Kapitel 3

Mathematische Optimierungsmodelle

Lösung von Linearen Optimierungsmodellen

Uwe H. Suhl
Lehrstuhl für Wirtschaftsinformatik
Freie Universität Berlin

Optimierungssysteme
Version 1.1 / SS 2008

Computational forms of LP Models

- LP models have the general form **(EMR)**
  - Min $c'x$, $bl \le Ax \le bu$, $l \le x \le u$, where $c, x, l, u \in R^n$, $bl, bu \in R^m$ and $A \in R^{m\times n}$
  - Elements of bl, bu, l, u may be plus or minus infinity
  - There can be free rows and columns where both bounds are -+ infinity
  - This form is also called the ***external model representation*** (EMR)
- This form is computationally unsuitable because row and column vectors may be linearly dependant and variables cannot be treated uniformly
- there are two possible computational forms which are based on adding m logical variables combined in a vector $s \in R^m$ **(IMR)**
  - Form 1: Min $c'x$, $A x + E_m s = bu$, $0 \le s \le bu - bl$, $l \le x \le u$
  - Form 2: min $c'x$, $A x + E_m s = 0$, $-bu \le s \le -bl$, $l \le x \le u$ (used in MOPS)
- Both forms lead to a universal form for a general LP problem
  - Min $c'x$, $A x = b$, $l \le x \le u$, $c, x, l, u \in R^{n+m}$, $b \in R^m$, $A \in R^{m\times(m+n)}$, rank(A) = m
  - Elements of l, u may be plus or minus infinity
- **This form is a equivalent to the following general form of an LP-Model**
  - Min $c'x$, $A x = b$, $c, x, l, u \in R^n$, $b \in R^m$, $A \in R^{m\times n}$, rank(A) = m, n > m
  - Elements of l, u may be plus or minus infinity
  - Thus, every LP-model can be converted to this form which is called the ***standard LP form***
- Another possibility is to transform (LP) by a linear translation such that $l=\underline{0}$ for variables which are not free
- The computational forms play only a role for the theoretical framework and for the implementation of the computational kernels in an optimization system

Umwandlungsprozesse für das LP-Standardformat

- **Min-Zielfunktion ⇒ Max-Zielfunktion: Multiplikation mit -1**
  - **min 3 *x*1 – 4 *x*2 ⇒ max –3 *x*1 + 4 *x*2**
- **Ungleichungen können durch Einführen von Schlupfvariablen in Gleichungen überführt werden; Beispiele:**
  - **2 *x*1 + *x*2 ≤ 7 ⇒ 2 *x*1 + *x*2 + *x*3 = 7, *x*3 ≥ 0 *Schlupfvariable***
  - **2 *x*1 + *x*2 ≥ 7 ⇒ 2 *x*1 + *x*2 - *x*3 = 7, *x*3 ≥ 0 *Schlupfvariable***
  - **2 ≤ 2 *x*1 + *x*2 ≤ 7 (range constraint) ⇒ 2 *x*1 + *x*2 + *x*3 = 7, 0 ≤ *x*3 ≤ 5 (7 – 2)**
  - **2 *x*1 + *x*2 = 7 ⇒ 2 *x*1 + *x*2 + *x*3 = 7, *x*3 = 0 *künstliche Variable (artificial)***
- **Gleichungen sind äquivalent zu zwei Ungleichungen; im LP-preprocessing kann aus 2 Ungleichungen u.U eine Range-Restriktion generiert werden**
  - **2 *x*1 + *x*2 ≤ 7 und 2 *x*1 + *x*2 ≥ 2 ⇒ 2 ≤ 2 *x*1 + *x*2 ≤ 7; Folge: Reduktion von m**
- **Das Gürtelbeispiel im Standardformat**

