eingang I zum Ausgang O gegangenen Entitäten. **Wenn Kapazität** C **eingestellt** ist, stoppt der Zähler das Weiterleiten von Entitäten, wenn diese Zahl erreicht ist. Ausgangsanschluss LT: Zeitpunkt, zu dem die letzte Entität durchging, Ausgangsanschluss N: Anzahl der durchgegangenen Entitäten. Provide zu I

Validierung M/M/1-Warteschlange

Warteschlangen-Vorbelastung = 98

Für Validierung von N_q wird t-Tests angewendet (in Excel). Anzahl simulierter Entitäten pro Simulationslauf: 100'000.

mu(N_q)
99

Data	Runs	10	20	50	100
89.8632733	mean	90.4206277	86.1507114	94.1990359	94.2116888
95.2672949	stddev	50.8787318	38.4521181	49.7156106	50.2376617
219.56733	t0	-0.0533236	-0.0747212	-0.0136569	-0.0095313
87.754165	p	0.95863896	0.94121765	0.9891592	0.9924144
123.847806					

Summary of p-values

Arrivals	Runs	10	20	50	100
100		0.988	0.965	0.967	0.993
1000		0.822	0.879	0.945	0.964
10000		0.889	0.972	0.987	0.980
100000		0.959	0.941	0.989	0.992

Fazit: sowohl eine Erhöhung der Anzahl Simulationsläufe als auch der Anzahl durchlaufener Entitäten verbessert die statistische Signifikanz.

Bestimmen der Einschwingzeit

- Keine automatische Methode ist stets richtig. Einfach und wirkungsvoll: **Grafische Methode nach Peter D. Welch.**
- Das k'te Ereignis beendet die Einschwingzeit
- Bestimmung von k:
  - 1 für verschiedene k's den empirischen Mittelwert eines Wertes ab k berechnen
  - 2 kleinste k wählen, für das sich Mittelwert nur noch wenig ändert
- Ist nur effektiv, wenn man mehrere Simulationsläufe betrachtet

Mittelwertberechnung

- Gesucht: Mittelwert von Attribut-Werten $w(t_i)$ für $i = 1, \ldots, n$ Problem: Arithmetisches Mittel ist falsch $\frac{1}{n}\sum_{i=1}^{n} w(t_i)$
- Grund: der zeitliche Anteil ist nicht berücksichtigt → Lösung: Zeitliches Mittel $\hat{\theta} = \frac{1}{t_n - t_1}\int_{t_1}^{t_n} w(t)\,dt$
- Vereinfachung, da sich Werte nur zu diskreten Zeitpunkten ändern $\hat{\theta} = \frac{1}{t_n - t_1}\sum_{i=1}^{n-1} w(t_i)\,(t_{i+1} - t_i)$ dabei beginnt Wert $w(t_i)$ bei t_i und endet bei t_{i+1}. Der letzte Wert $w(t_n)$ ist irrelevant, da er über keine Zeit in den Mittelwert einfliesst
- Beispiel: **Werte für Zeitdauern →** Lösung: Differenz zwischen Start-Zeitpunkt s_i und Ende-Zeitpunkt e_i. Mittelwert aus n Zeitdauern: Arithmetisches Mittel: $\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n}(e_i - s_i)$

Aussagekraft eines Mittelwerts $\hat{\theta}_n$

- Unmöglich, $n \to \infty$ zu betrachten, um den Erwartungswert θ exakt zu berechnen. → Dafür hilft ein weiteres Werkzeug: Wir bestimmen ein Konfidenzintervall (bzw. Konfidenzradius) für ein vorgegebenes Signifikanzniveau $\alpha \in (0,1)$.
- Umfasst $\hat{\theta}_n$ und beinhaltet θ zur Wahr' von $\left(1 - \frac{\alpha}{2}\right) \cdot 100\,\%$. «Wert ist mit 95% Wahr' durchschnittlich 100 ± 10»

Bestimmung eines Konfidenzintervalls

- Dafür sind unabhängige, normalverteilte Messwerte nötig. Tatsächlich ist ein empirischer Mittelwert $\hat{\theta}$ normalverteilt.
- Durch mehrere unabhängige Simulationsläufe erhalten wir mehrere unabhängige, normalverteilte Messwerte $\hat{\theta}_1, \ldots, \hat{\theta}_m$.
- Für den empirischen Mittelwert aus den empirischen Mittelwerten können wir also das Konfidenzintervall bestimmen.

Exkurs: Student's t-verteilte Zufallsvariable T_m

α	$t_{10,\alpha/2}$	$t_{20,\alpha/2}$	$t_{40,\alpha/2}$	$t_{100,\alpha/2}$	$t_{\infty,\alpha/2}$
0.1	3.169	2.845	2.704	2.626	2.576
0.05	3.581	3.153	2.971	2.871	2.807
0.01	4.587	3.850	3.551	3.390	3.291

- Wähle $t_{m-1,\frac{\alpha}{2}}$ sodass für t-verteilte Zufallsvariable T_m gilt, dass $\mathbb{P}[-t_{m-1,\frac{\alpha}{2}} \leq T_m \leq t_{m-1,\frac{\alpha}{2}} \approx 1-\alpha$
- In der (zweiseitigen) t-Tabelle können wir den passenden $t_{m-1,\frac{\alpha}{2}}$-Wert ablesen.
- **Bestimmung des Konfidenzintervalls.** Bereits bei kleinen m gilt für Θ_m Student's t-Verteilung mit $m-1$ Freiheitsgraden. Dann gilt bereits „mittelgrosse" m, dass $\mathbb{P}\left[\hat{\theta}_m - t_{m-1,\frac{\alpha}{2}} \cdot \sqrt{\frac{S_m^2}{m}} \leq \theta \leq \hat{\theta}_m - t_{m-1,\frac{\alpha}{2}} \cdot \sqrt{\frac{S_m^2}{m}}\right] \approx 1-\alpha$ empirischen Varianz $S_m^2 = \frac{1}{m-1}\sum_{i=1}^{m}(\hat{\theta}_i - \theta)^2$ Konfidenzradius ist somit $t_{m-1,\frac{\alpha}{2}} \cdot \sqrt{\frac{S_m^2}{m}}$. Je kleiner S_m^2 und je höher m, desto kleiner ist Radius.
- **Skalierung der empirischen Varianz S_m^2**: proportional zu $\frac{1}{m}$ (für Anzahl Simulationsläufe m), proportional zu $\frac{1}{\hat{n}}$ (für mittlere Anzahl Messwerte je Simulationslauf n)
- Beispiel: Wie viel mehr Simulationsläufe sind für Halbierung der Varianz nötig? → Doppelt so viele Simulationsläufe.
- Skalierung des Konfidenzradius α: proportional zu $\frac{1}{\sqrt{m}}$ und proportional zu $\frac{1}{\sqrt{\hat{n}}}$.
- Wie viel mehr Messungen sind für Halbierung von α nötig? ▶ Vierfache Simulationsläufe ($m' = 4m$) z. B. 90% → 95%.

Problematik der Einschwingzeit

Wie ist unser Umgang mit der Einschwingzeit?

- Länger simulieren (Dauert zu lange?)
- Sinnvolle Startwerte (Welche Werte sind sinnvoll?)
- Einschwingzeit abschneiden (Wann ist sie zu Ende?)

Da wir für ein Konfidenzintervall mehrere Wiederholungen brauchen, verstärkt sich das Problem: Um einen Performancewert $\mathcal{L}$ aus m Simulationsläufen zu bestimmen, werden m Einschwingzeiten „entsorgt".

