Linear Programming

- brewer's problem
- simplex algorithm
- implementation
- solving LPs
- linear programming

Reference: The Allocation of Resources by Linear Programming, Scientific American, by Bob Bland

Linear Programming

What is it? see ORF 307

- Quintessential tool for optimal allocation of scarce resources, among a number of competing activities.
- Powerful and general problem-solving method that encompasses:
  - shortest path, network flow, MST, matching, assignment...
  - Ax = b, 2-person zero sum games

Why significant?

- Widely applicable problem-solving model
- Dominates world of industry.
- Fast commercial solvers available: CPLEX, OSL.
- Powerful modeling languages available: AMPL, GAMS.
- Ranked among most important scientific advances of 20th century.

Ex: Delta claims that LP saves $100 million per year.

Applications

Agriculture. Diet problem.
Computer science. Compiler register allocation, data mining.
Electrical engineering. VLSI design, optimal clocking.
Energy. Blending petroleum products.
Economics. Equilibrium theory, two-person zero-sum games.
Environment. Water quality management.
Finance. Portfolio optimization.
Logistics. Supply-chain management.
Management. Hotel yield management.
Marketing. Direct mail advertising.
Manufacturing. Production line balancing, cutting stock.
Medicine. Radioactive seed placement in cancer treatment.
Operations research. Airline crew assignment, vehicle routing.
Physics. Ground states of 3-D Ising spin glasses.
Plasma physics. Optimal stellarator design.
Telecommunication. Network design, Internet routing.
Sports. Scheduling ACC basketball, handicapping horse races.

brewer's problem
simplex algorithm
implementation
solving LPs
linear programming

Toy LP example: Brewer's problem

Small brewery produces ale and beer.

- Production limited by scarce resources: corn, hops, barley malt.
- Recipes for ale and beer require different proportions of resources.

	corn (lbs)	hops (oz)	malt (lbs)	profit ($)
available	480	160	1190	
ale (1 barrel)	5	4	35	13
beer (1 barrel)	15	4	20	23

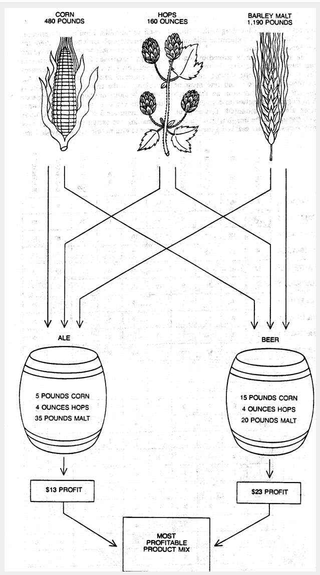
Brewer's problem: choose product mix to maximize profits.

all ale (34 barrels)	179	136	1190	442
all beer (32 barrels)	480	128	640	736
20 barrels ale 20 barrels beer	400	160	1100	720
12 barrels ale 28 barrels beer	480	160	980	800
more profitable product mix?				>800 ?

Brewer's problem: mathematical formulation

Small brewery produces ale and beer.

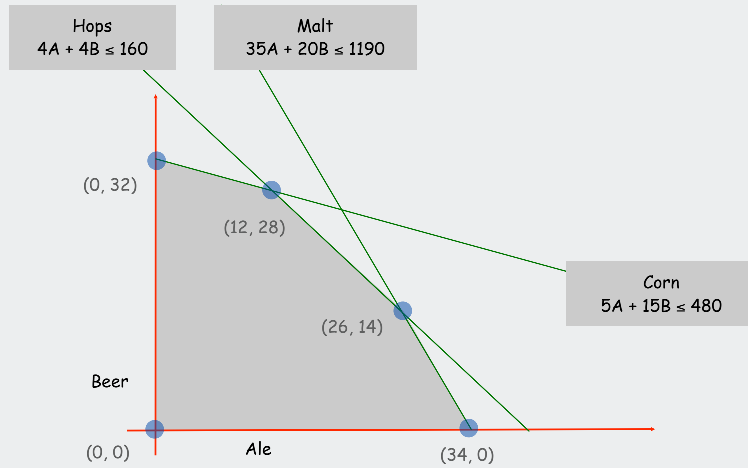
- Production limited by scarce resources: corn, hops, barley malt.
- Recipes for ale and beer require different proportions of resources.