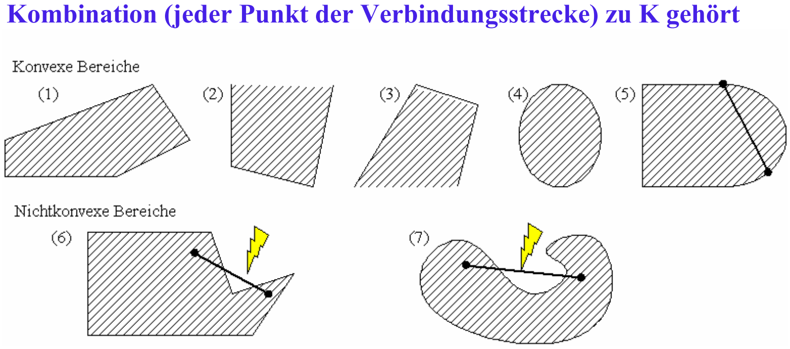
$$\begin{array}{l} \text{Max } z = 2\,x1 + 1{,}5\,x2 \\ \quad 2\,x1 + x2 + x3 = 1000 \\ \quad x1 + x2 + x4 = 800 \end{array}$$

0 ≤ x1 ≤ 400, 0 ≤ x2 ≤ 700, x3, x4 ≥ 0, wobei x3, x4 Schlupfvariablen sind

- ***Die Transformationen finden nur innerhalb des Optimierungssystems statt***
- ***Im Normalfall werden die Modelle in der EMR (s.o) formuliert und das Optimierungssystem transformiert die EMR in das interne Standardformat***

Konvexität

- **x_i seien k Vektoren des R^n und $\alpha_i \geq 0$ reelle Zahlen, i = 1,..,k; sei $x \in R^n$**
- $x = \sum_{j=1}^{k} \alpha_j x_j$ heißt Konvex-Kombination der x_i, wenn $\sum_{j=1}^{k} \alpha_j = 1$
- **Sind alle $\alpha_i > 0$, so heißt x eine *echte Konvex-Kombination* der x_i**
- **Eine Punktmenge $K \subset R^n$ heißt konvex, wenn mit je zwei Punkten x_1, x_2 auch jede Konvex-Kombination (jeder Punkt der Verbindungsstrecke) zu K gehört**



- **Die Menge der zulässigen Lösungen eines LP-Modells ist eine konvexe Menge, denn aus Ax_1=b und Ax_2=b, und $\alpha_1+\alpha_2 = 1$, $\alpha_i \geq 0$ folgt $A(\alpha_1 x_1+\alpha_2 x_2) = \alpha_1 b+\alpha_2 b = b$ und aus $\alpha_1 l \leq \alpha_1 x_1 \leq \alpha_1 u$, $\alpha_2 l \leq \alpha_2 x_2 \leq \alpha_2 u$ folgt $\alpha_1 l + \alpha_2 l \leq \alpha_1 x_1+\alpha_2 x_2 \leq \alpha_1 u+\alpha_2 u$, also $l \leq \alpha_1 x_1+\alpha_2 x_2 \leq u$**
- ***Ist eine optimale LP-Lösung nicht eindeutig (Regelfall bei praktischen Modellen), so gibt es unendlich viele optimale LP Lösungen, da $Z=c'(\alpha_1 x_1+\alpha_2 x_2)=\alpha_1 c' x_1+\alpha_2 c' x_2=\alpha_1 Z_{opt}+\alpha_2 Z_{opt}=Z_{opt}$***

