Suche in Graphen

Beispiele:
- –Routenplanung
- –Fahrplanauskunft
- –Suche nach einem Beweis
- –Suche nach Gewinnstrategie
- –Planung

Modell für Problemlösen:
- Gegeben:
  - –Graph G = [V,E]
  - –„Anfangszustand“ $z_0 \in V$
  - –Menge von „Zielzuständen“ $Z_f \subseteq V$
- Probleme:
  - –Existiert ein Weg von z_0 zu einem $z_f \in Z_f$
  - –Konstruiere einen Weg von z_0 zu einem $z_f \in Z$
  - –Konstruiere optimalen Weg von z_0 zu einem $z_f \in Z_f$
    (bzgl. eines gegebenen Optimalitätskriteriums)

Planung: Modellierung als Graph

Mögliche Aktionen: $A = \{a_1,...,a_n\}$

Zustände (Knoten im Graphen):

V = durch Aktionen entstehende Situationen

Ausgangsituation: Anfangszustand z_0

Situationen, in denen Planungsziel erreicht ist: Zielzustände Z_f

Zustandsübergänge (Kanten im Graphen):

E = Übergänge zwischen Situationen durch Aktionen

$= \{ [v,v',a] \mid v,v' \in V \; a \in A \wedge v \text{ wird durch } a \text{ in } v' \text{ überführt} \}$

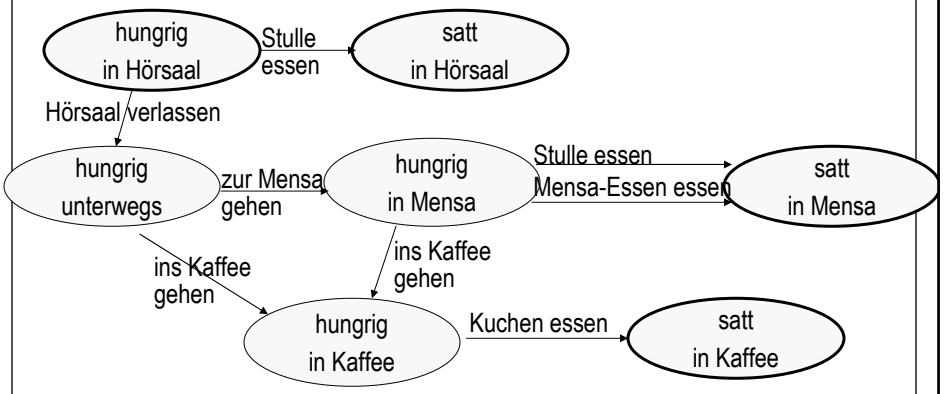
G ist ein Kanten-beschrifteter Graph mit Mehrfachkanten

$G = [V,E,f,A,\beta]$

mit $f([v,v',a]) = [v,v']$, $\beta([v,v',a]) = a$

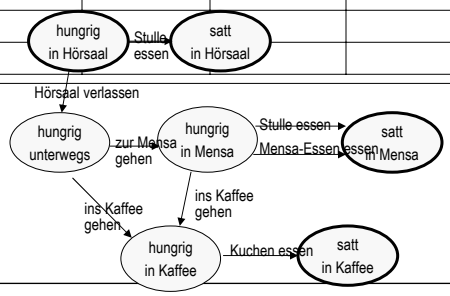
Planung: Modellierung als Graph

- Ausgangszustand: *hungrig, im Hörsaal*
- Bedingung an Zielzustände: *satt*



Übergangsmatrix

	Stulle essen	Hörsaal verlassen	zur Mensa gehen	Mensa-Essen essen	ins Kaffee gehen	Kuchen essen
hungrig, in Hörsaal	s., i.H.	h., u.				
hungrig, unterwegs			h., i.M.		h., i.K	
satt, in Hörsaal						
hungrig, in Mensa	s., i.M.			s., i.M.	h., i.K.	
hungrig, in Kaffee						s., i.K.
satt, in Mensa						
satt, in Kaffee						



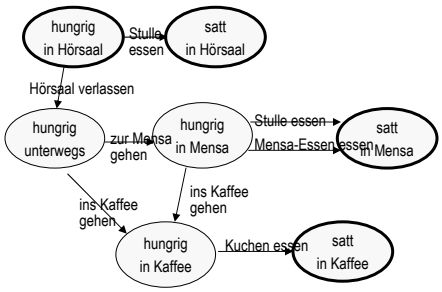
Übergangsmatrix

	Stulle essen	Hörsaal verlassen	zur Mensa gehen	Mensa-Essen essen	ins Kaffee gehen	Kuchen essen
hungrig, in Hörsaal	s., i.H.	h., u.				
hungrig, unterwegs			h., i.M.		h., i.K	
satt, in Hörsaal						
hungrig, in Mensa	s., i.M.			s., i.M.	h., i.K.	
hungrig, in Kaffee						s., i.K.
satt, in Mensa						
satt, in Kaffee						