Batch-Mittelwert-Methode

- zur Bestimmung von $\mathcal{L}$ mit Konfidenzintervall bei nicht-terminierenden Zufallszahlenreihen
- Brauchen dafür m Zeitreihen, die jeweils die Dauer t_n umfassen, und m ist z.B. in $[10, 30]$
- Simulation in stationären Phase verlängern, bis zur Zeit $(t_k + m \cdot t_n)$
- Zeit nach t_k dann in m Messreihen (Batches) unterteilen. Die Messreihe $i = 1, \ldots, m$ geht dann von $t_k + t_n \cdot (i-1)$ bis $t_k + t_n \cdot i$. ▶ Damit ergibt sich Performancewert $\mathcal{L}_i$
- Vorteil: nur noch der Simulationslauf bis t_k wird „entsorgt" (geringere Rechenzeit). Messreihen können korreliert sein.
- Dann sind $\mathcal{L}_1, \mathcal{L}_2, \ldots$ nicht mehr unabhängig.

Bestimmung der Korrelation von Batches

- Testen Autokorrelation aufeinanderfolgenden Batch-Werten (AR(1)-Modell): $\mathcal{L}_1$ mit $\mathcal{L}_2$, $\mathcal{L}_2$ mit $\mathcal{L}_3$, $\mathcal{L}_3$ mit $\mathcal{L}_4$ usw. bis $\mathcal{L}_{m-1}$ mit $\mathcal{L}_m$ → Auf diese Weise ist die AR(1)-Autokorrelation $\rho = \frac{\sum_{i=1}^{m-1}(\mathcal{L}_i - \hat{\mathcal{L}})(\mathcal{L}_{i+1} - \hat{\mathcal{L}})}{\sum_{i=1}^{m}(\mathcal{L}_i - \hat{\mathcal{L}})^2}$ mit $\hat{\mathcal{L}} = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}_i$

Korrelation von Batches: Effekte

- Falls AR(1)-Korrelation ρ vorhanden ist, schlägt Fishman Verdopplung der Batchlänge vor. Was kann passieren?
- Korrelation sinkt beim Verdoppeln der Batchlänge unter 0.05. → unkritisch.
- Negative Korrelation → unkritisch.
- Korrelation sinkt beim Erhöhen Batchlänge zuerst, steigt aber wieder → ganze Simulationsdauer erhöhen. Mildert Korrelationseffekt ab, Rechenzeit steigt jedoch an.

Analyse von Simulationsläufen in Quisim

- Steering Block "replications": Zahl der Simulationsläufe festlegen
- Steering Block "duration": Dauer eines Simulationslaufs festlegen
- Statistics Block "warmupDuration": Dauer Einschwingzeit festlegen
- Statistics Block "numBatches": Anzahl Batches festlegen
- Statistics Block "batchDuration": Batch-Dauer überschreiben (deaktiviert 'numBatches')

Es gilt stets: $batchDuration = \frac{duration - warmupDuration}{numBatches}$

Terminierende versus nicht-terminierende Simulationen

Bisher haben wir nicht-terminierende Simulationen betrachtet:

- Können prinzipiell unbeschränkt laufen.
- Performance-Werte beziehen sich dann auf stationäre Verhalten (sofern es existiert).
- Wann ist stationäre Verhalten erreicht?
  - Nach Einschwingen
  - Wenn Wahrscheinlichkeit, dass beliebiger Zustand erreicht wird, sich nicht mehr ändert.
- Wann ist Messreihe lang genug? Wenn Konfidenzintervall so klein ist, wie gewünscht ist (Skalierung für n Messwerte)

Terminierende Simulationen hingegen tauchen oft ebenfalls in der Praxis auf:

- Haben ein natürliches Ende, z. B. Schichtende, Ladenschluss.
- Performance-Werte beziehen sich dann auf die gesamte Simulationszeit.
  - Hier interessiert uns der gesamte Zeitbereich, es wird also keine Einschwingzeit entfernt.
  - Daher sind hier mehrere Simulationsläufe zu betrachten.

Systemvergleich

- Ziel: Statistisch abgesicherter Vergleich zweier Systeme in ihrer Performance $\mathcal{L}$.
- Problem: empirische Mittelwerte allein sind nicht aussagekräftig (Varianz!).
- Auch direkter Vergleich zweier Konfidenzintervalle kann irreführend sein. → Konfidenzintervall Differenz vergleichen.
- Hinweis: Einen Vergleich von mehr als zwei Systemen, kann auf einen paarweise Vergleich zurückgeführt werden.
- System A hat den Wert $\mathcal{L}_A$, System B den Wert $\mathcal{L}_B$. Empirische Differenz ist dann $\hat{\mathcal{D}} = \hat{\mathcal{L}}_A - \hat{\mathcal{L}}_B$.
- Frage: Wie ist das Konfidenzintervall $[a, b]$ für empirische Differenz
  - Negatives Intervall $[a, b]$: dann ist $\hat{\mathcal{L}}_A < \hat{\mathcal{L}}_B$
  - Positives Intervall $[a, b]$: dann ist $\hat{\mathcal{L}}_A > \hat{\mathcal{L}}_B$
  - $0 \in [a, b]$ (Intervall beinhaltet die 0): dann ist keine Aussage möglich. Abhilfen: Simulation verlängern, damit $[a, b]$ bzw. die Varianz kleiner wird. **Korrelation** zwischen einzelnen Simulationsläufen **erzeugen** (!)

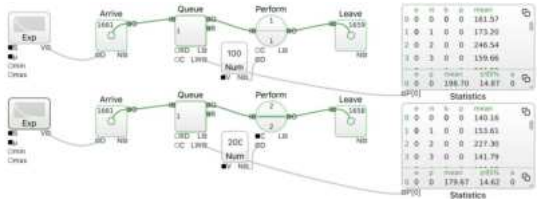
Weshalb will man eine Korrelation zwischen Simulationsläufen?

- Das Konfidenzintervall für $\hat{\mathcal{D}}$ ist bei m Simulationsläufen $\hat{\mathcal{D}} \pm t_{m-1,\alpha/2} \cdot \sqrt{\frac{Var[\hat{\mathcal{D}}]}{m}}$ mit $Var[\hat{\mathcal{D}}] = Var[\hat{\mathcal{L}}_A - \hat{\mathcal{L}}_B] = Var[\hat{\mathcal{L}}_A] + Var[\hat{\mathcal{L}}_B] - 2 \cdot CoVar[\hat{\mathcal{L}}_A, \hat{\mathcal{L}}_B]$, $CoVar[\hat{\mathcal{L}}_A, \hat{\mathcal{L}}_B] := (*)$ ist dabei 0, wenn $\hat{\mathcal{L}}_A, \hat{\mathcal{L}}_B$ unabhängig sind.
- Varianz wird verringert, wenn man positive Korrelation zwischen den Performancewerten $\hat{\mathcal{L}}_A, \hat{\mathcal{L}}_B$ erzeugt.

Korrelation zwischen den Performancewerten erzeugen

- Ziel: zwei Systeme vergleichen mit geringer Varianz in der empirischen Differenz
- Idee: positive Korrelation zwischen den Performancewerten $\hat{\mathcal{L}}_A, \hat{\mathcal{L}}_B$ erzeugen.
- Lösung: die Common Random Numbers (CRN) Methode → Synchronisierung der Zufallszahlenströme in der Simulation beider Systeme. z. B. nutzt man jeweils dieselben Ankunfts- und Bedienzeiten der Kunden.