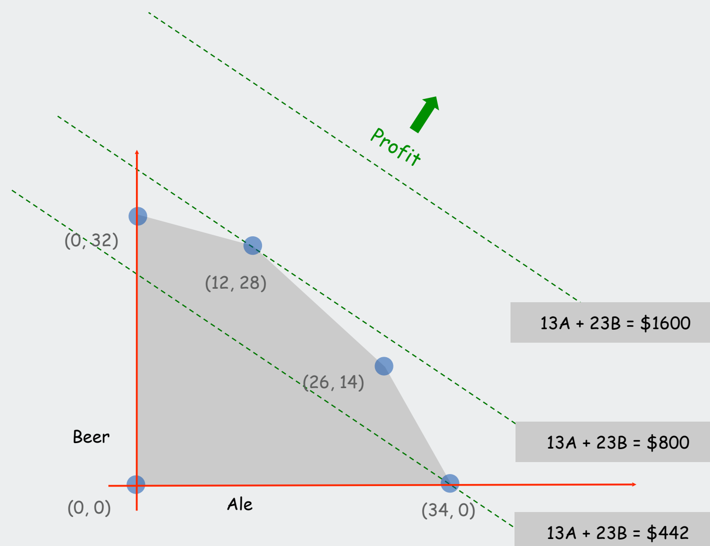
Mathematical formulation:

	ale		beer			
maximize	13A	+	23B			profit
subject to the constraints	5A	+	15B	≤	480	corn
	4A	+	4B	≤	160	hops
	35A	+	20B	≤	1190	malt
			A	≥	0	
			B	≥	0	

Brewer's problem: Feasible region

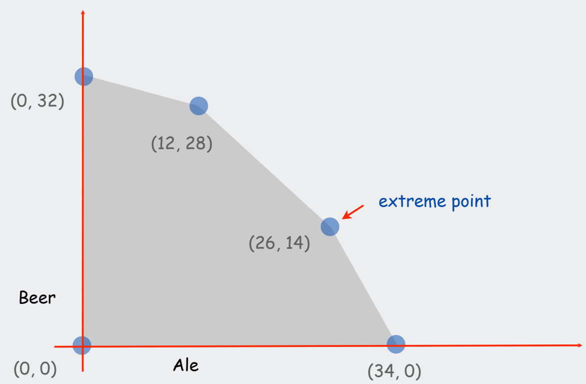


Brewer's problem: Objective function



Brewer's problem: Geometry

Brewer's problem observation. Regardless of objective function coefficients, an optimal solution occurs at an extreme point.



Standard form linear program

Input: real numbers a_{ij}, c_j, b_i.

Output: real numbers x_j.

n = # nonnegative variables, m = # constraints.

Maximize linear objective function subject to linear equations.

n variables

maximize	$c_1 x_1 + c_2 x_2 + \ldots + c_n x_n$
subject to the constraints (m equations)	$a_{11} x_1 + a_{12} x_2 + \ldots + a_{1n} x_n = b_1$
	$a_{21} x_1 + a_{22} x_2 + \ldots + a_{2n} x_n = b_2$
	...
	$a_{m1} x_1 + a_{m2} x_2 + \ldots + a_{mn} x_n = b_m$
	$x_1, x_2, \ldots, x_n \geq 0$

matrix version

maximize	$c^T x$
subject to the constraints	$A x = b$
	$x \geq 0$

"Linear" No x^2, xy, $\arccos(x)$, etc.

"Programming" " Planning" (term predates computer programming).

Converting the brewer's problem to the standard form

Original formulation.

maximize	13A	+	23B		
subject to the constraints	5A	+	15B	≤	480
	4A	+	4B	≤	160
	35A	+	20B	≤	1190
			A, B	≥	0

Standard form.

