Computergestützte Modellierung

Ieyasu Sugimoto

Institut für Statistik und Operations Research
Universität Graz

Lehrveranstaltungsablauf

- 6 Blöcke a 3,5 Std
- 3 Blöcke Stoffpräsentation
  - Einheiten 1, 2 und 3
- 3 Blöcke Präsentationen Programmieraufgaben
  - Einheit 4: Bsp. 1 – 15
  - Einheit 5: Bsp. 16 – 23
  - Einheit 6: Bsp. 24 – 30

Lehrveranstaltungsablauf

Programmieraufgaben

- Präsentation der Aufgabenstellung
- Vorstellung des Lösungsansatzes
- Lösung von Programmieraufgaben mittels Computerprogramm
- in 2-er Gruppen
- Programmiersprache kann selbst gewählt werden
- Programmierbeispiele sind je nach Schwierigkeitsgrad unterschiedlich bewertet.
- Unterstützend wird Programmiertutorium angeboten (Matlab)
- Einreichen bis Freitag vor jeweiliger Einheit (also eine Woche vorher) über `moodle.uni-graz.at`

Lehrveranstaltungsablauf

Programmieraufgaben

- Anwesenheit beider Gruppenmitglieder verpflichtend, falls man Beispiel eingereicht hat.
- Falls mehrere Gruppen das selbe Beispiel eingereicht haben, wird per Zufall entschieden, wer präsentieren darf.
- Für das Präsentieren gibt es einen Zusatzpunkt.
- Falls unterschiedliche Lösungswege abgegeben wurden, sollen diese auch präsentiert werden.
- Es können auch Teilpunkte eines Beispiels abgegeben bzw. präsentiert werden.

Lehrveranstaltungsablauf

Programmieraufgaben

- Falls ein Beispiel von keiner Person gelöst wurde, wird ein Teilnehmer der Lehrveranstaltung ausgewählt, um Flussdiagramm mit Hilfe der Kollegen an der Tafel zu erstellen - dafür werden auch Zusatzpunkte je nach ursprünglicher Punkteanzahl vergeben.
- Der Notenschlüssel wird am Angabenblatt mit den Programmierbeispielen angegeben.
- Es werden auch Punkte für Mitarbeit vergeben.

Outline

Simulation und Modellierung - Einführung

Was versteht man unter “Simulation und Modellierung”?

- Vereinfachung realer Systeme zu einem quantitativen Modell
- Umsetzung in Computerprogramm
- Arbeiten mit diesem Modell am Computer zur Bestimmung von Kennzahlen, Optimierung, zeitlichen Entwicklung . . .
- Gegensatz zu
  - analytischem Ansatz, wo Lösung rein theoretischer Natur
  - echtem Experiment, wo Vorgang in Realität beobachtet wird

Simulation und Modellierung - Einführung

Welche Vorteile gewinnt man durch Simulation und Modellierung?

- Kosten- und Zeitersparnis
- Analytische Lösung oft nicht möglich, da grundlegendes Modell zu komplex
- Flexibilität (Betrachtung der Modelle in Abhängigkeit von Parametern)
- Wiederholbarkeit mit unterschiedlichen Szenarien
- Bestehende Simulationssoftware kann benutzt werden

Simulation und Modellierung - Einführung

2 unterschiedliche Problemsituationen:

- Simuliert man das richtige Modell?
    - Wird die Realität ordnungsgemäß abgebildet?
    - Werden alle wichtigen Einflussfaktoren berücksichtigt?
    - Stimmt die verwendete Zufallsverteilung für die Wahrscheinlichkeitsberechnung?
- Simuliert man das Modell richtig?
    - Stimmt die technische Umsetzung mit dem Modell überein?

Anwendungsgebiete

Wo wird Simulation eingesetzt?

- Betriebswirtschaft
  - Warteschlangen
  - Lagerhaltung
- Technik
  - Karrosserieentwicklung/Crashtests
- Naturwissenschaften
  - Epidemie

→ oftmals verwendet um zeitliche Entwicklung darzustellen

Simulation und Modellierung - Einführung

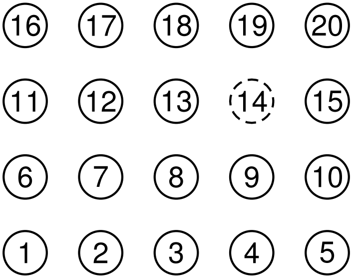
Beispiel: Epidemie (kommt aus der Naturwissenschaft, lässt sich aber auf wirtschaftliche Probleme anwenden), z. B.:

- Werbeaktion in Wohngebiet mit zufälligem Startpunkt
- Nachbarn erzählen sich im Zeitablauf mit gewisser Wahrscheinlichkeit von Werbeaktion
- Wahrscheinlichkeit kann im Zeitablauf abnehmen
- Durch Investitionen kann Wahrscheinlichkeit (Ansteckungsrate) beeinflußt werden
- Ziel: Ausbreitung abschätzen, um Investitionen gering zu halten
- Unterschiedliche Szenarien:
  - Ansteckungsrate verändert sich im Zeitablauf
  - Nachbarn können öfters informiert werden
  - Zeitdauer: Ende nach vorgegebener Zeit oder bis Information verebbt

Simulation und Modellierung - Einführung

Epidemie-Beispiel: Umsetzung

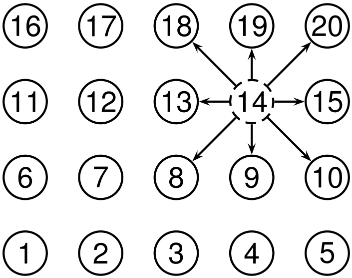
- Jeder Haushalt ein Punkt in einem Gittermodell
- Startpunkt bei Knoten 14



Simulation und Modellierung - Einführung

Epidemie-Beispiel: Umsetzung

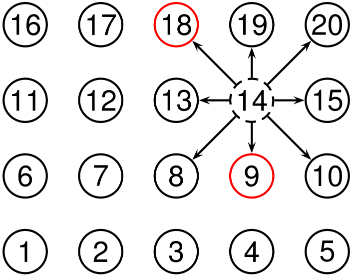
- Ausgehend vom Startpunkt können je Zeiteinheit nur die direkten Nachbarn informiert (angesteckt) werden



Simulation und Modellierung - Einführung

Epidemie-Beispiel: Umsetzung

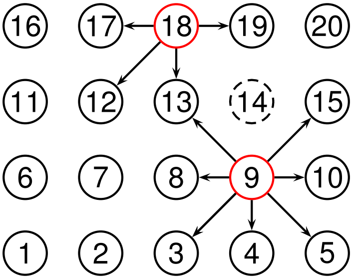
- Ausgehend vom Startpunkt werden je Zeiteinheit nur die direkten Nachbarn mit einer gewissen Wahrscheinlichkeit informiert werden
- Je höher die Wahrscheinlichkeit, desto eher werden Nachbarn informiert $\rightarrow$ diese kann von außen beeinflußt werden



Simulation und Modellierung - Einführung

Epidemie-Beispiel: Umsetzung

- Im nächsten Schritt wird von den neu hinzugefügten Knoten ausgegangen



Epidemie-Simulation

Umsetzung des Beispiels der Epidemie:

- Ausdehnung der Epidemie in einem gewissen Gebiet
- Darstellung des Gebiets in einer Matrix (oder Liste)
- Jedes Element der Matrix (oder Liste) entspricht einem Punkt innerhalb des untersuchten Gebiets
- Der Wert des Elements entspricht dem Zustand des Punktes $\rightarrow$ geeignete Codierung notwendig
  - Bsp.: mögliche Zustände eines Punktes: lebendig (0), infiziert (1) oder tot (2)
  - Der Wert jedes Elementes muss doppelt vorkommen, da Wert unterschiedlich für t oder $t + 1$
  - Über Codierung kann man einzelne Elemente aus der Simulation auch ausnehmen

Epidemie-Simulation (Fortsetzung)

Generelle Vorgehensweise (hier: Element ist nur eine Zeiteinheit infiziert und stirbt dann):

- Ist Punkt lebendig zur Zeit $t \rightarrow$ keine Auswirkung auf $t + 1$
- Ist Punkt infiziert zu Zeit $t \rightarrow$
  - er kann Nachbarn anstecken, dann sind diese in $t + 1$ infiziert $\rightarrow$ Zufallsgröße
  - er selbst ist in $t + 1$ tot.
- Ist Punkt tot zur Zeit $t \rightarrow$ keine Auswirkung auf $t + 1$
- Erhöhung des Zeitzählers $\rightarrow$ Werte von $t + 1$ werden zu Werte von t
- Berücksichtigen von Abbruchkriterium
- Berücksichtigen von Gebietsgrenzen (in welcher Richtung gibt es Nachbarn?)

