Vorlesungsreihe Simulation betrieblicher Prozesse

Simulation von diskreten Modellen

Prof. Dr.-Ing. Thomas Wiedemann
email: wiedem@informatik.htw-dresden.de

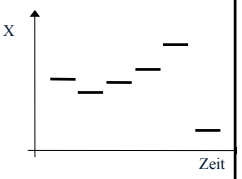
HOCHSCHULE FÜR TECHNIK UND WIRTSCHAFT DRESDEN (FH)
Fachbereich Informatik/Mathematik

Simulation von diskreten Modellen

- Einführung
- Abgrenzung zu kontinuierlichen Modellen
- Typische Anwendungsgebiete diskreter Simulatoren
- Aufbau diskreter Simulationssysteme
  - Grundarchitektur
  - Arten der Zeitsteuerung
- Software für die diskrete Simulation

Einführung in die diskrete Simulation

- Bei der Klassifizierung der Simulationsmodelle nach der Art des Zustandsüberganges der Systemparameter wird zwischen kontinuierlichen und diskontinuierlichen Systemen unterschieden.
- Bei den diskontinuierlichen Systemen, im folgenden kurz als **diskrete Systeme** bezeichnet, erfolgen die Zustandsübergänge nur zu ganz bestimmten Zeitpunkten. Zwischen den einzelnen Zeitpunkten bleiben die Zustandsgrößen konstant.



- Typische Anwendungsfälle zeitdiskreter Systeme ergeben sich bei
  - der allgemeinen Teilefertigung,
  - der Lagerhaltung abzählbarer Produkte,
  - in der Telekommunikation bei der Vermittlung einzelner Gespräche,
  - und im Verkehr.
- Wie bereits bei der Klassifizierung der Simulationsmodelle ausgeführt wurde, kann aus dem Zeitverhalten eines realen Systems nicht zwingend auf den Charakter des Modells geschlossen werden.

Typische Merkmale diskreter Simulationsmodelle

Ansatzpunkte zur Abgrenzung der diskreten von der kontinuierlichen Simulation:

PRO diskrete Simulation:

- eine zu betrachtende **Individualität der System- und Modellobjekte**
- Bei einer sehr geringen Anzahl von Systemobjekten (<30) spielt das Verhalten einzelner Objekte eine große Rolle.
- Ein stark typenabhängiges Verhalten erfordert eine getrennte Beschreibung für jeden Objekttyp. Bei kontinuierlichen Modelle führt dieser zur Aufspaltung der Modellvariablen und stark anwachsender Komplexität.
- Objekte mit Kombinationen von Eigenschaftstypen (z.B. Verhalten in Abhängigkeit von Farbe / Gewicht) sind mit einer Aufspaltung in Variablentypen kaum sinnvoll modellierbar

Pro kontinuierliche Simulation

- Stetige Werteverläufe (schwer realisierbar bei diskreten Simulationssystemen)
- sehr lange Betrachtungszeiträume (Langzeitstudien)
- bei sehr vielen Systemobjekten nivellieren sich die einzelnen zeitdiskreten Ereignisse und lassen sich kontinuierliche Modelle günstiger berechnen

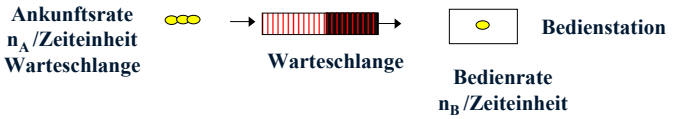
Anwendungsgebiete diskreter Simulationsuntersuchungen

Bedien- und Warteschlangensysteme

- häufigstes Einsatzgebiet diskreter Simulationswerkzeuge
- Untersuchungen haben gezeigt, daß sich die Werkstücke in üblichen Fertigungseinrichtungen bis zu 90% der Zeit in Warteschlangen befinden.
- Handel und Dienstleistungssektor versuchen im Interesse der Kundenzufriedenheit die Anzahl und Länge der Warteschlangen zu reduzieren.

Beschreibung von Bedien- und Warteschlangen

- Als Ausgangsgrößen werden die Ankunftsrate der zu bedienenden Objekte (Personen, Teile , Fahrzeuge etc.) und die Bedienrate (z.B. in 20 Personen pro Stunde) verwendet.



