# Modellierung verteilter Systeme

## Grundlagen der Programm und Systementwicklung

Wintersemester 2010/11

Prof. Dr. Dr. h.c. Manfred Broy

Unter Mitarbeit von Dr. M. Spichkova, J. Mund, P. Neubeck

Lehrstuhl Software & Systems Engineering

# Zustandssicht

## Systeme als Zustandsmaschinen

# Einleitung

- Systeme besitzen in der Regel Zustände
- Verhalten eines Systems lässt sich durch Übergänge zwischen Zuständen modellieren
- Übergänge können gekoppelt sein mit (ausgelöst werden durch)
  - Aktionen
  - Ein- und Ausgabe
- Beispiele
  - Rechner und Schaltungen
    - Gatterzustand und Schaltvorgang
  - Imperative, zuweisungsorientierte Programme
    - Variablenwerte, Programmzeiger und Zuweisungen, Sprünge
  - Anwendung bei formalen Sprachen
    - Reguläre Sprachen als endliche Automaten, Buchstaben auf Übergängen

# Definition: Nichtdeterministische Zustandsmaschine (ZM)

- Eine nichtdetermistische Zustandsmaschine $(\Delta, \Sigma_0)$ besteht aus
  - einer *Zustandsmenge* $\Sigma$,
  - *Anfangszuständen* $\Sigma_0 \subseteq \Sigma$ (oder einem Anfangszustand $\sigma_0 \in \Sigma$) und
  - einer *Zustandsübergangsfunktion* $\Delta: \Sigma \rightarrow \wp(\Sigma)$

- Die ZM $(\Delta, \Sigma_0)$ heißt
  - *total*, wenn $\forall \sigma \in \Sigma : \Delta(\sigma) \neq \varnothing$
  - ansonsten *partiell*
  - *deterministisch*, wenn $\forall \sigma \in \Sigma : |\Delta(\sigma)| \leq 1$ und $|\Sigma_0| = 1$
  - ansonsten *nichtdeterministisch*

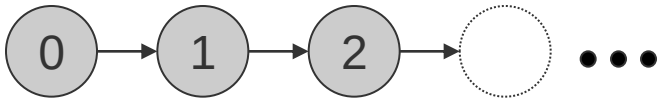
- Eine *Spur* oder *Ablauf* der ZM $(\Delta, \Sigma_0)$ ist eine endliche oder unendliche Folge von Zuständen $\sigma_0, \sigma_1, \ldots$ mit $\sigma_0 \in \Sigma_0$ und $\sigma_{i+1} \in \Delta(\sigma_i)$

# Beispiel: Zähler

- Zustandsmenge entspricht den natürlichen Zahlen
- Startzustand ist die Null
- Übergang zur nächst größeren Zahl

$$\Delta_{counter}(x) = \{x + 1\}$$

- Interpretation als Programm: Zahl als Variable und wiederholtes inkrementieren




- Genau ein unendlicher Ablauf: 0,1, 2, …

# Beispiel: Sortierer

- Zustand besteht aus zwei Sequenzen (s, t) natürlicher Zahlen
- Startzustände sind (⟨⟩, t) mit beliebiger Sequenz t
- Die durch t gegebene Sequenz soll sortiert werden, so dass schließlich ein Zustand (s, ⟨⟩) erreicht wird, wobei s der Sequenz entspricht, die durch Sortieren aus t entsteht.

- Forderung:
  - Von jedem Startzustand (⟨⟩, t) wird
  - nach endlich vielen Schritten der Endzustand (s, ⟨⟩) erreicht,
  - wobei s der Sequenz entspricht, die durch Sortieren aus t entsteht.