Deterministisch vs. Stochastisch

- Deterministische Simulation
  - Zur Lösung von deterministischen Problem
  - Inputparameter stehen schon vor der Simulation fest
  - neuerlicher Durchlauf liefert exakt das selbe Ergebnis
- Stochastische Simulation
  - Zur Lösung stochastischer Problem, d.h. der Ablauf von Ereignissen ist von zufälligen Einflüssen abhängig.
  - Trotz konstanter Inputfaktoren kommt neuerlicher Durchlauf nicht notwendigerweise zu dem selben Ergebnis
  - Erst bei mehreren Durchläufen aussagekräftiges Ergebnis
  - Stabilisierungsphase am Anfang beachten

Diskret vs. Kontinuierlich

- Diskrete Systeme (Discrete Event Dynamic Systems, DEDS models)
    - Zu gewissen, vorher nicht bekannten Zeitpunkten treten Zustandsänderungen ein
    - Simulationszeit verändert sich in diskreten Zeitabständen
- Kontinuierliche Systeme (Continuous Time Dynamic Systems, CTDS models)
    - Im Idealfall treten Zustandsänderungen kontinuierlich ein
    - In Realität werden auch hier diskrete Zeitsprünge verwendet, da man die Zustände nicht zu jedem Zeitpunkt in einem kontinuierlichen Modell beobachten kann.
- Mischung aus Diskreten und Kontinuierlichen Systemen
    - z. B. Fließband, wo Produkte erhitzt werden.

Deterministische Simulation

Beispiel deterministische Simulation

- Keine stochastischen Elemente
- Ablauf steht schon vorher fest

- Bsp.: Produktionsmodell
  - Betrieb produziert ein Produkt (1 EH Rohstoff $\rightarrow$ 1 EH des Produkts)
  - Lagerbestand abhängig von Nachfrage
  - Bestellmenge entspricht Abgang aus Rohstofflager
  - Wochenweise Simulation

Deterministische Simulation

Produktionsmodell (Fortsetzung): Variablen

- Variablen zu Beginn d. Woche t:
  - $R(t)$: Bestand im Rohstofflager
  - $F(t)$: Bestand im Auslieferungslager
  - $B(t)$: Bestellungen
  - $T(t)$: gewünschter Lagerbestand (target)
- Veränderungen innerhalb der Woche:
  - $P(t, t+1)$: Produktion innerhalb der Woche t
  - $D(t, t+1)$: Verkaufte Stückzahl (delivery)
  - $M(t, t+1)$: Gelieferte Rohstoffmenge
  - $X(t, t+1)$: eingegangene Bestellungen

Deterministische Simulation

Produktionsmodell (Fortsetzung): Zusammenhänge

1. offene Bestellungen: $B(t+1) = B(t) + X(t, t+1) - D(t, t+1)$
2. Rohstoffbestand: $R(t+1) = R(t) + M(t, t+1) - P(t, t+1)$
3. Auslieferungslager: $F(t+1) = F(t) + P(t, t+1) - D(t, t+1)$
4. gewünschter Lagerbestand: 5x durchschnittliche Nachfrage der letzten 4 Wochen (willkürlich)
   - $T(t+1) = \frac{5}{4} \cdot (X(t, t+1) + X(t-1, t) + X(t-2, t-1) + X(t-3, t-2))$
5. Verkauf: $D(t, t+1) = \min(F(t), B(t))$
6. Einkauf: $M(t, t+1) = P(t-1, t)$
7. Produktion: $P(t, t+1) = \min\{\max\{T(t) - F(t) + D(t, t+1), 0\}, R(t)\}$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Simulationsablauf

- 5 Wochen konstante Startphase (stabiles System):
  - Produktion, Verkauf, Rohstoffeingang, existierende Bestellungen, neue Bestellungen = 50
  - gewünschter Lagerbestand: 250 ($\frac{5}{4} \cdot 4 \cdot 50$)
  - Ausliegerungslager: 250
  - Rohstoffbestand: 150
- Simulationsstart: in Woche 2 Sonderangebot $\rightarrow$ in Woche 2 verdoppeln sich die Bestelleingänge und reduzieren sich zu 0 in Woche 3
- danach wieder stabiles System

Welche Auswirkungen hat bei den gegebenen Produktionsabläufen das Modell?

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Ausgangslage

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100							
3	0							
4	50							
5	50							
6	50							
7	50							
8	50							
9	50							
10	50							

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Verkauf

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100							50

$$D(t, t+1) = \min(F(t), B(t)) = \min(250, 50) = 50$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - offene Bestellungen

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t,t+1)$	B $(t,t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t,t+1)$	P $(t,t+1)$	D $(t,t+1)$
1	50	50	250	250	150	50	50	50
2	100	100						50

$$B(t,t+1) = B(t) + X(t,t+1) - D(t,t+1) =$$

$$= 50 + 100 - 50 = 100$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - gewünschter Lagerbestand

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5					50

$$T(t, t+1) = \frac{5}{4} \cdot (X(t, t+1) + X(t-1, t) + X(t-2, t-1) + X(t-3, t-2)) =$$

$$= \frac{100 + 50 + 50 + 50}{4} \cdot 5 = 312.5$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Produktion

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t{+}1)$	B $(t, t{+}1)$	$T(t{+}1)$	$F(t{+}1)$	$R(t{+}1)$	M $(t, t{+}1)$	P $(t, t{+}1)$	D $(t, t{+}1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5				50	50

$$P(t, t+1) = \min\{\max\{T(t) - F(t) + D(t, t+1), 0\}, R(t)\} =$$

$$= \min \begin{cases} \max \begin{cases} 250 - 250 + 50 \\ 0 \end{cases} \\ 150 \end{cases} = 50$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Auslieferungslager

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t,t{+}1)$	B $(t,t{+}1)$	$T(t{+}1)$	$F(t{+}1)$	$R(t{+}1)$	M $(t,t{+}1)$	P $(t,t{+}1)$	D $(t,t{+}1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250			50	50

$$F(t+1) = F(t) + P(t,t+1) - D(t,t+1) =$$

$$= 250 + 50 - 50 = 250$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Einkauf

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250		50	50	50

$$M(t+1) = P(t-1, t) = 50$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Rohstoffbestand

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50

$$R(t+1) = R(t) + M(t, t+1) - P(t, t+1) =$$

$$= 150 + 50 - 50 = 150$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Verkauf

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0							100

$$D(t, t+1) = \min(F(t), B(t)) = \min(250, 100) = 100$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - offene Bestellungen

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0						100

$$B(t, t+1) = B(t) + X(t, t+1) - D(t, t+1) =$$

$$= 100 + 0 - 100 = 0$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - gewünschter Lagerbestand

