6 Ganzzahlige und kombinatorische Optimierung

Abschnitt 2: Lineare Optimierung
mit Entscheidungsvariablen $\mathbf{x}$ $\mathbf{R}^n$
im Standardfall $\mathbf{x}$ $\mathbf{R}_+^n$

Vom Problem her ggf. als Problemlösung
ganzzahlige Lösungen gefragt ("Stück")
mit Entscheidungsvariablen $\mathbf{x}$ $\mathbf{Z}^n$
in häufigem Standardfall $\mathbf{x}$ $\mathbf{Z}_+^n$
(nichtnegativ, ganzzahlig)

→ **ganzzahliges Optimierungsproblem**
(→ Alternativenmenge nicht kontinuierlich)

Zu dieser Klasse auch gerechnet
Probleme mit zweiwertigen
"binären"Entscheidungsvariablen
$\mathbf{x}$ $\{0,1\}^n$
modellierungstechnisch oft für "entweder/oder"

Praktisch (selbstverständlich) auch auftretend
"gemischte Probleme" kontinuierlich/ ganzzahlig/ binär
mit uU speziell angepaßten Verfahren

Nicht kontinuierliche + endliche Alternativenmenge
→ **kombinatorisches Optimierungsproblem**
(Menge Anordnungen endlich vieler Objekte,
+ Bewertung Anordnungen + Zielrichtung)

Abschnitt 3: Minimalgerüste,
kürzeste Wege,
Zuordnungsproblem sind komb. Opt.Probleme

auch: Transport-, Umladeproblem
bei ganzzahligen
Mindest-, Höchst-,
Nachfrage-, Angebots-Mengen

waren "**leicht**" zu lösen: mit **polynomialem** Aufwand

Probleme dieses Abschnitts zT
"**schwer**" zu lösen: NP-hart,
praktisch: mit **exponentiellem** Aufwand

Probleme umfassen ua
Zuordnungsprobleme (incl. Stundenplan),
Reihenfolgeprobleme (incl. Rundreise, Maschinenbelegung)
Gruppierungsprobleme (incl. Losgrößenplanung)
Auswahlprobleme (incl. Rucksackproblem)

Viele davon formulierbar / formalisierbar als

ganzzahlige / binäre lineare Modelle

und behandelbar mit allg. Verfahren der ganzz. Optimierung

aber (wie zuvor:) spezifische Problemklassen
haben spezifische Struktur,
erlauben uU spezifische Methoden

6.1 Ganzzahlige Optimierung

Standardform eines (rein) ganzzahligen
(linearen) Optimierungsproblems:

$\mathbf{G_o}$: min $Z(\mathbf{x}) = \mathbf{c}^T \mathbf{x}$
udN $\mathbf{A}\, \mathbf{x} \le \mathbf{b}$
$\mathbf{x}$ $\mathbf{Z}_+^n$

mit ganzzahligen **A**-Elementen
ganzzahligen **b**-Elementen
ganzzahligen **c**-Elementen (nicht wesentlich)

n: Zahl Entscheidungsvariablen
m: Zahl Nebenbedingungen **A** ist m x n

bzw. in erweiterter Fassung (wie gewohnt)
unter Einführung von Schlupfvariablen
(nicht mehr separat bezeichnet):

$\mathbf{G_e}$: min $Z(\mathbf{x}) = \mathbf{c}^T \mathbf{x}$
udN $\mathbf{A}\, \mathbf{x} = \mathbf{b}$
$\mathbf{x}$ $\mathbf{Z}_+^n$

n:=n+m Zahl Variablen (Entscheidungs- + Schlupf-)
m: Zahl Nebenbedingungen **A** ist m x n,
+ ia vorausgesetzt: rg **A** = m (< n)

Naheliegende Frage:
(da wir entsprechendes kontin. Problem lösen können:)
Hilft Lösung des (bekannten) lin. Optimierungsmodells?

Wenn einige (oder alle) Nebenbedingungen
eines Optimierungsproblems "gelockert" / "gestrichen",
spricht man von **relaxiertem Problem** / **Relaxation**

Ein G_e zugeordnetes relaxiertes Problem ist
das (uns wohlbekannte) Problem

$\mathbf{L_e}$: min $Z(\mathbf{x}) = \mathbf{c}^T \mathbf{x}$
udN $\mathbf{A}\, \mathbf{x} = \mathbf{b}$
$\mathbf{x}$ $\mathbf{R}_+^n$

Hat L_e ganzzahlige (zulässige) Basislösungen,
dann ist optimale Lösung L_e auch optimale Lösung G_e
(welche zB mit dem Simplex-Verfahren bestimmbar)

Wann ist dies der Fall?
dazu:

- quadratische ganzzahlige Matrix **B**
heißt **unimodular** wenn
| det **B** | = 1

- (nicht notwendig quadratische) ganzzahlige Matrix **A**
heißt **total unimodular** wenn
jede quadratische nichtsing. Teilmatrix von **A** unimodular
(insbesondere jedes Element a_{ij} {0,1,-1})

- (vgl. Abschn. 2.8.1:)
$\mathbf{x}^T = (\mathbf{x}_B^T, \mathbf{x}_N^T)$ Basislösung L_e
$\mathbf{x}_B, \mathbf{x}_N$ Vektoren Basis-, Nichtbasis-Variable
$\mathbf{B}$ zugehörige Basismatrix

$\mathbf{B}\, \mathbf{x}_B = \mathbf{b}$
$\mathbf{x}_B = \mathbf{B}^{-1}\, \mathbf{b}$

unter Nutzung der Cramer'schen Regel:

$$\mathbf{x}_B = \frac{\mathbf{B}^{adj}}{\det \mathbf{B}}\, \mathbf{b}$$

wo $\mathbf{B}^{adj}$ die sog. Adjungierte von $\mathbf{B}$

$$\mathbf{B}^{adj} = \begin{pmatrix} \beta_{11} & \dots & \beta_{m1} \\ \dots & & \dots \\ \beta_{1m} & \dots & \beta_{mm} \end{pmatrix}$$

mit Elementen

$\beta_{ij} = (-1)^{i+j}\, D_{ij}$

und D_{ij} Determinante der (m-1) x (m-1) Matrix (Minor), die aus **B** durch Streichen Zeile i + Spalte j entsteht

ist **A** total unimodular, dann

| det **B** | = 1

$\mathbf{B}^{adj}$ ganzzahlig

\+ **b** ganzzahlig

$\mathbf{x}_B$ ganzzahlig, $\mathbf{x}_N$=0, **x** ganzzahlig

Satz 6.1.01 : Ganzzahlige Lösung Relaxation

Ist in dem relaxierten Optimierungsproblems L_e
- die Matrix **A** total unimodular
- der Vektor **b** ganzzahlig

dann sind alle Basislösungen von L_e ganzzahlig

(und optimale Basislösung damit
Lösung des ganzzahligen "Originals" G_e)

Prüfung einer beliebigen ganzzahligen Matrix **A** auf totale Unimodularität

- zwar mühsam (**nicht** empfohlen)

- aber für interessante Spezialfälle gegeben:
  Inzidenzmatrix Digraph ist total unimodular
  Probleme Abschnitt 3 (Wege, Flüsse, Umladepr., …)
  nur ganzzahlige Basislösungen,
  falls "sonstige" Parameter ganzzahlig

Fortsetzung Frage:
(da wir zugehöriges
relaxiertes - kontinuierliches - Problem lösen können:)
Hilft Lösung des (bekannten) lin. Optimierungsmodells?