Systemvergleich mit synchronisierten Zufallszahlen

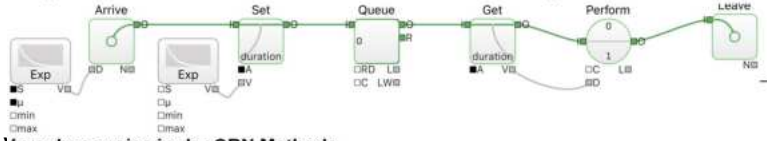


- System B: Ein M/D/2 Modell mit halber Bedienrate $\mu = \frac{1}{200}$ (konstant), gleicher Exp-Seed
- Zwar lässt sich bereits erkennen, dass die mittlere Wartezeit verschieden ist, jedoch überlappen die Konfidenzintervalle von System A und B. So ist keine Aussage möglich.
- Mittlere Wartezeit ist in beiden Systemen oft beide gemeinsam hoch oder niedrig.
- Die Performance hängt also von den konkreten Kundenankünften ab.
- Differenz variiert hingegen recht wenig und daher ist die Varianz klein.
- Wir erkennen, dass das 95%-Konfidenzintervall der Differenz eindeutig im positiven Bereich ist. Somit hat System B klar kleinere Wartezeiten.

Replication i	$\hat{\mathcal{L}}_A$	$\hat{\mathcal{L}}_B$	$\hat{\mathcal{D}}$
1	161.6	140.2	21.4
2	173.2	153.6	19.6
3	246.5	227.3	19.2
4	159.7	141.8	17.9
5	200.3	181.0	19.4
6	256.3	235.4	20.9
7	172.8	157.3	15.5
8	178.7	156.5	22.3
9	240.6	223.0	17.7
10	180.7	162.5	18.1
11	185.7	166.9	18.8
12	205.1	187.8	17.3
13	215.8	197.7	18.2
14	192.3	173.8	18.5
15	251.8	230.9	21.0
16	201.2	183.1	18.1
17	206.3	186.7	19.7
18	144.9	128.8	16.1
19	183.8	164.9	18.9
20	216.5	194.2	22.3
mean	198.7	179.7	19.0
stdev	31.8	31.2	1.8
radius	14.9	14.6	0.9
lower	183.8	165.0	18.2
upper	213.6	194.3	19.9

Vorgehensweise in der CRN Methode

- Zufallszahlenstrom in beiden Systemen gleich eingeben
- Pseudozufallszahlengeneratoren erlauben das Setzen eines Seeds
- Gleicher Seed ⇒ gleicher Pseudozufallszahlenstrom.
- Zum selben Objekt gehörende Werte stets auch derselben Zufallszahl zuordnen (erhöht Korrelation noch weiter!), z.B. denselben Kunden in beiden Systemen dieselbe Ankunfts-/Bedienzeit zuordnen durch Setzen Kundenattribut bei Ankunft im System
- Eigenschaften von Entitäten können in Quisim durch Attribute gesetzt werden:



Vorgehensweise in der CRN Methode

- Zwischen verschiedenen Aspekten ist CRN nicht sinnvoll, z.B. Auftragseingänge und Maschinenwartungszeiten sollten unabhängige Zufallszahlenströme erhalten.
- CRN erhöht Korrelation nicht immer! (vgl. „nicht-monotone Simulationen"). Korrelation kann sogar kleiner werden.
- Daher: vorab testen, ob die CRN Methode angebracht ist.

CRN Methode in Quisim

- Je Random-Block kann ein Seed gesetzt werden. Dessen Zufallszahlenstrom wird in Simulationslauf mit der Replikationsnummer $i = 0, \ldots, numReplications - 1$ zurückgesetzt auf $seed + i$.
- Der Steering-Block bestimmt $Anzahl\ Simulationsläufe = numEditions \cdot numReplications$.
- Mit jeder neuen Edition beginnt Quisim wieder mit Replikation $i = 0$.
- Gibt globalen Zufallszahlenstrom, aus dem sich Random-Blöcke bedienen, wenn sie selbst keinen Seed gesetzt haben.
- Für globalen Zufallszahlenstrom wird Seed Steering-Block gesetzt. Ist kein globaler Seed gesetzt, wird aktuelle Uhrzeit als Seed verwendet

Fallstudien

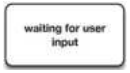
Erstellen von Simulationsmodellen aus UML-Zustandsdiagrammen

UML (Unified Modeling Language) ist Modellierungssprache, die verwendet wird, um **System prägnant** zu **beschreiben**. UML-Diagramme werden im Bereich der Softwaretechnik weit verbreitet verwendet. Ein Zustandsdiagramm modelliert das **Verhalten Systems** oder **Objekts** als Reaktion auf Ereignisse. Zustandsdiagramm besteht hauptsächlich aus

- Zuständen
- Ereignissen
- Übergängen
- Aktionen

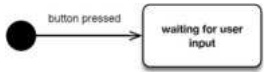
Zustände

Zustände beschreiben **Phase** oder **Zustand** des **Systems**. Werden durch **Rechteck** dargestellt, in dem **Name** des **Zustands** geschrieben steht.



Ereignisse

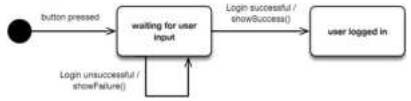
- Ereignis kann **Signal** oder **Veränderung** in Umgebung des **Systems** sein. Ereignisse können zu einer Änderung des Systemzustands führen oder auch nicht.
- Im Diagramm ist Knopf gedrückt das Ereignis, das den Übergang vom Anfangszustand zum Zustand «Warten auf Benutzereingabe» auslöst.



Übergänge

Übergänge werden als Pfeile von einem Zustand zum anderen dargestellt. Sie können drei Indikatoren aufweisen:

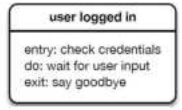
- **Ereignis**, das den Übergang **auslöst**
- **Wächterbedingung**, die wahr sein muss, damit Übergang stattfindet
- **Aktionsanweisung**, die als Ergebnis Übergangs ausgeführt wird



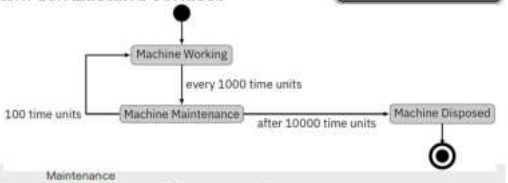
Aktionen

Zustandsaktionen werden aufgerufen, wenn sich System im bestimmten Zustand befindet. Es gibt 3 Arten von Zustandsaktionen:

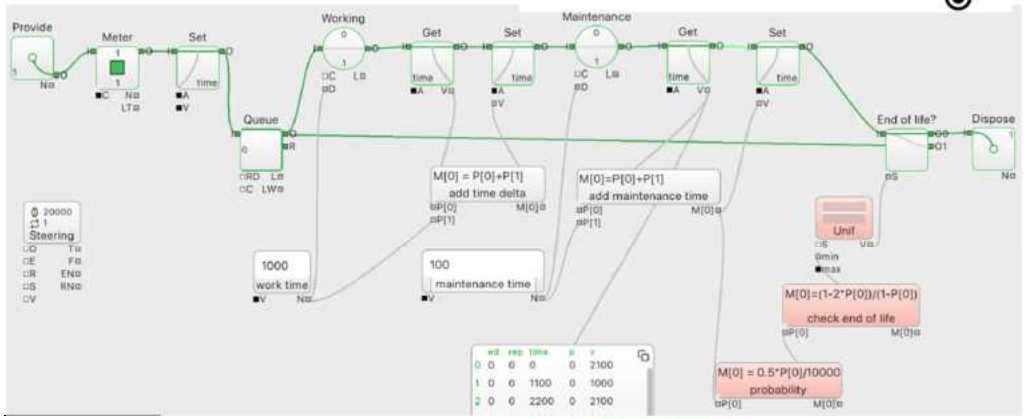
- **Eintrittsaktionen**: Werden ausgeführt, wenn System **Zustand betritt**
- **Ausführungsaktionen**: Werden kontinuierlich ausgeführt, **während** System im **Zustand** ist
- **Austrittsaktionen**: Sie werden ausgeführt, wenn das System den **Zustand verlässt**



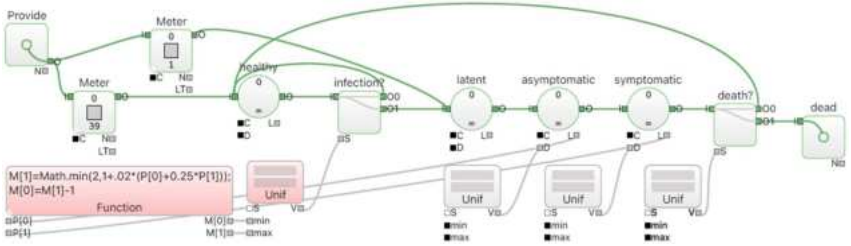
Ein einfaches Zustandsdiagramm einer Maschine.



mit probabilistischer Entsorgungszeit
disposal probability
p = 0.5 · totalTime/10000



Infektion (p = 0.02 · (a + 0.25 · s)), wobei a die Zahl der asymptomatischen und s die syptomatischen Personen im System ist. Der Gedanke ist, dass die Ansteckwahrscheinlichkeit bei 2 % ist und 25% der symptomatischen Personen anwesend sind.