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t{+}1)$	B $(t, t{+}1)$	$T(t{+}1)$	$F(t{+}1)$	$R(t{+}1)$	M $(t, t{+}1)$	P $(t, t{+}1)$	D $(t, t{+}1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250					100

$$T(t, t{+}1) = \frac{5}{4} \cdot (X(t, t{+}1) + X(t{-}1, t) + X(t{-}2, t{-}1) + X(t{-}3, t{-}2)) =$$

$$= \frac{0 + 100 + 50 + 50}{4} \cdot 5 = 250$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Produktion

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250				150	100

$$P(t, t+1) = \min\{\max\{T(t) - F(t) + D(t, t+1), 0\}, R(t)\} =$$

$$= \min \begin{cases} \max \begin{cases} 312.5 - 250 + 100 \\ 0 \end{cases} \\ 150 \end{cases} = 150$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Auslieferungslager

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	300			150	100

$$F(t+1) = F(t) + P(t, t+1) - D(t, t+1) =$$

$$= 250 + 150 - 100 = 300$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Einkauf

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	300		50	150	100

$$M(t+1) = P(t-1, t) = 50$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Rohstoffbestand

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	300	50	50	150	100

$$R(t+1) = R(t) + M(t, t+1) - P(t, t+1) =$$

$$= 150 + 50 - 150 = 50$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Verkauf

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t,t{+}1)$	B $(t,t{+}1)$	$T(t{+}1)$	$F(t{+}1)$	$R(t{+}1)$	M $(t,t{+}1)$	P $(t,t{+}1)$	D $(t,t{+}1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	300	50	50	150	100
4	50							0

$$D(t,t+1) = \min(F(t), B(t)) = \min(300, 0) = 0$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - offene Bestellungen

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t,t+1)$	B $(t,t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t,t+1)$	P $(t,t+1)$	D $(t,t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	300	50	50	150	100
4	50	50						0

$$B(t,t+1) = B(t) + X(t,t+1) - D(t,t+1) =$$
$$= 0 + 50 - 0 = 50$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - gewünschter Lagerbestand

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	300	50	50	150	100
4	50	50	250					0

$$T(t, t+1) = \frac{5}{4} \cdot (X(t, t+1) + X(t-1, t) + X(t-2, t-1) + X(t-3, t-2)) =$$

$$= \frac{50 + 0 + 100 + 50}{4} \cdot 5 = 250$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Produktion

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t,t+1)$	B $(t,t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t,t+1)$	P $(t,t+1)$	D $(t,t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	300	50	50	150	100
4	50	50	250				0	0

$$P(t,t+1) = \min\{\max\{T(t) - F(t) + D(t,t+1), 0\}, R(t)\} =$$

$$= \min \begin{cases} \max \begin{cases} 250 - 300 + 0 \\ 0 \end{cases} \\ 50 \end{cases} = 0$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Auslieferungslager

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	300	50	50	150	100
4	50	50	250	300			0	0

$$F(t+1) = F(t) + P(t, t+1) - D(t, t+1) =$$

$$= 300 + 0 - 0 = 300$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Einkauf

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	300	50	50	150	100
4	50	50	250	300		150	0	0

$$M(t+1) = P(t-1, t) = 150$$

Deterministische Simulation

Produktionsmodell (Fortsetzung): Berechnungen - Rohstoffbestand

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t,t+1)$	B $(t,t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t,t+1)$	P $(t,t+1)$	D $(t,t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	300	50	50	150	100
4	50	50	250	300	200	150	0	0

$$R(t+1) = R(t) + M(t,t+1) - P(t,t+1) =$$
$$= 50 + 150 - 0 = 200$$

Deterministische Simulation

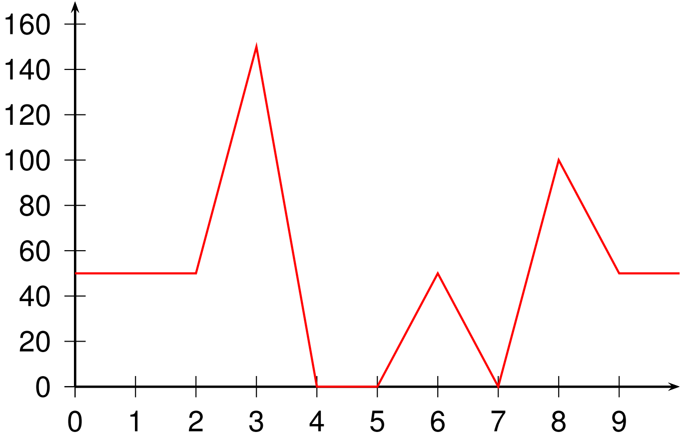
Produktionsmodell (Fortsetzung): Berechnungen

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	300	50	50	150	100
4	50	50	250	300	200	150	0	0
5	50	50	250	250	200	0	0	50
6	50	50	187.5	250	150	0	50	50
7	50	50	250	200	200	50	0	50
8	50	50	250	250	100	0	100	50
9	50	50	250	250	150	100	50	50
10	50	50	250	250	150	50	50	50

Deterministische Simulation

Produktionsmodell (Fortsetzung): Auswertung

- Problematik mit hoher Schwankungsbreite bei Produktion



Deterministische Simulation

Produktionsmodell II: neue Berechnung der Produktionsmenge

- Einkauf basierend auf Verhältnis von gewünschtem Lagerbestand und Bestand im Auslieferungslager: $M(t, t+1) = 50 \cdot (T(t)/F(t))$
- Produktionsmenge: ein Drittel der Rohstoffmenge wird umgewandelt $P(t, t+1) = R(t)/3$
- Rest bleibt gleich
    - 5 Wochen konstant (stabiles System)
    - Simulationsstart: in Woche 2 Sonderangebot $\rightarrow$ in Woche 2 verdoppeln sich die Bestelleingänge und reduzieren sich zu 0 in Woche 3
    - danach wieder stabiles System

Welche Auswirkungen hat bei den gegebenen Produktionsabläufen das Modell?

Deterministische Simulation

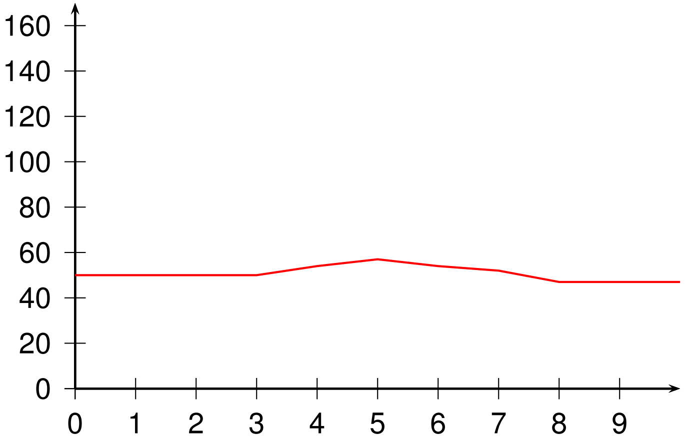
Produktionsmodell II (Fortsetzung): Berechnungen

Reihenfolge	ext. Input	2	3	5	7	6	4	1
Woche	X $(t, t+1)$	B $(t, t+1)$	$T(t+1)$	$F(t+1)$	$R(t+1)$	M $(t, t+1)$	P $(t, t+1)$	D $(t, t+1)$
1	50	50	250	250	150	50	50	50
2	100	100	312.5	250	150	50	50	50
3	0	0	250	200	163	63	50	100
4	50	50	250	254	171	63	54	0
5	50	50	250	261	163	49	57	50
6	50	50	187.5	265	157	48	54	50
7	50	50	250	268	140	35	52	50
8	50	50	250	264	140	47	47	50
9	50	50	250	261	141	47	47	50
10	50	50	250	258	142	48	47	50

Deterministische Simulation

Produktionsmodell II (Fortsetzung): Auswertung

- Problematik mit hoher Schwankungsbreite bei Produktion gelöst



Stochastische Simulation

- Bei deterministischem Modell waren Abläufe vorgegeben:
  - Änderungen bei Bestelleingängen bekannt
  - Materialversorgung gesichert, Produktion konstant, Nachfrage bekannt . . .
- In stochastischen Modellen können sich diese Parameter ändern
- $\rightarrow$ Zufallszahlen

Stochastische Simulation - Beispiel

Beispiel Instandhaltungsmodell

- Maschine mit 2 Motoren
- Ausfallswahrscheinlichkeiten (abhängig von Tagen sei der letzten Reparatur):

Tage seit letzter Reparatur	1	2	3	4	5	6	> 6
p_i	0.05	0.15	0.2	0.3	0.2	0.1	0

- Reparatur eines Motors kostet 50 GE
- Wartung eines Motors kostet 25 GE

Stochastische Simulation - Beispiel

Beispiel Instandhaltungsmodell Fortsetzung

- 2 Strategien
  - Strategie 1: Reparatur, wenn Motor ausfällt
  - Strategie 2: Bei Ausfall eines Motors wird dieser repariert und der andere Motor automatisch gewartet
  - Durch Simulation über 50 Tage entscheiden, welche Strategie besser ist.

