Generalized Assignment Problem (GAP)

This Hypercard stack was prepared by:
Dennis L. Bricker,
Dept. of Industrial Engineering,
University of Iowa,
Iowa City, Iowa 52242
e-mail: dbricker@icaen.uiowa.edu

author

In the classical **Assignment Problem**, we must find the least-cost, one-to-one assignment of **n** jobs to **n** machines:

AP

Minimize $\sum_{i=1}^{m}\sum_{j=1}^{n} C_{ij}X_{ij}$

subject to

$\sum_{j=1}^{n} X_{ij} = 1$ **for all i** ← *Each machine is assigned only one job!*

$\sum_{i=1}^{m} X_{ij} = 1$ **for all j** ← *Each job is assigned to one machine*

$X_{ij}\ \varepsilon\ \{0,1\}$ **for all i & j**

Suppose that job #j requires a_{ij} units of processing time on machine #i, which has a total of b_i time units available.

Then the constraint $\sum_{j=1}^{n} X_{ij} = 1$ **for all i**

which says that exactly one job is assigned to machine #i

is replaced by $\sum_{j=1}^{n} a_{ij} X_{ij} \le b_i$ **for all i**

which says that the total time required to process the jobs assigned to machine #i cannot exceed b_i, the time available.

Thus, the Generalized Assignment Problem is

GAP

Minimize $\sum_{i=1}^{m}\sum_{j=1}^{n} C_{ij}X_{ij}$

subject to

$\sum_{j=1}^{n} a_{ij}X_{ij} \le b_i$ **for all i** ← *no more than the available capacity of machine i is used*

$\sum_{i=1}^{m} X_{ij} = 1$ **for all j** ← *each job must be assigned to exactly one machine*

$X_{ij}\ \varepsilon\ \{0,1\}$ **for all i & j**

Example:

4 jobs must be processed, each on one of 3 machines which are available

Data:

machine \ job:	1	2	3	4	hrs available
1	6	4	9	7	15
2	3	2	6	4	8
3	5	4	7	6	12

Time Req'd (hrs)

Data:

The cost of processing a job varies, according to the machine to which it is assigned:

machine \ job:	1	2	3	4
1	18	15	20	16
2	10	9	15	10
3	12	12	18	13

Cost

Minimize $18X_{11}+15X_{12}+20X_{13}+16X_{14}+10X_{21}+9X_{22}+15X_{23}+10X_{24}+12X_{31}+12X_{32}+18X_{33}+13X_{34}$

subject to

$$
\begin{aligned}
&6X_{11}+4X_{12}+9X_{13}+7X_{14} \le 15\\
&3X_{21}+2X_{22}+6X_{23}+4X_{24} \le 8\\
&5X_{31}+4X_{32}+7X_{33}+6X_{34} \le 12\\
&X_{11}+X_{21}+X_{31} = 1\\
&X_{12}+X_{22}+X_{32} = 1\\
&X_{13}+X_{23}+X_{33} = 1\\
&X_{14}+X_{24}+X_{34} = 1
\end{aligned}
$$

$X_{ij}\ \varepsilon\ \{0,1\}$ **for all i & j**

Lagrangian Relaxation:

Lagrangian relaxation is based upon the fact that, if we relax (ignore) one set of constraints of the GAP, the problem that remains is much easier to solve.

The assignment problem is an LP with the property that every basic solution is integer... so that AP is very easy to solve!

In the case of the Generalized Assignment Problem, not all basic solutions are integer-valued, so that solving the problem with the integer restrictions ignored, i.e., as an LP, generally yields a non-integer solution

(i.e., jobs may be split between two or more machines.)

If we relax the machine availability constraints, however, the problem that remains is rather trivial to solve:

$$\begin{aligned} &\textbf{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} C_{ij}X_{ij} \\ &\textbf{subject to} \\ &\quad \sum_{i=1}^{m} X_{ij} = 1 \quad \textbf{for all } j \longleftarrow \textit{each job must be assigned to exactly one machine} \\ &\quad X_{ij} \in \{0,1\} \quad \textbf{for all } i \ \& \ j \end{aligned}$$

