

Using a Modeling Language for Efficient and Reliable Optimization in Logistics and Related Applications



Robert Fourer

AMPL Optimization

Industrial Engineering & Management Sciences,
Northwestern University

www.ampl.com — 4er@ampl.com, 4er@northwestern.edu

International Symposium on Mathematics of Logistics
Tokyo, 29 November 2011

Outline

Motivation

- ❖ The optimization modeling cycle
- ❖ Optimization modeling languages
- ❖ **AMPL** modeling language

Introductory example

Users

- ❖ Commercial
 - * Business areas
 - * Project examples: ZARA, Norske Skog
- ❖ Government
- ❖ Research & teaching

Future directions

The Optimization Modeling Cycle

Steps

- ❖ Communicate with problem owner
- ❖ Build model
- ❖ Build datasets
- ❖ Generate optimization problems
- ❖ Feed problems to solvers
- ❖ Solve
- ❖ Process results for analysis & reporting
- ❖ ***Repeat!***

Goals

- ❖ Do this quickly and reliably
- ❖ Get results before client loses interest
- ❖ Deploy for application

Optimization Modeling Languages

Two forms of an optimization problem

- ❖ Modeler's form
 - * Mathematical description, easy for people to work with
- ❖ Algorithm's form
 - * Explicit data structure, easy for solvers to compute with

Idea of a modeling language

- ❖ **A computer-readable modeler's form**
 - * You write optimization problems in a modeling language
 - * Computers translate to algorithm's form for solution

Advantages of a modeling language

- ❖ Faster modeling cycles
- ❖ More reliable modeling and maintenance

Algebraic Modeling Languages

Formulation concept

- ❖ Define data in terms of sets & parameters
 - * Analogous to database keys & records
- ❖ Define decision variables
- ❖ Minimize or maximize a function of decision variables
- ❖ Subject to equations or inequalities that constrain the values of the variables

Advantages

- ❖ Familiar
- ❖ Powerful
- ❖ Implemented

The AMPL Modeling Language

Features

- ❖ Algebraic modeling language
- ❖ Variety of data sources
- ❖ Choice of control modes
 - * Interactive modeling-building and analysis
 - * Predefined control through integrated scripting language

Advantages

- ❖ Powerful, general expressions
- ❖ Natural, easy-to-learn design
- ❖ Efficient processing

Solvers for AMPL

Features

- ❖ Support for all popular optimization engines
 - * **Gurobi** used in the following examples
- ❖ Detection of all supported problem types
- ❖ Access to all algorithmic options

Introductory Example

Multicommodity transportation

- ❖ Products available at factories
- ❖ Products needed at stores
- ❖ Plan shipments at lowest cost

Practical restrictions

- ❖ Cost has fixed and variables parts
- ❖ Shipments cannot be too small
- ❖ Factories cannot serve too many stores

. . . require a mixed-integer solver like Gurobi

Multicommodity Transportation

Given

- O Set of origins (factories)
- D Set of destinations (stores)
- P Set of products

and

- a_{ip} Amount available, for each $i \in O$ and $p \in P$
- b_{jp} Amount required, for each $j \in D$ and $p \in P$
- l_{ij} Limit on total shipments, for each $i \in O$ and $j \in D$
- c_{ijp} Shipping cost per unit, for each $i \in O, j \in D, p \in P$
- d_{ij} Fixed cost for shipping any amount from $i \in O$ to $j \in D$
- s Minimum total size of any shipment
- n Maximum number of destinations served by any origin

Multicommodity Transportation

Mathematical Formulation

Determine

X_{ijp} Amount of each $p \in P$ to be shipped from $i \in O$ to $j \in D$

Y_{ij} 1 if any product is shipped from $i \in O$ to $j \in D$
0 otherwise

to minimize

$$\sum_{i \in O} \sum_{j \in D} \sum_{p \in P} c_{ijp} X_{ijp} + \sum_{i \in O} \sum_{j \in D} d_{ij} Y_{ij}$$