Eckpunkte – ein wenig Theorie

- Es seien $a_1,...,a_n$ die Spaltenvektoren der Matrix A, $a_i \in R^m$ und F die Menge der zulässigen Lösungen $F=\{x|x\in R^n, Ax=b, l\leq x\leq u\}$ des LP-Problems
- F kann leer, beschränkt oder unbeschränkte (konvexe) Teilmenge des R^n sein
- $Ax=b$ ist äquivalent zu $\Sigma\, a_i\, x_i = b$, wenn $x = (x_i)_{i=1,...,n}$
- *Ein Punkt $x \in F$ heißt **Eckpunkt** (kurz Ecke), wenn sich x **nicht** als echte Konvexkombination zweier verschiedener Punkte von F darstellen lässt*
- 2-dimensionaler Fall: Ecken sind die Endpunkte einer Verbindungsgeraden
- $x\in F$ ist genau dann Ecke von F, wenn in $\Sigma\, a_i\, x_i = b$ die Spaltenvektoren a_i die zu Werten $l_i<x_i<u_i$ gehören, linear unabhängig sind (ohne Beweis)
- Ist $x\in F$ Ecke, dann gilt für ***höchstens m Werte*** $l_i<x_i<u_i$ (ohne Beweis); für die übrigen Werte gilt $x_i\in\{l_i, u_i\}$ - wenn keine freien Variablen vorkommen (oB)
- Eine Ecke bei der für ***weniger als m Werte*** gilt $l_i<x_i<u_i$, heißt ***entartete Ecke***
- Ist F nicht leer, so ist die Menge der Ecken von F nicht leer und endlich (oB)
- Eine Optimallösung eines LP-Modells wird an einer Ecke angenommen (oB)
- Einer (entarteten) Ecke von F können m linear unabhängige Spaltenvektoren mit $l_i<x_i<u_i$ (Normalfall) bzw. $l_i\leq x_i\leq u_i$ und $\Sigma\, a_i\, x_i = b$ zugeordnet werden
- Die ***Indexmenge JB***, $|JB|=m$ dieser Spalten heißt ***Basis*** und $(a_i)_{i\in JB}$ ***Basismatrix***

Anzahl der Eckpunkte eines LP-Modells

- Anzahl der Ecken eines LP-Modells mit m Restriktionen und n Variablen (zulässig oder unzulässig):

$$\binom{n+m}{m} = \frac{(n+m)!}{m!n!} = \frac{\overbrace{(n+m)(n+m-1)...(n+1)}^{m\ \text{Terme}}}{m(m-1)...1} \geq \frac{(2m)(2m-1)...(m+1)}{m\quad(m-1)\quad...\quad 1} >> 2^m,\ \text{falls}\ n\geq m$$

$$= \binom{m+n}{n} = \frac{(m+n)(m+n-1)...(m+1)}{n(n-1)..1} \geq \frac{(2n)(2n-1)...(n+1)}{n\quad(n-1)\quad...\quad 1} >> 2^n,\ \text{falls}\ m\geq n$$

$$\Rightarrow \binom{m+n}{m} = \binom{m+n}{n} >> \min(2^m, 2^n)$$

- Daher ist eine Berechnung der Eckpunkte und deren Enumeration für praktische Modelle mit Tausenden von Variablen und Restriktionen nicht praktikabel
- Die Simplex-Methode ist ein Verfahren, das nur einen kleinen Bruchteil der Ecken bestimmt und Optimalität beweist
- Die Implementierungen sind über Jahrzehnte weiterentwickelt worden und hocheffizient

Basis, primal and dual feasibility

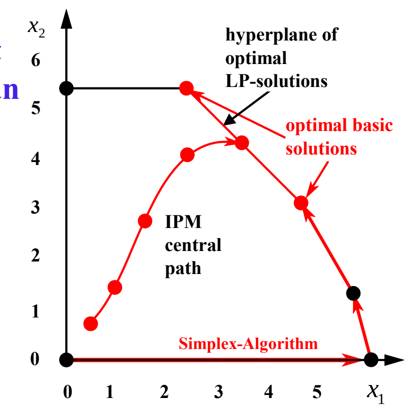
- A *basis* is an ordered index set $JB \subset J=\{1,\ldots,n+m\}$ with $|JB|=m$, such that the columns a_j, $j \in JB$ of A are linear independent
- JB defines an mxm *basis matrix* B of columns a_j of A, $j \in JB$; B has rank m
- NB = J-JB index set of nonbasic variables, N the corresponding Matrix
- (JB,JN) defines a partition of x in $x=(x_B,x_N)'$, $l=(l_B,u_N)'$, $c=(c_B,c_N)'$
- A *basic solution* is defined as follows:
  - the values of $x_N(i)$ are either l_i or u_i if one is finite, if $x_N(i)$ is free $x_N(i) = 0$
  - the basic solution values are uniquely determined by setting the nonbasic variables
  - primal part: $x_B = B^{-1}(b - Nx_N)$ since $Ax = Bx_B + Nx_N = b$
  - dual part: $\pi' = c_B' B^{-1}$ (dual costs), $d_j = c_j - \pi' a_j$, $j \in J$ (reduced costs)
  - B^{-1} is never computed; x_B and π are computed by solving two triangular systems based on $L \bullet U = P\ B\ Q$, where P, Q are permutation matrices and L resp. U are lower resp. Upper triangular
- A basic solution is *primal feasible* iff $l_B \leq x_B \leq u_B$
- A basic solution is *dual feasible* iff $d_j=0$ for $j \in JB$, $d_j \geq 0$ if $x_j=l_j$, $d_j \leq 0$ if $x_j=u_j$ and $d_j=0$ if j corresponds to a free variable, fixed variables may have any d_j value
- A basic solution is *optimal* if it is primal and dual feasible
- Theorem: If there exists an optimal solution to (LP) then there exists an optimal basic solution
- Theorem applies also to interior point methods: we want for various reasons an optimal basic solution; this is done by a specialized simplex (x-over)