- Gesucht werden die mittlere Warteschlangenlänge, die mittlere Wartezeit pro zu bedienendem Objekt und bei stochastischen Einflüssen auch die maximale Warteschlangenlänge. Bei Bedarf werden auch verschiedene Strategien zur Organisation und Steuerung der Warteschlangen geprüft.

Anwendungsgebiete II

Investitionsplanung von Fertigungssystemen

- Kapazitätsberechnungen unter Berücksichtigung von stochastischen Einflußgrößen
- Bestimmung von Lagergrößen (Pufferlager, Materiallager)
- Layoutplanung zur Optimierung des innerbetrieblichen Transports
- Personalplanung

Operative Fertigungsplanung

- Reihenfolgeplanung für oder anstelle von Fertigungsleitständen mit dem Ziel der Auslastungsmaximierung und Einhaltung von Lieferzeiten
- Kostensimulation zur Minimierung der Fertigungskosten
- Personaleinsatzplanung
- Zunehmende Bedeutung bei der Supply-Chain-Analyse

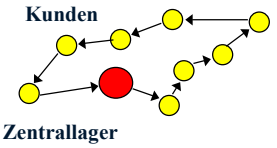
Zu modellierende Objekte

- Maschinen, (Werkzeuge) Transportmittel
- Produkte, Fertigungshilfsmittel
- Bedienpersonal, Wartungspersonal

Anwendungsgebiete III

Logistik

- In sehr vielen Bereichen der Logistik und des Transports sind sowohl die Anforderungen (z.B. der Kunden) wie auch die Rahmenbedingungen (Wetter, Verkehrsstaus) stochastischer Natur.
- Bei gleichzeitiger Optimierung in sich widersprüchlicher Ziele (niedrige Kosten, hohe Kundenzufriedenheit, Zuverlässigkeit, Verfügbarkeit von Spezialfahrzeugen) sind Standardverfahren wie die Simplexmethode kaum sinnvoll einsetzbar.
- Zur Bestimmung optimaler Fahrtrouten und Servicestrategien kann die diskrete Simulation sehr gut eingesetzt werden.
- Häufig erfolgt dabei der Test verschiedener Fahrtrouten und Strategien anhand von simulativen Planspielen unter Einbeziehung von zufälligen Modellparametern.



Grundlegende Bestandteile diskreter Simulationssysteme

Anstelle einer mathematischen Beschreibung erfolgt bei diskreten Simulationssystemen eine Systemnachbildung durch algorithmische Beschreibungen:

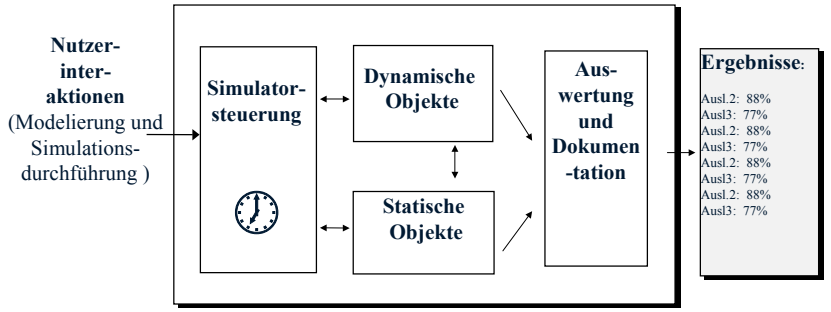
- Das Verhalten der Systemobjekte wird programmtechnisch nachgebildet.
- Eigenschaften der Systemobjekte werden auf Softwareobjekte mit entsprechenden Attributen übertragen.
- Der enge Zusammenhang zwischen Systemobjekt und Softwareobjekt war auch der Ausgangspunkt für die erste Realisierung objektorientierter Programmierung mit der der Simulationssprache SIMULA bereits in den 60er Jahren.

Die Modellobjekte (Entities) lassen sich in folgende Modellbestandteile unterteilen :

- **Statische (permanente) Objekte**, häufig auch als Einrichtungen (oder treffender engl. Facilities) bezeichnet, dienen zur Nachbildung der Bedienstationen, Bearbeitungseinrichtungen oder sonstigen Systemobjekten mit Servicefunktionen.
- **Dynamische Objekte** (engl. auch Transactions) realisieren die Darstellung sich dynamisch durch das System bewegenden Personen, Fahrzeuge oder Teile.