Stochastische Simulation - Beispiel

Beispiel Instandhaltungsmodell Fortsetzung

1. Strategie

- Zufallszahlen für Motoren generieren und Lebensdauer bestimmen
- Anzahl der Motoren für zwei Einheiten unabhängig bestimmen, bis Summe der Lebensdauern bei beiden 50 erreicht
- Anzahl mit Reparaturkosten multiplizieren

Stochastische Simulation - Beispiel

Beispiel Instandhaltungsmodell Fortsetzung

2. Strategie

- Zufallszahlen für Motoren generieren und Lebensdauer bestimmen
- Anzahl der Motoren für zwei Einheiten abhängig bestimmen $\rightarrow$ es gibt nur eine (gemeinsame) Gesamtlebensdauer
- Falls Lebensdauer bei beiden Motoren unterschiedlich$\rightarrow$ Gesamtlebensdauer um kleinere Lebensdauer erhöhen, Gesamtkosten um Reparaturkosten (1x) und Wartungskosten (1x) erhöhen
- Falls Lebensdauer bei beiden Motoren gleich $\rightarrow$ Gesamtlebensdauer um Einzellebensdauer erhöhen, Gesamtkosten um Reparaturkosten (2x) erhöhen.

Stochastische Simulation - Beispiel

- Bestimmung der Lebensdauer der Motoren

Kumulierte Wahrscheinlichkeiten	1	2	3	4	5	6
h_i	0.05	0.2	0.4	0.7	0.9	1

Zufallszahl

14159265358979323846264338327950288419716939937510582
09749445923078164062862089986280348253421170679821480
86513282306647093844609550582231725359408128481117450
28410270193852110555964462294895493038196442881097566
59334461284756482337867831652712019091456485669234603

Strukturierte Programmierung

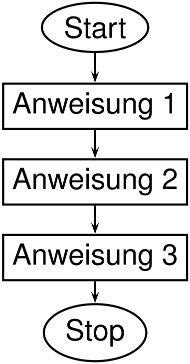
Programmablauf besteht aus folgenden Kontrollstrukturen:

- Sequenz: hintereinander ausgeführte Anweisungen
- Selektion: Anweisungen werden nur unter bestimmten Voraussetzungen ausgeführt
- Repetition: Anweisungen werden mehrfach hintereinander ausgeführt, bis ein bestimmter Zustand erreicht ist

Abbildung über Flussdiagramm

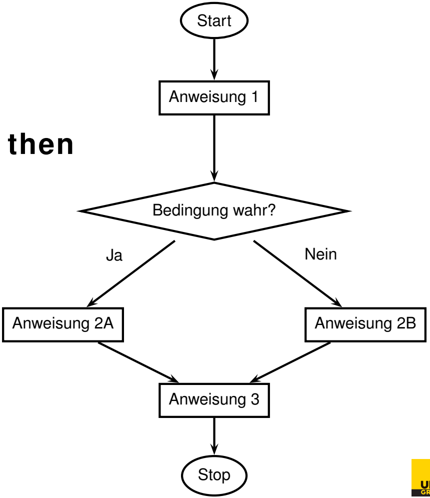
Strukturierte Programmierung: Sequenz

- Anweisungen: Ausführen von Operationen/Funktionen
- Eingabe/Ausgabe



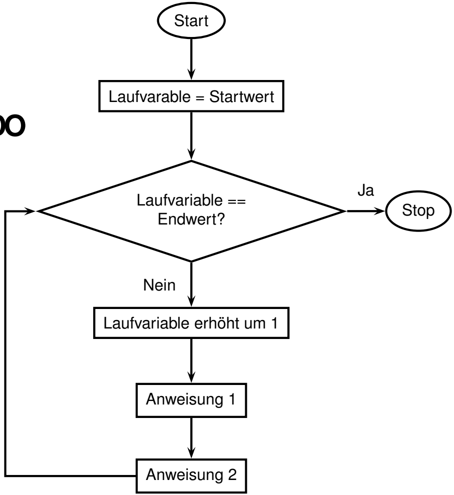
Strukturierte Programmierung: Selektion

```
if (Bedingung == wahr) then
   Anweisung
else if (Bedingung == wahr) then
   Anweisung
else
   Anweisung
end if
```



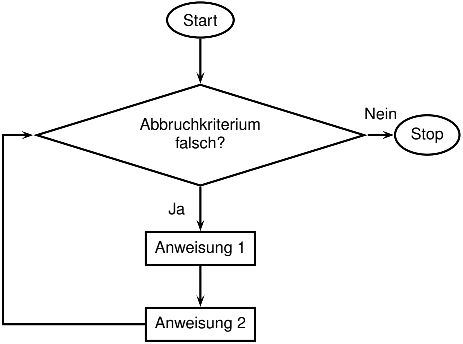
Strukturierte Programmierung: Repetition

```
FOR Laufvariable :=
    Startwert TO Endwert DO
BEGIN
  Anweisung 1;
  Anweisung 2;
END;
```



Strukturierte Programmierung: Repetition

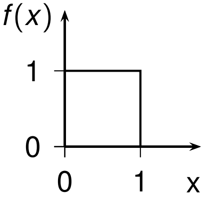
```
WHILE (Abbruchkriterium == falsch)
DO
BEGIN
   Anweisung 1;
   Anweisung 2;
END;
```



Zufallszahlen

- Elementarer Bestandteil von Simulationen
  - echte Zufallszahlen
  - Pseudozufallszahlen: Zufallszahlen werden von deterministischen Algorithmen erzeugt, sind also nicht wirklich zufällig
- sollen der Realität entsprechen
- Basis bilden gleichverteilte Zufallszahlen im Bereich [0; 1]
  - X heißt gleichverteilt in $[0;1]$, falls

$$f(x) = \begin{cases} 1 & \text{wenn } x \in [0;1] \\ 0 & \text{sonst} \end{cases}$$



Pseudozufallszahlen

- Konstruktion von Zufallszahlen beruht meist auf (nicht umkehrbarer) rekursiver Funktion
- Forderungen an guten Pseudo-Zufallszahlengenerator:
  1. Jede Zahl $\{1, \ldots, n\}$ soll mit gleich hoher Wahrscheinlichkeit $1/n$ vorkommen, bzw. jedes Intervall $[a, b]$ in $[0, 1]$ hat die ungefähre Häufigkeit $b - a$.
  2. Die k-Tupel, die aus k aufeinanderfolgenden Zahlen bestehen, sollen gleichverteilt sein (z. B. Paare von aufeinanderfolgenden Zahlen sollen gleich häufig sein).