Total variable cost plus total fixed cost

Mathematical Formulation

Subject to

$$\sum_{j \in D} X_{ijp} \leq a_{ip} \quad \text{for all } i \in O, p \in P$$

Total shipments of product p out of origin i
must not exceed availability

$$\sum_{i \in O} X_{ijp} = b_{jp} \quad \text{for all } j \in D, p \in P$$

Total shipments of product p into destination j
must satisfy requirements

Mathematical Formulation

and subject to

$$\sum_{p \in P} X_{ijp} \leq l_{ij} Y_{ij} \quad \text{for all } i \in O, j \in D$$

When there are shipments from origin i to destination j , the total may not exceed the limit, and Y_{ij} must be 1

$$\sum_{p \in P} X_{ijp} \geq s Y_{ij} \quad \text{for all } i \in O, j \in D$$

When there are shipments from origin i to destination j , the total amount of shipments must be at least s

$$\sum_{j \in D} Y_{ij} \leq n \quad \text{for all } i \in O$$

Number of destinations served by origin i must be at most n

AMPL Formulation

Symbolic data

```
set ORIG;    # origins
set DEST;    # destinations
set PROD;    # products

param supply {ORIG,PROD} >= 0; # availabilities at origins
param demand {DEST,PROD} >= 0; # requirements at destinations
param limit  {ORIG,DEST} >= 0; # capacities of links

param vcost  {ORIG,DEST,PROD} >= 0; # variable shipment cost
param fcost  {ORIG,DEST} > 0;      # fixed usage cost

param minload >= 0;                # minimum shipment size
param maxserve integer > 0;       # maximum destinations served
```

AMPL Formulation

Symbolic model: variables and objective

```
var Trans {ORIG,DEST,PROD} >= 0;    # actual units to be shipped
var Use {ORIG, DEST} binary;        # 1 if link used, 0 otherwise

minimize Total_Cost:
    sum {i in ORIG, j in DEST, p in PROD} vcost[i,j,p] * Trans[i,j,p]
+ sum {i in ORIG, j in DEST} fcost[i,j] * Use[i,j];
```

$$\sum_{i \in O} \sum_{j \in D} \sum_{p \in P} c_{ijp} X_{ijp} + \sum_{i \in O} \sum_{j \in D} d_{ij} Y_{ij}$$

Multicommodity Transportation

AMPL Formulation

Symbolic model: constraint

```
subject to Supply {i in ORIG, p in PROD}:  
    sum {j in DEST} Trans[i,j,p] <= supply[i,p];
```

$$\sum_{j \in D} X_{ijp} \leq a_{ip}, \text{ for all } i \in O, p \in P$$

AMPL Formulation

Symbolic model: constraints

```
subject to Supply {i in ORIG, p in PROD}:  
    sum {j in DEST} Trans[i,j,p] <= supply[i,p];  
  
subject to Demand {j in DEST, p in PROD}:  
    sum {i in ORIG} Trans[i,j,p] = demand[j,p];  
  
subject to Multi {i in ORIG, j in DEST}:  
    sum {p in PROD} Trans[i,j,p] <= limit[i,j] * Use[i,j];  
  
subject to Min_Ship {i in ORIG, j in DEST}:  
    sum {p in PROD} Trans[i,j,p] >= minload * Use[i,j];  
  
subject to Max_Serve {i in ORIG}:  
    sum {j in DEST} Use[i,j] <= maxserve;
```

AMPL Formulation

Explicit data independent of symbolic model

```
set ORIG := GARY CLEV PITT ;
set DEST := FRA DET LAN WIN STL FRE LAF ;
set PROD := bands coils plate ;

param supply (tr):  GARY  CLEV  PITT :=
                    bands  400   700   800
                    coils  800  1600  1800
                    plate  200   300   300 ;

param demand (tr):
                    FRA  DET  LAN  WIN  STL  FRE  LAF :=
bands  300  300  100  75  650  225  250
coils  500  750  400  250  950  850  500
plate  100  100   0   50  200  100  250 ;

param limit default 625 ;
param minload := 375 ;
param maxserve := 5 ;
```