Könnte "kontinuierliche" Lösung L_e bestimmt werden,
\+ "zulässig" gerundet werden (ganzzahlig, in zul. Bereich),
→ als Lösung G_e dienen?

Antwort:

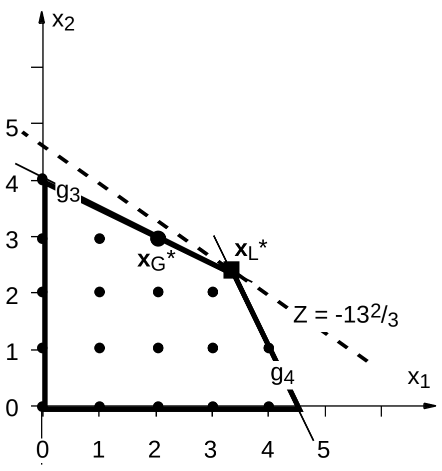
- (wie folgendes Beispiel zeigen wird:)
  ia nein

- bei großen Elementen des Lösungsvektoirs
  uU ja (bei "heuristischen" Verfahren tauchen
  hinreichend viele
  "approximative" Lösungen auf)

Beispiel 6.1.02: Quantifiziertes ganzzahliges Modell

min	$Z(x_1,x_1) = -2x_1 -3x_2$	
udN	$x_1 +2x_2 \le 8$	(g_3)
	$2x_1 +x_2 \le 9$	(g_4)
	$x_1,x_1 \quad \mathbf{Z}_+$	

graphische Veranschaulichung:



- zulässig für G_o sind Gitterpunkte (des zul. Bereichs L_o)
- optimale Lösung für G_o ist $\mathbf{x}_G^*$, für L_o $\mathbf{x}_L^*$
- "Runden" von $\mathbf{x}_L^*$ erzeugt (offensichtlich)
  nicht die korrekte Lösung $\mathbf{x}_G^*$
- Entfernung $|\mathbf{x}_L^* - \mathbf{x}_G^*|$ kann ia durchaus groß sein !

Grundidee der sog. **Schnittebenenverfahren** ist

- Fortsetzung der Nutzung der Simplex-Verfahren

- (schrittweise) Beseitigung von
  unzulässigen (nichtganzzahligen) Lösungen
  durch Einführung zusätzlicher Restriktionen
  (→ Nebenbedingungen, welche
  Teile zulässiger Menge "wegschneiden")
  derart daß
  - gefundene optimale (nichtganzzahlige) Lösung
    nicht (mehr) zulässig
  - alle ganzzahligen Lösugen (weiterhin) zulässig

Schnittebenenverfahren auf Gomory zurückgehend

- allgemein einsetzbar
  für Modelle der ganzzahligen Optimierung

- in (deutlich unterschiedlichen) Varianten existierend

- hier in "Grundform" vorgestellt (später mehr dazu)
  eingebettet in Behandlung Bsp. 6.1.02

Im Folgenden Notation Tableaus leicht geändert

- (-)**b**-Spalte "ganz links" (0-te Spalte der Anordnung)

- Z-Zeile "ganz oben" (0-te Zeile der Anordnung)

Initialtableau Bsp. 6.1.02

(-)**b**	$\mathbf{x_1}$	$\mathbf{x_2}$	
0	-2	-3	**Z**
-8	1	2	$\mathbf{-x_3}$
-9	2	1	$\mathbf{-x_4}$

und Lösungstableau (nach Simplex-Anwendung)

(-)**b**	$\mathbf{x_4}$	$\mathbf{x_3}$	
-41/3	1/3	4/3	**Z**
-7/3	-1/3	2/3	$\mathbf{-x_2}$
-10/3	2/3	-1/3	$\mathbf{-x_1}$

optimale Basislösung L

$(\mathbf{x}_L{}^*)^T = (x_1,x_2,x_3,x_4) = (10/3,7/3,0,0)$

$Z = -41/3$

nicht G-zulässig (Ganzzahligkeit verletzt)

Zwischenüberlegung

bezeichne (analog Abschn. 2.8.1)

B := { i ; x_i Basisvariable}
die Indexmenge der (momentanen) Basisvariablen

N := { i ; x_i Nichtbasisvariable}
die Indexmenge der (moment.) Nichtbasisvariablen

jede Gleichung (Zeile) des Tableaus lautet explizit
(Indizierung **nicht** nach Spalten/Zeilen-Indizes Tableau)

$$-b_k + \sum_{l \quad N} a_{kl}x_l = -x_k \qquad k \quad B$$

$$x_k + \sum_{l \quad N} a_{kl}x_l = b_k \qquad k \quad B$$

mit der (momentanen) Basislösung (NBVs = 0)

$\mathbf{x}_B = \mathbf{b}_B$

reelle Zahl a **R** besitzt Darstellung ("Abrundung")

$a = \lfloor a \rfloor + r \qquad \lfloor a \rfloor \ \mathbf{Z},\ r \ \mathbf{R},\ 0 \le r < 1$

so auch

(6.1.03) $b_k = \lfloor b_k \rfloor + r_k$

$a_{kl} = \lfloor a_{kl} \rfloor + r_{kl}$

und Tableaugleichung insgesamt

$$x_k + \sum_{l \quad N} \lfloor a_{kl} \rfloor x_l - \lfloor b_k \rfloor = -\sum_{l \quad N} r_{kl}x_l + r_k$$

mit (für alle ganzzahligen Lösungen)
ganzzahliger linker Seite
ganzzahliger rechter Seite

zusätzlich rechte Seite $\le r_k$
(wegen x.≥0 bei Zulässigkeit)

rechte Seite ganzzahlig
\+ $\le r_k$
\+ $0 < r_k < 1$ (bei NGZ-Lösung x_k)

$$-\sum_{l \quad N} r_{kl}x_l + r_k \le 0$$

ausdrückbar, mithilfe neuer "Gomory"- (Schlupf-) Variabler, durch Restriktion

(6.1.04) $r_k - \sum_{l \quad N} r_{kl}x_l = -x_{n+1}$

$x_{n+1} \ge 0$

momentane Basislösung ($\mathbf{x}_N = \mathbf{0}$, $r_k>0$) verletzt Restriktion
Zufügen der Nebenbedingung "schneidet Lösung weg"

dabei (s. Weg) alle ganzzahligen Lösungen erhalten

Fortsetzung Beispiel:

Basislösung L ist NGZ
in x_2-Zeile mit b_2=7/3 r_2=1/3
in x_1-Zeile mit b_1=10/3 r_1=1/3

aus einer dieser Zeilen ist neue Restriktion zu formen:
- Zeile mit maximalem r.
- falls nicht eindeutig, Zeile kleinsten Matrix-Index'

x_2-Zeile (Matrix-Index 1)
Tableau mit zusätzlicher Restriktion:

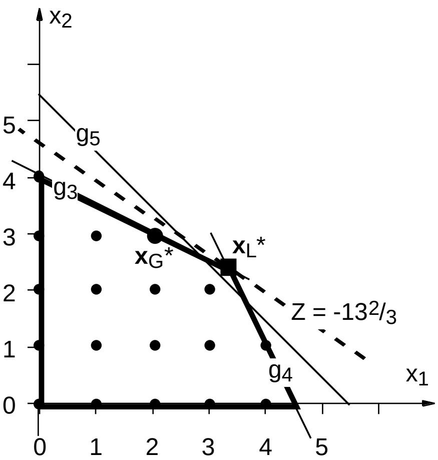
(-)**b**	$\mathbf{x_4}$	$\mathbf{x_3}$	
-41/3	1/3	4/3	**Z**
-7/3	-1/3	2/3	$\mathbf{-x_2}$
-10/3	2/3	-1/3	$\mathbf{-x_1}$
1/3	-2/3	-2/3	$\mathbf{-x_5}$

