Optimierung in dynamischer Umgebung

(Dozent: PD Dr. Ulf Lorenz)

TECHNISCHE UNIVERSITÄT DARMSTADT

Literatur und Danksagung

TECHNISCHE UNIVERSITÄT DARMSTADT

Literatur:

s. Webseiten der Veranstaltung

Dank

Für Anregungen und die Erlaubnis Unterlagen nutzen zu dürfen, möchte ich mich bedanken bei:
Prof. Dr. Meyer auf der Heide, Uni Paderborn,
Prof. Dr. Schindelhauer von der Uni Freiburg,
Prof. Dr. Ziegler, TU Darmstadt

Vorstellung

TECHNISCHE UNIVERSITÄT DARMSTADT

PD Dr. rer. nat. Ulf Lorenz

Eckdaten:

• *21.4.69, verheiratet, 3 Kinder

• Akademischer Oberrat in der Mathematik an der TU Darmstadt

• Arbeitsgruppe Diskrete Optimierung

• Fachbereich Mathematik der Technischen Universität Darmstadt

Forschungs-/Interessensgebiete:

- Diskrete Optimierung, Algorithmen

- Optimierung unter Unsicherheit

- Spiele

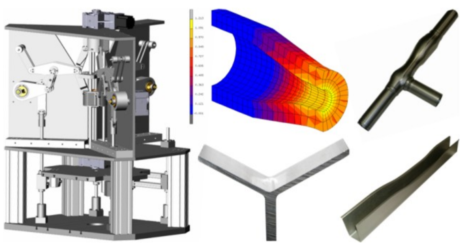
Übersicht II: Ausgangssituation Optimierung

Häufige Annahme für Planung und Entscheidungsfindungen in logistischen Prozessen und in Herstellungsprozessen:

- im vorhinein bestimmbare Daten

Beobachtende und auf Planabweichungen fexibel reagierende Kontrollstrategie

- existiert entweder nicht oder
- wird von der Planung getrennt betrachtet.



Folgen der Unsicherheit

- große Planabweichungen

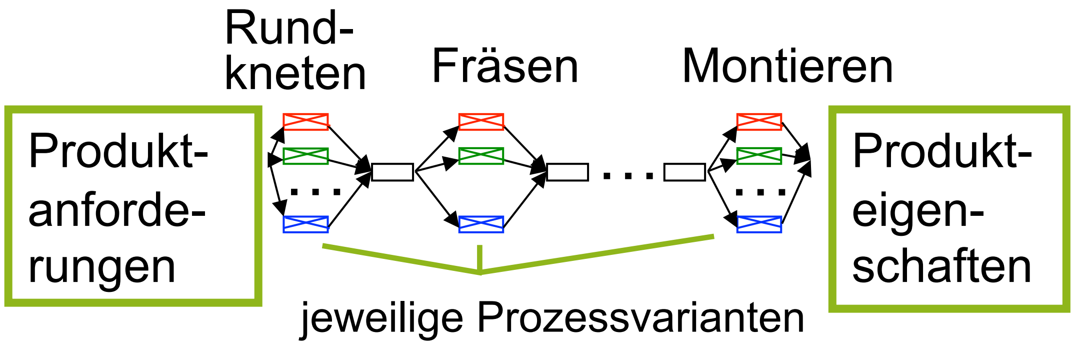
oder

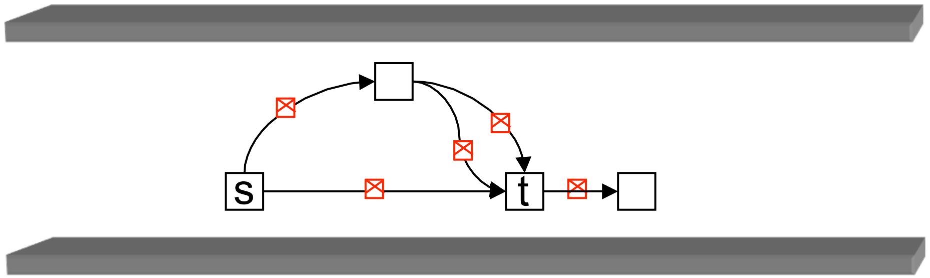
- hohe Sicherheitsbeiwerte
- Überdimensionierung
- große Pufferbildung