Aufbau von diskreten Simulationssystemen

- **Die Simulationssteuerung und dabei insbesondere die Zeitsteuerung** ist verantwortlich für die korrekte Nachbildung der Wechselwirkungen zwischen den statischen und dynamischen Objekten.
- Weitere Module dienen zur Aufzeichnung des Simulationsverlaufs für die spätere statistische Auswertung oder zur Animation von Abläufen.



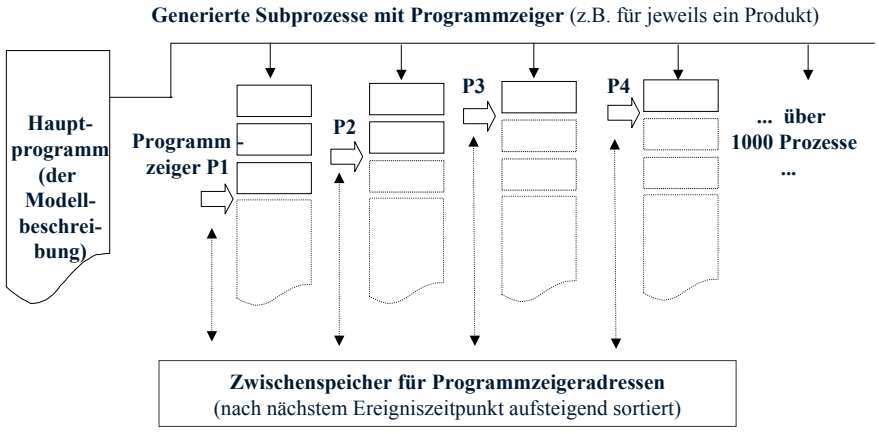
Anforderungen an die Simulatorsteuerung

- Die Wechselwirkungen von Objekten in diskreten Simulationsmodellen sind in der Regel von einer hohen Parallelität der Vorgänge geprägt.
- Zu einem beliebigen Zeitpunkt laufen mehrere Prozesse unabhängig voneinander ab. Die Reihenfolge der Programmabarbeitung ergibt sich dabei allein aus der zeitlichen Folge der Einzelereignisse (engl. Events)
- Die Abbildung der parallelen Prozesse auf ein Ein-Prozessorsystem geht über die Anforderungen eines normalen Multitaskings hinaus. Aufgrund meist sehr vieler, dafür aber sehr kleiner Einzelereignisse muß das Umschalten von einem auf den anderen Prozeß sehr effizient und schnell erfolgen.
- Erste, spezialisierte Simulationssprachen verfügten über Koroutinenfähigkeit, d.h. die Eigenschaft, innerhalb einer sequentiellen Beschreibung des Objektverhaltens auf andere Prozesse umschalten zu können. Auf der Ebene der Assembler-programmierung läßt sich Korroutinenfähigkeit durch eine Manipulation der Rückkehradresse einer Funktion auf dem Prozessorstack erreichen.
- Heutige, moderne Programmiersprachen wie C++ oder DELPHI verfügen leider über keine Koroutinenfähigkeit. Eine direkte Modellierung und Simulation diskreter Systeme ist daher nur mit erhöhtem Aufwand möglich.

Quasiparalleler Ablauf der Simulation

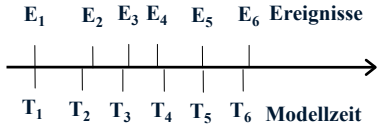
Optionen zur Parallelisierung des Programmlaufs

- Zerlegung des Simulationsprogramms in reine Funktionsaufrufe und Verwaltung der Funktionen über Vektoren von Pointern auf Funktionen
- Verwendung von Threads für eine kleine Anzahl von Prozessen (noch rerlativ aufwendig und langsam- vgl. System SILK von Kilgore [Kilg99])



Zeitsteuerung mit fixen Zeitinkrementen

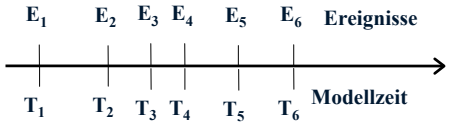
- In Analogie zur Schrittweite Δt bei der kontinuierlichen Simulation kann auch bei der diskreten Simulation mit einem vorgegebenen Zeitinkrement gearbeitet werden.
- **Zu jedem Zeitpunkt T_i** wird überprüft ob entsprechende Ereignisse eingetreten sind.