Algorithms for solving LPs - Comparison

- three computational competitive algorithms
  - Primal simplex
  - Dual simplex
  - Interior point (barrier)
  - There are problem classes where each algorithm is best
- Simplex (primal, dual)
  - computes basic solutions – very important for IP to exploit strong LP formulation
  - excellent warm start capabilities important for branch-and-bound in solving IPs
  - Sparsity of model and LU-factorization can be exploited in optimal way - resulting in lower main memory requirement, important for very large models if 2 GB address limit
  - Primal simplex is most reliable engine – can be slow on highly degenerate problems
- interior point
  - Main kernel is the symbolic factorization of a sparse positive definite symmetric system
  - Number of expensive iterations is between 30-50; after 150 iters. MOPS uses simplex
  - Iteration speed depends mainly on the size (fill-in) of the sparse cholesky factorization
  - Key is dense processing: Intel MKL offers highly optimized BLAS used in MOPS IPM
  - needs a crossover from an optimal LP-solution to an optimal basic solution
  - No efficient warm start – rules out use in branch-and-bound
- During last decade dual simplex has become a strong contender

Prinzip der Lösungsverfahren

- Beim Simplexverfahren bewegt man sich bei jeder Iteration von einer Ausgangs-Ecke zu einem benachbarten Eckpunkt, wobei das Verfahren sicherstellt, dass der Zielfunktionswert sich *monoton verbessert (primal)*
- Man bewegt sich also nach der Bestimmung einer zulässigen Anfangslösung grundsätzlich auf der Oberfläche des Polytops der zulässigen Lösungen
- Bei degenerierten Ecken gibt es mehrere zugehörige Basislösungen; daher kommt es vor, dass man Iterationen ausführt, wo man an einer Ecke bleibt und lediglich die Basis wechselt – der ZF-Wert ändert sich nicht
- Bei innere Punkte Verfahren (IPM) bewegt man sich grundsätzlich im Inneren eines Polytops
- Eine optimale Lösung ist im Regelfall keine Basislösung, d.h. wird nicht an einer Ecke erzielt
- Aus verschiedenen Gründen möchte man jedoch eine optimale Basislösung
- Dies erreicht man mit nach geschalteten Basis-Identifikationsalgorithmen (x-over)