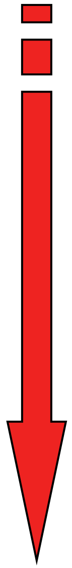
Übersicht II: Arbeitsprogramm im SFB 805

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Modellierung und Optimierung







$$\max_{y_1^{(u,v)} \in \mathbb{N}^{n_1}} c_1^T y_1 + \mathrm{Erw}[Q_2(y_1^{(u,v)}; \overline{\eta}_2)]$$

$$\text{s.t.} \sum_{v \in \delta^+(s)} y_1^{(s,v)} = 1$$

$$\text{mit } Q_t\left(y_{t-1}^{(u,v)}; \overline{\eta}_t(\omega)\right) := \max_{y_t^{(u,v)} \in \mathbb{R}^{n_t}} c_t(\omega)^T y_t^{(u,v)} + \mathrm{Erw}[Q_{t+1}\left(y_t^{(u,v)}; \overline{\eta_{t+1}}\right)]$$

$$\text{s.t.} \sum_{v \in (\delta^+(u)(\omega))} y_t^{(u,v)} - \sum_{u \in (\delta^-(v)(\omega))} y_{t-1}^{(u,v)} = 0$$

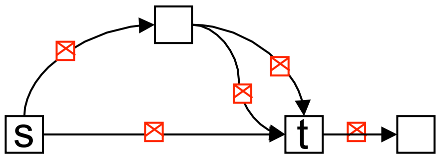
n-stufige, modulare Prozesskette mit verschiedenen Entscheidungs-alternativen

Netz von Entscheidungs-alternativen

mehrstufiges stochastisches Optimierungsmodell

Übersicht II: Andere Modellierungen

Modellierung und Optimierung, Alternativen



$\exists x_1 \forall x_2 \exists x_3 \forall x_4 \exists x_5 \ldots$

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 \leq b_1$$
$$a_{21}x_1 + a_{22}x_2 + a_{24}x_4 \leq b_2$$
$$a_{31}x_1 + a_{32}x_2 + a_{35}x_5 \leq b_3$$

$\exists x_1 \forall x_2 \exists x_3 \forall x_4 \exists x_5 \ldots$

$$\begin{aligned} a_{11}x_1x_2 + a_{13}x_3 & \leq b_1 \\ a_{21}x_1 + a_{22}x_2x_4 & \leq b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{35}x_5 & \leq b_3 \end{aligned}$$

Algorithmen:

- ganzzahlig (random / worst case):
  „von Aussen nach Innen"
  mittels backtracking

- kontinuierlich (random / worst case):
  „von Innen nach Aussen"
  mittels Variablemelemination
  oder
  „von Aussen nach Innen"
  mittels Dekomposition

Übersicht

- Einführung in Komplexitätstheorie

- Dynamic Graph Reliability Probleme
- Schach: Lösungsalgorithmen und Näherungsideen
- Go: UCT Lösungsverfahren
- Sokoban, Rushhour und Stackingprobleme

- Satz von Savich, speziell: NPSPACE = DPSPACE

- Stochastic Programming
- Quantifizierte Lineare Programme

Übersicht I

- Einführung in Komplexitätstheorie
  - Unentscheidbarkeit

    Welche Probleme können mit einem Algorithmus gelöst werden? Was ist überhaupt ein “Algorithmus”, was ist ein “Problem”?

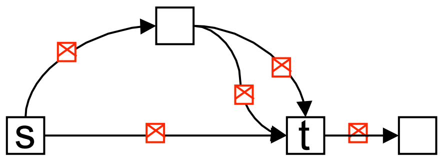
  - verschiedene Maschinenmodelle und formale Sprachen
  - Algorithmen, Komplexitätsklassen P, NP, PSPACE
  - Reduktionstechnik

Übersicht

- Das Dynamic Graph Reliability Problem

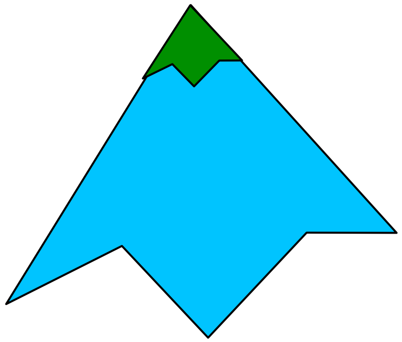
  geg.: ein DAG (gerichteter Graph ohne Kreise); Regeln, nach denen Kanten im Graphen ausfallen; Startknoten, Endknoten

  ges.: Gibt es eine Gewinnstrategie, die einem Walker im Startknoten erlaubt, an den Endknoten zu gelangen?