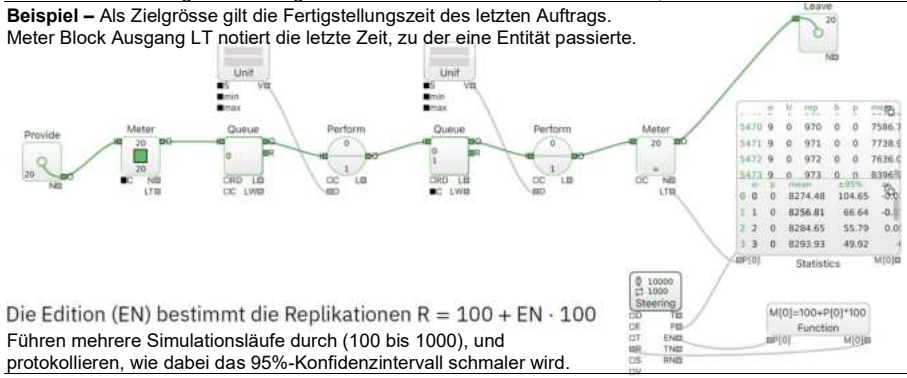
Produktionssysteme

- Produktionssysteme sind komplex aber andererseits meist einigermassen geschlossen.
- Simulation ist ein hilfreiches Werkzeug, um Produktionssysteme zu analysieren und zu verbessern.
- Was sind die grundlegenden Ablaufformen?
  - **Parallel Resources:** Aufträge werden auf mehrere Ressourcen verteilt, die **parallel** arbeiten.
  - **Flow Shop Produktion:** Aufträge gehen auf mehrere Ressourcen **nacheinander**
  - **Job Shop Produktion:** Aufträge gehen auf mehrere Ressourcen **nacheinander**; **Reihenfolge** ist auftragsspezifisch

Flow Shop Produktion

- Die Bestimmung einer Reihenfolge kann als Optimierungsproblem aufgefasst werden.
- Ist über die Auftragsdaten wenig bekannt und sind diese stark stochastisch, werden Simulationen interessant.

Beispiel – Als Zielgrösse gilt die Fertigstellungszeit des letzten Auftrags.
Meter Block Ausgang LT notiert die letzte Zeit, zu der eine Entität passierte.



Die Edition (EN) bestimmt die Replikationen R = 100 + EN · 100

Führen mehrere Simulationsläufe durch (100 bis 1000), und protokollieren, wie dabei das 95%-Konfidenzintervall schmaler wird.

Systemvergleich

- Neues System mit bspw. Reparaturen. Da wir Seeds gesetzt haben, können wir einen Systemvergleich machen.
- Speichern in a und b jeweils die Mittelwerte aller Replikationen ab. Wir bilden dann die Differenz D.
- Damit erhalten wir zwanzig 95%-Konfidenzintervalle für D. Keines davon enthält die 0.
- Es ist also eindeutig, dass das System ohne Defekte besser ist.

Vollständiger Versuchsplan

- Für jeden Parameter (Faktor) eine Liste möglicher Werte festlegen (diskreter Wertebereich)
- ▶ Versuchsplan testet alle möglichen Kombinationen aus Parameterwerten. z.B. gibt es bei k Parametern je 2 möglichen Werten („Stufen“) genau 2^k Kombinationen („2^k-Versuchsplan“)
- Bei $k = 2$ Faktoren gibt es $2^2 = 4$ Versuche.
- Wir tragen bei jedem Versuch −1 oder +1 ein, je nachdem, ob der Faktor aktiv/inaktiv (bzw. hoch/niedrig) ist.
- Nun kann man per «Kontrastmethode» den «Effekt» jedes Faktors berechnen.

Versuch	Faktor A	Faktor B	Ergebnis $\mathcal{L}$
1	−1	−1	82.02
2	+1	−1	14.02
3	−1	+1	2.33
4	+1	+1	0.89

2^k-Versuchsplan: Kontrastmethode

Effekt EZ des Parameters Z auf das System

- zeigt durchschnittliche Änderung eines Performance-Werts $\mathcal{L}$. Effekt entsteht durch Stufenwechsel des Parameters unter Beibehaltung aller anderen Parameter. Effekt wird über alle 2^k Versuche hinweg betrachtet.
- Vorgehensweise: Versuche unterteilen in Menge Z_{+1}, bei denen Parameter Z auf der hohen Stufe ist, und in Menge Z_{-1}. Dann ist $E_Z = \frac{1}{|Z_{+1}|}\sum_{v\in Z_{+1}}\mathcal{L}_v - \frac{1}{|Z_{-1}|}\sum_{v\in Z_{-1}}\mathcal{L}_v$
- $E_A = \frac{1}{|A_{+1}|}\sum_{v\in A_{+1}}\mathcal{L}_v - \frac{1}{|A_{-1}|}\sum_{v\in A_{-1}}\mathcal{L}_v = \frac{1}{2}(14.02 + 0.89) - \frac{1}{2}(82.02 + 2.33) = -34.72$
- Mit $E_A = -34.72$ lässt sich sagen, dass das System bei hoher Stufe von Faktor A durchschnittlich einen um -34.72 kleineren Performancewert hat.
- $E_B = \frac{1}{|B_{+1}|}\sum_{v\in B_{+1}}\mathcal{L}_v - \frac{1}{|B_{-1}|}\sum_{v\in B_{-1}}\mathcal{L}_v = \frac{1}{2}(2.33 + 0.89) - \frac{1}{2}(82.02 + 14.02) = \frac{1}{2}(3.22) - \frac{1}{2}(96.04) = -46.41$

Versuch	Faktor A	Faktor B	Ergebnis $\mathcal{L}$	Menge A	Menge B
1	−1	−1	82.02	A_{-1}	B_{-1}
2	+1	−1	14.02	A_{+1}	B_{-1}
3	−1	+1	2.33	A_{-1}	B_{+1}
4	+1	+1	0.89	A_{+1}	B_{+1}

2^k-Versuchsplan: Wechselwirkungs-Effekt

Wechselwirkungs-Effekt E_{ZX} zwischen zwei Parametern Z und X

- zeigt die durchschnittliche Änderung eines Performance-Werts $\mathcal{L}$
- Vorgehensweise: Versuche unterteilen in Menge Z_{X+1}, bei denen Z und X beide auf der jeweils tieferen oder höheren Stufe sind; und in Menge Z_{X-1}. Dann ist $E_{ZX} = \frac{1}{|ZX_{+1}|}\sum_{v\in ZX_{+1}}\mathcal{L}_v - \frac{1}{|ZX_{-1}|}\sum_{v\in ZX_{-1}}\mathcal{L}_v$
- $E_{AB} = \frac{1}{2}(82.02 + 0.89) - \frac{1}{2}(14.02 + 2.33) = 35.78$. Setzt man also genau eins der beiden Faktoren auf die hohe Stufe, ist $\mathcal{L}$ durchschnittlich um 35.78 höher.