  Bsp.:
  - $1 - 2 - 3 - 4 - 5 - 1 - 2 - 3 - 4 - 5 \rightarrow$ gleich hohe WK für alle Zahlen
  - $(1 - 2)(2 - 3)(3 - 4)(4 - 5)(5 - 1)(1 - 2) \ldots \rightarrow$ nur fünf verschiedene Paare (von 25 möglichen)

Pseudozufallszahlen

Midsquare Methode:

- primitiver Algorithmus zur Erzeugung von Zufallszahlen, entwickelt von John v. Neumann

- Für eine vierstellige Zufallszahl startet man mit einer beliebigen vierstelligen Zahl.
- Diese Zahl wird quadriert => man erhält eine achtstellige Zahl. Sollte das Resultat nicht achtstellig sein, werden 0-er vorangestellt.
- Die mittleren vier Zahlen sind nächste Zufallszahl und bilden den Ausgangswert für die nächste Quadratur.

Pseudozufallszahlen

Midsquare Methode - Beispiel zweistellige Zufallszahl

- Startwert: 25
- Quadratur: $\rightarrow$ 625
- 0 voranstellen $\rightarrow$ 0625
- mittleren zwei Zahlen sind nächste Zufallszahl $\rightarrow$ 62
- Quadratur: $\rightarrow$ 3844
- mittleren zwei Zahlen sind nächste Zufallszahl $\rightarrow$ 84

Pseudozufallszahlen

Kongruenzgeneratoren: $z_{i+1} = f(z_{i-k}, \ldots, z_i)$, mit $k \geq 0$, $i > k$.

- Allgemeiner Kongruenzgenerator:

$$y_i = \left(\left(\sum_{k=1}^{n} a_k y_{i-k} \right) + b \right) \mod m$$

- $n \in \mathbb{N}^+$: Anzahl der Zustandswerte
- $m \in \{2, 3, 4, \ldots\}$: Modul (Restwert der Division durch m)
- $a_1, \ldots, a_n \in \{0, \ldots, m-1\}$: Multiplikatoren
- $b \in \{0, \ldots, m-1\}$: Inkrement
- $y_1, \ldots, y_n \in \{0, \ldots, m-1\}$: Startwert

Für Zufallszahlen in $[0, 1]$: $u_i = \frac{y_i}{m}$

Pseudozufallszahlen

linearer Kongruenzgenerator: $n = 1$

$$y_i = a\, y_{i-1} + b \mod m$$

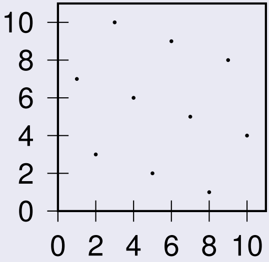
- Voraussetzungen für maximale Periodenlänge $L = m$ (Knuth 1997):
  1. Das Inkrement b ist zum Modul m teilerfremd.
  2. Jeder Primfaktor von m teilt $a - 1$.
  3. Wenn m durch 4 teilbar ist, dann auch $a - 1$.
- falls $m = 2^b$, dann ist $L = m$, wenn c ungerade und $a \equiv 1 \mod 4$
- multiplikativer Kongruenzgenerator: $b = 0$
- gemischter Kongruenzgenerator: $b \neq 0$

Pseudozufallszahlen

- Periodenlänge nicht ausreichend
- k-Tupel müssen gleichverteilt sein

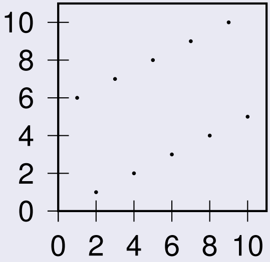
Bsp1: $y_{i+1} = 7\,y_i \mod 11$

voller Zyklus:
$1, 7, 5, 2, 3, 10, 4, 6, 9, 8$
2-Tupel: (1,7),(7,5),(5,2),...



Bsp2: $y_{i+1} = 6\,y_i \mod 11$

voller Zyklus:
$1, 6, 3, 7, 9, 10, 5, 8, 4, 2$
2-Tupel: (1,6),(6,3),(3,7),...



Pseudozufallszahlen

Parameterbeispiele:

- $m = 2^{35}$, c ungerade, $a \equiv 1 \mod 4$
- $m = 2^{31} - 1$, c beliebig, $a = 1$

Derzeit bester Zufallszahlengenerator:
Mersenne-Twister (Makoto Matsumoto, Takuji Nishimura 97,98)

- $m = 2^{28}$
- $a = 532393$
- Periode = $2^{19937} - 1$
- 623 Ebenen

Pseudozufallszahlen

Erzeugung von

- nicht $(0,1)$-gleichverteilter Zufallszahlen z aus $(0,1)$-gleichverteilter Zufallszahl u, bzw.
- nicht $[0,1]$-gleichverteilter Zufallszahlen z aus $[0,1]$-gleichverteilter Zufallszahl u:

$$z = (b-a)u + a$$

Falls Randstellen nicht inkludiert $\rightarrow$ über Abfragen ausschließen

Bsp.: In $[a,b] = [5,8]$-gleichverteilte Zufallszahl

u	$(b-a)u+a$
0.0	5.0
0.1	5.3
0.2	5.6
...	...
1.0	8.0

Pseudozufallszahlen

Endliche diskrete Verteilung zu gegebenener Wahrscheinlichkeitsverteilung

ξ_k	ξ_1	ξ_2	$\dots$	ξ_n
p_k	p_1	p_2	$\dots$	p_n

- $Q_0 = 0$
- $Q_j := \sum_{k=1}^{j} p_k \qquad$ für $j = 1, \dots, n$
- zu u_i bestimme j: $Q_{j-1} \leq u_i < Q_j$
- $z := \xi_j$

Beispiel: Würfel

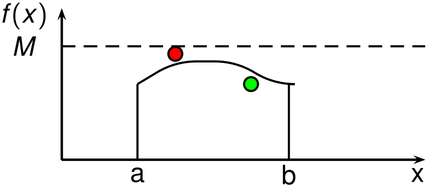
ξ_k	1	2	$\dots$	6
p_k	$1/6$	$1/6$	$\dots$	$1/6$
Q_k	$1/6$	$2/6$	$\dots$	$6/6$

Pseudozufallszahlen

Erzeugung von ZZ mittels Verwerfungsmethode

- Dichtefunktion f bekannt mit
  - f nach oben beschränkt ($f(x) < M$)
  - endliches Intervall ($x \in [a, b]$)

1. Generation von 2 gleichverteilten Zufallszahlen:
   - ZZ1 im Bereich $[a, b]$
   - ZZ2 im Bereich $[0, M]$
2. ZZ1 in f einsetzen
3. wenn $ZZ2 > f(ZZ1) \rightarrow$ Verwerfe ZZ1, ansonsten behalte ZZ1



Pseudozufallszahlen

Erzeugung von ZZ mittels Inversionsmethode

- Erzeugung stetig verteilter Zufallszahlen zur Verteilungsfunktion $F(x)$
- Voraussetzung: $F(x)$ ist invertierbar mit Umkehrfunktion F^{-1}

1. Erzeuge stetig gleichverteilte Zufallszahl u (zwischen 0 und 1)
2. $ZZ = F^{-1}(u)$

Beispiel:

$$F(x) = \begin{cases} 0 & x < 0 \\ 1 - \frac{1}{1+x^2} & x \geq 0 \end{cases}$$

Erzeuge $u(0,1)$

$$ZZ = \sqrt{\frac{1}{1-u} - 1}$$

Pseudozufallszahlen

Erzeugung von Exponentialverteilten Zufallszahlen

- Parameter $\lambda > 0$

$$F(x) = \begin{cases} 0 & x < 0 \\ 1 - e^{-\lambda \cdot x} & \text{sonst} \end{cases}$$