Basispunkt nicht zulässig: x_5=-b_5<0

Veranschaulichung: →

Veranschaulichung:

neue Schnittebene	$1/3 = 2/3x_4 + 2/3x_3$	
mit (s. Initialtableau)	$x_3 = -x_1 - 2x_2 + 8$	
	$x_4 = -2x_1 - x_2 + 9$	
dh in x_1/x_2-Koordinaten	$11 = 2x_1 + 2x_2$	(g_5)



Zwischenüberlegung zu "Basispunkt nicht zulässig"

Nach Gomory-Erweiterung aus (zB:) Zeile k
Unzulässigkeit Basispunkt immer gegeben,
da $x_{n+1} = -b_{n+1} = -r_k < 0$

Andererseits im dualen Modell (vgl Abschn. 2.6)
Zulässigkeit Basispunkt immer gegeben,
da $c_j > 0$, j N wegen Optimalität der L-Lösung

Demnach: Aufgabe der Optimierung
des Gomory-erweiterten Modells
(mittels Simplex-Schritten)
besser in dualem Bereich vornehmen,
(+ duale Zulässigkeit erhalten)

Erinnerung:

zu primalem Modell (in Standard-/Minimierungs-Form)

min $Z(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$
udN $\mathbf{A}\,\mathbf{x} \le \mathbf{b}$
$\mathbf{x} \ge \mathbf{0}$

ist duales Modell (in Standard-/Minimierungs-Form)

min $-Z'(\mathbf{y}) = \mathbf{b}^T\mathbf{y}$
udN $(-\mathbf{A})^T\,\mathbf{y} \le \mathbf{c}$
$\mathbf{y} \ge \mathbf{0}$

Simplexschritt im dualen Bereich
normalerweise mit "dualer Simplexmethode":
auf gegebenem (primalem) Tableau,
unter entsprechender Algorithm.-Anpassung

Fortsetzung Beispiel:

Statt dualer Simplexmethode
(wg "Gewohntheit":) - explizite Übertragung ins Duale
- normaler Simplexschritt im Dualen

aus primal:

(-)**b**	x_4	x_3	
-41/3	1/3	4/3	**Z**
-7/3	-1/3	2/3	$-x_2$
-10/3	2/3	-1/3	$-x_1$
1/3	-2/3	-2/3	$-x_5$

——> wird dual:

	(-)**c**	y_2	y_1	y_5	
(0)	41/3	7/3	10/3	-1/3	**-Z'**
(1)	-1/3	1/3	-2/3	2/3	$-y_4$
(2)	-4/3	-2/3	1/3	2/3	$-y_3$

auf dualem Modell:
Simplex-Schritt:
- (duale) Pivotspalte: neue Bedingung y_5
- (duale) Pivotzeile: Erhalt'g Zulässigk't y_4

		(-)**c**	y_2	y_1	y_4	
(0'):	(0)+1/2(1)	27/2	5/2	3	1/2	**-Z'**
(1'):	3/2(1)	-1/2	1/2	-1	3/2	$-y_5$
(2'):	(2)-(1)	-1	-1	1	-1	$-y_3$

primal <—— dual

(-)**b**	x_5	x_3		
-27/2	1/2	1	**Z**	NGZ
-5/2	-1/2	1	$-x_2$	r_2=1/2
-3	1	-1	$-x_1$	
-1/2	-3/2	1	$-x_4$	r_4=1/2

neue (Gomory-) Zeile
aus x_2-Zeile

(-)**b**	x_5	x_3		
-27/2	1/2	1	**Z**	NGZ
-5/2	-1/2	1	$-x_2$	r_2=1/2
-3	1	-1	$-x_1$	
-1/2	-3/2	1	$-x_4$	r_4=1/2

↓ neue Restriktion aus x_2-Zeile

(-)**b**	x_5	x_3	
-27/2	1/2	1	**Z**
-5/2	-1/2	1	$-x_2$
-3	1	-1	$-x_1$
-1/2	-3/2	1	$-x_4$
1/2	-1/2	0	$-x_6$

——> dual

	(-)**c**	y_2	y_1	y_4	y_6	
(0)	27/2	5/2	3	1/2	-1/2	**-Z'**
(1)	-1/2	1/2	-1	3/2	1/2	$-y_5$
(2)	-1	-1	1	-1	0	$-y_3$

Simplex-Schritt $\mathbf{y_6}/\mathbf{y_5}$:

		(-)**c**	y_2	y_1	y_4	y_5	
(0'):	(0)+(1)	13	3	2	2	1	**-Z'**
(1'):	2(1)	-1	1	-2	3	2	$-y_6$
(2'):	(2)	-1	-1	1	-1	0	$-y_3$

primal <——

(-)**b**	x_6	x_3	
-13	1	1	**Z**
-3	-1	1	$-x_2$
-2	2	-1	$-x_1$
-2	-3	1	$-x_4$
-1	-2	0	$-x_5$

Lösung ganzzahlig zulässig
optimal
mit Basispunkt
$(\mathbf{x}_G^*)^T = (x_1,x_2,x_3,x_4,x_5,x_6) = (2,3,0,2,1,0)$
Z = -13

Zusammenfassung Schnittebenenverfahren
(in vorgestellter Form)

- Lösung relaxierten Problems mittels Simplex-Verfahren
  - Unbeschränktheit, Leerheit zulässigen Bereichs "unterwegs erkannt"
  - weiter mit optimaler Lösung der Relaxation
- Folge von Schritten, basierend auf Vorgängertableau
  - Prüfung der Lösung auf Ganzzahligkeit
    falls gegeben: opt. L-Lösung ist optimale G-Lösung
  - Aufspaltung der NGZ-b_i gemäß (6.1.03)
    k: r_k=max r_i ist Ausgangszeile für zus. Restriktion
    falls nicht eindeutig: kleinster Matrix-Index (daraus)
  - Aufspaltung Matrix-Elemente k-Zeile gemäß (6.1.03),
    Zufügung Restriktion gemäß (6.1.04)
  - dualer Simplexschritt

    (duale) Pivotspalte: k

    (duale) Pivotzeile: gemäß Erhaltung (dualer) Zulässigkeit (vgl. Abschn. 2)

Gomory selbst führte 3 **Schnittebenenverfahren** ein

(1) etwa wie vorgestelltes Schema,
Nachteil insbesondere in Form von Rundungsfehlern

(2) Rechnung völlig im Bereich der ganzen Zahlen
(erreicht durch Maßnahme, bei der Pivot-Elemente =+1/-1)
Anfangs- (L-) Phase vermieden
(erreicht durch zusätzliche Anfangsrestriktion)

(3) gemischt-ganzzahliges Verfahren

\+ weitere Varianten seither entwickelt

Beweise der Endlichkeit der Verfahren existieren,
beruhend auf detaillierteren Regeln
- der Auswahl der Gomory-Basiszeilen
- der (dualen) Pivot-Zeilen/-Spalten

(obige "lose" Regeln garantieren Endlichkeit nicht)

Aufwand:

- worst-case exponentiell (s. Simplex)
  faktisch fallabhängig "gut" / "schlecht"
- prinzipielle Schwäche reiner Schnittebenenverfahren:
  weggeschnittener Bereich relativ klein
  Tendenz zu vielen Iterationen
- (trotz breiter Bekanntheit:)
  reine Schnittebenenverfahren für ernsthafte
  (große) Probleme eher nicht mehr angewandt
- "branch-and-bound"-Techniken (s. später)
  als überlegen eingestuft,
  insbesondere als "branch-and-cut":
  in Kombination mit Schnittebenenverfahren