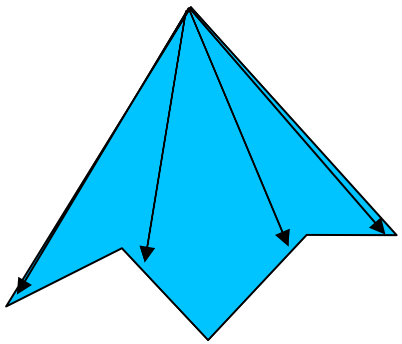


Übersicht

- Schach: Lösungsalgorithmen und Näherungsideen

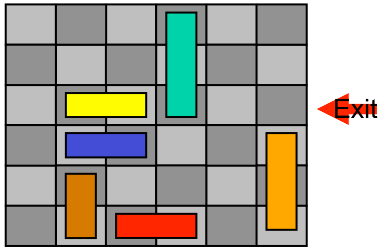


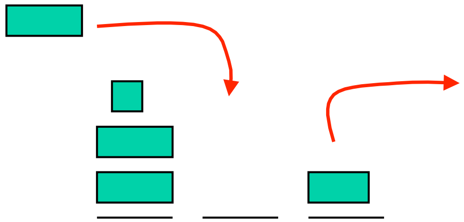
- Go: UCT Lösungsverfahren



Übersicht

- Sokoban, Rushhour und Stackingprobleme





- Satz von Savich, speziell: NPSPACE = DPSPACE
- Stochastic Programming

$$\max_{y_1^{(u,v)} \in \mathbb{N}^{n_1}} c_1^T y_1 + \mathrm{Erw}[Q_2(y_1^{(u,v)}; \overline{\eta_2})]$$

$$\text{s.t.} \sum_{v \in \delta^+(s)} y_1^{(s,v)} = 1$$

$$\text{mit } Q_t\left(y_{t-1}^{(u,v)}; \overline{\eta_t}(\omega)\right) := \max_{y_t^{(u,v)} \in \mathbb{R}^{n_t}} c_t(\omega)^T y_t^{(u,v)} + \mathrm{Erw}[Q_{t+1}\left(y_t^{(u,v)}; \overline{\eta_{t+1}}\right)]$$

$$\text{s.t.} \sum_{v \in (\delta^+(u)(\omega))} y_t^{(u,v)} - \sum_{u \in (\delta^-(v)(\omega))} y_{t-1}^{(u,v)} = 0$$

- Quantifizierte Lineare Programme

Es fängt ganz harmlos an

- Ein Alphabet Σ besteht aus einer endlichen Menge von Zeichen, z.B.
  - $\Sigma_1 = \{a,b,c\}$
  - $\Sigma_2 = \{0,1\}$
  - $\Gamma = \{0,1,x,y,z\}$
- Eine Zeichenkette (String/Wort) ist eine endliche Folge (Tupel) von Zeichen, z.B.
  - $w = abba$
    - Notation: $w_1=a$, $w_2=b$, $w_3=b$, $w_4=a$
    - Die Länge eines Worts wird mit $|w|$ beschrieben: $|w| = 4$
- Σ^* bezeichnet die Menge aller Zeichenketten über Alphabet Σ
  - z.B.: "abba" $\in \{a,b\}^*$
  - Die leere Zeichenkette wird mit ε bezeichnet.
    - Es gilt: $|\varepsilon| = 0$
- Eine Teilmenge von Σ^* wird als Sprache bezeichnet