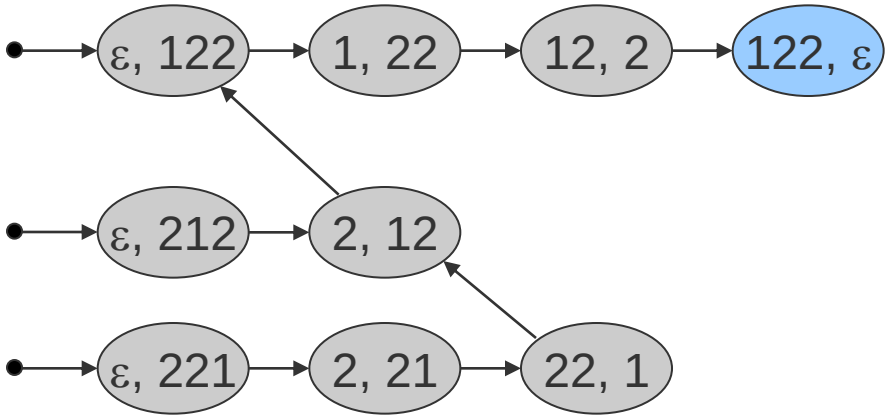
# Beispiel: Sortierer - erzeugt aufsteigende Sortierung

$$\Sigma_{sort} = \mathbb{N}^* \times \mathbb{N}^*$$

$$\Delta_{sort}(\langle\rangle, \langle n\rangle \circ t) = \{(\langle n\rangle, t)\}$$

$$\Delta_{sort}(s \circ \langle n \rangle, \langle m\rangle \circ t) = \textbf{if } n > m \textbf{ then } \{(s, \langle m\rangle \circ \langle n\rangle \circ t)\} \textbf{ else } \{(s \circ \langle n\rangle \circ \langle m\rangle, t)\} \textbf{ fi}$$

- Diagramm für Permutationen von ⟨1, 2, 2⟩



# Definition: Erreichbare Zustände

- Ein Zustand heißt *erreichbar*, wenn er in einem Ablauf enthalten ist.
- Induktive Konstruktion der Menge $R_\infty \subseteq \Sigma$ aller *erreichbaren Zustände*

$$\Sigma_{i+1} = \Delta(\Sigma_i)$$

$$R_\infty = \bigcup_{i \geq 0} \Sigma_i$$

- Andere Charakterisierung:
  - $R_\infty$ ist die kleinste Zustandsmenge, die folgende Ungleichung für die Menge R erfüllt

$$R \supseteq \Sigma_0 \cup \bigcup_{\sigma \in R} \Delta(\sigma)$$

# Definition: Permanente Invariante & Stabiles Prädikat

- Wir übertragen die Beweismethode der Induktion auf Zustandsmaschinen:

- Ein Prädikat auf Zuständen p: $\Sigma \rightarrow$ B heißt *(permanente) Invariante*, wenn p für alle erreichbaren Zustände gilt

- Ein Prädikat auf Zuständen p: $\Sigma \rightarrow$ B heißt *stabil*, wenn für alle Zustände $\sigma$ und $\sigma$' gilt:

$$p(\sigma) \wedge \sigma' \in \Delta(\sigma) \Rightarrow p(\sigma')$$

# Zusammenhang von Stabilität und Invarianz

- Sei p: $\Sigma \rightarrow$ B ein Prädikat auf Zuständen einer ZM ($\Delta$, $\Sigma_0$), dann gilt: Ist p stabil und gilt für jeden Anfangszustand $\sigma_0 \in \Sigma_0$, dann ist p eine permanente Invariante.

- Sei p: $\Sigma \rightarrow$ B eine permanente Invariante und q: $\Sigma \rightarrow$ B ein Prädikat auf Zuständen. Gilt für jeden (erreichbaren) Zustand $\sigma$

$$p(\sigma) \Rightarrow q(\sigma)$$

  dann ist auch q eine permanente Invariante.

# Definition: Nichtdet. ZM mit markierten Übergängen

- Eine nichtdeterministische ZM mit markierten Übergängen $(\Delta, A, \Sigma_0)$ besteht
  - aus einer Zustandsmenge $\Sigma$,
  - einer *Aktionsmenge* A,
  - Anfangszuständen $\Sigma_0 \subseteq \Sigma$ (oder einem $\sigma_0 \in \Sigma$) und einer
  - Zustandsübergangsfunktion $\Delta_a: \Sigma \rightarrow \wp(\Sigma)$ für jedes $a \in A$.
- Eine Aktion $a \in A$ heißt ***bereit*** im Zustand $\sigma \in \Sigma$, wenn

$$\Delta_a(\sigma) \neq \emptyset$$

gilt.