Versuch	Faktor A	Faktor B	Wechsel A·B	Ergebnis $\mathcal{L}$
1	−1	−1	+1	82.02
2	+1	−1	−1	14.02
3	−1	+1	−1	2.33
4	+1	+1	+1	0.89

Bei mehreren Faktoren lässt sich eine mehrstufige Wechselwirkung analysieren. Beispiel: drei Faktoren A, B, C.

- Die Wechselwirkung zwischen A und B ist AB, zwischen B und C ist es BC zwischen A und C ist es AC.
- Wie gesehen wird auch hier die Multiplikationsmethode angewandt, also BC ist B·C.
- Ebenso lässt sich ein für alle drei die Wechselwirkung betrachten mittels ABC.

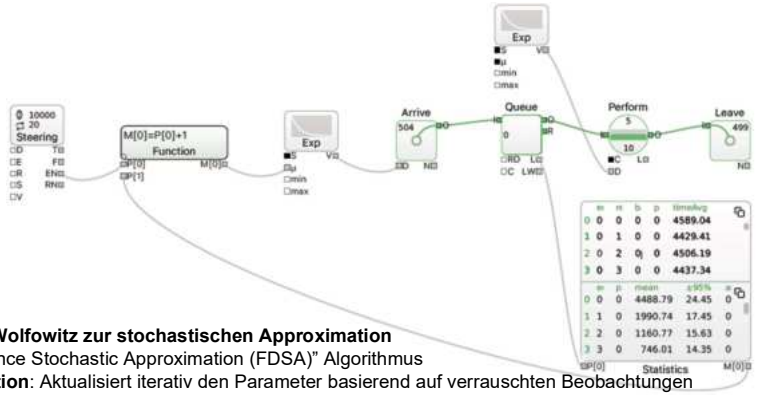
Einführung in die Simulationsoptimierung

- Simulationsoptimierung bedeutet, die **besten Eingabeparameter** für ein durch Simulation modelliertes System zu finden, wobei die **Zielfunktion** anhand der **Simulationsergebnisse** bewertet wird.
- **Herausforderungen**:
  - **Rauschen**: Simulation unterliegt stochastischen Effekten und kann verrauscht sein → erschwert Bewertung Zielfun.
  - **Rechenaufwand**: lange rechnende Simulationen erfordern effiziente Optimierungsalgorithmen.
- **Gängige Methoden**:
  - **Gitter-Suche**: **evaluiert** eine **diskrete Menge** von Werten.
  - **Stochastische Approximation**: Methoden wie der Kiefer-Wolfowitz-Algorithmus zum **Umgang mit Rauschen**.
  - **Metaheuristiken**: Techniken wie **genetische Algorithmen** und **Simulated Annealing**.

Gitter-Suche: Beispiel

- Eine Menge möglicher Parameterwerte wird verwendet, um das System zu bewerten
- Gegeben ist ein M/M/10-Warteschlangensystem mit einer mittleren Servicezeit $1/\mu = 100$.
- Ziel mittlere Zwischenankunftszeit $1/\lambda$ bestimmen, sodass Kosten minimieren. $\hat{L}$ durchschn. Warteschlangenlänge.
- Kosten für gegebenen Parameter $1/\lambda$ und durch Simulation berechnete $\hat{L}$ werden durch $\mathcal{L}(1/\lambda) = \hat{L} + 1/\lambda$ angegeben.

Umsetzung Quisim:



Algorithmus von Kiefer und Wolfowitz zur stochastischen Approximation

- auch genannt "Finite Difference Stochastic Approximation (FDSA)" Algorithmus
- **Stochastische Approximation**: Aktualisiert iterativ den Parameter basierend auf verrauschten Beobachtungen
- **Gradientenschätzung** unter Verwendung von finiten Differenzen
- **Abnehmende Schrittweite** für effiziente Konvergenz
- Hauptmerkmale:
  - Geeignet für komplexe oder verrauschte Zielfunktionen
  - Entwickelt für simulationsbasierte Optimierung
  - Einfach zu implementieren und auf verschiedene Probleme anwendbar
- Konvergenz:
  - Optimale Konvergenzrate ist $1/\sqrt[3]{i}$ für Iteration $i \to \infty$
  - CRN kann auf $1/\sqrt{i}$ für $i \to \infty$ verbessert werden

Parameter:

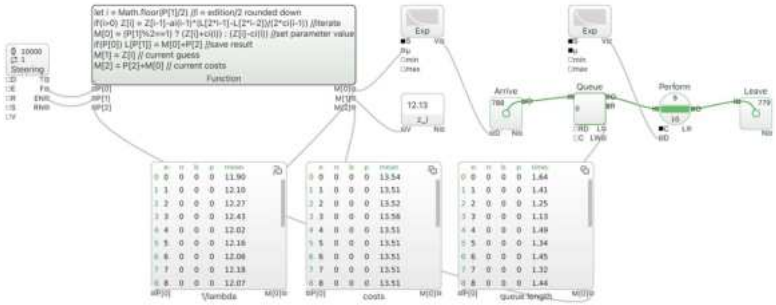
- Anpassungsparameter a, c individuell für Problemstellung
- Schrittweite a_i: abnehmende Folge, z.B. $a_i = a/i$
- Störungsgrösse c_i: abnehmende Folge, z.B. $c_i = c/\sqrt[3]{i}$ oder $c_i = c/\sqrt[4]{i}$
- Bedingungen: $a_i \to 0, \sum_{i=0}^{\infty} a_i = \infty, c_i \to 0, \sum_{i=0}^{\infty} \frac{a_i^2}{c_i^2} < \infty$

Algorithmus:

1. Initialisiere mit Anfangswert z_1. Setze $i \leftarrow 1$
2. Simuliere um $\mathcal{L}(z_i + c_i)$ und $\mathcal{L}(z_i - c_i)$ zu bewerten
3. Gradientenschätzung mit $\hat{g}_i = \frac{\mathcal{L}(z_i+c_i) - \mathcal{L}(z_i-c_i)}{2c_i}$
4. Aktualisiere Parameter auf $z_{i+1} = z_i - a_i \hat{g}_i$
5. Iteriere $i \mapsto i+1$ und wiederhole ab Schritt 2 bis zur Konvergenz.

Kiefer-Wolfowitz für das M/M/10-Beispiel

- Bevor wir den Algorithmus implementieren, führen wir eine Gitter-Suche durch, um guten Anfangswert zu finden
- Wir verwenden hier nicht Mittelwert der Mittelwerte, daher bewerten wir nur eine Replikation.
- Wir zeigen die Kostenfunktion in einer zusätzlichen Ausgabe an.
- Ein guter Anfangswert für den Parameter $1/\lambda$ scheint 12 zu sein.