- Add slack variable to convert each inequality to an equality.
- Now a 5-dimensional problem.

maximize	Z									
subject to the constraints	13A	+	23B					− Z	=	0
	5A	+	15B	+ S_C					=	480
	4A	+	4B		+ S_H				=	160
	35A	+	20B			+ S_M			=	1190
			A, B, S_C, S_H, S_M						≥	0

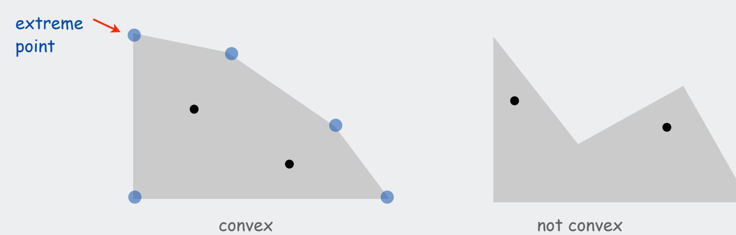
Geometry

A few principles from geometry:

- inequality: halfplane (2D), hyperplane (kD).
- bounded feasible region: convex polygon (2D), convex polytope (kD).

Convex set. If two points a and b are in the set, then so is $\frac{1}{2}(a + b)$.

Extreme point. A point in the set that can't be written as $\frac{1}{2}(a + b)$, where a and b are two distinct points in the set.



Geometry (continued)

Extreme point property. If there exists an optimal solution to (P), then there exists one that is an extreme point.

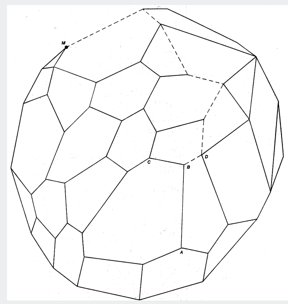
Good news. Only need to consider finitely many possible solutions.

Bad news. Number of extreme points can be exponential!

n-dimensional hypercube

Greedy property. Extreme point is optimal iff no neighboring extreme point is better.

local optima are global optima



Simplex Algorithm

Simplex algorithm. [George Dantzig, 1947]

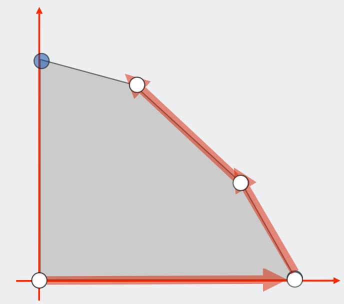
- Developed shortly after WWII in response to logistical problems, including Berlin airlift.
- One of greatest and most successful algorithms of all time.

Generic algorithm.

- Start at some extreme point.
- Pivot from one extreme point to a neighboring one.
- Repeat until optimal.

never decreasing objective function

How to implement? Linear algebra.



brewer's problem

simplex algorithm

implementation

solving LPs

linear programming

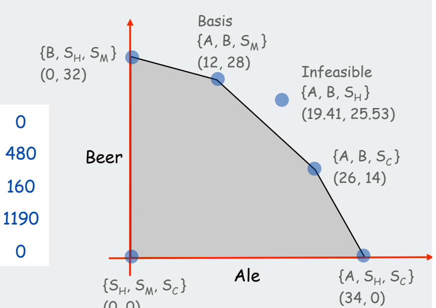
Simplex Algorithm: Basis

Basis. Subset of m of the n variables.

Basic feasible solution (BFS). Set n - m nonbasic variables to 0, solve for remaining m variables.

- Solve m equations in m unknowns.
- If unique and feasible solution $\Rightarrow$ BFS.
- BFS $\Leftrightarrow$ extreme point.

maximize	Z								
subject to the constraints	13A	+	23B				− Z	=	0
	5A	+	15B	+ S_C				=	480
	4A	+	4B		+ S_H			=	160
	35A	+	20B			+ S_M		=	1190
			A, B, S_C, S_H, S_M					≥	0



Simplex Algorithm: Initialization

$$
\begin{array}{llcccccccl}
\text{maximize} & Z & & & & & & & & \\
\text{subject to the constraints} & 13A & + & 23B & & & & - Z & = & 0 \\
 & 5A & + & 15B & + S_C & & & & = & 480 \\
 & 4A & + & 4B & & + S_H & & & = & 160 \\
 & 35A & + & 20B & & & + S_M & & = & 1190 \\
 & & & A, B, S_C, S_H, S_M & & & & & \geq & 0
\end{array}
$$

Basis = $\{S_C, S_H, S_M\}$
$A = B = 0$
$Z = 0$
$S_C = 480$
$S_H = 160$
$S_M = 1190$