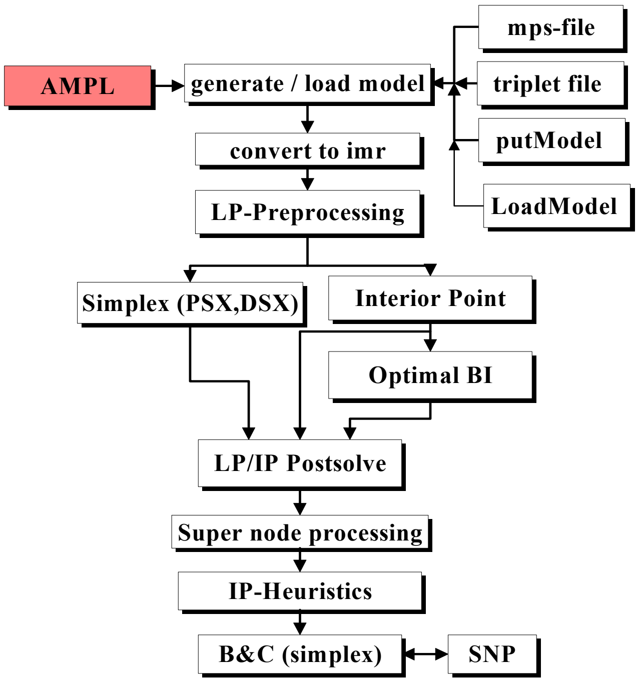
MOPS©

- Optimierungssystem zur schnellen Lösung von LP and IP-Modellen
- Zentrale Bedeutung hat Lösungszeit und die Lösbarkeit von IP-Modellen
- Basis sind komplexe mathematische Algorithmen: Grundlagenforschung
- Gehört zu den vier schnellsten Systemen weltweit – einziges in Deutschland
- Wird seit 1987 laufend verbessert: mathematische Konzepte, Algorithmen, Datenstrukturen, Software Technologie (Intel MKL), Alladin
- umfasst ca. 1000 Prozeduren und entstand in vielen Jahren Forschung und Entwicklung – Publikationen in intern. wissenschaftlichen Zeitschriften
- Intensive Kooperation mit dem LS-L. Suhl: A. Koberstein Dual Simplex (PhD-Arbeit), P. Christophel Heuristik, F. Wesselmann Cuts und Cut-Pool,…
- Kooperationen mit: Csaba Meszaros, ung. Akad. der Wiss.
- Zentrale Bedeutung haben
  - LP-Kernels (LPP, IPM, DSX, PSX)
  - mathematische Techniken zur Verschärfung von LP-Relaxationen (SNP)
  - B&B/C-Technologie, Heuristiken

year	version	Shell (5563 x 6181) LP	secs
1991	1.4	I486 (25Mhz), DOS	612.4
1996	2.8	HP-735, HP-UX	17.6
2001	5.0	PIII (500), Win98	3.9
2003	6.3	PIV (2.2) Win 2000	0.8
2007	9.x	PIV (3.0) Win XP	0.6

year	version	Shell (5563 x 6181) IP	secs
1994	2.0	PII (500), LIFO-MIP	1794.3
1995	2.5	PII (500), NON LIFO	450.1
1999	4.0	PIV (2,2) IPM, new cuts	75.2
2003	6.3	PIV (2.2) refinements	39.6
2007	9.x	PIV (3.0) Gomory, MIR cuts,	9.8

MOPS Architecture



Performance of LP-Software (Simplex): Influence Factors

- the LP-model (structure and data),
- the number of pivots (basis changes) which is very much dependent on the pricing strategies: Easy LP-models require k•m iterations where k is a small number in a typical range 2-5. For hard (practical) LP models k can be in the range 10-20,
- LP-preprocessing, in case that a large part of the LP-model is redundant and can be removed from the active LP-model,
- pricing strategies (row and column selection) to reduce number of iterations,
- degeneracy handling methods during primal and dual simplex to reduce the number of degenerate simplex iterations,
- starting basis produced by a crash heuristic or an advanced basis from a similar model,
- iteration speed which depends among other things on an efficient way to exploit sparsity and hypersparsity in the computation of vectors; Data structures and other implementation details play a key role for the iteration speed,
- low level implementation details such as data locality to access data stored in the cache of a computing system, and compilation techniques to exploit the hardware of the machines.