6.2 Methoden der kombinatorischen Optimierung

Allgemeine Prinzipien + Methoden
zur Bewältigung/Lösung schwerer Probleme

- in Anwendbarkeit nicht auf komb. Optimierung beschränkt
- dennoch gern in diesem Kontext diskutiert:
  kombinatorische Optimierung "typischerweise" schwer

Offensichtliche Methode (endlicher Zulässigkeitsraum)
ist **vollständige Enumeration** (exakte Methode):

- systematische Untersuchung aller zulässigen Punkte
  (systematisch: zB mit Entscheidungsbaum, s.später)
- bester Punkt liefert Lösung

  exponentieller Aufwand (worst case **und** praktisch)

Illustration exponentiell (und damit "schwer"):

Lösungsraum (binäre Optimierung)	$M = \{0,1\}^n$
Problemgröße	$n = 50$
Größenordnung Aufwand	$2^{50} \approx 10^{15}$
bei zeitl. Aufwand /Punkt	1 msec
Größenordnung Zeitaufwand	2^{50} msec ≈ 3200 Jahre

nur für kleine Probleme praktikabel

Allgemeine Ideen zur Lösung schwerer Probleme

- **exakte Methoden**
  worst case nach wie vor exponentiell
  praktisch uU polynomial /
  je nach Größe zumindest.tolerierbar

  * Begrenzung Lösungsraum
    uU schrittweise (Bsp Schnittebenenverf.)
  * "teile und herrsche" (i.allg. Sinn)
    Zerlegung Problem in kleinere (i.allg. repetitiv)
    derart daß
    - aus Lösung Teilproblemen
      Lösung Gesamtproblem konstruierbar
      mit insgesamt geringerem Aufwand
    - Lösung Gesamtproblem "in"
      Lösung eines Teilproblems zu suchen
      mit geringem Aufwand für gewisse Teilprobleme
      (Bsp. B&B, Branch-and-Bound)
    - (in gewissem Sinne "dynamische Optimierung", s.später)
  * (clevere) Kombinationen aus obigen Ideen
    (Bsp. Branch-and-Cut)

- **heuristische Methoden**
  erstrebenswerterweise polynomial

  aber "approximativ" in folgendem Sinne
  - ohne Garantie für optimale Lösung
  - aber ia ("praktisch häufig") gute Lösung
    erstrebenswerterweise mit Schranken
    erstrebenswerterweise "enge" Schranken

  heuristische Methoden meistens problemspezifisch,
  häufig mit Charakteristika
  - Eröffnung — Bestimmung initialer Lösung (oft: zulässiger Lösung)
  - Verbesserung — iterative Suche (oft: lokal, in Nachbarschaft)
  - Abbruch — vorzeitige Beendigung (oft: Anspruch-/ Grenz-abhängig)
  - (clevere) Kombinationen aus diesen

ZU B&B-METHODEN (Erinner'g, vgl. Datenstrukturen)

Sei ein Maximierungsproblem
P_0 zu lösen (Minimierungsproblem analog)

P_0: $\max Z(\mathbf{x} ; \mathbf{x}\ M_0)$ zulässige Lösungsmenge M_0

B&B beruht auf den Lösungsprinzipien

- Branching
- Bounding

Branching:

- zerteile Problem P_0 in Teilprobleme (TPs) $P_1,\ldots,P_k$ mit Lösungsmengen $M_1,\ldots,M_k$
  derart daß

  $M_0 = {}_{i=1}^{k} M_i$

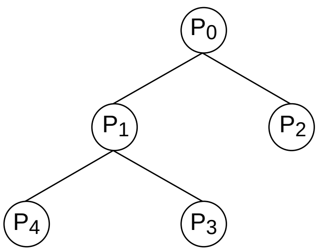
  $M_i \cap M_j =$ $\quad i \neq j$

  dh: Problemzerteilung: "Zerlegung" Lösungsmenge
  wo: Disjunktheit TP-Lösungsmengen nur "möglichst", (zulässige) TP-Lösungsmenge uU leer

- wenn erforderlich:
  zerteile Probleme P_i weiter in Teilprobleme $P_{k+1},\ldots$

- insgesamt resultiert Entscheidungs-(Wurzel-)Baum, in dem Teilprobleme als Knoten notiert, und Teilprobleme eines Problems
  vom zugeordneten Knoten per "branching" erreicht

- Lösung aller Blattprobleme liefert (sicher) Gesamtlösung



Beispiel: binäre Zerlegung (häufig)

Bounding:

- für optimalen Lösungswert Z* untere Schranke U bekannt

  $U \leq Z^*$

  - sowohl initial:
    schlimmstenfalls $-\infty$
    oder (besser) aus Heuristik

  - als auch während Ablauf B&B:
    fortlaufend bestmögliche Vergrößerung vorzunehmen

- für Teilprobleme P_i obere Z-Schranke O_i zu ermitteln

  $O_i \geq \max Z(\mathbf{x} ; \mathbf{x} \quad M_i)$

  im Gleichheitsfall ist Optimum O_i^* auf M_i gefunden

  $O_i^* := \max Z(\mathbf{x} ; \mathbf{x} \quad M_i)$

  Schranken-/Wert-Ermittlung aus

  - entweder (direkt:) Heuristik

  - oder (Verzweigung:) Betrachtung P_i-Teilprobleme
    wo aus $M_i = \quad M_k$
    folgt $O_i \leq \max O_k$
    $O_i^* = \max O_k^*$

- Teilprobleme gelten als
  - weiter zu untersuchen, zu verzweigen
  - **ausgelotet** (fathomed), abgearbeitet

- Teilproblem heißt **ausgelotet** (fathomed) wenn

  - $O_i < U$, optimale Lösung kann nicht in M_i liegen
    ($O_i^* < U$) P_i nicht weiter untersuchen / verzweigen
    Abbruch "Zweig" durch "bounding"

  - $O_i^* > U$ beste M_i-Lösung gefunden, besser als U
    $U := O_i^*$ als Vergrößerung U

  - $M_i =$ P_i hat keine zulässige Lösung

- B&B-Varianten arbeiten zusätzlich mit
  - oberer Schranke O
  - unteren Schranken U_i

B&B-Verfahren sind bzgl folgender Komponenten zu konkretisieren (geschieht weitgehend problemspezifisch)

- Regel Initialisierung: initale Schranken U, O (Heuristiken, zB Relaxation)

- Regeln TP-Schranken: Schranken O_i, U_i (Heuristiken, zB Relaxation)

- Regeln zur Reihenfolge der Auswahl zu verzweigender TP und zur Auslotung
  (DepthFirstSearch,
  BreadthFirstSearch,
  MaximumUpperBound,…)

- Regeln zur Problemaufteilung, Verzweigung
  (zwei dh binär, mehrere, wie …)

ZU HEURISTISCHEN VERFAHREN

Eröffnungsverfahren (wenn vorgesehen)
bestimmen initiale (zulässige) Lösung
Verfahren greedy
bzw vorausschauend

Verbesserungsverfahren

- starten mit zulässiger Lösung **x**

- iterieren über
  - Nachbarschaftsbestimmung
    NB(**x**) := {$\mathbf{x}_i$; $\mathbf{x}_i$ gemäß Regel}
    wo (Transformations-)Regeln
    Umgebung "kleiner" Veränderungen **x** definieren
  - Untersuchung in Nachbarschaft NB(**x**)
    festzulegen: welche $\mathbf{x}_i$, in welcher Reihenfolge, wielange, welcher nächste Aufsetzpunkt