AMPL Formulation

Explicit data (continued)

```
param vcost :=
  [*,*,bands]:  FRA  DET  LAN  WIN  STL  FRE  LAF :=
    GARY    30   10   8   10   11   71   6
    CLEV    22   7   10   7   21   82  13
    PITT    19  11  12  10  25   83  15
  [*,*,coils]:  FRA  DET  LAN  WIN  STL  FRE  LAF :=
    GARY    39  14  11  14  16  82   8
    CLEV    27   9  12   9  26  95  17
    PITT    24  14  17  13  28  99  20
  [*,*,plate]:  FRA  DET  LAN  WIN  STL  FRE  LAF :=
    GARY    41  15  12  16  17  86   8
    CLEV    29   9  13   9  28  99  18
    PITT    26  14  17  13  31 104  20 ;
param fcost:   FRA  DET  LAN  WIN  STL  FRE  LAF :=
  GARY  3000 1200 1200 1200 2500 3500 2500
  CLEV  2000 1000 1500 1200 2500 3000 2200
  PITT  2000 1200 1500 1500 2500 3500 2200 ;
```

Multicommodity Transportation

AMPL Solution

Model + data = problem instance to be solved

```
ampl: model multmipG.mod;
ampl: data multmipG.dat;
ampl: option solver gurobi;
ampl: solve;

Gurobi 4.6.0: optimal solution; objective 235625
404 simplex iterations
45 branch-and-cut nodes

ampl: display Use;

Use [*,*]

:      DET FRA FRE LAF LAN STL WIN      :=
CLEV   1   1   1   0   1   1   0
GARY   0   0   0   1   0   1   1
PITT   1   1   1   1   0   1   0
;
```

Multicommodity Transportation

AMPL Solution

Examine results

```
AMPL: display {i in ORIG, j in DEST}
AMPL?   sum {p in PROD} Trans[i,j,p] / limit[i,j];

:      DET    FRA    FRE    LAF    LAN    STL    WIN    :=
CLEV   1      0.6    0.88   0     0.8    0.88   0
GARY   0      0      0     0.64   0     1      0.6
PITT   0.84   0.84   1     0.96   0     1      0
;

AMPL: display Max_Serve.body;

CLEV   5
GARY   3
PITT   5
;

AMPL: display TotalCost,
AMPL?   sum {i in ORIG, j in DEST} fcost[i,j] * Use[i,j];

TotalCost = 235625
sum {i in ORIG, j in DEST} fcost[i,j]*Use[i,j] = 27600
```

AMPL “Sparse” Network

Indexed over sets of pairs and triples

```
set ORIG;    # origins
set DEST;    # destinations
set PROD;    # products

set SHIP within {ORIG,DEST,PROD};
           # (i,j,p) in SHIP ==> can ship p from i to j
set LINK = setof {(i,j,p) in SHIP} (i,j);
           # (i,j) in LINK ==> can ship some products from i to j

.....

var Trans {SHIP} >= 0;    # actual units to be shipped
var Use {LINK} binary;    # 1 if link used, 0 otherwise

minimize Total_Cost:
    sum {(i,j,p) in SHIP} vcost[i,j,p] * Trans[i,j,p]
+ sum {(i,j) in LINK} fcost[i,j] * Use[i,j];
```

Multicommodity Transportation

AMPL “Sparse” Network

Constraint for dense case

```
subject to Supply {i in ORIG, p in PROD}:  
    sum {j in DEST} Trans[i,j,p] <= supply[i,p];
```

Constraint for sparse case

```
subject to Supply {i in ORIG, p in PROD}:  
    sum {(i,j,p) in SHIP} Trans[i,j,p] <= supply[i,p];
```