LP Preprocessing overview in

repeat
  repeat
    repeat
      Singleton row tests
      Singleton column tests
      Primal feasibility tests
      Cheap dual tests
    until no more reductions are possible
      Dual feasibility tests
      Duplicate column tests
      Duplicate row tests
    until no more reductions are possible
    Doubleton row elimination
    Bound relaxation
    Scaling of model with geometric mean
    Finding of linearly dependent rows
    Elimination of free variables by pivoting for sparsity and numerical stability
  until max. number of passes reached or no more reductions are possible

- Postsolve computes an optimal primal and dual solution to the original model (LP & IP) from the opt. sol. of the reduced model and a stored log
- LP-Preprocessing und Postsolve are very complex software modules

LP-Preprocessing in

- State-of-the-art; joint development with Csaba Meszaros
- Paper published in OR Spektrum 25 (4), November 2003, Springer
- LP-Preprocessing is also very important for IP because the reduced problem size benefits the solution of LPs in b&b

model	dimensions	m	n	nz
	original	10264	29264	67856
Schering	Active v3	10165	29165	67662
	Active v7	6062	16759	34664
	Original	33690	28122	114039
Bayer	Active v3	24907	21318	93720
	Active v7	15345	9537	42940
	Original	17431	81896	457990
	Original	217238	172158	997990
tough	Active v7	8633	99725	129119
	Original	105127	154699	233288
Ken18	Active v7	46887	96783	233288
	Original	391609	295801	1270825
Gams	Active v7	291708	197898	1158840
	Original	104376	450915	1864215
xs01	Active v7	32143	374744	2316087

64-Bit Plattform versus 32 Bit Plattform

- Pentium IV (32-Bit) hat ein Adresslimit von 2 GB (mit Ausnahmen: 3 GB)
- Neue Pentium D bzw. Prozessoren mit Intel EMT64 Adressierung und Win64 XP heben dieses Limit auf

Name	Rows	variables	Int. vars.	nonzeros
Halle01	14994	53249	40387	113069
Halle02	21939	87128	67367	184111
Halle03	29031	118768	91957	250675
Mun28	52303	478823	429088	981163
MDVS	61254	573300	515530	1174095
Mun01	163142	1479833	1330580	3031285
Mun02	196263	3366143	3183593	6834311

Name	**MOPS (secs) 3,4 GHz, WinXP) for initial LP**		
Name	Primal simplex	Dual simplex	Barrier + x-over
Halle01	281.69	21.28	33.19
Halle02	Nt	61.38	73.06
Halle03	Nt	176.05	108.81
MDVS	Nt	3051.58	1519.33
Mun28	Nt	4266.36	798.24
Mun01	Nt	15012.08	nem
Mun02	Nt	>20000	nem

- Auf 32 Bit System ist Mun01 und Mun02 wegen des virtuellen Speicherlimits mit IPM nicht lösbar; Zeiten auf PIV 3,4 GHz, EMT64 und Win64 XP:

Name	**default ordering**		**multisection ordering**	
Name	Nonzeros in Cholesky	Barrier time (sec)	Nonzeros in Cholesky	Barrier time (sec)
MDVS	63,648,566	3567.72	45,470,777	1603,23
Mun28	26,887,134	1206.81	22,688,525	798.24
Mun01	84,710,597	5689.22	71,388,781	3490.20
Mun02	nem	nem	210,899,940	36247,23

- Die Cholesky Factorisierung für Mun02 braucht allein 1,6 GB

IPM on Intel Core™ 2 Duo (32 Bit mode)

- IP-Model:

	Rows	variables	Int. vars.	nonzeros
MDVS	**61254**	**573300**	**515530**	**1174095**

- Results on Intel Xeon 3,4 GHz, 2 MB L2 Cache

	CP-time secs
IPM +x-over	**1509.13**
Dual simplex 8.x	**2997.38**

- Symmetric multiprocessing of MOPS 9.x on 2,4 GHz, Intel Core ™ 2 duo, 4 MB L2 Cache:

	CP-time secs	
# processors	**1**	**2**
IPM +x-over	**1565.98**	**954.91**
Dual simplex 8.x	**1955.72**	**1955.72**

- Conclusions
  - The new Core ™ 2 duo architecture offers the same performance as Xeon processors (single processor mode) but at much lower clock speed
  - IPM greatly benefits from symmetric multiprocessiong and vectorization (40% reduction)
  - The simplex method cannot take any significant advantages from vectorization and multiprocessing