- brechen ab bei (festzulegendem) Kriterium

Eröffnungs- und Verbesserungsverfahren können deterministisch oder stochastisch ausgestaltet sein

Typische Schwäche reiner Verbesserungsverfahren liegt in Lokalität der Suche (bestimmt durch Def NB)
Beschränkung auf lokale Optima

Abhilfe durch "Metastrategien",
Stichworte: Simulated Annealing
/ Threshold Accepting / Tabu Search
Genetische / Evolutionäre Algorithmen
(nicht weiter diskutiert)

6.3 Ausgewählte kombinatorische Probleme

Beträchtliche Menge spezifischer Problemkreise
"in diesem Kontext" behandelt, ua:

- Knapsack-Probleme (*)
- Travelling Salesman (Tourenplanung) - Probleme
- Verschnittprobleme
- Scheduling- (Maschinenbelegungs-) Probleme (*)
- Ressorcen- / Projekt- Planungsprobleme
- …

Unsere (bescheidene) Auswahl liegt bei (*)

6.3.1 Das Knapsack-Problem
(Rucksackproblem)

(sicher bekannte:) anschauliche
Aufgabenstellung:

In Rucksack verschiedene Gegenstände einpackbar

- Gegenstände haben Gewicht,
  Maximalgewicht nicht zu überschreiten (Restriktion)
- Gegenstände haben Nutzen,
  Gesamtnutzen zu maximieren (Zielfunktion)

Problemlösung(smethoden)

- von hoher theoretischer Bedeutung
  (Testfeld für Methodik, Methoden, Techniken,
  Aufwandsabschätzung, …; breit untersucht)
- von gewisser praktischer Bedeutung
  (Frachtladung, Produktionsplanung, Maschinenbelegung)

Problemformalisierung
(als binäres, endliches kombinatorisches Optim.Problem)

- n Gegenstände
  mit Gewichten $a_j > 0$ $j=1,\dots,n$
  und Werten $c_j > 0$ $j=1,\dots,n$
- Maximalgewicht $A > 0$
  + sinnvollerweise: $\max a_j \le A$ (sonst: kleineres Problem)
  $\text{sum } a_j > A$ (sonst: Lösung gefunden)
- (gepackte) Werte-Summe zu maximieren

Lösung (und Lösungsversuche) charakterisiert durch
Indikatorvariable x_j {0,1} $j=1,\dots,n$
zeigen an, ob Gegenstand im Rucksack $x_j = 1$
("entweder/oder") oder nicht $x_j = 0$

Problem-Parameter und Lösung (wie üblich) vektoriell
$\mathbf{c} = (c_1,\dots,c_n)^T$ **a**, **x** analog

Problem formal

(R) max $Z = \mathbf{c}^T \mathbf{x}$
udN $\mathbf{a}^T \mathbf{x} \le A$
$\mathbf{x}$ $\{0,1\}^n$

mit zulässiger Lösung $\mathbf{x} = \mathbf{0}$
Problem ist NP-hart, Entscheidungsvariante NP-vollständig

Für Folgendes
sinnvolle Ordnung Gegenstände nach "spezifischem Wert":
$c_1/a_1 \ge \dots \ge \dots c_n/a_n$ (in WE / GE)

HEURISTISCHE LÖSUNGEN
(hier, wie typischerweise, problemspezifisch)

Eröffnung:

Relaxation von R gemäß
$\mathbf{x}$ $\{0,1\}^n$ $\mathbf{0} \le \mathbf{x} \le \mathbf{1}$ (komponentenweise)
ist lineares Optimierungsproblem ("**LR**")

Lösungsweg bekannt

Sogar:
Optimale Lösung LR unmittelbar verfügbar:

"jede Volumeneinheit Rucksack mit maximal möglichem
spezifischen Wert"
(nach gew. Ordnung) erste k-1 Gegenstände in Rucksack,
k-ten Gegenstand teilweise, restliche nicht

Optimale Lösung LR formal
(Zielfunktionswert Z_{LR}, Gewicht A_{LR}) :

$$k: \quad \sum_{j=1}^{k-1} a_j \le A, \qquad \sum_{j=1}^{k} a_j > A$$

$$x_j^{LR} = 1 \qquad j = 1,\dots,k-1$$

$$x_k^{LR} = \frac{A - \sum_{j=1}^{k-1} a_j}{a_k}$$

$$x_j^{LR} = 0 \qquad j = k+1,\dots,n$$

$$Z_{LR} = \sum_{j=1}^{k-1} c_j + \left(A - \sum_{j=1}^{k-1} a_j\right)\frac{c_k}{a_k} \qquad A_{LR} = A$$

daraus Näherungslösung R
(Zielfunktionswert Z_R, Gewicht A_R) :

$$x_j^R = x_j^{LR} \qquad j \ne k$$

$$x_k^R = 0$$

$$Z_R = \sum_{j=1}^{k-1} c_j \qquad A_R = \sum_{j=1}^{k-1} a_j$$

ist "suboptimal",
Z_{LR} ist obere Schranke für optimales Z_R^*

Verbesserung:

Freigebliebener Rest A' := A - A_R
kann mit Kandidaten aus Gegenstandsmenge {j;j=k+1,…,n}
gefüllt werden

Greedy-Heuristik für "in Richtung der Anordnung":

```
A':=A-AR;
ZH:=ZR;

for j=k+1 step 1 until n do
if a[j]≤A'then
  begin x[j]:=1; ZH:=ZH+c[j]; A':=A'-a[j]; end
else x[j]:=0;

AH:=A-A';

{ZH ist untere Schranke für ZR*, AH zugeh. Gewicht}
```

woraus insgesamt als Schranken folgt
$Z_H \le Z_R{}^* \le Z_{LR}$
(es gibt schärfere; hier nicht verfolgt)

Nachüberlegung:
Greedy (Verbesserungs-) Heuristik,
von A' := A, Z_H:= 0 startend,
über j = 1,…,n laufend
liefert sowohl Z_H / A_H als auch Z_{LR} / A_{LR} (=A)

Aufwand (Eröffnung + Verbesserung)
- von initialer Sortierung dominiert
- dh (bekannterweise:) O(n log n)

BRANCH-AND-BOUND LÖSUNGEN
(auch dies, wie typischerweise, problemspezifisch)

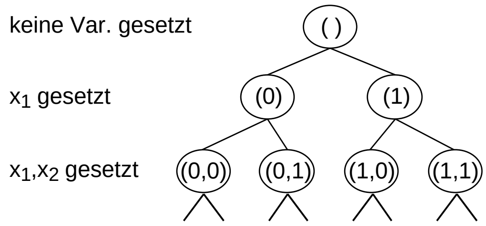
Gegenstände geordnet: 1,…,n
sinnvoll (wie zuvor): nach spezifischem Wert
$c_1/a_1 \ge \ldots \ge \ldots c_n/a_n$
mit Indikatorvariable x_i {0,1}

- Problemaufteilung

  - Problem / Teilproblem (Knoten Entscheidungsbaum)
    charakterisiert durch (Teil-)Vektor Indikatorvariable
    $(x_1,\ldots,x_s)$ 1≤s≤n
    oder leerer Vektor
    mit Bedeutung $x_1,\ldots,x_s$ gesetzt: =0: nicht im Rucks.
    =1: im Rucksack
    $x_{s+1},\ldots,x_n$ frei: Entscheid'g offen
    oder leere Liste