AMPL “Sparse” Network

All constraints

```
subject to Supply {i in ORIG, p in PROD}:
    sum {(i,j,p) in SHIP} Trans[i,j,p] <= supply[i,p];

subject to Demand {j in DEST, p in PROD}:
    sum {(i,j,p) in SHIP} Trans[i,j,p] = demand[j,p];

subject to Multi {i in ORIG, j in DEST}:
    sum {(i,j,p) in SHIP} Trans[i,j,p] <= limit[i,j] * Use[i,j];

subject to Min_Ship {i in ORIG, j in DEST}:
    sum {(i,j,p) in SHIP} Trans[i,j,p] >= minload * Use[i,j];

subject to Max_Serve {i in ORIG}:
    sum {(i,j) in LINK} Use[i,j] <= maxserve;
```

Multicommodity Transportation

AMPL “Sparse” Network

1st dataset: shipments allowed

```
set SHIP :=  
  (*,*,bands):  FRA  DET  LAN  WIN  STL  FRE  LAF :=  
    GARY      +   +   +   +   +   -   +  
    CLEV      +   -   +   -   +   +   +  
    PITT      -   +   +   +   +   +   +  
  (*,*,coils):  FRA  DET  LAN  WIN  STL  FRE  LAF :=  
    GARY      +   +   +   +   +   +   -  
    CLEV      +   +   -   +   +   +   +  
    PITT      +   +   +   +   +   +   +  
  (*,*,plate): FRA  DET  LAN  WIN  STL  FRE  LAF :=  
    GARY      +   +   -   +   +   -   +  
    CLEV      +   +   +   +   +   +   +  
    PITT      -   +   +   -   +   +   + ;
```

AMPL “Sparse” Network

2nd dataset: shipments allowed

```
set SHIP :=  
  (*,*,bands):  FRA  DET  LAN  WIN  STL  FRE  LAF :=  
    GARY      +   +   +   +   +   -   -  
    CLEV      -   +   +   -   +   +   +  
    PITT      +   -   +   +   +   +   +  
  (*,*,coils):  FRA  DET  LAN  WIN  STL  FRE  LAF :=  
    GARY      +   +   +   +   +   +   +  
    CLEV      +   +   -   +   +   +   +  
    PITT      +   +   +   +   +   +   +  
  (*,*,plate): FRA  DET  LAN  WIN  STL  FRE  LAF :=  
    GARY      -   +   +   +   +   -   +  
    CLEV      +   +   +   +   +   +   +  
    PITT      +   +   -   -   +   +   + ;
```

Multicommodity Transportation

AMPL “Sparse” Network

Same model, different data

```
AMPL> model multmipT.mod;
AMPL> data multmipT1.dat;
AMPL> solve;
Gurobi 4.6.0: optimal solution; objective 247725
108 simplex iterations
13 branch-and-cut nodes
AMPL> reset data;
AMPL> data multmipT2.dat;
AMPL> solve;
Gurobi 4.6.0: optimal solution; objective 237775
79 simplex iterations
AMPL>
```

Multicommodity Transportation

AMPL Scripting

Script to test sensitivity to service limit

```
model multmipG.mod;
data multmipG.dat;

option solver gurobi;

for {m in 7..1 by -1} {
    let maxserve := m;
    solve;
    if solve_result = 'infeasible' then break;
    display maxserve, Max_Serve.body;
}
```

Multicommodity Transportation

AMPL Scripting

Run showing sensitivity to serve limit

```
ampl: include multmipServ.run;

Gurobi 4.6.0: optimal solution; objective 233150
maxserve = 7
CLEV 5   GARY 3   PITT 6

Gurobi 4.6.0: optimal solution; objective 233150
maxserve = 6
CLEV 5   GARY 3   PITT 6

Gurobi 4.6.0: optimal solution; objective 235625
maxserve = 5
CLEV 5   GARY 3   PITT 5

Gurobi 4.5.0: infeasible
```

AMPL Scripting

Script to generate n best solutions

```
param nSols default 0;
param maxSols;

model multmipG.mod;
data multmipG.dat;

set USED {1..nSols} within {ORIG,DEST};

subject to exclude {k in 1..nSols}:
    sum {(i,j) in USED[k]} (1-Use[i,j]) +
    sum {(i,j) in {ORIG,DEST} diff USED[k]} Use[i,j] >= 1;

option solver gurobi;

repeat {
    solve;
    display Use;
    let nSols := nSols + 1;
    let USED[nSols] := {i in ORIG, j in DEST: Use[i,j] > .5};
} until nSols = maxSols;
```