Zeilen: Zustände z

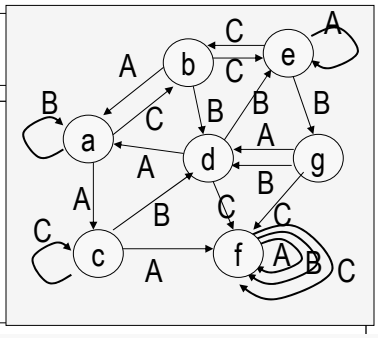
Spalten: Aktionen a

Matrix-Element: von z durch a erreichter Zustand z´



- Transitionssystem
- Automat
- Akzeptor

Transitionssystem



T = [Z,X,δ] mit

Z : Zustandsmenge

X : Eingangssignale

δ: $Z \times X \rightarrow Z$ Überführungsfunktion

Graph mit Knoten für Zustände z aus Z
und Kanten für Übergänge von z nach $\delta(z,x)$

Erweiterung: δ: $Z \times X^* \rightarrow Z$

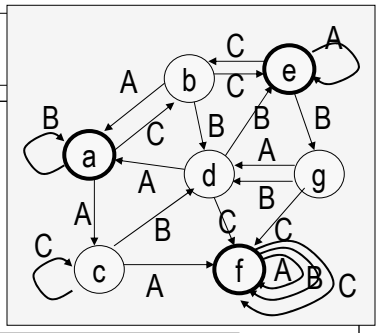
$\delta(z,x_1...x_n)$ ist der von z mit der Folge $x_1...x_n$ erreichte Zustand

$\delta(z,\lambda)=z$

$\delta(z,x_1...x_n\ x)= \delta(\delta(z,x_1...x_n), x)$

Rekursive Definition in freier Halbgruppe (Baum)

Akzeptor



T = [Z,X,δ] mit

„Anfangszustand“ $z_0 \in Z$

Menge von „Zielzuständen“ $Z_f \subseteq Z$

Akzeptierte Sprache:

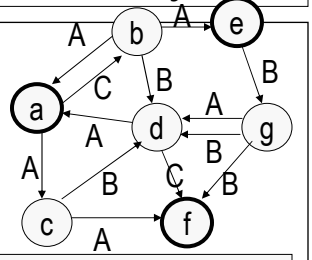
$L(T, z_0, Z_f) = \{ x_1...x_n \mid \delta(z_0,x_1...x_n) \in Z_f \} \quad \subseteq X^*$

L ist <u>regulär</u>,

genau dann, wenn T = [Z,X,δ], $z_0 \in Z$, $Z_f \subseteq Z$ existieren

mit X,Z endlich und L = $L(T, z_0, Z_f)$.

Nicht-deterministischesTransitionssystem



T = [Z,X,f] mit

Z : Zustandsmenge

X : Eingangssignale

$f : Z \times X \rightarrow 2^Z$ Überführungsfunktion

Erweiterung: $f: 2^Z \times X^* \rightarrow 2^Z$

$f(M,\lambda)=M$

$f(M,x_1...x_n\ x)= f(f(M,x_1...x_n), x)$

$f(z,x_1...x_n)$ sind die von z mit der Folge $x_1...x_n$ erreichten Zustände

Akzeptierte Sprache:

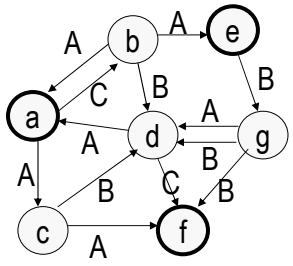
$L(T, z_0, Z_f) = \{ x_1...x_n \mid f(z_0,x_1...x_n) \cap Z_f \neq \varnothing \}$

regulär, falls X,Z endlich

Transitionssystem: Akzeptor

```
accept(P):-initial(Z),accept(Z,P).

accept(Z,[X|P]) :- delta(Z,X,Z1),accept(Z1,P).
accept(Z,[ ])   :- final(Z).
```




```
initial(a).
final(e).
final(f).
```

```
delta(a,x_c,b).
delta(a,x_A,c).
delta(b,x_A,a).
delta(b,x_A,e).
delta(b,x_B,d).
...
delta(g,x_B,f).
```

„Nicht-deterministisch"

Komplexität (Anzahl der Zustände/Knoten)

- 8-er Puzzle: 9! Zustände
  davon 9!/2 = 181.440 erreichbar
- 15-er Puzzle: 16! Zustände
  davon 16!/2 erreichbar
- ungarischer Würfel: $12 \cdot 4{,}3 \cdot 10^{19}$ Zustände
  1/12 davon erreichbar: $4{,}3 \cdot 10^{19}$
- Türme von Hanoi: 3^n Zustände für n Scheiben
  lösbar in $(2^n) - 1$ Zügen
- Dame: ca 10^{40} Spiele durchschnittlicher Länge
- Schach: ca 10^{120} Spiele durchschnittlicher Länge
- Go: 3^{361} Stellungen

Suchverfahren in Graphen

Graph: G = [Z,E] mit

- Anfangszustand $z_0 \in Z$
- Zielzuständen $Z_f \subseteq V$

Probleme:

- Speicher reicht nicht für vollständigen Zustandsraum
- Aufwand für Erkennen von Wiederholungen

Lösungsmethode:

„Expansion des Zustandsraumes“:

Schrittweise Konstruktion und Untersuchung von Zuständen

„konstruieren – testen – vergessen“

Expansionsstrategien

- **Richtung**
  - Vorwärts, beginnend mit z_0
    (forward chaining, data driven, bottom up)
  - Rückwärts, beginnend mit Z_f
    (backward chaining, goal driven, top down)
  - Bidirektional
- **Ausdehnung**
  - Tiefe zuerst
  - Breite zuerst
- **Zusatzinformation**
  - blinde Suche
  - heuristische Suche