  - Zerteilung Problem / Teilproblem (falls beabsichtigt)
    durch Setzung der "nächsten" Indikatorvariablen
    x_{s+1} =0 Teilproblem $(x_1,\ldots,x_s,0)$
    =1 Teilproblem $(x_1,\ldots,x_s,1)$

  - es resultiert (demnach) binärer Entscheidungsbaum



- Initialisierung

  für Gesamtproblem (keine Variable gesetzt)
  bekannt aus (durchzuführender) greedy-Heuristik "GH"
  $Z_H \le Z^*$ Z* optimaler Lösungswert in $\mathbf{x}_H$
  U := Z_H untere Schranke Gesamtproblem
  $(Z^+,\mathbf{x}^+):=(Z_H,\mathbf{x}_H)$
  temporäres Optimum U
  ab hier in Z^+ "verwahrt"
  (+ zugehöriger Punkt $\mathbf{x}^+$)
  (Gesamt-)
  Problem ( ) zerteilen

- Teilproblem-Schranken

- für Teilproblem
  $\mathbf{t} := (x_1,\ldots,x_s)^T$
  sind "zugewiesenes Füllgewicht" A_t
  \+ "erreichter Packwert" Z_t bekannt zu

  $$A_t := \sum_{i=1}^{s} x_i a_i \qquad Z_t := \sum_{i=1}^{s} x_i c_i$$

- für **t** lassen sich (wenn bei Untersuchung **t** erforderlich)
  obere Schranken O_t für Z ermitteln

  mittels GH
  von A'=A-A_t, Z_H:= Z_t startend,
  über j = s+1,…,n laufend
  mit Z_{LR} als Resultat

  $O_t := Z_{LR}$

- Teilproblem-Untersuchung

- für Teilproblem
  $\mathbf{t} := (x_1,\ldots,x_s)^T$
  ist Füllgewicht A_t und Packwert Z_t bekannt

- Untersuchung **t** führt zu Fällen:

  (1) $A_t > A$ Lösungsmenge leer (nur für x_s=1, vgl. (3))
  TP ausgelotet

  (2) $A_t = A$ volle Zuweisung erreicht (nur für x_s=1)
  TP ausgelotet
  (2a) $Z_t > Z^+$ neues temporäres Optimum
  $(Z^+,\mathbf{x}^+):=(Z_t,\mathbf{t})$

  (3) $A_t < A$ Optimum potentiell in **t**

  (3a) s=n Aufteilung nicht mehr möglich
  (3a1) $Z_t > Z^+$ neues temporäres Optimum
  $(Z^+,\mathbf{x}^+):=(Z_t,\mathbf{t})$, ausgelotet

  (3b) s<n potentiell Aufteilung,
  Ermittlung oberer Schranke O_t
  (3b1) $O_t \le Z^+$ Optimum nicht in **t**
  TP ausgelotet
  (3b2) $O_t > Z^+$ Optimum potentiell in **t**
  TP zerteilen in
  $(x_1,\ldots,x_s,0)^T$ und $(x_1,\ldots,x_s,1)^T$

- uU backtracking-Maßnahmen (Speichereffizienz)

- Reihenfolge zu untersuchender TPs
  (durch Zerteilung entstanden)

  unterschiedliche Strategien sinnvoll / empfohlen

  - nach MUB wg resultierender Chance,
    Z^+ schnellstmöglich zu erhöhen
    mehr Zweige abzuschneiden

  - DFS in Abwandlung LIFO wg resultierender Chance,
    zerteiltes Problem bzgl. Schrankenermittlung
    "in nur leichter Abwandl'g" des Vaters zu untersuchen
    Schrankenermittlung schneller

- Terminierung
  wenn kein TP mehr zu untersuchen
  mit optimaler Lösung

  $(Z^*,\mathbf{x}^*) := (Z^+,\mathbf{x}^+)$

Knapsack - B&B
in vielen Varianten / mit vielen Verfeinerungen
ausgiebig untersucht (vgl. Literatur)

6.3.2 Scheduling-Probleme
(Maschinenbelegungs-Probleme)

Terminologie **Scheduling** / Maschinenbelegungsplanung:
Jobs (Aufgaben, Aufträge)
sind von
Maschinen (Geräten, Personen, Einrichtungen)
zu bearbeiten

Breit interpretierbar
- Fertigungssvorgänge in Werkstätten (Fabriken)
- Ab-/Anflugvorgänge auf Rollbahnen
  (und ähnliche Abfertigungen)
- Berechnungen / Bearbeitungen auf Prozessoren
  (Rechen- und Kommunikationssysteme)
- Kundenwünsche in Service-Einrichtungen
  (Warenhaus, Post, Bank, …)

Allgemeine Fragestellung:
- wann soll
- welcher Job (bzw einer seiner Arbeitsvorgänge: **Tasks**)
- auf welcher Maschine / welchen Maschinen

bearbeitet werden, wenn
- bestimmte Zielfunktion gegeben
- und zu optimieren ist

Beispiele für Zielfunktionen sind
- Zeitspanne bis Fertigstellung letzter Job
- mittlere Wartezeit (Zeit ohne Bearbeitung) Job
- Überschreitung Fertigstellungstermin
- Gesamtkosten Bearbeitung
- …

Viele spezielle Klassen von Problemen
bzgl "Art" von Jobs / Maschinen, so

- alle Jobs initial bekannt
  / Jobs im Lauf der Zeit auftauchend
- Jobs nicht unterbrechbar / unterbrechbar
  mit Verlust / ohne Verlust
- Jobs gleichwertig / wichtig vs unwichtig

Problemlösungen polynomial / exponentiell
mit Charakteristikum (in diesem Bereich),
daß harmlos erscheinende Problem(typ)änderung
aus "leichtem" Problem "schweres" Problem machen kann

BEGRIFFE UND ERSTE RESULTATE

n Jobs $JM := \{1, \dots, n\}$
auf
m Maschinen $MM := \{M_1, \dots, M_m\}$
zu bearbeiten

(zunächst:)
- alle Jobs bei Einplanungsvorgang bekannt

- jeder Job zu jedem Zeitpunkt
  auf maximal 1 Maschine bearbeitet
- jede Maschine zu jedem Zeitpunkt für maximal 1 Job aktiv

- keine Unterbrechungen

(potentielle) Charakteristika / Attribute von Jobs

- **Bearbeitungsdauer** processing time
  $p_{ij} \geq 0$ benötigte Arbeitszeit für Job j auf M_i

- **Bereitstellungstermin** release date
  $r_j \geq 0$ frühester Bearbeitungsbeginn für Job j

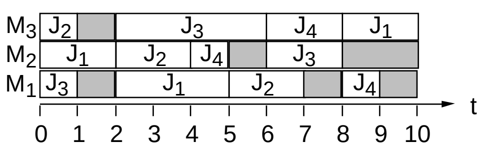
- **Fälligkeitstermin** due date
  $d_j \geq 0$ spätestes (gewünscht.) Bearbeitungsende j

- **Gewicht** weight
  $w_j \geq 0$ Wichtigkeit / Priorität j im Vgl. zu anderen
  $w_j > w_k$: j wichtiger als k

Nach Festlegung aller Zeitintervalle,
in denen Maschinen Jobs zugewiesen,
existiert
Schedule / Bearbeitungsplan / Belegungsplan

- für jeden Job:
  Folge von Zeitintervallen von Tasks auf Maschinen
- für jede Maschine
  Job(-Task)reihenfolge