AMPL Scripting

Run showing 3 best solutions

```
ampl: include multmipBest.run;
Gurobi 4.6.0: optimal solution; objective 235625
:      DET FRA FRE LAF LAN STL WIN      :=
CLEV   1   1   1   0   1   1   0
GARY   0   0   0   1   0   1   1
PITT   1   1   1   1   0   1   0 ;
Gurobi 4.6.0: optimal solution; objective 237125
:      DET FRA FRE LAF LAN STL WIN      :=
CLEV   1   1   1   1   0   1   0
GARY   0   0   0   1   0   1   1
PITT   1   1   1   0   1   1   0 ;
Gurobi 4.6.0: optimal solution; objective 238225
:      DET FRA FRE LAF LAN STL WIN      :=
CLEV   1   0   1   0   1   1   1
GARY   0   1   0   1   0   1   0
PITT   1   1   1   1   0   1   0 ;
```

AMPL's Users

Business

- ❖ Customer areas
- ❖ Project examples
 - * ZARA (clothing retailer)
 - * Norske Skog (paper manufacturer)

Government

Academic

AMPL's Users

Business Customer Areas

Transportation

- ❖ Air, rail, truck

Production

- ❖ Planning
 - * steel
 - * automotive
- ❖ Supply chain
 - * consumer products
 - * industrial products

Finance

- ❖ Investment banking
- ❖ Insurance

Natural resources

- ❖ Electric power
- ❖ Gas distribution
- ❖ Mining

Information technology

- ❖ Telecommunications
- ❖ Internet services

Consulting practices

- ❖ Management
- ❖ Industrial engineering

AMPL's Users

Business Customer Examples

Two award-winning projects

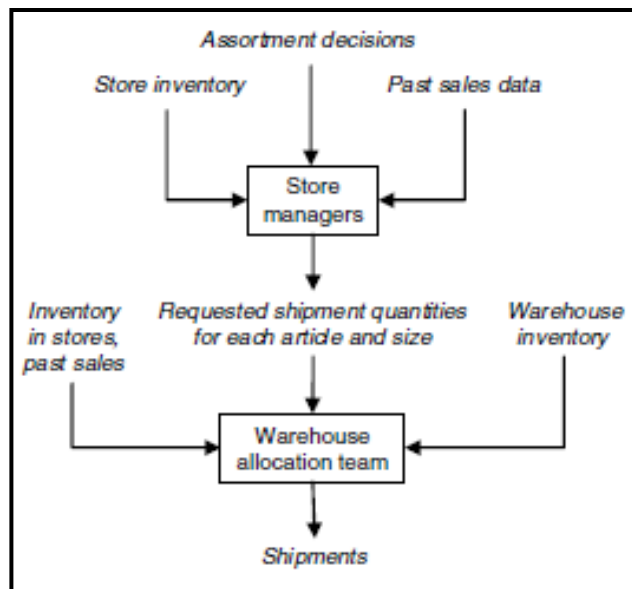
- ❖ ZARA (clothing retailer)
- ❖ Norske Skog (paper manufacturer)