Sprachbeschreibungen und Maschinen

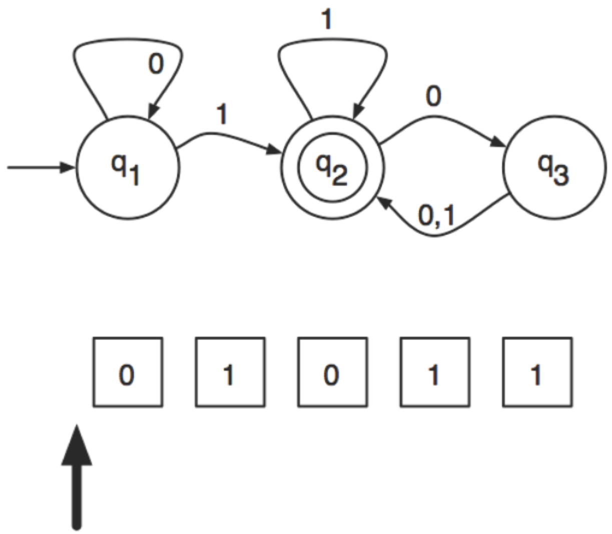
TECHNISCHE UNIVERSITÄT DARMSTADT

- Eine Sprache $L \subseteq \Sigma^*$ muss nun irgendwie beschrieben werden.
  - z.B. durch einen *regulären Ausdruck*: (0*10*)
    - $\emptyset$ ist ein regulärer Ausdruck.
    - ε ist ein regulärer Ausdruck.
    - $\forall\ a_i \in \Sigma$ ist a_i ein regulärer Ausdruck.
    - Sind x und y reguläre Ausdrücke, so auch $x \cup y$, (xy) und x^*.
    - Es gibt keine weiteren regulären Ausdrücke.

  - z.B. durch eine **Problembeschreibung**:
    - **Definition**: Ein *Entscheidungsproblem* ist ein input-output Tupel mit
      geg.: Kodierung eines Inputs einer Instanz, mittels Alphabet Σ
      ges.: ja/nein
    - Die Teilmenge aller Inputs, für die die Antwort "ja" ist, ist offenbar eine Sprache

Sprachbeschreibungen und Maschinen

- Die Frage, ob ein $w \in \Sigma^*$ ein Wort aus einer Sprache $L \subseteq \Sigma^*$ ist, kann unterschiedlich schwierig zu lösen sein
  - Bsp. 1: In einem sehr einfachen Fall durch einen endlichen Automaten:
    0*1 (1 | 0(0|1))*



Sprachbeschreibungen und Maschinen

Formal ist ein deterministischer endlicher Automat (DEA) ein 5-Tupel

Def: Ein (deterministischer) endlicher Automat (**DFA**) ist ein 5-Tupel (Q,Σ,δ,q_0,F), wobei

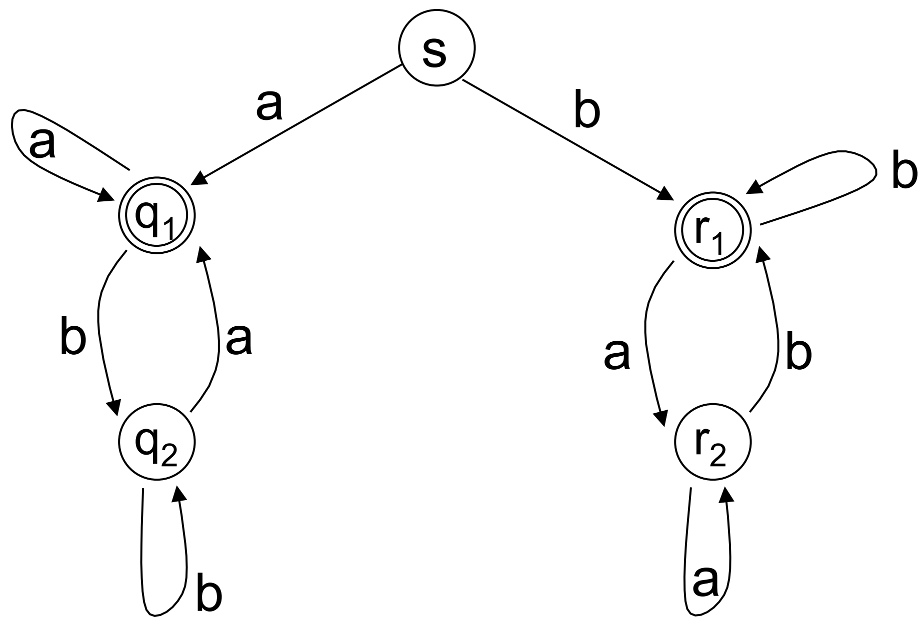
- Q eine endliche Menge von Zuständen ist,
- Σ ein endliches Alphabet
- $\delta: Q\times\Sigma\rightarrow Q$ die Übergangsfunktion,
- q_0 der Startzustand und
- $F\subseteq Q$ die Menge akzeptierender Endzustände.