Festgehalten zB in Balken- / **Gantt-Diagrammen**



Beispiel
- 4 Jobs (mit ihren Tasks)
- auf 3 Maschinen

Schedule heißt **zulässig**,
wenn alle Restriktionen Scheduling-Problem erfüllt

Schedule heißt **optimal**,
wenn zulässig und Zielfunktion optimiert

Aus (Art der) job-Bearbeitung j resultierende "Kosten" erfaßt
(wie gewohnt, mannigfaltig interpretierbar)
durch monoton wachsende (iS nicht fallend) **Kostenfunktion**

$f_j : \mathbf{R}_+ \to \mathbf{R}$

$f_j(t)$ anfallende Kosten J_j bei Beendigung zu Zeitpunkt t

Mit Schedule feststehende Beurteilungsgrößen Job j zB

- **Abschlußzeitpunkt** completion time C_j
- **Durchlaufzeit** / Verweilzeit turnaround time / residence time $V_j := C_j - r_j$
- **Verspätung** lateness $L_j := C_j - d_j$

Zielfunktionen manngifaltiger Art, manche mit "Kennungen"

zB	Kennung
$\max\limits_{j=1}^{n} f_j(C_j)$	f_{max}
$\sum\limits_{j=1}^{n} f_j(C_j)$	

welche ia zu minimieren sind
"MiniMax"-, MiniSum"-Probleme

oft spezielle

Zielfunktionen	Kennung
$\max\limits_{j=1}^{n} C_j$	C_{max}
$\max\limits_{j=1}^{n} L_j$	L_{max}

oder auch
$\Sigma\, C_j$, $\Sigma\, w_jC_j$, $\Sigma\, w_j U_j$
wo $U_j = 1$ für $C_j > d_j$
$U_j = 0$ sonst

Kennzeichnung Scheduling-Probleme
(ua) mit Tripel $\alpha|\beta|\gamma$
mit α für Maschinenkonfiguration
zB 1 Einzelmaschine
P2 2 identische parallele Maschinen
P unbeschränkt viele par. Maschinen
β für Bearbeitungsspezifika
zB pmtn Jobunterbrechungen (preemptions) zugelassen
r_j Bereitstellungstermine vorgegeben
prec Bearb'gsreihenfolgen (precedence) vorgegeben

γ für Zielfunktion
zB f_{max}, C_{max}, L_{max}

Konkrete Beispiele
$1|tree|\Sigma\, w_jC_j$
$1|pmtn,r_j|f_{max}$
$P2||C_{max}$

EIN-MASCHINEN-PROBLEME

- Jobs nicht (in Tasks) unterteilt,
- alle Jobs von selber Maschine zu bearbeiten
  Notation $p_{ij} \to p_j$

Satz 6.3.01 : Ein-Maschinen-Schedule

Jedes Ein-Maschinen-Problem ohne Bereitstellungstermine besitzt einen optimalen Schedule ohne Leerzeiten.

Dabei ist jeder optimale Plan eines Problems
ohne Unterbrechungen
auch optimal für das Problem
mit zugelassenen Unterbrechungen.

- f_j monoton steigend
- Zielfunktion monoton steigend, zu minimieren
- jede "Lücke" im Schedule aufzufüllen,
  jede "Lücke" in Bearbeitungsintervall aufzufüllen

Konkrete Probleme:

- $1||C_{max}$ (MiniMax) trivial, da für jede lückenlose Bearbeitung
  $C_{max} = \Sigma\, p_j$ konstant optimal

- $1||L_{max}$ (MiniMax) EDD-Regel: Earliest Due Date
  Jobs nach nichtfallenden
  Fälligkeitsterminen d_j bearbeitet

  Aufwand (sortieren): O(n log n)

EDD ist optimal für $1||L_{max}$

- in allen Schedules tauchen alle jobs auf
  sei Schedule Π^* EDD
  Schedule Π nicht EDD

  jobs j, k mit $d_k \le d_j$
  welche
  in Π unmittelbar aufeinanderfolgen
  $\Pi=(\ldots,j,k,\ldots)$
  in Π^* in umgekehrter Reihenfolge auftauchen
  $\Pi^*=(\ldots,k,\ldots,j,\ldots)$

  Vertauschung j, k in Π
  $d_k \le d_j$, $C_k \ge C_j$
  $C_k\text{-}d_k \ge C_j\text{-}d_j$
  * verändert L_{max} nicht Beschreibung
  * oder verkleinert L_{max}

- Π^* aus Π durch endlich viele Vertausch'gen erreichbar

  $L_{max}(\Pi) \ge L_{max}(\Pi^*)$,
  Π^* optimal

- $1|prec|f_{max}$ (MiniMax) — Beschreibung prec
  zB durch zyklenfreien Digraphen mit
  Jobs als Knoten,
  direkten Folgevorschriften als Kanten

- $1|r_j|L_{max}$ (MiniMax) — ist exponentiell
  $1 | | L_{max}$ war polynomial
  $1 | r_j , p_j=1 | L_{max}$ ist polynomial
  $1 | pmtn , prec , r_j | f_{max}$ ist polynomial

- $1||\Sigma C_j$ (MiniSum) — SPT-Regel: Shortest Process. Time First
  Jobs nach nichtfallenden
  Bearbeitungsdauern p_j bearbeitet
  Aufwand (sortieren): $O(n \log n)$

  SPT ist optimal für $1||\Sigma C_j$

  - jobs j, k mit $p_j < p_k$

  - job j unmittelbar vor k, zu A startend:
    $C:=C_j + C_k = (A + p_j) + (A + p_j + p_k)$
    Vertauschung jobs j, k:
    $C':=C_k' + C_j' = (A + p_k) + (A + p_k + p_j) > C$

    Abweichung von SPT vergrößert ΣC_j
    (für $p_j = p_k$ keine Veränderung)

- $1|| [\Sigma(C_j-r_j)]/n$ Zielfunktion ist mittlere Durchlaufzeit (MiniSum)
  SPT ist wieder optimal

  multiplikative, additive Konstanten Zielfunktion
  ändern Optimalitätsregeln nicht

- …

MEHRERE PARALLELE MASCHINEN

- Jobs nicht (in Tasks) unterteilt

- alle Jobs von selbem Typ Maschine bearbeitbar

- m (zeitparallel arbeitende) Exemplare M-Typ vorhanden
  Unterschiede ggf in "Geschwindigkeiten" s_{ij}
  so daß Ausführungszeit $s_{ij}\, p_{ij}$
  ≠ "processing time" p_{ij}
  dh maschinenspezifisch

  * $s_{ij} \equiv c$ identische parallele Maschinen
    Kennzeichnung: P
    → $s_{ij} := 1$, $p_j := p_{ij}$ gesetzt

  * $s_{ij} \equiv s_i$ uniforme parallele Maschinen
    Kennzeichnung: Q

- Jobs von irgendeiner Maschine zu bearbeiten
  (nicht von mehreren gleichzeitig)

Große Bedeutung beim Scheduling paralleler Prozessoren,
aber Probleme oft exponentiell
(BS-Entwicklungen sind schon "daran" gescheitert)

- $P2||C_{max}$ bereits exponentiell

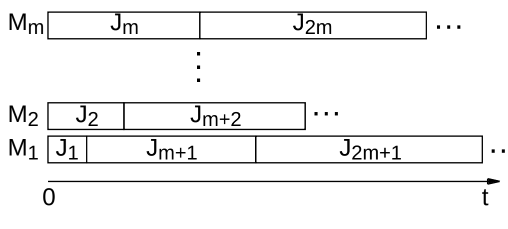
- $P2||\Sigma\, w_jC_j$ ebenfalls

Bei Zulassung von Unterbrechungen
oft polynomiale exakte Lösungen

- $P||\Sigma C_j$ (MiniSum) — beliebig (aber endlich: m) viele
  identische parallele Maschinen