*. . . finalists for INFORMS Edelman Award
for practice of Operations Research*

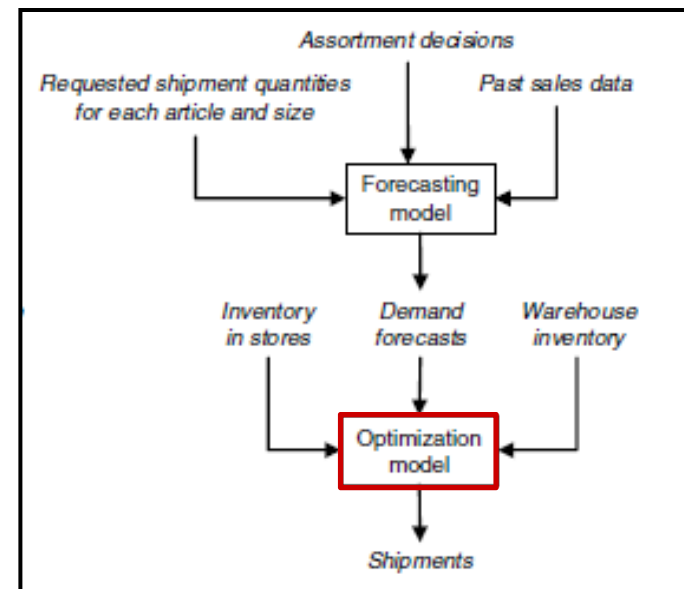
ZARA: Clothing Retailer

Optimization of worldwide shipments

❖ Legacy process



New Process



- ❖ Piecewise-linear AMPL model with integer variables
- ❖ Run once for each product each week
- ❖ Decides how much of each size to ship to each store
- ❖ Increases sales 3-4%

AMPL's Users

ZARA's Formulation

Given

$$S = S^+ \cup S^-$$

Set of sizes partitioned into major & regular sizes

J Set of stores

and

W_s Inventory of size s available at the warehouse

I_{sj} Inventory of size s available in store j

P_j Selling price in store j

K Aggressiveness factor (value of inventory remaining in the warehouse after the current shipments)

λ_{sj} Demand rate for size s in store j

N_{sj} Approximation set for size s in the inventory-to-sales function approximation for store j

AMPL's Users

ZARA's Formulation

Determine

x_{sj} (integer) shipment quantity of each size $s \in S$
to each store $j \in J$ for the current replenishment period

z_j (≥ 0) approximate expected sales across all sizes
in each store $j \in J$ for the current period

to maximize

$$\sum_{j \in J} P_j z_j + K \sum_{s \in S} (W_s - \sum_{j \in J} x_{sj})$$

Expected sales plus value of items remaining in warehouse

subject to

$$\sum_{j \in J} x_{sj} \leq W_s \quad \text{for all } s \in S$$

Total shipments of size s
must not exceed amount available in warehouse

AMPL's Users

ZARA's Formulation

and subject to

$$\begin{aligned} z_j &\leq (\sum_{s \in S^+} \lambda_{sj})y_j + \sum_{s \in S^-} \lambda_{sj}v_{sj} && \text{for all } j \in J \\ y_j &\leq a_i \lambda_{sj} (I_{sj} + x_{sj} - i) + b_i \lambda_{sj} && \text{for all } j \in J, s \in S^+, i \in N_{sj} \\ v_{sj} &\leq a_i \lambda_{sj} (I_{sj} + x_{sj} - i) + b_i \lambda_{sj} && \text{for all } j \in J, s \in S^-, i \in N_{sj} \\ v_{sj} &\leq y_j && \text{for all } j \in J, s \in S^- \end{aligned}$$

Relationship between sales
and store inventory after shipments

AMPL's Users

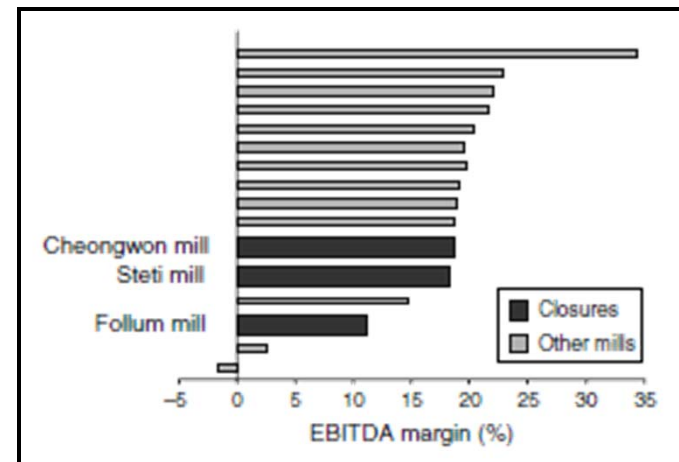
Norske Skog: Paper Manufacturer

Optimization of production and distribution

- ❖ Australasia
- ❖ Europe
 - * 640 binary variables
 - * 524,000 continuous variables
 - * 33,000 constraints