Simplex Algorithm: Pivot 1

$$
\begin{array}{llcccccccl}
\text{maximize} & Z & & & & & & & & \\
\text{subject to the constraints} & 13A & + & 23B & & & & - Z & = & 0 \\
 & 5A & + & 15B & + S_C & & & & = & 480 \\
 & 4A & + & 4B & & + S_H & & & = & 160 \\
 & 35A & + & 20B & & & + S_M & & = & 1190 \\
 & & & A, B, S_C, S_H, S_M & & & & & \geq & 0
\end{array}
$$

Basis = $\{S_C, S_H, S_M\}$
$A = B = 0$
$Z = 0$
$S_C = 480$
$S_H = 160$
$S_M = 1190$

Substitute: $B = (1/15)(480 - 5A - S_C)$

$$
\begin{array}{llcccccccl}
\text{maximize} & Z & & & & & & & & \\
\text{subject to the constraints} & (16/3)A & + & 23B & - (23/15)S_C & & & - Z & = & -736 \\
 & (1/3)A & + & B & + (1/15)S_C & & & & = & 32 \\
 & (8/3)A & & & - (4/15)S_C & + S_H & & & = & 32 \\
 & (85/3)A & & & - (4/3)S_C & & + S_M & & = & 550 \\
 & & & A, B, S_C, S_H, S_M & & & & & \geq & 0
\end{array}
$$

Basis = $\{B, S_H, S_M\}$
$A = S_C = 0$
$Z = 736$
$B = 32$
$S_H = 32$
$S_M = 550$

Simplex Algorithm: Pivot 1

$$
\begin{array}{llcccccccl}
\text{maximize} & Z & & & & & & & & \\
\text{subject to the constraints} & 13A & + & 23B & & & & - Z & = & 0 \\
 & 5A & + & 15B & + S_C & & & & = & 480 \\
 & 4A & + & 4B & & + S_H & & & = & 160 \\
 & 35A & + & 20B & & & + S_M & & = & 1190 \\
 & & & A, B, S_C, S_H, S_M & & & & & \geq & 0
\end{array}
$$

Basis = $\{S_C, S_H, S_M\}$
$A = B = 0$
$Z = 0$
$S_C = 480$
$S_H = 160$
$S_M = 1190$

Why pivot on B?

- Its objective function coefficient is positive
  (each unit increase in B from 0 increases objective value by $23)
- Pivoting on column 1 also OK.

Why pivot on row 2?

- Preserves feasibility by ensuring RHS ≥ 0.
- Minimum ratio rule: min { 480/15, 160/4, 1190/20 }.

Simplex Algorithm: Pivot 2

$$
\begin{array}{llcccccccl}
\text{maximize} & Z & & & & & & & & \\
\text{subject to the constraints} & (16/3)A & & & - (23/15)S_C & & & - Z & = & -736 \\
 & (1/3)A & + & B & + (1/15)S_C & & & & = & 32 \\
 & (8/3)A & & & - (4/15)S_C & + S_H & & & = & 32 \\
 & (85/3)A & & & - (4/3)S_C & & + S_M & & = & 550 \\
 & & & A, B, S_C, S_H, S_M & & & & & \geq & 0
\end{array}
$$

Basis = $\{B, S_H, S_M\}$
$A = S_C = 0$
$Z = 736$
$B = 32$
$S_H = 32$
$S_M = 550$

Substitute: $A = (3/8)(32 + (4/15)S_C - S_H)$

$$
\begin{array}{llccccccl}
\text{maximize} & Z & & & & & & & \\
\text{subject to the constraints} & & & - S_C & - 2S_H & & - Z & = & -800 \\
 & & B & + (1/10)S_C & + (1/8)S_H & & & = & 28 \\
 & A & & - (1/10)S_C & + (3/8)S_H & & & = & 12 \\
 & & & - (25/6)S_C & - (85/8)S_H & + S_M & & = & 110 \\
 & & A, B, S_C, S_H, S_M & & & & & \geq & 0
\end{array}
$$

Basis = $\{B, S_H, S_M\}$
$S_C = S_H = 0$
$Z = 800$
$B = 28$
$A = 12$
$S_M = 110$

Simplex algorithm: Optimality

Q. When to stop pivoting?
A. When all coefficients in top row are non-positive.

Q. Why is resulting solution optimal?
A. Any feasible solution satisfies system of equations in tableaux.
- In particular: $Z = 800 - S_C - 2\,S_H$
- Thus, optimal objective value $Z^* \le 800$ since $S_C, S_H \ge 0$.
- Current BFS has value 800 $\Rightarrow$ optimal.