Aufgaben aus Praktika

Ereignisliste

Zwischenankunftszeiten:	7, 2, 2, 2, 5, 2
Bediendauern:	4, 3, 3, 4, 2, 3

Betrachten M/M/1-Modell. Aufgabe ist, nun Mechanismus der Abarbeitung der Simulation in Tabellenform gemäss Vorlesungsunterlagen zu beschreiben. Zu Beginn ist dabei kein Kunde im System. Nutzen Sie dabei die folgend gegebenen Zufallszahlenströme. Legen Sie fest und dokumentieren Sie bitte, wie Sie mit gleichzeitigen Ereignissen umgehen.
Gleichzeitigen Ereignissen → Ende vor Ankunft

Event	Aktuelle Simulationszeit	Nächste Ankunft	Nächstes Bedienungsende	Zustand Warteschlangenlänge	Zustand Kasse A: aktiv, F: frei
Start	0	7	-	0	F
Ankunft	7	9	11	0	A
Ankunft	9	11	11	1	A
Ende	11	11	14	0	A
Ankunft	11	13	14	1	A
Ankunft	13	18	14	2	A
Ende	14	18	17	1	A
Ende	17	18	21	0	A
Ankunft	18	20	21	1	A
Ankunft	20	-	21	2	A
Ende	21	-	23	1	A
Ende	23	-	26	0	A
Ende	26	-	-	0	F

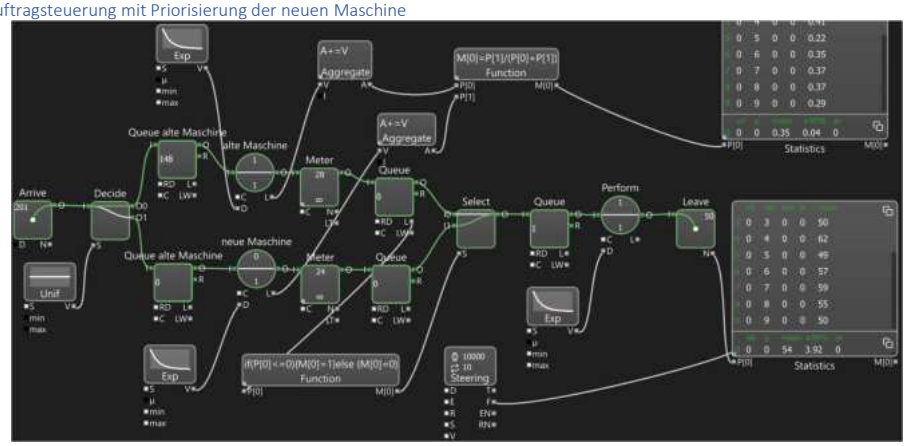
Zeitpunkt t	Länge der Warteschlange	Dauer Intervall	Fläche
0	0	9	0
9	1	2	2
11	0	0	0
11	1	2	2
13	2	1	2
14	1	3	3
17	0	1	0
18	1	2	2
20	2	1	2
21	1	2	2
23	0	3	0
26			15

Beschreiben daraus abgeleitet in Tabellenform die Zeitpunkte, bei denen sich die Länge der Warteschlange ändert sowie die zugehörige Länge der Warteschlange.

$$N_q = \frac{15}{26} = 0.577$$

Dynamische Auftragsteuerung mit Priorisierung der neuen Maschine

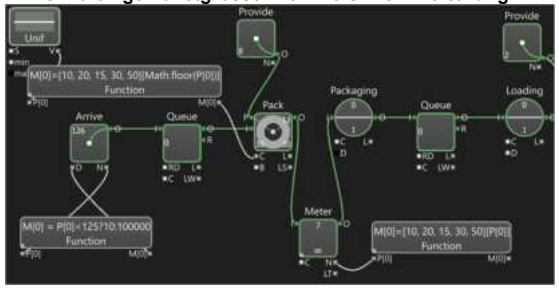
Mit dem Select Block werden wenn immer möglich zuerst die Teile von der neuen Maschine zur Weiterverwendung genommen. Sollte die neue Maschine gerade kein fertiges Teil haben, nimmt der Block eines der alten Maschine, sodass im nachgelagertem System weitergearbeitet werden kann und kein «Starving» passiert.

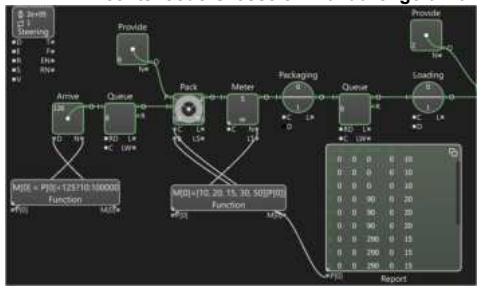


AB7 – Unterschiedliche Paketgrösse

Verwenden Modell logistics5.qui und ändern es, sodass es unterschiedliche Bestellmengen je Artikel abbilden kann. Die Bestellmengen für die fünf Artikel sind 10, 20, 15, 30, 50.

Links: Zufällige Paketgrösse mit Hilfe Uniformverteilung **Rechts: Jede Grösse einmal durchgeführt**

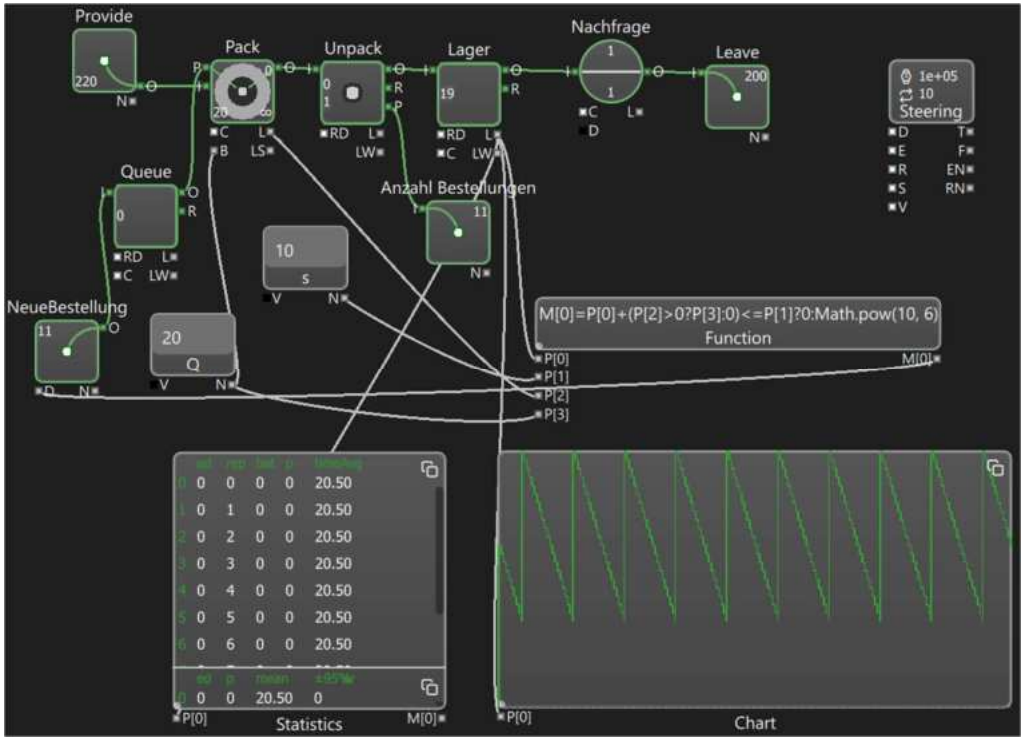




AB8 – Lagerhaltungsmodell

Haben ein Produkt in einem Lager. Für dieses Produkt wird eine (s, Q)Lagerhaltungspolitik wie folgt definiert:

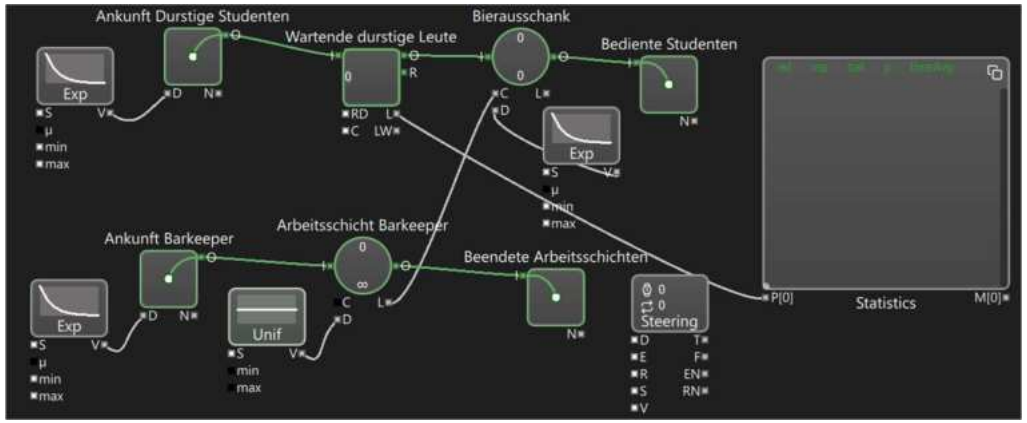
- Der verfügbare Bestand eines Produkts ist physischer Bestand im Lager plus stock-intransit (Bestand in Transit). Sobald der verfügbare Bestand kleiner oder gleich s ist, wird eine Nachbestellung von Q Produkten ausgelöst.
- Übung 1. Erstellen Sie ein Quisim-Modell, das die (s, Q)-Lagerhaltungspolitik für einen einzelnen Artikel implementiert. Die Nachfrage sei deterministisch mit 1 Artikel pro 500 simulierte Zeiteinheiten. Darüber sei s = 10 und Q = 20. Nehmen Sie an, dass die Lieferzeit null ist.