  SPT-Analogon (Conway):
  Jobs geordnet nach nichtfallenden Bearb.Z.
  $p_1 \le \ldots \le p_n$
  Einplanung in dieser Reihenfolge,
  für frühest verfügbare Maschine
  (falls nicht eindeutig, kleinster M-Index)

  - Conway-Regel liefert Schedule:



  Conway-Regel ist optimal für $P||\Sigma C_j$

  - M_i seien n_i Jobs zugewiesen $i=1,\ldots,m$; $\Sigma n_i =n$
    mit aufeinander folgenden Bearbeitungen der
    Jobs $j_{i,1},\ldots,j_{i,ni}$
  - auf M_i sind die Abschußzeitpunkte
    $C_{i,1}=p_{i,1}$, $C_{i,2}=p_{i,1}+p_{i,2}$, …, $C_{i,ni}=p_{i,1}+\ldots+p_{i,ni}$
    und Gesamtsumme

    $$\Sigma\, C_j = \sum_{i=1}^{m} \sum_{k=1}^{n_i} \left(n_i - k + 1\right) p_{i,k}$$

  - Summe Produkte aus je Bearb.Zeiten (jede 1mal)
    Faktoren (ganzzahlig)
    minimiert für Faktoren nichtsteigend,
    Bearbeitungszeiten nichtfallend

Satz 6.3.02 : MiniSum-Probleme mit / ohne Unterbrechung

Ist ein Schedule optimal für $P||\Sigma\, w_jC_j$,
dann ist er auch optimal für $P|pmtn|\Sigma\, w_jC_j$

Bei identischen Maschinen läßt sich
gewichtete Summe Abschlußzeiten
nicht durch Unterbrechung + Verschiebung verkleinern

- $P|pmtn|\Sigma C_j$ (MiniSum) — optimal mit Conway-Regel lösbar

- $P||C_{max}$ (MiniMax) — analog SPT scheint hier empfehlenswert
  LPT Largest Processing Time First
  (große zuerst zum Schluß kleine
  gut verteilbar)
  Idee trägt nicht, exakte $P||C_{max}$-Lösung
  ist exponentiell
  LPT aber gute Heuristik
  (maximaler rel. Fehler (1-1/m)/3 , oB)

dagegen:
- $P|pmtn|C_{max}$ (MiniMax) in $O(n)$ exakt lösbar

  - offensichtlich
    ist $C_{max} \ge$ maximale Bearbeitungszeit Job
    und $C_{max} \ge$ mittlerer Belegungszeit der Maschinen

    $$C_{max} \ge \max\left(\max_{j=1}^{n} p_j,\ \frac{1}{m}\sum_{j=1}^{n} p_j\right) =: C'$$

- folgender Schedule (McNaughton)
  realisiert C' ist optimal

- gesamte Bearbeitungszeit
  $mC' = \Sigma p_i + \max(L,0)$
  mit $L := m(\max p_i) - \Sigma p_i$

- McNaughton-Schedule:
  Jobs (in beliebiger Reihenfolge)
  lückenlos an Maschinen verweisen
  (1 Maschine nach der anderen)
  bis C' erreicht
  (je Maschine)
  dann Unterbrechung
  (Verweis Rest an nächste Maschine)

  C' realisiert
  mit maximal m-1 Unterbrechungen
  Leerzeiten der Gesamtlänge L

- …

FLOWSHOP- UND JOBSHOP-PROBLEME

Betrachtung von

- $m > 1$ Maschinen $M_1,\ldots,M_m$
- Jobs j, die aus Tasks / Arbeitsvorgängen O_{ij} bestehen
  wo Zuordnung von Maschine M_i vorgeschrieben
  so daß zu Task O_{ij} Bearbeitungsdauer p_{ij} bekannt
- Reihenfolge Tasks von Jobs j nicht vorgeschrieben
  "OPENSHOP" Kennzeichnung: O
- Reihenfolge Tasks von Jobs j vorgeschrieben,
  für alle Jobs j identisch (→ Maschinen entspr. numeriert)
  "FLOWSHOP" Kennzeichnung: F
- Reihenfolge Tasks von Jobs j vorgeschrieben,
  nicht notwendig für alle Jobs j identisch
  "JOBSHOP" Kennzeichnung: J

Aus dem O-Bereich:

- Die meisten Probleme in diesem Bereich sind exponentiell
  so zB $O2||L_{max}$
  $O||\Sigma C_j$
  $O|pmtn|\Sigma C_j$
- $O2||C_{max}$ polynomial lösbar in O(n)
  und gleichzeitig auch $O2|pmtn|C_{max}$
- $O3||C_{max}$ ist bereits wieder exponentiell

Aus dem F-Bereich:

- Jobs durchlaufen Maschinen in identischer Reihenfolge
  oBdA $M_1, \ldots, M_m$
- Ein **Permutationsplan** permutation schedule
  ist ein spezieller Typ von F-Schedule,
  in dem alle Jobs
  auf allen Maschinen
  in gleicher Folge bearbeitet
  Permutation der Job-Indizes 1,…,n
  + Start ersten Jobs dieser Permutation auf M_1 zu t=0
  + keine unnötigen Leerzeiten
  legen Permutationsplan fest

Satz 6.3.03 : Existenz Permutationspläne

- Die Probleme $F2||C_{max}$ und $F3||C_{max}$
  besitzen optimale Permutationspläne (oB)

Folgende Regel (Johnson) liefert
optimalen Permutationsplan für $F2||C_{max}$
(in Sonderfällen auch für $F3||C_{max}$)

Sei abkürzend
$a_j := p_{1j}$ $b_j := p_{2j}$

Johnson's Regel:
Permutationsplan ist optimal, bei dem
Job j genau dann vor Job k wenn
$\min(a_j,b_k) \le \min(b_j,a_k)$

+ Verfahren für Permutationsplan, mit Aufwand O(n log n):

Johnson-Verfahren für Permutationsplan $F2||C_{max}$

- Jobmenge JM := {1,…,n}
- Teilmengen A := {j JM; $a_j<b_j$}
  B := {j JM; $a_j \ge b_j$} = JM\A
- Ordnung A nach nichtfallenden a_j
  B nach nichtsteigenden b_j
- Schedule j A (in dieser Ordnung)
  gefolgt von j B (in dieser Ordnung)

- $F||C_{max}$ m>2 ist exponentiell
  mit Schwierigkeiten selbst für B&B

  "CDS"-Heuristik (nicht exakt)
  (Campbell/Dudek/Smith)
  wendet m-1 Schritte Johnson
  auf jeweils $F2||C_{max}$ an

Aus dem J-Bereich:

"es wird immer schwieriger":
→ aktuelle Forschungsgebiete

- $J2||C_{max}$ mit Jackson-Algorithmus
  (Abwandlung Johnson-Regel)
  in O (n log n) lösbar

- "darüber hinaus": exponentiell, "faktisch" exponentiell

  zB 1989 erstmalige B&B-Lösung
  für 10 Job / 10 Maschinen-Problem $J||C_{max}$

Scheduling-Probleme dieses Abschnitts
als "statische" Probleme behandelt

alle Informationen (zB alle Jobs) bekannt
→ deterministischer Schedule zu finden

In praxi tauchen "immer wieder"
(bisher unbekannte) Jobs auf
müssen "dynamische" Probleme gelöst werden

(Später:) weiter bei "stochastischen Scheduling-Problemen"

LEER

LEER

LEER