$$
\begin{array}{lrrrrrrrrrr}
\text{maximize} & Z & & & & & & & & & \\
\text{subject to the constraints} & & & & - & S_C & - & 2S_H & & - Z & = & -800 \\
& & B & + & & (1/10)\,S_C & + & (1/8)\,S_H & & & = & 28 \\
& A & & - & & (1/10)\,S_C & + & (3/8)\,S_H & & & = & 12 \\
& & & - & & (25/6)\,S_C & - & (85/8)\,S_H & + & S_M & = & 110 \\
& & & & A, & B, & S_C, & S_H, & S_M & & \ge & 0
\end{array}
$$

Basis = $\{A, B, S_M\}$
$S_C = S_H = 0$
$Z = 800$
$B = 28$
$A = 12$
$S_M = 110$

brewer's problem
simplex algorithm
implementation
solving LPs
linear programming

Simplex tableau

Encode standard form LP in a single Java 2D array

$$
\begin{array}{lrrrrrrrrrr}
\text{maximize} & Z & & & & & & & & \\
\text{subject to the constraints} & 13A & + & 23B & & & & - Z & = & 0 \\
& 5A & + & 15B & + S_C & & & & = & 480 \\
& 4A & + & 4B & & + S_H & & & = & 160 \\
& 35A & + & 20B & & & + S_M & & = & 1190 \\
& & & A, B, S_C, S_H, S_M & & & & & \ge & 0
\end{array}
$$

5	15	1	0	0	480
4	4	0	1	0	160
35	20	0	0	1	1190
13	23	0	0	0	0

m	A	I	b
1	c	0	0
	n	m	1

Simplex tableau

Encode standard form LP in a single Java 2D array (solution)

$$
\begin{array}{lrrrrrrrrrr}
\text{maximize} & Z & & & & & & & & & \\
\text{subject to the constraints} & & & & - & S_C & - & 2S_H & & - Z & = & -800 \\
& & B & + & & (1/10)\,S_C & + & (1/8)\,S_H & & & = & 28 \\
& A & & - & & (1/10)\,S_C & + & (3/8)\,S_H & & & = & 12 \\
& & & - & & (25/6)\,S_C & - & (85/8)\,S_H & + & S_M & = & 110 \\
& & & & A, & B, & S_C, & S_H, & S_M & & \ge & 0
\end{array}
$$

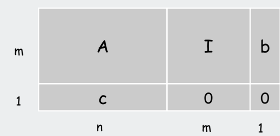
0	1	1/10	1/8	0	28
1	0	1/10	3/8	0	12
0	0	25/6	85/8	1	110
0	0	−1	−2	0	−800

m	A	I	b
1	c	0	0
	n	m	1

Simplex algorithm transforms initial array into solution

Simplex algorithm: Bare-bones implementation

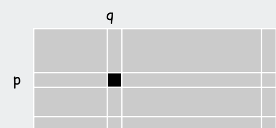
Construct the simplex tableau.




```
public class Simplex
{
   private double[][] a;   // simplex tableaux
   private int M, N;

   public Simplex(double[][] A, double[] b, double[] c)
   {
      M = b.length;
      N = c.length;
      a = new double[M+1][M+N+1];
      for (int i = 0; i < M; i++)
         for (int j = 0; j < N; j++)
            a[i][j] = A[i][j];
      for (int j = N; j < M + N; j++) a[j-N][j] = 1.0;
      for (int j = 0; j < N;     j++) a[M][j]   = c[j];
      for (int i = 0; i < M;     i++) a[i][M+N] = b[i];
   }
```


Simplex algorithm: Bare-bones Implementation

Pivot on element (p, q).