AB9.1 – Barkeeper Frackwoche

Ausschank Frackwoche. Hier soll in einem Schichtmodell gearbeitet werden. Erstellen für Bedienung Durstigen ein M/M/s-Modell. Annahme unbeschränkt viele Zapfhähne und Barkeeper parallel ausschenken. Verwenden folgenden Annahmen:

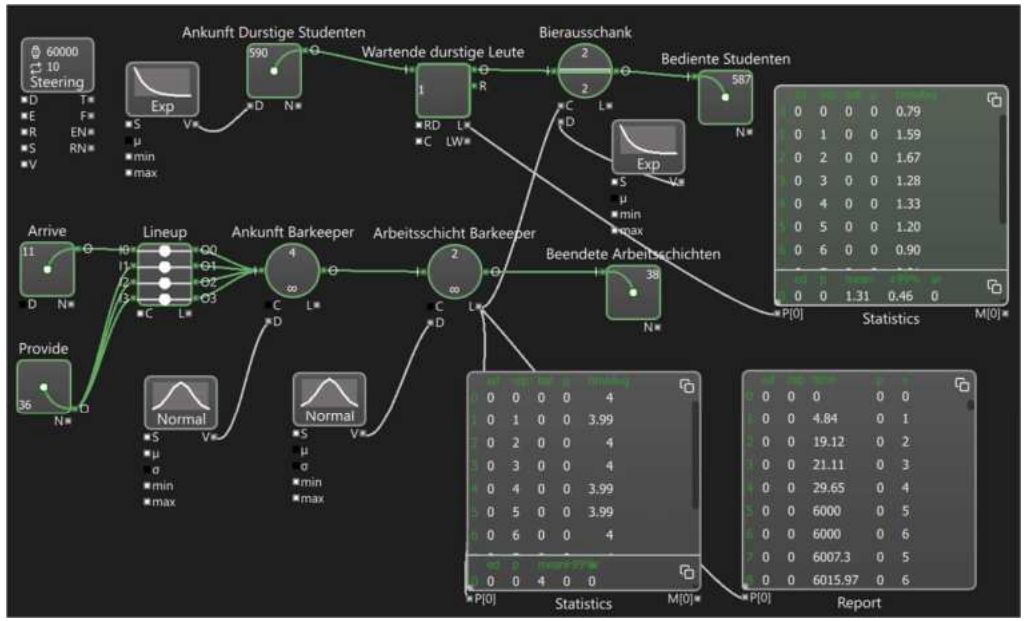
- Durstigen kommen exponentiell verteilten Zwischen-Ankunftszeiten Durchschnitt 100 Zeiteinheiten an.
- Bedienzeiten exponentiell verteilt mit Durchschnitt von 300 Zeiteinheiten.
- Anzahl Barkeeper kann sich im Laufe der Zeit ändern. Barkeeper arbeiten zufällig: Sie kommen unabhängig voneinander an die Bar und verlassen sie unabhängig voneinander wieder.
- Arbeitszeitspannen Barkeeper sind uniform verteilt zwischen 1000 und 5000 Zeiteinheiten
- Die Zwischen-Ankunftszeiten der Barkeeper sind exponentiell verteilt mit einem Durchschnitt von a.
- Wählen Sie a so aus, dass die mittlere Warteschlangenlänge fast nie länger als 10 ist.



AB9.2 – Barkeeper Frackwoche

In einem zweiten Szenario sollen die Schichtzeiten der Barkeeper geordneter ablaufen. Arbeitszeiten sind normalverteilt, d.h. eine Stunde mit einer Varianz von 10 Minuten (1000 Einheiten). Ankunftszeiten Barkeeper sollen auch normalverteilt sein, d.h. die Schicht-Startzeiten, mit einer Varianz von 10 Minuten (1000 Einheiten). Wie viele Barkeeper benötigen Sie pro Schicht, damit die mittlere Warteschlangenlänge fast nie mehr als 10 ist?

➔ Der Schichtbeginn ist deterministisch, jedoch kommen die einzelnen Barkeeper für diese Schichte zufällig (Normalverteilung) an.



Aufgaben aus Probe-SEP

Aufg. 1

a) Geben Sie drei Vorteile einer DES im Vergleich zur analytischen Theorie, die in FAP behandelt wurde.

- **Komplexität**: DES kann komplexe Systeme modellieren, bei denen analytische Lösung nicht möglich oder sehr aufwändig ist.
- **Flexibilität**: DES erlaubt die Berücksichtigung von realistischen, zeitabhängigen oder nicht-stationären Bedingungen.
- **Detailgenauigkeit**: DES kann Prozesse mit individuellen Entitäten und detaillierten Interaktionen simulieren, was bei der analytischen Theorie oft abstrahiert wird.
- **Dynamik und Zeitabhängigkeit**: DES kann Zeitverläufe und dynamische Effekte abbilden, die sich mit FAP oft nicht darstellen lassen, wie z. B. Stosszeiten oder saisonale Schwankungen.
- **Szenarienanalyse**: Es lassen sich leicht verschiedene „Was-wäre-wenn"-Szenarien simulieren, um die Auswirkungen von Änderungen zu bewerten.
- **Unabhängigkeit von Verteilungen**: DES kann mit beliebigen Verteilungen arbeiten, während FAP oft Annahmen über Exponentialverteilungen oder andere spezifische Verteilungen macht.
- **Visualisierung**: DES-Modelle erlauben eine visuelle Darstellung der Abläufe, was das Verständnis erleichtert.

b) Geben Sie zwei Beispiele, die nicht mit FAP aber mit einem DES-Modell gelöst werden können.

- Ein System mit zeitvariablen Ankunftsraten oder Bearbeitungszeiten.
- Ein Produktionsprozess mit komplexen Regeln (z. B. Priorisierungen oder Parallelprozesse).
- Ein Produktionsprozess mit mehreren Abteilungen, bei dem Rückstände oder Verzögerungen in einer Abteilung andere Abteilungen beeinflussen.
- Ein komplexer Transport- oder Logistikprozess mit mehreren Knotenpunkten und Lagerstandorten.
- Ein Callcenter mit variabler Verfügbarkeit von Mitarbeitern (z. B. Pausen, Schichten).
- Ein Warteschlangensystem mit Prioritätsregeln, bei dem einige Kunden schneller bedient werden als andere.

Aufg. 2

a) Erklären Sie den Unterschied zwischen einem stationären und einem transienten Zustand.

- **Stationärer Zustand**: Das System hat ein Gleichgewicht erreicht, in dem statistische Eigenschaften (z. B. durchschnittliche Wartezeit) konstant bleiben.
- **Transienter Zustand**: Das System befindet sich in der Anfangsphase oder einer Übergangsphase, in der statistische Eigenschaften noch variieren.

b) Erläutern Sie den Begriff Warmup-Periode/Einschwingzeit.