Sprachbeschreibungen und Maschinen

δ	s	q_1	q_2	r_1	r_2
a	q_1	q_1	q_1	r_2	r_2
b	r_1	q_2	q_2	r_1	r_1

$A := (Q, \Sigma, \delta, q_0, F)$,
$Q := \{s, q_1, q_2, r_1, r_2\}$,
$\Sigma := \{a,b\}$,
$F := \{q_1, r_1\}$
$q_0 = s$

$L(A) = \{w_1 w_2 \dots w_n :$
$w_i \in \{a,b\}, w_1 = w_n\}$



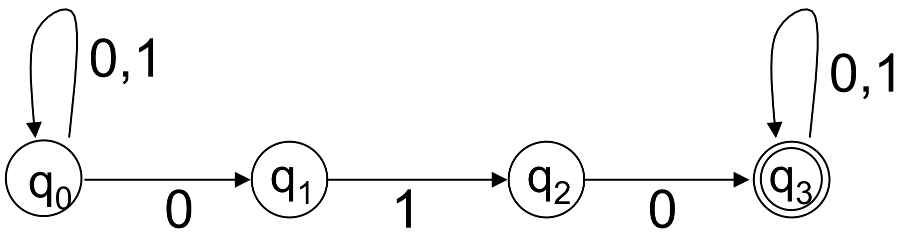
Sprachbeschreibungen und Maschinen

Formal ist ein nichtdeterministischer endlicher Automat (NEA) (ohne ε-Übergänge) ein 5-Tupel

Def: Ein (nichtdeterministischer) endlicher Automat (**NFA**) ist ein 5-Tupel (Q,Σ,δ,q_0,F), wobei

- Q eine endliche Menge von Zuständen ist,
- Σ ein endliches Alphabet
- δ: $Q\times\Sigma\rightarrow 2^Q$ die Übergangsfunktion,
- q_0 der Startzustand und
- $F\subseteq Q$ die Menge akzeptierender Endzustände.

Bsp.:



	0	1
q_0	$\{q_0,q_1\}$	$\{q_0\}$
q_1	-	$\{q_2\}$
q_2	$\{q_3\}$	-
q_3	$\{q_3\}$	$\{q_3\}$

Sprachbeschreibungen und Maschinen

Satz: Sei N ein NFA und L = L(N). Dann gibt es einen DFA A mit L(A) = L

Beweis: Sei $N = (Q,\Sigma,\delta,q_0,F)$. Die folgende Konstruktion heißt auch "Potenzmengenkonstruktion". Um $A = (Q',\Sigma,\delta',q'_0,F')$ zu definieren setzen wir:

- $Q' = 2^Q$ (Potenzmenge von Q)
- $q'_0 = \{q_0\}$
- $F' = \{R \in Q' \mid R \cap F \neq \{\}\}$
- $\delta'(R,a) = \bigcup_{r\in R} \delta(r,a) = \{q \in Q \mid \text{es gibt ein } r \in R \text{ mit } q \in \delta(r,a)\}$

Dann gilt:

$$
\begin{aligned}
w \in L(N) &\Leftrightarrow \delta(q_0,w) \cap F \neq \{\} \\
&\Leftrightarrow \delta(q'_0,w) \in F' \\
&\Leftrightarrow w \in L(A)
\end{aligned}
$$

Hierbei bedeutet $\delta(q,w)$, dass die Übergangsfunktion δ mehrfach auf das Wort w angwendet wird, Buchstabe für Buchstabe und startend bei Zustand q.