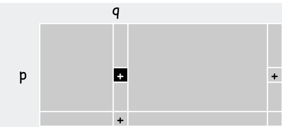

```
public void pivot(int p, int q)
{
   for (int i = 0; i <= M; i++)
      for (int j = 0; j <= M + N; j++)
         if (i != p && j != q)
            a[i][j] -= a[p][j] * a[i][q] / a[p][q];

   for (int i = 0; i <= M; i++)
      if (i != p) a[i][q] = 0.0;

   for (int j = 0; j <= M + N; j++)
      if (j != q) a[p][j] /= a[p][q];
   a[p][q] = 1.0;
}
```

zero out column q

scale row p

Simplex Algorithm: Bare Bones Implementation

Simplex algorithm.




```
public void solve()
{
   while (true)
   {
      int p, q;
      for (q = 0; q < M + N; q++)
         if (a[M][q] > 0) break;
      if (q >= M + N) break;

      for (p = 0; p < M; p++)
         if (a[p][q] > 0) break;
      for (int i = p+1; i < M; i++)
         if (a[i][q] > 0)
            if (a[i][M+N] / a[i][q] < a[p][M+N] / a[p][q])
               p = i;
      pivot(p, q);
   }
}
```

find entering variable q (positive objective function coefficient)

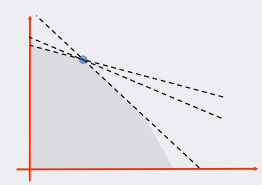
find row p according to min ratio rule

min ratio test

Simplex algorithm: Degeneracy

Degeneracy. New basis, same extreme point.

"stalling" is common in practice



Cycling. Get stuck by cycling through different bases that all correspond to same extreme point.

- Doesn't occur in the wild.
- Bland's least index rule guarantees finite # of pivots.

LP Duality: Economic Interpretation

Brewer's problem. Find optimal mix of beer and ale to maximize profits.

maximize	13A	+	23B		
subject to the constraints	5A	+	15B	≤	480
	4A	+	4B	≤	160
	35A	+	20B	≤	1190
			A, B	≥	0

$A^* = 12$
$B^* = 28$
OPT = 800

Entrepreneur's problem. Buy resources from brewer at min cost.

- C, H, M = unit price for corn, hops, malt.
- Brewer won't agree to sell resources if 5C + 4H + 35M < 13
 or 15C + 4H + 20M < 23

minimize	480C	+	160H	+	1190M		
subject to the constraints	5C	+	4H	+	35M	≥	13
	15C	+	4H	+	20M	≥	23
			C, H, M			≥	0

$C^* = 1$
$H^* = 2$
$M^* = 0$
OPT = 800

LP Duality

Primal and dual LPs. Given real numbers a_{ij}, b_i, c_j, find real numbers x_j, y_i that solve (P) and (D).

maximize	$c^T x$
subject to the constraints	$A x \leq b$
	$x \geq 0$

P: n variables, m equations

minimize	$b^T y$
subject to the constraints	$A^T y \geq c$
	$y \geq 0$

D: m variables, n equations

Duality Theorem. [Gale-Kuhn-Tucker 1951, Dantzig-von Neumann 1947]
If (P) and (D) have feasible solutions, then max = min.

LP Duality: Sensitivity Analysis

Q. How much should brewer be willing to pay (marginal price) for additional supplies of scarce resources?
A. Corn \$1, hops \$2, malt \$0.

Q. How do I compute marginal prices (dual variables)?
A. Simplex solves primal and dual simultaneously!

0	1	1/10	1/8	0	28
1	0	1/10	3/8	0	12
0	0	25/6	85/8	1	110
0	0	−1	−2	0	−800

objective row of final simplex tableaux provides optimal dual solution!

Q. New product "light beer" is proposed. It requires 2 corn, 5 hops, 24 malt. How much profit must be obtained from light beer to justify diverting resources from production of beer and ale?
A. Breakeven: 2 (\$1) + 5 (\$2) + 24 (\$0) = \$12 / barrel.

Simplex Algorithm: Running Time

Remarkable property. In practice, simplex algorithm typically terminates after at most 2(m+n) pivots.

- No pivot rule that is guaranteed to be polynomial is known.
- Most pivot rules known to be exponential (or worse) in worst-case.

Pivoting rules. Carefully balance the cost of finding an entering variable with the number of pivots needed.

brewer's problem
simplex algorithm
implementation
solving LPs
linear programming

Simplex Algorithm: Implementation Issues

To improve the bare-bones implementation

- Avoid stalling.
- Choose the pivot wisely.
- Watch for numerical stability.
- Maintain sparsity. ← requires fancy data structures
- Detect infeasiblity
- Detect unboundedness.
- Preprocess to reduce problem size.