- Die Warmup-Periode bezeichnet die Anfangszeit einer Simulation, in der sich das System noch nicht im stationären Zustand befindet. Daten aus dieser Phase werden häufig ausgeschlossen, um Verzerrungen zu vermeiden.

Aufg. 3

a) Erläutern Sie den Unterschied zwischen zeitgewichtetem Mittelwert und arithmetischem Mittelwert.

- **Zeitgewichteter Mittelwert**: Berücksichtigt die Dauer, über die ein bestimmter Wert konstant bleibt.
- **Arithmetischer Mittelwert**: Einfache Durchschnittsberechnung ohne Berücksichtigung der Dauer der Werte.

b) Geben Sie jeweils ein Beispiel, wo ein zeitgewichteter Mittelwert und ein arithmetischer Mittelwert verwendet wird.

- **Zeitgewichteter Mittelwert**: Durchschnittliche Anzahl Kunden in Warteschlange, gewichtet nach der Verweildauer.
- **Arithmetischer Mittelwert:** Durchschnittliche Bearbeitungszeit einer Kundenanfrage.

Aufg. 4

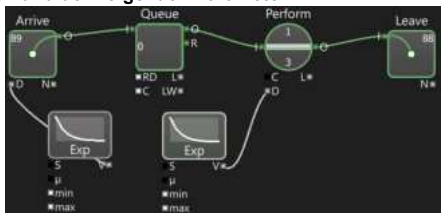
Erklären Sie den Unterschied zwischen einem Arrive- und einem Provide-Block in Quisim.

- **Arrive-Block**: Erzeugt Entitäten (z. B. Kunden) in Simulation. Die Ankunftsrate wird durch Verteilungsparameter definiert. Diese können einer statistischen Verteilung folgen und damit zufällig gezogen/generiert werden. Gibt auch fixe deterministische Zwischenankunftszeiten.
- **Provide-Block:** Stellt jederzeit eine neue Entität bereit so bald eine benötigt wird. Gibt kein Starving von Seite des Provide-Blocks her (bei anderen Systemteilen trotzdem möglich).

Aufg. 5

a) Modellieren Sie eine M/M/3-Queue auf Papier mit Quisim-Blöcken und den folgenden Parametern:

- Durchschnittliche Zwischenankunftszeiten = 100 Zeiteinheiten
- Durchschnittliche Bearbeitungszeiten = 90 Zeiteinheiten



b) Erreicht diese M/M/3-Queue mit den oben genannten Parametern einen steady-state-Zustand?

- Ja, da gemäss FAP Theorie gilt: Systemauslastung $u = \frac{\lambda}{c \cdot \mu} = \frac{1/100}{3 \cdot 1/90} = 0.3$ Erreichen steady-state, da u kleiner als 1 ist.

c) Wie würden Sie das Modell validieren?

- Vergleich der Simulationsergebnisse mit analytischen Ergebnissen (z. B. Wartezeiten aus der M/M/c-Theorie).
- Überprüfung der Implementierung durch Tests mit bekannten Szenarien.

Aufg. 6

Erstellen Sie eine Event-Liste für eine M/M/1-Queue mit den folgenden Zwischenankunfts- und Bearbeitungs-Zeiten: → siehe erste Aufgabe in dieser Zusammenfassung.

Aufg. 7

Gegeben ist folgendes Quisim-Modell einer M/M/s-Queue mit einer dynamischen Anzahl von Workstations.

Erklären Sie, wieso dieses Modell fehlerhaft ist. (kein Bild enthalten…)

Aufg. 8

Modellieren Sie folgenden Bezahl-Prozess in Quisim: Nach Eingang einer Bestellung spaltet sich der Bezahl-Prozess in zwei parallele Subprozesse auf:
1) Im einen Subprozess wird geprüft, ob die Ware an Lager ist. Dieses Prozessschritt dauert konstant 100 Zeiteinheiten.
2) Im anderen Subprozess wird die Zahlung verarbeitet. Dieser Prozessschritt ist exponential-verteilt mit Durchschnitt 1000 Zeiteinheiten.
Sind beide Subprozesse abgeschlossen, so wird die Bestellung ausgelöst und das Paket versendet. Das Versenden des Pakets ist exponential-verteilt mit durchschnittlich 1500 Zeiteinheiten.

Weitere Fragen

Erklären Sie den Begriff „stochastisches Modell". Wie unterscheidet es sich von einem deterministischen Modell?

- **Stochastisches Modell**: Nutzt Zufallsvariablen zur Modellierung von Unsicherheiten (z.B. Zufallszeiten, Ankunftsraten).
- **Deterministisches Modell**: Hat festgelegte Werte ohne Variabilität (z. B. konstante Zeiten).

Wie verändert sich die Leistung eines M/M/1-Systems, wenn die Ankunftsrate λ die Bedienrate μ übersteigt?

- System wird instabil und die Warteschlange wächst unendlich an, da der Server nicht alle Ankünfte bedienen kann.

Was ist der Unterschied zwischen einer Gitter-Suche und einem stochastischen Optimierungsalgorithmus wie dem Kiefer-Wolfowitz-Algorithmus?

- **Gitter-Suche**: Testet alle Kombinationen von Parametern in einem definierten Bereich.
- **Kiefer-Wolfowitz**: Schätzt Gradienten stochastisch und optimiert effizienter bei verrauschten Zielfunktionen.

Welche Bedeutung haben Konfidenzintervalle in der Simulation? Wie können sie in Quisim durch die Batch-Mittelwert-Methode bestimmt werden?

- Konfidenzintervalle zeigen die statistische Unsicherheit. Die Batch-Mittelwert-Methode teilt die Simulation in Batches, berechnet Mittelwerte und verwendet diese zur Bestimmung der Intervalle.

In einem Job-Shop-Modell sollen die Aufträge unterschiedliche Reihenfolgen für die Maschinen haben. Wie kann dies in Quisim modelliert werden?

- Verwendung von Attributen wie „Reihenfolge" in Kombination mit Get- und Set-Blöcken.

Wie können Sie Modell validieren, wenn keine analytischen Ergebnisse zum Vergleich verfügbar sind.

- Vergleich mit realen Daten oder Experteneinschätzungen.
- Sensitivitätsanalyse durchführen.

Wie gehen Sie mit einer negativen Autokorrelation in den Ergebnissen einer Simulation um?

- Simulation in Batches aufteilen und Mittelwerte verwenden, um unabhängige Daten zu erzeugen.

Stand der Arbeit

SW	Vorlesung	AB/Aufgabe	Skript	ZF
1	☒	☒	☐	☒
2	☒	☒	☐	☒
3	☒	☒	☐	☒
4	☒	☒	☐	☒
5	☒	☒	☐	☒
6	☒	☒	☐	☒
7	☒ 2. Seite 83	☒	☐	☒
8	☒ 3. Seite 13/19	☒	☐	☒
9	☒ 4. Seite 34	☒	☐	☒
10	☒	☒	☐	☒
11	☒ 4. Seite 60	☒	☐	☒
12	☒ 5. Seite 22	☒	☐	☒
13	☒ 5. Seite 47	☒	☐	☒
14	☒ 5. Seite 70	☒	☐	☒

Modulauflagen

- Abgabe von Übungsaufgaben während der Lektion.
- Der Rundungszusatz zur SEP-Note deckt eine Spanne von 0.6 Notenpunkten ab:
  - –0.2 falls <70% der Übungspunkte erreicht werden
  - +0.4 falls ≥80% der Übungspunkte erreicht werden und ansonsten 0 Notenpunkte.
- Abschlussprüfung am Ende des Semesters
- Schriftliche Prüfung (90 Minuten)
- Testet theoretische Kompetenz und Modellierungswissen
- Prüfung ohne System/Python, alle Unterlagen auf Papier
- Max. 10 Seiten handschriftlicher Notizen erlaubt