- Die Realisierung des Simulationssteuerung ist bei diesem Verfahren relativ einfach, da nur die Modellzeit mit dem Inkrement monoton steigend hochgezählt wird.
- Weniger einfach ist die Modellierung von zufälligen Ereignissen, da bei jedem Takt das Eintreten des Ereignisses überprüft werden muß. Bei komplexeren Verteilungsfunktionen muß dabei die Vergangenheit berücksichtigt werden, was zu einem beträchtlichen Aufwand in der Modellbeschreibung führt.
- **Das Verfahren der fixen Zeitinkremente wird nur selten und meist nur bei spezialisierten, hoch aggregierten Modellen verwendet, bei denen wenige Zustandsvariable sich sehr häufig ändern.**

Zeitsteuerung mit variablen Zeitinkrementen

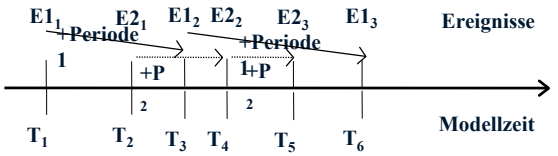
- Die einzelnen Zeitpunkte T_i können auch durch Auswertung des Modells ermittelt werden.
- Die Modellzeit wird damit nicht mehr durch die modellexterne Simulationssteuerung vorgegeben, sondern wird asynchron durch die Abläufe im Modell gesetzt.
- Ein Modellzeitpunkt entspricht mindestens einem Ereignis !



- Bei der **ereignisorientierten Simulation** wird durch die Simulationssteuerung ein **Systemkalender** (engl. Future event list) mit den nächsten möglichen Ereignissen geführt.
- Durch Bestimmung des Minimums aller anstehenden Ereigniszeitpunkte wird der nächste Modellzeitpunkt bestimmt und die Simulationsuhr entsprechend gesetzt.

Berechnung des Systemkalenders

- Die „Termine“ im Systemkalender werden durch eine Vorausbestimmung des nächsten identischen Ereignisses bei der Abarbeitung des gerade aktuellen Ereignisses berechnet:
  - Bei unabhängigen, periodischen Ereignissen wird einfach die Periode zur aktuellen Simulationszeit addiert und das Ergebnis als nächster Ereigniszeitpunkt in den Systemkalender eingetragen.
  - Analog wird bei zufälligen Ereignissen bei der Abarbeitung des gerade ausgelösten Ereignisses durch die Addition eines per Zufallszahlenzahlengenerator und Verteilungsfunktion bestimmten Intervalles zur aktuellen Modellzeit das nächste zufällige Ereignis gleichen Typs in den Systemkalender eingetragen.
  - Bereits feststehende Ereignisse wie geplante Betriebsunterbrechungen oder das Simulationsende können natürlich bereits zum Simulationsbeginn fest in den Systemkalender eingetragen werden. Die einzelnen Zeitpunkte T_i können auch durch Auswertung des Modells ermittelt werden.



Besonderheiten bei abhängigen Ereignissen

- Durch den Systemkalender wird die korrekte Abarbeitung der unabhängigen Ereignisse garantiert. Die Betrachtung und programmtechnisch effiziente Bearbeitung der abhängigen Ereignisse ist dagegen wesentlich aufwendiger.
- Bei der klassischen ereignisorientierten Simulation wird nach der Abarbeitung aller unabhängigen Ereignisse eines Zeitpunktes **eine Prüfung aller abhängigen Ereignisse** durchgeführt. Bei einem Zutreffen der für die Abhängigkeit relevanten Bedingung wird das Ereignis ausgeführt.
- Typische Abhängigkeiten aus der Sicht der realen Systeme sind
  - **Das Warten auf die Auslösung** - damit das Ereignis stattfindet, müssen ein oder mehrere andere Ereignisse eingetreten sein.
  - **Blockieren** - der Zustand einer Modellgröße oder eines Prozesses wird bei Eintreten eines anderen Ereignisses in Entsprechung einer Unterbrechung oder eines Abbruches verändert (z.B. wird eine Maschine blockiert wenn das Ausgangslager voll ist; der Maschinenstatus ist damit zwar „Bearbeitung beendet“ aber die nächste Bearbeitung kann nicht beginnen, da daß Teil noch auf der Maschine ist)
- In der Praxis werden vielfach alle prinzipiell möglichen Ereignisse in der Kalenderliste geführt und es wird $T_i=\infty$ bei allen abhängigen Ereignissen eingetragen. Falls ein abhängiges Ereignis eintreten kann, wird die aktuelle Systemzeit eingesetzt und die zentrale Simulationssteuerung führt die zugehörigen Zustandsänderungen des Modells durch.