Wir schreiben dann enabled(a, $\sigma$)

# Definition: Nichtdet. ZM mit markierten Übergängen

- Für $\sigma' \in \Delta_a(\sigma)$ schreiben wir kurz

$$\sigma \xrightarrow{a} \sigma'$$

- Eine ZM mit markierten Übergängen ist deterministisch,
  wenn für jede Aktion $a \in A$ und jeden Zustand $\sigma \in \Sigma$ die Menge der Nachfolgezustände

$$\{\sigma' \in \Sigma : \sigma' \in \Delta_a(\sigma)\}$$

  höchstens ein Element enthält.

# Beispiel: Keller

Wir modellieren einen Keller über der Datenmenge Data mit

- der Zustandsmenge Data*
- den Aktionen
  $A_{stack}$ = {initStack, pop} $\cup$ {top(d) : d $\in$ Data} $\cup$ {push(d) : d $\in$ Data}
- und der Übergangsfunktion

$$\Delta_{initStack}(s) = \{\varepsilon\}$$

$$\Delta_{pop}(s) = \{s' : \exists d \in Data : s = \langle d \rangle \circ s'\}$$

$$\Delta_{push(d)}(s) = \{\langle d \rangle \circ s\}$$

$$\Delta_{top(d)}(s) = \begin{cases} \{s\} & \text{falls } first(s) = d \\ \varnothing & \text{sonst} \end{cases}$$

# Zustandsdiagramme

- Ein *Zustandsdiagramm* ist ein gerichteter Graph.
- Knoten heißen *Kontrollzustände* und stellen Mengen von Zuständen dar.
- Eine Zusicherung in einem Knoten charakterisiert dessen Zustandsmenge
- Bei markierten Zustandsmaschinen werden Kanten mit Aktionen markiert
- Kanten können mit einer Vorbedingung Q($\sigma$) und einer Nachbedingung R($\sigma$, $\sigma$‘) genau spezifiziert werden.
- Der Anfangszustand ist mit einem Pfeil mit Punkt als Anfang gekennzeichnet.

# Modellierung der Zustände mit Variablen/Attributen

- Menge von Variablennamen (Identifikatoren) V
- Menge von Variablenwerten M
- Jede Variablenbelegung

$$\sigma: V \rightarrow M$$

  beschreibt einen Zustand
- Zustandsraum ist $\Sigma = (V \rightarrow M)$
- Den Variablen können Sorten zugeordnet sein
- Prädikate über Zustände (Zusicherungen) werden über Variablen formuliert

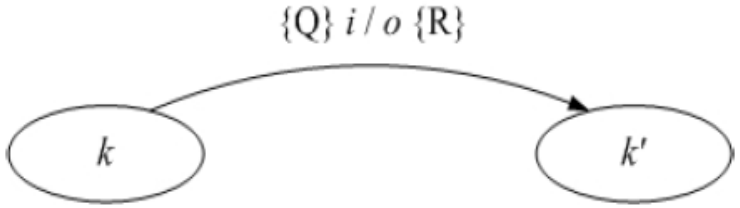
# Zustandssicht

## Systeme als Zustandsmaschinen

# Definition: Zustandsmaschinen mit Ein- und Ausgabe

- Eine nichtdet. ZM $(\Delta, \Sigma_0)$ mit *Eingabealphabet* I und *Ausgabealphabet* O besteht aus einer Zustandsmenge $\Sigma$, Anfangszuständen $\Sigma_0 \subseteq \Sigma$ (oder einem $\sigma_0 \in \Sigma$) und einer Zustandsübergangsfunktion $\Delta: \Sigma \times I \rightarrow \wp(\Sigma \times O)$

- Solche ZM heißen auch *Mealy-Automaten*
  - Ausgabe hängt von Eingabe und Zustand ab
- Spezialfall *Moore-Automaten*
  - Ausgabe hängt nur vom Zustand ab
- Für $(\sigma', o) \in \Delta_a(\sigma, i)$ schreiben wir kurz $\sigma' \xrightarrow{i/o} \sigma$
- Notation für Übergänge in Diagrammen