- erzeuge stetig gleichverteilte Zufallszahl u ($\in [0, 1]$)

$$ZZ = -\frac{1}{\lambda} \ln(ZZ1)$$

Inversionsmethode:

$$\begin{aligned} 1 - e^{-\lambda \cdot x} &= y \\ e^{-\lambda \cdot x} &= 1 - y \\ -\lambda \cdot x &= \ln(1 - y) \\ x &= -\frac{1}{\lambda} \ln(1 - y) \end{aligned}$$

Pseudozufallszahlen

Erzeugung von Normalverteilten $N(0,1)$ bzw. $N(\mu,\sigma)$ Zufallszahlen

- $N(0,1)$:
  - erzeuge stetig gleichverteilte Zufallszahlen u_1,u_2 ($\in [0,1]$)

$$ZZ1 := \sqrt{-2\ln u_1} \cdot \sin(2\pi u_2)$$
$$ZZ2 := \sqrt{-2\ln u_1} \cdot \cos(2\pi u_2)$$

  - (nur eine der beiden Methoden verwenden)
- $N(\mu,\sigma)$:
  - erzeuge Zufallszahl zz $N(0,1)$-verteilt (siehe oben)
  - $ZZ1 := \sigma \cdot zz + \mu$

Pseudozufallszahlen

Erzeugung von Geometrisch verteilten Zufallszahlen

$$Q_z(k) = p(1-p)^{k-1} \quad k = 1, 2, \ldots$$

- erzeuge stetig gleichverteilte Zufallszahl u ($\in [0, 1]$)

$$z := \text{int}\left[\frac{\ln u}{\ln(1-p)}\right] + 1$$

Pseudozufallszahlen

Erzeugung von Binomialverteilten ($B(n, p)$) Zufallszahlen

- Erzeuge n stetig gleichverteilte Zufallszahlen u_i:

$$z := |\{i|u_i < p\}|$$

- d.h. z ist die Zahl der u_i, die kleiner als p sind

Pseudozufallszahlen

Erzeugung von Poisson-verteilten Zufallszahlen mit Parameter $\lambda > 0$

- Erzeuge sukzessive Zufallszahlen u_i, bis

$$\prod_{i=1}^{k} u_i \leq e^{-\lambda} < \prod_{i=1}^{k-1} u_i$$

$k - 1$ ist $P(\lambda)$-verteilte Zufallszahl

- theoretischer Hintergrund: Anzahl an stochastisch unabhängigen identisch mit Parameter λ exponentialverteilten Zufallsvariablen, die benötigt werden, damit ihre Summe das erste Mal größer als 1 ist, ist Poisson-verteilt mit λ.

Pseudozufallszahlen

Zusammenhang Poisson-Verteilung $\leftrightarrow$ Exponentialverteilung über Warteschlange

- Wenn Zwischenankunftszeiten einer Warteschlange exponentialverteilt mit Erwartungswert 1 $\rightarrow$ Anzahl der Ankünfte im Intervall $[0, \lambda]$ poissonverteilt mit Parameter λ
- $\rightarrow$ Simulation von exponentialverteilten Zwischenankunftszeiten y_i mit Parameter 1, bis Summe der Zwischenankunftszeiten die Intervallgrenze λ erreicht $\rightarrow$ Anzahl an erzeugten Zufallszahlen ist poissonverteilt mit Parameter λ

$$\sum_{i=1}^{k} y_i \leq \lambda < \sum_{i=1}^{k+1} y_i$$

Chi-Quadrat-Anpassungs-Test

χ^2-Test auf Akzeptanz von n erzeugten Zufallszahlen für Q_0

1. Zerlegung des Wertebereiches der zugrundeliegenden Wahrscheinlichkeitsverteilung in r disjunkte Teilmengen (sinnvoll: $r \approx \sqrt(n))$ $A_1, \ldots, A_r$, sodass für die Wahrscheinlichkeiten $p_i = Q_0(A_i)$ gilt $np_i \geq 5$, für $i = 1, \ldots, r$.
2. Berechne aus den n Zufallszahlen $\xi_1, \ldots, \xi_n$ die Anzahl der Werte in A_i:
$$n_i = |\{\xi_j | \xi_j \in A_i\}|$$
3. Berechne die Testgröße:
$$\chi_0^2 = \sum_{i=1}^{r} \frac{(n_i - np_i)^2}{np_i}$$
4. Bestimme aus der Tabelle der χ^2-Verteilung mit $r-1$ Freiheitsgraden zu α die Ablehngrenze χ_{r-1}^{α}
5. Falls $\chi_0^2 \leq \chi_{r-1}^{\alpha}$, akzeptiere Zufallszahlen, ansonsten verwerfen

Chi-Quadrat-Anpassungs-Test: Beispiel

Beispiel: 100 gleichverteilte Zufallszahlen zwischen 0 und 1

1. Zerlegung in r Teilmengen
   - $np_i \geq 5$
   - $n = 100 \rightarrow p_i \geq 0.05$
   - $\rightarrow$ höchstens 20 Klassen
2. Bei 10 Klassen:

	$[0, 0.1[$	$[0.1, 0.2[$	$[0.2, 0.3[$	$[0.3, 0.4[$	$\ldots$	$[0.9, 1]$
ist	9	15	12	10	$\ldots$	12
soll (np_i)	10	10	10	10	$\ldots$	10

3. $\chi_0^2 = \sum_{i=1}^{r} \frac{(n_i - np_i)^2}{np_i} = \frac{1}{10} + \frac{25}{10} + \frac{4}{10} + 0 + \ldots + \frac{4}{10}$

- $\alpha = 0.05 \rightarrow$ mit WK 0.05 wird H_0 abgelehnt, obwohl sie richtig ist.
- laut Tabelle: $\chi_9^{0.05} = 16.92$
- falls $\chi_0^2 \leq \chi_9^{0.05} \rightarrow$ Nicht verwerfen

Runtest auf Unabhängigkeit

- Besitzen $l+1$ Zufallszahlen $u_1, u_2, \ldots, u_l + 1$ die Eigenschaft $u_1 < u_2 < \ldots < u_l > u_l + 1$, so spricht man von einem Run der Länge l
- Bezeichnet R die Zufallsvariablen "Länge des Runs", so besitzt R die Wahrscheinlichkeitsverteilung

$$P(R = l) = \frac{1}{l!} - \frac{1}{(l+1)!}$$

  mit Erwartungswert $E(R) = e - 1$

- Für $u_1, u_2, \ldots, u_l + 1$ bestimme
  - K : Anzahl der Runs und
  - k_l Anzahl der Runs mit Länge l
- Teste mit Chi-Quadrat-Anpassungstest

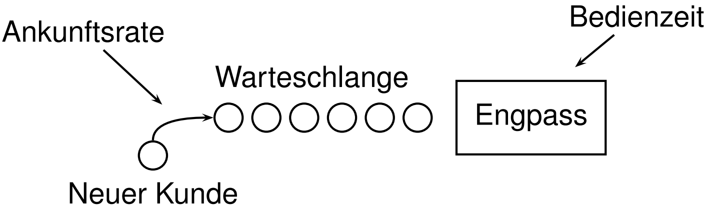
Runtest: Beispiel

Beispiel bei 39 Runs:

Länge	1	2	3	4
k_l	24	10	4	1
$p_i = P(R = l)$	$\frac{1}{1!} - \frac{1}{2!} = \frac{1}{1} - \frac{1}{2} = \frac{1}{2}$	$\frac{2}{6}$	$\frac{3}{24}$	$\frac{4}{120}$
np_i	19.5	13	4.875	1.625

- $$\chi_0^2 = \frac{5,5^2}{19.5} + \frac{3^2}{13} + \frac{0.875^2}{4.875} + \frac{0.625^2}{1.625} = 2.64$$
- $$\chi_3^{0.95} = 7.82 \rightarrow \text{ nicht ablehnen}$$