Aktivitätsorientierte Zeitsteuerung

- Die relativ einfache Methodik der ereignisorientierten Zeitsteuerung wird ineffizient, wenn sehr große Modelle vorliegen und sehr viele abhängige Ereignisse getestet werden müssen (großer Systemkalender).
- Als Alternative bietet sich die Extraktion und Aufbereitung der Bedingungen an. Die Konzentration auf die Abprüfung der Bedingungen für Aktivitäten läßt Raum für weitere Optimierungen innerhalb der Simulationssteuerung:
  - Bei einer Zerlegung komplexer, mehrstufiger Bedingungen in eine Reihe sequentiell durchführbarer Tests kann bei logischen UND-Verknüpfungen die Unterbedingung mit der geringsten Eintrittswahrscheinlichkeit an den Anfang gestellt werden.
  - Aufgrund der ausschließlichen Betrachtung von Ereignissen während der Simulation können auch die Bedingungen nur im Resultat von Ereignissen erfüllt werden.
  - Nach einem Negativtest einer Bedingung wird die Abhängigkeit der Bedingung in alle der Prüfung zugrunde liegende Zustandsvariablen eingetragen und der Termin der nächsten Überprüfung der Bedingung auf Unendlich gesetzt.
  - Kommt es bei einer der Entities zu einer Änderung der Zustandsvariablen wird bei dieser Änderung die Kalenderzeit der abhängigen Entities wieder auf die aktuelle Modellzeit gesetzt und die Bedingung damit wieder komplett geprüft. Besonders bei sehr großen Modellen, bei welcher das jeweilige Einzelereignis relativ selten eintritt, kann dies zu einer drastischen Reduzierung der Rechnerzeiten führen. Nachteilig ist natürlich der erhöhte Aufwand zur Verwaltung der Abhängigkeiten.

Vergleich der Zeitsteuerungsarten

Beim Test aller Bedingungen zu jedem Zeitpunkt kommt es aufgrund des Verhältnisses

- Gesamtzahl aller Tests = Anzahl aller Modellzeitpunkte * Anzahl aller Bedingungen

und der etwa linearen Abhängigkeit der Zeitpunkte und Bedingungen von der Elementanzahl zu

Gesamtzahl aller Tests ~ (k_t * Anzahl Modellelemente) * (k_B * Anzahl Modellelemente)

und damit zu einem **quadratischen Wachstum des Rechenaufwandes** in Abhängigkeit von der Anzahl der Modellelemente (k_t und k_b sind modellabhängige Koeffizenten).

Bei der Optimierung der Zeitsteuerung durch die zuvor beschriebene Methode der Inaktivierung von Bedingungstests ändert sich das Verhältnis zu

Gesamtzahl ~ (k_{t2} * Anzahl Modellelemente) * (K_{B2} *Mittlere Zahl ausgelöster Ereignisse)

welches in der Regel zu einem etwa linearen Wachstum des Rechenaufwandes führt, auf jeden Fall aber günstiger als die allgemeine Methode ist.

- Diese relativ allgemeine Schätzung des Rechenaufwandes zeigt aber auch, daß bei der Beurteilung von Simulationssystemen die Frage des Verhaltens bei sehr großen Modellen nicht einfach von kleineren Modellen her extrapoliert werden kann.
- Je nach der (meist unbekannten) intern implementierten Zeitsteuerung kann das Wachstum der Rechenzeiten völlig unterschiedlichen Gesetzen unterliegen. (Zusätzlich können natürlich auch noch Probleme bei Speicherverwaltung und eventuell notwendigen Swapping auf die Platte zu völligen Einbrüchen bei der Performance führen).