Commercial solvers routinely solve LPs with millions of variables and tens of thousands of constraints.

LP solvers: toy problems

Use MS Excel or OR-Objects.

```
import drasys.or.mp.*;
import drasys.or.mp.lp.*;

public class LPDemo {
   public static void main(String[] args) throws Exception {

      Problem prob = new Problem(3, 2);
      prob.getMetadata().put("lp.isMaximize", "true");
      prob.newVariable("x1").setObjectiveCoefficient(13.0);
      prob.newVariable("x2").setObjectiveCoefficient(23.0);
      prob.newConstraint("corn").setRightHandSide( 480.0);
      prob.newConstraint("hops").setRightHandSide( 160.0);
      prob.newConstraint("malt").setRightHandSide(1190.0);

      prob.setCoefficientAt("corn", "x1",  5.0);
      prob.setCoefficientAt("corn", "x2", 15.0);
      prob.setCoefficientAt("hops", "x1",  4.0);
      prob.setCoefficientAt("hops", "x2",  4.0);
      prob.setCoefficientAt("malt", "x1", 35.0);
      prob.setCoefficientAt("malt", "x2", 20.0);

      DenseSimplex lp = new DenseSimplex(prob);
      System.out.println(lp.solve());
      System.out.println(lp.getSolution());
   }
}
```


LP solvers: commercial strength

AMPL. [Fourer, Gay, Kernighan] An algebraic modeling language.
CPLEX solver. Industrial strength solver.

separate data from model

```
set PROD := beer ale;
set INGR := corn hops malt;

param: profit :=
ale  13
beer 23;

param: supply :=
corn  480
hops  160
malt 1190;

param amt: ale beer :=
corn        5  15
hops        4   4
malt       35  20;
```

beer.dat

beer.mod

```
set INGR;
set PROD;
param profit {PROD};
param supply {INGR};
param amt {INGR, PROD};
var x {PROD} >= 0;

maximize total_profit:
   sum {j in PROD} x[j] * profit[j];

subject to constraints {i in INGR}:
   sum {j in PROD} amt[i,j] * x[j] <= supply[i];
```

```
[cos226:tucson] ~> ampl
AMPL Version 20010215 (SunOS 5.7)
ampl: model beer.mod;
ampl: data beer.dat;
ampl: solve;
CPLEX 7.1.0: optimal solution; objective 800
ampl: display x;
x [*] :=  ale 12  beer 28;
```

History

1939. Production, planning. [Kantorovich]
1947. Simplex algorithm. [Dantzig]
1950. Applications in many fields.
1979. Ellipsoid algorithm. [Khachian]
1984. Projective scaling algorithm. [Karmarkar]
1990. Interior point methods.

- Interior point faster when polyhedron smooth like disco ball.
- Simplex faster when polyhedron spiky like quartz crystal.

200x. Approximation algorithms, large scale optimization.

Linear programming

Linear "programming" is the process of developing a model to solve the problem at hand.

Identify variables
Define inequalities and equations

Easy part: convert to standard form

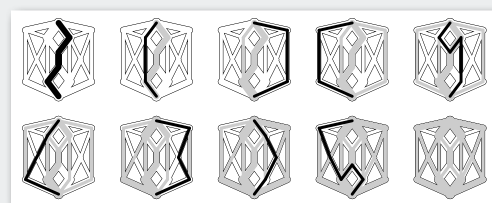
Examples:

- max flow
- assignment
- scheduling
- shortest paths
- ...

Max flow

Weighted digraph: edge weights represent capacities

- single source (no edges in)
- single sink (no edges out)



Problem: compute flow through edges

- flow less than capacity in each edge
- inflow equals outflow at each vertex (except source and sink)
- maximize flow from source to sink

Applications:

- distribution of oil through network of pipes
- distribution of goods in trucks through highways

Linear programming formulation of maxflow

Got a maxflow problem?

Approach 1: Use a specialized algorithm to solve it

- Algs in Java, Chapter 22
- vast literature
- worst-case performance close to VE
- performance on real problems little understood
- easy linear-time algorithm could exist

Approach 2: LP is a direct mathematical representation of the problem

- one variable for each edge
- inequalities saying that flow does not exceed capacity
- equalities saying that flow is preserved at vertices
- maximize outflow from source

Got an LP solver?
Maybe easier to use it than to implement specialized algorithm

Assignment Problem

Assign N jobs to N machines to minimize total cost
where c_{ij} = cost of assignment job j to machine i.