Optimization of shutdown decisions worldwide

- ❖ Multiple scenarios
- ❖ Numerous sensitivity analyses
- ❖ **Key role of AMPL models**
 - * Implemented in a few weeks
 - * Modified to analyze alternatives
 - * Run interactively at meetings



AMPL's Users

Norske Skog's Formulation

Given sets

- N Number of mills
- M_n Set of machines at mill N
- M Number of paper machines
- J Number of products
- L Number of raw material sources
- R Number of raw materials
- K Number of customers
- P Number of recipes

Norske Skog's Formulation

Given capital parameters

- l_n fixed cost of mill n running for one period
(excluding machine fixed costs)
- f_m fixed cost of machine m running for one period
- θ_m proportion of fixed running costs saved from
a temporary shutdown on machine m
- q_m minimum time that machine m must be shut
before savings accrue
- ϕ_m amortized cost of a permanent closure of machine m

Norske Skog's Formulation

and operating parameters

- g_{mjk} variable freight cost for shipping product j from machine m to customer k
- a_{mjp} capacity of machine m making product j using recipe p
- c_{mjp} variable cost incurred by producing one tonne of product j using recipe p on machine m
- h_{mjrp} tonnes of raw material r required to make one tonne of product j using recipe p on machine m
- π_{mrl} procurement, transportation, and process cost of raw material r from source l for machine m
- W_{rl} supply of raw material r at source l
- d_{jk} demand for product j by customer k
- s_{jk} sales price for product j by customer k

Norske Skog's Formulation

Make capital decisions

- δ_n 1 if mill n closes, 0 otherwise
- μ_m 1 if machine m shuts down permanently, 0 otherwise
- u_m time that machine m has been shut down
- ξ_m 1 if machine m has been shut down long enough
to accrue savings, 0 otherwise
- v_m time that qualifies for savings on machine m

Norske Skog's Formulation

and operating decisions

- x_{mjp} tonnes of product j made on machine m
using recipe p
- y_{mjk} tonnes of product j made on machine m and
delivered to customer k
- w_{mrl} tonnes of raw material r from source l
used by machine m
- σ_{mp} 1 if recipe p is used on machine m , 0 otherwise

Norske Skog's Formulation

Maximize

$$\begin{aligned} & \sum_{m=1}^M \sum_{j=1}^J (\sum_{k=1}^K (s_{jk} - g_{mjk}) y_{mjk} - \sum_{p=1}^P c_{mjp} x_{mjp}) \\ & - \sum_{m=1}^M \sum_{l=1}^L \sum_{r=1}^R \pi_{mrl} w_{mrl} \\ & + \sum_{m=1}^M \theta_m f_m v_m \\ & - \sum_{n=1}^N (l_n (1 - \delta_n) + \lambda_n \delta_n) \\ & - \sum_{m=1}^M (f_m (1 - \mu_m) + \phi_m \mu_m) \end{aligned}$$

Income from sales,
minus raw material, production and distribution costs,
plus savings from shutdowns,
minus fixed operating and shutdown costs

Norske Skog's Formulation

Subject to

$$\sum_{j=1}^J \sum_{p=1}^P \frac{x_{mjp}}{a_{mjp}} = 1 - u_m \quad \text{for } m = 1, \dots, M$$

Capacity used equals capacity available

$$\sum_{k=1}^K y_{mjk} = \sum_{p=1}^P x_{mjp} \quad \text{for } j = 1, \dots, J, m = 1, \dots, M$$

Amounts produced equal amounts shipped

$$\sum_{m=1}^M y_{mjk} \leq d_{jk} \quad \text{for } j = 1, \dots, J, k = 1, \dots, K$$

Amounts produced do not exceed demand

$$\sum_{j=1}^J \sum_{p=1}^P h_{mjrp} x_{mjp} = \sum_{l=1}^L w_{mrl} \quad \text{for } m = 1, \dots, M, r = 1, \dots, R$$