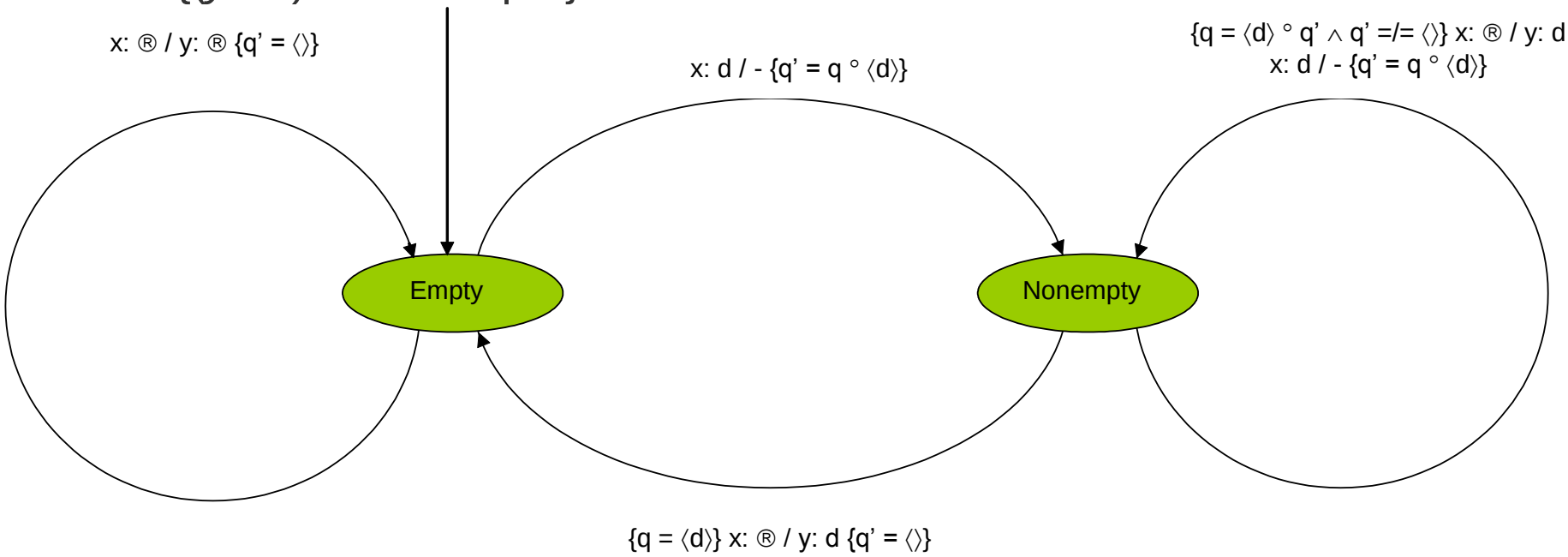
# Beispiel: Warteschlange als Zustandsmaschine

**Sort** Input = Data $\cup\{®\}$

**Sort** Output = Data $\cup\{®\}$

I = $\{(x : d) : d \in \text{Input}\}$

O = $\{(y : d) : d \in \text{Output}\}$



# Darstellung von Zustandsmaschinen

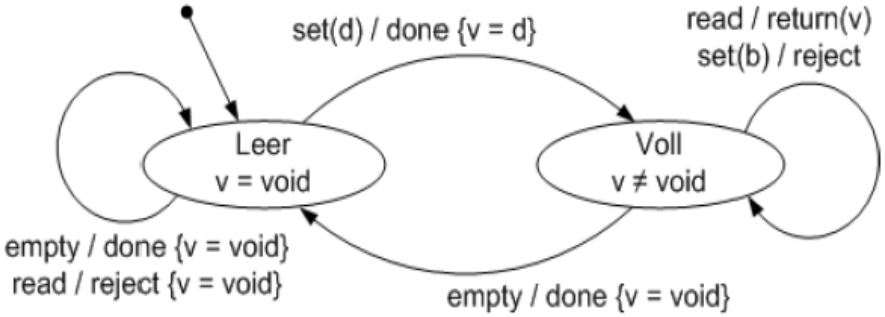
- Analytisch mit Mitteln der Mathematik und Logik
- Graphisch mit Zustandsübergangsdiagrammen
- Tabellarisch
- Als Matrix
- In der Notation einer Programmiersprache