In Lagrangian Relaxation, we assign a "Lagrange Multiplier" to every relaxed constraint, and shift the constraint to the objective:

$$\begin{aligned} &\textbf{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} C_{ij}X_{ij} + \sum_{i=1}^{m} v_i\left(\sum_{j=1}^{n} a_{ij}X_{ij} - b_i\right) \\ &\textbf{subject to} \\ &\quad \sum_{i=1}^{m} X_{ij} = 1 \quad \textbf{for all } j \longleftarrow \textit{each job must be assigned to exactly one machine} \\ &\quad X_{ij} \in \{0,1\} \quad \textbf{for all } i \ \& \ j \end{aligned}$$

This simplifies to

$$\left(-\sum_{i=1}^{m} v_i b_i\right) + \sum_{j=1}^{n}\left\{ \textbf{minimum} \sum_{i=1}^{m}\left(C_{ij} + a_{ij}v_i\right)X_{ij}\right\}$$

$$\begin{aligned} &\textbf{subject to} \\ &\quad \sum_{i=1}^{m} X_{ij} = 1 \quad \textbf{for all } j \longleftarrow \textit{each job must be assigned to exactly one machine} \\ &\quad X_{ij} \in \{0,1\} \quad \textbf{for all } i \ \& \ j \end{aligned}$$

That is, we must assign every job to ***exactly one*** machine, but we can ignore the machine capacity constraints.

The solution is obviously to assign each job to the machine which can process it most cheaply.

Consider our example GAP:

Generalized Assignment Problem

*** Example GAP ***

Costs

Machine i	Job 1	Job 2	Job 3	Job 4
1	18	15	20	16
2	10	9	15	10
3	12	12	18	13

Lambda = 0.75
Target Z* = 50 ← *target value & stepsize parameter for subgradient optimization of dual*

Resources Used

Machine	Jobs				Available
i	1	2	3	4	b
1	6	4	9	7	15
2	3	2	6	4	8
3	5	4	7	6	12

Iteration # 1

Multiplier vector U = 0 0 0
Objective function of relaxation:

machine \ job	1	2	3	4
1	18	15	20	16
2	10	9	15	10
3	12	12	18	13

in the case of each job, it can be done most cheaply by machine #2

Dual value is 44
Variables selected from GUB sets are:
2 2 2 2
Resources used are: 0 15 0, (Available: 15 8 12)
Subgradient of Dual Objective is ¯15 7 ¯12
Stepsize is 0.0918367

Iteration # 2

Iteration # 2

Multiplier vector U = 0 0.642857 0
Objective function of relaxation:

machine \ job	1	2	3	4
1	18	15	20	16
2	11.9286	10.2857	18.8571	12.5714
3	12	12	18	13

Dual value is 47.6429
Variables selected from GUB sets are:
2 2 3 2
Resources used are: 0 9 7, (Available: 15 8 12)
Subgradient of Dual Objective is ¯15 1 ¯5
Stepsize is 1.76786

Iteration # 3

Multiplier vector U = 0 2.41071 0
Objective function of relaxation:

machine \ job	1	2	3	4
1	18	15	20	16
2	17.2321	13.8214	29.4643	19.6429
3	12	12	18	13

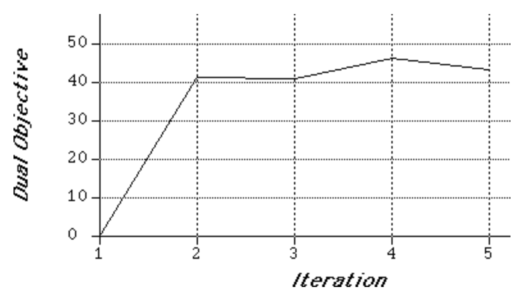
Dual value is 35.7143
Variables selected from GUB sets are:
3 3 3 3
Resources used are: 0 0 22, (Available: 15 8 12)
Subgradient of Dual Objective is ¯15 ¯8 10
Stepsize is 0.065331

Iteration # 4

Multiplier vector U = 0 1.88807 0.65331
Objective function of relaxation:

machine \ job	1	2	3	4
1	18	15	20	16
2	15.6642	12.7761	26.3284	17.5523
3	15.2666	14.6132	22.5732	16.9199

Dual value is 41.0984
Variables selected from GUB sets are:
3 2 1 1
Resources used are: 16 2 5, (Available: 15 8 12)
Subgradient of Dual Objective is 1 ¯6 ¯7
Stepsize is 0.07763



Integrality Property

A Lagrangian relaxation exhibits the "Integrality Property" if, when the integer restriction is relaxed, the resulting problem will still possess an integer solution.