Warteschlangensimulation



- Kunden werden nach unterschiedlichen Kriterien bedient

Warteschlangensimulation

- Gesuchte Parameter:
  - mittlere Wartezeit
  - Auslastungsgrad
  - mittlere Schlangenlänge
- simulierte Parameter:
  - Ankunftsrate, Zwischenankunftszeit
  - Bedienzeit
- Simulationsdauer/Abbruchkriterium:
  - Zeit
  - Kundenanzahl

Warteschlangesimulation

Version 1:

- Zwischenankunftszeit zufällig gleichverteilt in Minuten im Intervall $\{1;5\}$
- Bedienzeit zufällig gleichverteilt in Minuten im Intervall $\{1;3\}$
- Abbruchkriterium: 50 Kunden
- Abarbeitung in FIFO Reihenfolge

Warteschlangesimulation

- Zufallsparameter (*RND* gleichverteilt zwischen $(0, 1)$)
  - Zwischenankunftszeit für Kunde i:

$$a_i = (\text{int})(RND * 5) + 1$$

  - Bedienzeit für Kunde i:

$$b_i = (\text{int})(RND * 3) + 1$$

$(\text{int})x \rightarrow$ Abrundung von x (Streichung der Nachkommastellen)
Da Nachkommastellen abgeschnitten werden, ist Intervall um 1 größer

Warteschlangensimulation

- Ankunftszeit von Kunde i:

$$t_i = \sum_{j=1}^{i} a_j$$

- Endzeit von Kunde i:

$$f_1 = t_1 + b_1$$

$$f_i = \max\{f_{i-1}, t_i\} + b_i$$

Warteschlangesimulation

- Wartezeit von Kunde i

$$w_i = \max\{f_{i-1} - t_i, 0\}$$

  - $f_{i-1} \leq t_i \rightarrow$ keine Wartezeit, $f_i = t_i + b_i$
  - $f_{i-1} > t_i \rightarrow$ Wartezeit, $f_i = f_{-i} + b_i$

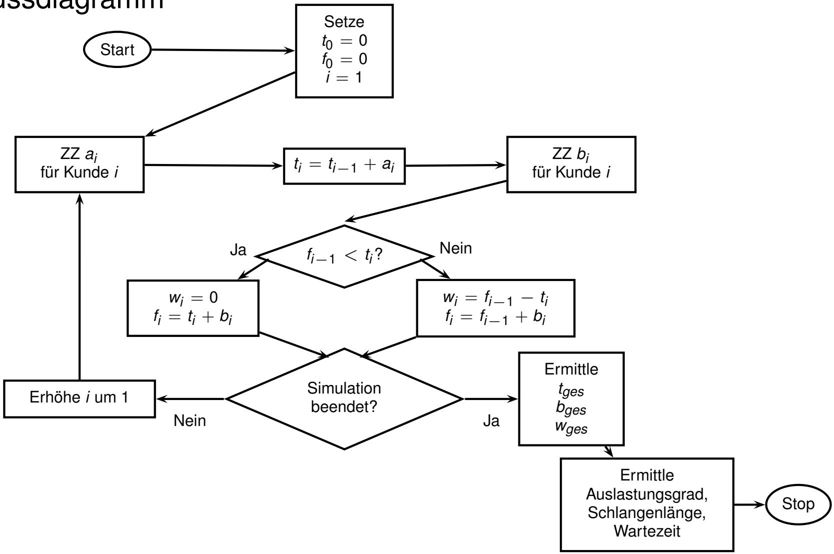
Warteschlangensimulation

Parameter

- mittlere Wartezeit: $\frac{w_{ges}}{n}$
- Auslastungsgrad: $\frac{b_{ges}}{t_{ges}}$
- mittlere Schlangenlänge: $\frac{w_{ges}}{t_{ges}}$

Warteschlangensimulation

Flussdiagramm



Warteschlangensimulation

ID	a_i	b_i	t_i	f_i	w_i
1	2	1	2	3	0
2	3	2	5	7	0
3	1	3	6	10	1
4	1	3	7	13	3
5	1	2	8	15	5
6	5	1	13	16	2
7	3	2	16	18	0
...	...	...	...	...	...
$\sum$		14			11

Nach 7 Kunden:

- mittlere Wartezeit: $\frac{w_{ges}}{n} = \frac{11}{7}$
- Auslastungsgrad: $\frac{b_{ges}}{f_n} = \frac{14}{18}$
- mittlere Schlangenlänge: $\frac{w_{ges}}{f_n} = \frac{11}{18}$

Warteschlangensimulation

Version 2:

- Mit WK p kommt ein neuer Kunde in einem fixen Zeitintervall (z. B. pro Minute) an
- Wenn zumindest ein Kunde anwesend ist, wird mit WK q ein Kunde fertig bedient.
- Ankünfte und Bedienungen sind über die Perioden hin unabhängige Ereignisse
- Abbruchkriterium: n Perioden

Warteschlangensimulation

Parameter für jeden Zeitraum i berechnen:

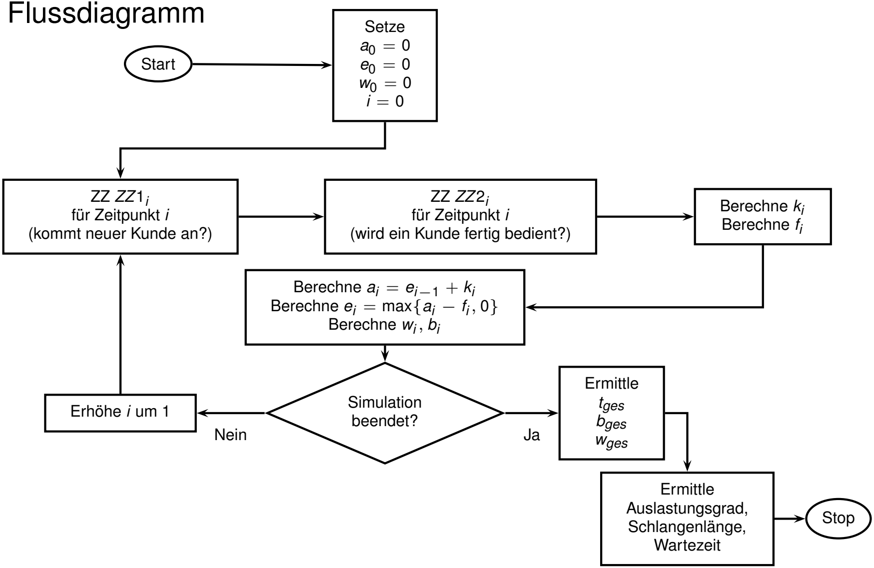
- Summe anwesender Kunden zu Beginn von Zeitpunkt i: a_i
- Neuer Kunde zu Zeitpunkt i: K_i
- Kunde fertig bedient zu Zeitpunkt i: f_i
- Summe anwesender Kunden zu Ende von Zeitpunkt i: e_i
- Summe Wartezeit zu Zeitpunkt i: $w_i = \max\{a_i - 1, 0\}$
- Falls zumindest ein Kunde anwesend zu Zeitpunkt i: Bedienzeit $b_i = 1$

Warteschlangensimulation

- Zufallszahlen $ZZ1_i$ und $ZZ2_i$ (gleichverteilt zwischen 0 und 1)
  - Wenn $ZZ1_i < p \rightarrow$ neuer Kunde kommt zu Zeitpunkt i an $\rightarrow k_i = 1$, ansonsten $k_i = 0$
  - Wenn $ZZ2_i < q \rightarrow$ wenn Kunde anwesend, dann wird einer zu Zeitpunkt i fertig bedient $\rightarrow f_i = 1$, ansonsten $f_i = 0$
- $a_i = e_{i-1} + k_i$
- $e_i = \max\{a_i - f_i, 0\}$