Raw material used equals raw material purchased

$$\sum_{m=1}^M w_{mrl} \leq W_{rl} \quad \text{for } l = 1, \dots, L, r = 1, \dots, R$$

Raw material purchased does not exceed amount available

Norske Skog's Formulation

and subject to

$$\sum_{p=1}^P \sigma_{mp} = 1 - \mu_m \quad \text{for } m = 1, \dots, M$$

$$x_{mjp} \leq a_{mjp} \sigma_{mjp} \quad \text{for } j = 1, \dots, J, m = 1, \dots, M, p = 1, \dots, P$$

$$\delta_n \leq \mu_m \quad \text{for } m \in M_n, n = 1, \dots, N$$

$$v_m \leq \xi_m \quad \text{for } m = 1, \dots, M$$

$$v_m \leq 1 - \mu_m \quad \text{for } m = 1, \dots, M$$

$$v_m \leq u_m - q_m \xi_m \quad \text{for } m = 1, \dots, M$$

Definitions of zero-one variables

AMPL's Users

Government Customers

Financial agencies

- ❖ United States
- ❖ Canada
- ❖ Sweden

U.S. departments

- ❖ Census Bureau
- ❖ Army Corps of Engineers

U.S. research centers

- ❖ Argonne National Laboratory
- ❖ Sandia National Laboratories
- ❖ Lawrence Berkeley Laboratory

AMPL's Users

Academic Customers

Research

- ❖ Over 250 university installations worldwide
- ❖ Nearly 1000 citations in scientific papers
 - * engineering, science, economics, management

Teaching

- ❖ Linear & nonlinear optimization
 - * Graph optimization
 - * Stochastic programming
- ❖ Operations Research
- ❖ Specialized courses
 - * Supply chain modeling
 - * Electric power system planning
 - * Transportation logistics
 - * Communication network design & algorithms

AMPL's Users

Free AMPL for Courses

Streamlined for quick setup

- ❖ One-page application form for each course offering
- ❖ AMPL & solvers in one compressed file for each platform
 - * *Accepts problems of any size*
- ❖ Freely install on any computer supportin the course
- ❖ Freely distribute to students for their own computers
 - * Times out after your specified course end date

Includes top-quality solvers

- ❖ Gurobi, CPLEX, KNIRO, MINOS, SNOPT

Available now

- ❖ More information: www.ampl.com/courses.html
- ❖ Application form: www.ampl.com/AMPLforCourses.pdf

Future Directions for AMPL

Core development

- ❖ Further set operations
- ❖ Enhanced scripting
- ❖ More natural formulations

Interface development

- ❖ Integrated development environment
- ❖ Callable version
 - * embedding in large applications
 - * deployment to end users
- ❖ Support of new solver types
- ❖ Extended database support

Business expansion

- ❖ Cloud computing services

Readings

- ❖ R. Fourer, “Modeling Languages versus Matrix Generators for Linear Programming.” *ACM Transactions on Mathematical Software* **9** (1983) 143–183.
- ❖ R. Fourer, D.M. Gay, B.W. Kernighan, “A Modeling Language for Mathematical Programming.” *Management Science* **36** (1990) 519–554.
- ❖ R. Fourer, D.M. Gay, B.W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press, Belmont, CA (first edition 1993, second edition 2003).
- ❖ R. Fourer, “Algebraic Modeling Languages for Optimization.” Forthcoming in Saul I. Gass and Michael C. Fu (eds.), *Encyclopedia of Operations Research and Management Science*, Springer (2012).
- ❖ G. Everett, A. Philpott, K. Vatn, R. Gjessing, “Norske Skog Improves Global Profitability Using Operations Research.” *Interfaces* **40**, 1 (Jan–Feb 2010) 58–70.
- ❖ F. Caro, J. Gallien, M. Díaz, J. García, J.M. Corredoira, M. Montes, J.A. Ramos, J. Correa, “Zara Uses Operations Research to Reengineer Its Global Distribution Process.” *Interfaces* **40**, 1 (Jan–Feb 2010) 71–84.