# Beispiel: Speicherzelle

- Analytische Darstellung

$$I = \{set(d) : d \in Data\} \cup \{read, empty\}$$
$$O = \{return(d) : d \in Data\} \cup \{done, rejected\}$$
$$\Sigma = Data \cup \{void\}$$
$$\Delta(void, set(d)) = \{(d, done)\}$$
$$\Delta(d, read) = \{(d, return(d))\}$$
$$\Delta(d, empty) = \{(void, done)\}$$
$$\Delta(d, set(d')) = \{(d, rejected)\}$$
$$\Delta(void, read) = \{(void, rejected)\}$$
$$\Delta(void, empty) = \{(void, done)\}$$
$$\sigma_0 = void$$

- Graphische Darstellung



# Tabellarische Darstellung

- Eingabe in Zeilen, Zustände in Spalten
- Deterministischer Mealy-Automat
  - *Übergangstabelle:* Zielzustand im Kreuzungspunkt
  - *Ausgabetabelle:* Ausgabezeichen im Kreuzungspunkt

Übergangstabelle

| | $S_0$ | $S_1$ | $S_2$ |
|---|---|---|---|
| $x_1$ | $S_2$ | $S_0$ | $S_0$ |
| $x_2$ | $S_0$ | $S_2$ | $S_1$ |

Ausgabetabelle

| | $S_0$ | $S_1$ | $S_2$ |
|---|---|---|---|
| $x_1$ | $y_1$ | $y_1$ | $y_2$ |
| $x_2$ | $y_1$ | $y_2$ | $y_1$ |

- Deterministischer Moore-Automat
  - Übergangstabelle, Kennzeichnung der Spalten mit Ausgabe

Gekennzeichnete Übergangstabelle

| | $S_0/y_1$ | $S_1/y_1$ | $S_2/y_3$ | $S_3/y_2$ | $S_4/y_3$ |
|---|---|---|---|---|---|
| $x_1$ | $S_1$ | $S_4$ | $S_4$ | $S_2$ | $S_2$ |
| $x_2$ | $S_3$ | $S_1$ | $S_1$ | $S_0$ | $S_0$ |

# Tabellarische Darstellung ff.

- *Logikbasierte Tabelle*
  - Bsp. Speicherzelle gemäß der Formel
    $\Delta(\sigma, a)=\{(\sigma', b)\}$

| σ | a | σ' | b |
|---|---|---|---|
| void | read | void | reject |
| void | set(d) | d | done |
| void | empty | void | done |
| d | empty | void | done |
| d | read | d | return(d) |
| d | set(d') | d | reject |

  - Jede Zeile definiert einen Übergang
  - Richtige Auswahl der logischen Formel erlaubt sehr kompakte Darstellung

# Matrix-Darstellung

- Mealy-Automat
  - $|\Sigma| = n$; $n \times n$-Matrix ($m_{ij}$)
  - Existiert Übergang dann $m_{ij} = x/y$
  - Sonst $m_{ij} = -$

$$S_i \xrightarrow{x/y} S_j$$

| | $S_0$ | $S_1$ | $S_2$ |
|---|---|---|---|
| $S_0$ | $x_2/y_1$ | $-$ | $x_1/y_1$ |
| $S_1$ | $x_1/y_1$ | $-$ | $x_2/y_2$ |
| $S_2$ | $x_1/y_2$ | $x_2/y_1$ | $-$ |

- Moore-Automat
  - $|\Sigma| = n$; $n \times n$-Matrix ($m_{ij}$); n-Vektor ($v_i$)
  - Existiert Übergang dann $m_{ij} = x$ und $v_i = y$
  - Sonst $m_{ij} = -$

$$S_i \xrightarrow{x/y} S_j$$

| | $S_0$ | $S_1$ | $S_2$ | $S_3$ | $S_4$ | |
|---|---|---|---|---|---|---|
| $S_0$ | $-$ | $x_1$ | $-$ | $x_2$ | $-$ | $y_1$ |
| $S_1$ | $-$ | $x_2$ | $-$ | $-$ | $x_1$ | $y_1$ |
| $S_2$ | $-$ | $x_2$ | $-$ | $-$ | $x_1$ | $y_3$ |
| $S_3$ | $x_2$ | $-$ | $x_1$ | $-$ | $-$ | $y_2$ |
| $S_4$ | $x_2$ | $-$ | $x_1$ | $-$ | $-$ | $y_3$ |