Warteschlangensimulation

Flussdiagramm



Warteschlangensimulation

t	neuer Kunde	Summe Kunden Anfang	Kunde fertig	Summe Kunden Ende	Wartenden Kunden	Bedienung
1	1	1	0	1	0	1
2	1	2	0	2	1	1
3	1	3	1	2	2	1
4	0	2	0	2	1	1
5	1	3	1	2	2	1
6	0	2	1	1	1	1
7	0	1	1	0	0	1
8	0	0	0	0	0	0
...	...	...	...	...	...	...
$\sum$	4				7	7

Nach 7 Kunden:

mittlere Wartezeit: $\frac{w_{ges}}{n} = \frac{7}{4}$

Auslastungsgrad: $\frac{b_{ges}}{t_{ges}} = \frac{7}{8}$

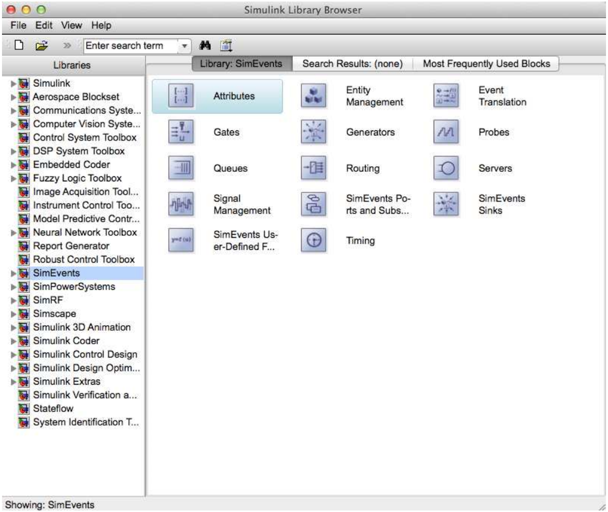
mittlere Schlangenlänge: $\frac{w_{ges}}{t_{ges}} = \frac{7}{8}$

Simulink

- Kommerzielle Software der Firma MathWorks
- Interaktive, graphische Entwicklungsumgebung
- Vorgefertigte und erweiterbare Bibliotheken
- In Matlab integriert
- Bibliotheken werden mittels Matlab-Funktionen bzw. Matlab-Code erweitert
- Ähnlich wie in Matlab Basisbibliotheken und Erweiterungen

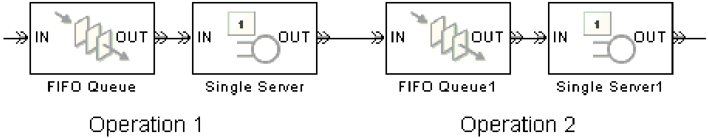
Simulink

Library Browser



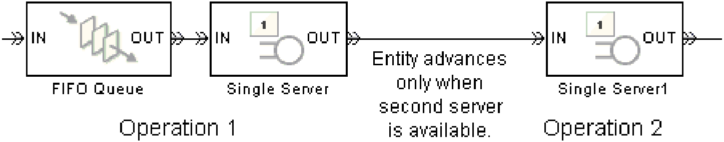
Simulink

Beispiel: Zwei Server mit zwei Warteschlangen



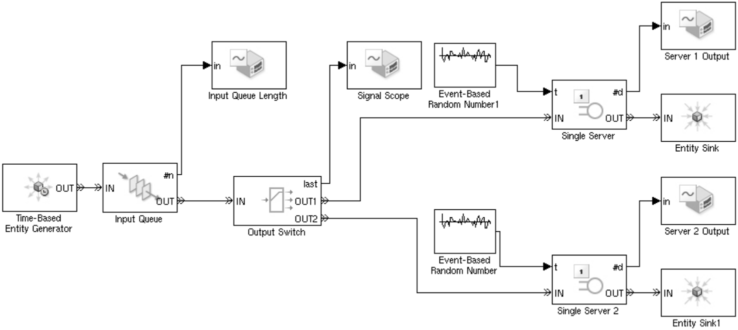
Simulink

Beispiel: Zwei Server mit einer Warteschlange



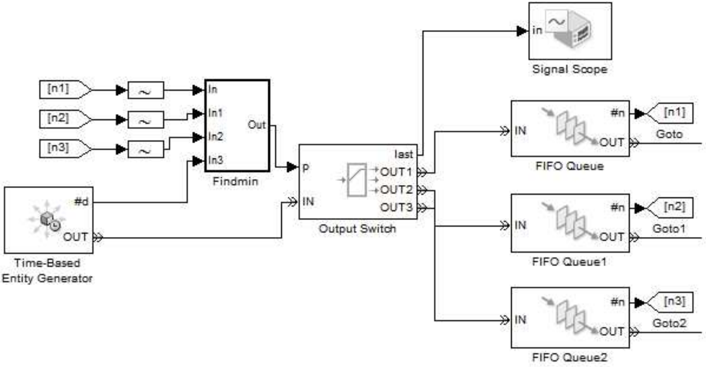
Simulink

Beispiel: Zwei parallele Server



Simulink

Drei Server mit kürzester Warteschlange



Kopplung Simulation und Optimierung

Vorteile Simulation

- Berücksichtigung praxisrelevanter Komplexität
- Berücksichtigung von real auftretenden Zufallsereignissen (Mengen- und Zeitschwankung, Ausfälle) durch Stochastik

Vorteile Optimierung

- Auffindung besserer (im Idealfall optimaler) Lösungen
- Einblick die Lösungsfindung

Klassifizierungen von Kopplungen

Verschiedene Kriterien für Klassifizierung von Verknüpfungen zwischen Simulation und Optimierung $\rightarrow$ Entscheidend für Kopplungsart ist gegenseitige Abhängigkeit der Simulations- und Optimierungsprozedure

- Hierarchische Architektur $\rightarrow$ eine Methode ist dominant und steuert die andere Methode
- Sequentielle Architektur: Ergebnisse der einen Methode ist abgeschlossene Methode
- Mischung aus den beiden Architekturen

Optimierung in Simulation integriert

- Hierarchische Architektur
- Falls Simulation dominant $\rightarrow$ Optimierung löst Problemstellung aufgrund des aktuellen Status des Simulationsmodell und gibt Daten für nächsten Simulationslauf zurück
- Bsp.: Bewertung einer Reihenfolgeoptimierung bei Engpass (vgl. Beispiel Warteschlange; bislang davon ausgegangen, dass Engpass mittels FIFO Sequenz abgearbeitet wird; muss nicht der Fall sein $\rightarrow$ Abarbeitung wird optimiert, mit neuer Sequenz wird weitersimuliert)

Simulation als Bewertungsfunktion der Optimierung

- Hierarchische Architektur
- Simulation stellt Ergebnisdaten für Optimierung zur Verfügung
- Simulation ist Prognosefunktion
- Bsp.: Problem wird mittels Genetischem Algorithmus gelöst $\rightarrow$ Fitness-Function wird mittels Simulation berechnet.

Simulationsergebnisse als Startwert der Optimierung

- Sequentielle Architektur
- Zunächst wird über Gesamtzeitraum simuliert
- Ergebnisse werden als Eingangsgröße bei Optimierung verwendet
- Bsp.: Zur Abschätzung von initialen Kapazitätsgrenzen kann eine Simulation verwendet weden; diese Kapazitätsgrenzen können dann mittels Optimierung neu berechnet werden.

Optimierungsergebnisse zur Konfiguration von Simulation

- Sequentielle Architektur
- Optimierung wird vorher durchgeführt, Simulation dient zur Überprüfung des Ergebnisse
- Oft können nicht alle Faktoren in einem Modell zur mathematischen Optimierung abgebildet werden (oder es ist nicht sinnvoll, da danach z. B. Rechenzeit zu lange wird) $\rightarrow$ Modell auf elementare Kriterien beschränken
- Lösung mittels Simulation auf Realisierbarkeit (Machbarkeitsstudie) verifizieren.