	1'	2'	3'	4'	5'
1	3	8	9	15	10
2	4	10	7	16	14
3	9	13	11	19	10
4	8	13	12	20	13
5	1	7	5	11	9

cost = 3 + 10 + 11 + 20 + 9 =53

	1'	2'	3'	4'	5'
1	3	8	9	15	10
2	4	10	7	16	14
3	9	13	11	19	10
4	8	13	12	20	13
5	1	7	5	11	9

cost = 8 + 7 + 20 + 8 + 11 = 44

Hungarian algorithm solves in time proportional to N^3
Simplex is fast enough in practice

Assignment Problem: Applications

Natural applications.

- Match jobs to machines.
- Match personnel to tasks.
- Match PU students to writing seminars.

Non-obvious applications.

- Vehicle routing.
- Signal processing.
- Virtual output queueing.
- Multiple object tracking.
- Approximate string matching.
- Enhance accuracy of solving linear systems of equations.

Assignment Problem: LP Formulation

N^2 variables
one corresponding
to each cell

2N equations
one per row
one per column

Interpretation: if $x_{ij} = 1$, then
assign job j to machine i

maximize

$$c_{11}x_{11} + c_{12}x_{12} + c_{13}x_{13} + c_{14}x_{14} + c_{15}x_{15} +$$
$$c_{21}x_{21} + c_{22}x_{22} + c_{23}x_{23} + c_{24}x_{24} + c_{25}x_{25} +$$
$$c_{31}x_{31} + c_{32}x_{32} + c_{33}x_{33} + c_{34}x_{34} + c_{35}x_{35} +$$
$$c_{41}x_{41} + c_{42}x_{42} + c_{43}x_{43} + c_{44}x_{44} + c_{45}x_{45} +$$
$$c_{51}x_{51} + c_{52}x_{52} + c_{53}x_{53} + c_{54}x_{54} + c_{55}x_{55}$$

subject
to the constraints

$$x_{11} + x_{12} + x_{13} + x_{14} + x_{15} = 1$$
$$\ldots$$
$$x_{51} + x_{52} + x_{53} + x_{54} + x_{55} = 1$$
$$x_{11} + x_{21} + x_{31} + x_{41} + x_{51} = 1$$
$$\ldots$$
$$x_{51} + x_{52} + x_{53} + x_{54} + x_{55} = 1$$
$$x_{11}, \ldots, x_{55} \geq 0$$

Theorem. [Birkhoff 1946, von Neumann 1953] All extreme points of the above polyhedron are {0-1}-valued.

Corollary. Can solve assignment problem by solving LP since LP algorithms return an optimal solution that is an extreme point.

Ultimate problem-solving model (in practice)

1. Many practical problems are easily formulated as LPs
2. Commercial solvers can solve those LPs quickly

More constraints on the problem?
- specialized algorithm may be hard to fix
- can just add more inequalities to LP

New problem?
- may not be difficult to formulate LP
- may be very difficult to develop specialized algorithm

Today's problem?
- similar to yesterday's
- edit tableau, run solver

Too slow?
- could happen
- doesn't happen

Ultimate problem-solving model (in theory)

Ultimate problem-solving model?
- Shortest path.
- Maximum flow.
- Assignment problem.
- Min cost flow.
- Multicommodity flow.
- Linear programming.
- Semidefinite programming.
- …
- Integer programming (or any NP-complete problem).

tractable

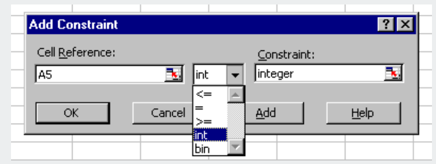
intractable (conjectured)

Does P = NP? No universal problem-solving model exists unless P = NP.

Perspective

LP is near the deep waters of NP-completeness.
- Solvable in polynomial time.
- Known for ≈ 25 years.

Integer linear programming.
- LP with integrality requirement.
- NP-hard.



An unsuspecting MBA student transitions from tractable LP to intractable ILP in a single mouse click.