Prozeßorientierte Simulation

Eine weitere Option der Zeitsteuerung realisiert die prozeßorientierte Simulation :

- Statt externer Zeitsteuerung wird die Steuerung direkt durch die Modellelemente realisiert. Man sieht das Modell aus der Sicht der Elemente .
- Die Zeitsteuerung hat bei der prozessorientierten Simulation weniger steuernden und treibenden Charakter. Sie ist verantwortlich für die Synchronisation der einzelnen Prozesse. **Jeder Prozeß führt seine eigene Simulationszeit** und meldet den Wunsch nach Weiterschaltung an die zentrale Zeitsteuerung. Diese wertet alle ankommenden Anforderungen aus und gewährt der Anforderung mit dem kleinsten Zeitpunkt die Weiterschaltung. In Analogie zur Bedingungsprüfung werden bei der prozess-orientierten Simulation auch überfällige Prozesse wieder aufgerufen.
- Der Aufbau prozeßorientierter Simulatoren erfordert bereits bei der Implementierung der Basisroutinen eine geeignete Programmiersprache, da die Prozeßunterprogramme als nichthierarchische Koroutinen mit beliebigen Ein- und Austrittspunkten codiert werden müssen. Dies ist z.B. mit Fortran oder C nicht möglich. Es entstanden daher speziell auf die Belange prozeßorientierter Simulation zugeschnittene Programmiersprachen wie SIMULA oder SIMSCRIPT.
- Der prozessorientierte Ansatz ist deutlich günstiger anwendbar bei der Beschreibung von Modellen mit sehr kompliziertem Verhalten der einzelnen dynamischen Elemente und **für verteilte Simulationssysteme.**

Weitere Aufgaben der Simulationssteuerung

Start und Beendigung der Simulation

Zum definiertem Abbruch der Simulation haben sich folgende Verfahren bewährt:

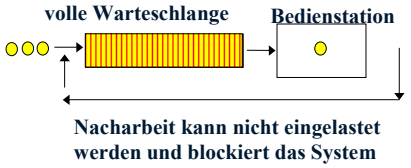
- auf der Basis einer gewissen Anzahl von Durchläufen dynamischer Objekte,z.B. Simulationsende nach dem Passieren von 1000 Fahrzeugen oder der Bearbeitung von 500 Teilen,
- nach einer Zeitspanne innerhalb der Modellzeit, z.B. nach einer simulierten Zeit von 5 Wochen
- nach einer bestimmten Rechenzeit in der Echtzeit, dies ist besonders beim Test fehlerhafter Modelle günstig, da somit auf jeden Fall abgebrochen wird.

Unterdrückung von Ein- und Ausschwingvorgängen

- Bei komplexen Modellen können nicht alle Zustandsvariablen bereits zu Simulationsbeginn auf die Betriebswerte gesetzt werden. Das Modell muß sich erst "einlaufen" und verfälscht stark die berechneten Statistiken. Zur Vermeidung dieses Problems stellen viele Simulationssysteme Funktionen zum expliziten Sperren oder Starten der statistischen Aufzeichnung bereit. Im einfachsten Fall kann ein Rücksetzten nach der Einschwingphase erfolgen.
- Erkennung der Einlaufphase durch Messung des Trends von Referenzvariablen

Erkennung von internen Modellblockierungen

- Bei Rückkopplungen oder gleichzeitiger Benutzung von Ressourcen (Transportmittel, Arbeitsmittel etc.) kann es zur einem Blockieren (Dead lock) des Modells kommen.



- Falls die Blockierung das gesamte Modell betrifft, ist dieser Zustand in der Kalenderliste mit ausschließlich auf Unendlich terminierten Ereignissen zu erkennen. Die meisten Systeme geben daraufhin eine Warnung aus und brechen die Simulation ab.
- Zur Behebung der Modellblockierung müssen in Anlehnung an die reale Situation z.B. zusätzliche Pufferlager für die Rückkopplungen mit absolut höchster Priorität bei der Weitergabe geschaffen werden.
- Blockierungen in Teilmodellen sind meist nur durch Beobachtungen erkennbar und müssen von echten Ruhephasen abgrenzt werden.