The optimal value of the associated Lagrangian dual problem, if the Lagrangian relaxation has the integrality property, is identical to that of the LP relaxation.

$$\left(-\sum_{i=1}^{m} v_i b_i\right) + \sum_{j=1}^{n} \left\{ \text{minimum} \sum_{i=1}^{m} \left(C_{ij} + a_{ij} v_i\right) X_{ij} \right\}$$

subject to

$$\sum_{i=1}^{m} X_{ij} = 1 \quad \text{for all } j$$

$$\cancel{X_{ij} \in \{0,1\} \quad \text{for all } i \text{ \& } j}$$

$$0 \le X_{ij} \le 1 \quad \text{for all } i \text{ \& } j$$

This Lagrangian relaxation does exhibit the "Integrality Property".

Consider again the original GAP:

$$\text{Minimize } \sum_{i=1}^{m}\sum_{j=1}^{n} C_{ij}X_{ij}$$

subject to

$$\sum_{j=1}^{n} \mathbf{a}_{ij}X_{ij} \le \mathbf{b}_i \quad \text{for all } i$$

no more than the available capacity of machine i is used

$$\sum_{i=1}^{m} X_{ij} = 1 \quad \text{for all } j$$

each job must be assigned to exactly one machine

$$X_{ij}\ \varepsilon \left\{0,1\right\} \quad \text{for all } i \ \& \ j$$

Before, we relaxed the machine resource constraints. Suppose that we relax instead the Multiple-Choice (GUB) constraints: $\sum_{i=1}^{m} X_{ij} = 1$ for all j

This gives us the Lagrangian Relaxation:

$$\text{Minimize } \sum_{i=1}^{m}\sum_{j=1}^{n} C_{ij}X_{ij} + \sum_{j=1}^{n}\mathbf{u}_j\left(\sum_{i=1}^{m}X_{ij} - 1\right)$$

subject to

$$\sum_{j=1}^{n} \mathbf{a}_{ij}X_{ij} \le \mathbf{b}_i \quad \text{for all } i$$

no more than the available capacity of machine i is used

$$X_{ij}\ \varepsilon \left\{0,1\right\} \quad \text{for all } i \ \& \ j$$

The Lagrangian Relaxation

$$\text{Minimize } \sum_{i=1}^{m}\sum_{j=1}^{n} C_{ij}X_{ij} + \sum_{j=1}^{n}\mathbf{u}_j\left(\sum_{i=1}^{m}X_{ij} - 1\right)$$

subject to

$$\sum_{j=1}^{n} \mathbf{a}_{ij}X_{ij} \le \mathbf{b}_i \quad \text{for all } i$$

$$X_{ij}\ \varepsilon \left\{0,1\right\} \quad \text{for all } i \ \& \ j$$

is rewritten

$$\left(-\sum_{j=1}^{n}\mathbf{u}_j\right) + \text{Minimize } \sum_{i=1}^{m}\sum_{j=1}^{n}\left(C_{ij} + \mathbf{u}_j\right)X_{ij}$$

subject to

$$\sum_{j=1}^{n} \mathbf{a}_{ij}X_{ij} \le \mathbf{b}_i \quad \text{for all } i$$

$$X_{ij}\ \varepsilon \left\{0,1\right\} \quad \text{for all } i \ \& \ j$$

This separates into one knapsack problem for each machine:

$$\text{Minimize } \sum_{j=1}^{n}\left(C_{ij} + \mathbf{u}_j\right)X_{ij}$$

subject to

$$\sum_{j=1}^{n} \mathbf{a}_{ij}X_{ij} \le \mathbf{b}_i$$

$$X_{ij}\ \varepsilon \left\{0,1\right\} \quad \text{for all } j$$

From the sum of the optimal values of the knapsacks, we then subtract $\sum_{j=1}^{n}\mathbf{u}_j$

to get a ***lower bound*** on the optimum of the GAP.

Consider our example GAP again:

Iteration # 1

```
Current multipliers:
                    i: 1 2 3 4
                 w[i]: 0 0 0 0
Solving knapsack problems:
  Machine #1
     NO items
  Machine #2
     NO items
  Machine #3
     NO items

*** Dual value is 0 ***
(Improvement: 999)
# of times jobs are assigned: 0 0 0 0
Subgradient of Dual Objective is 1 1 1 1
Stepsize is 11.25
```

since we have relaxed the requirement that each job be assigned to a machine!

each multiplier will be increased an equal amount

Iteration # 2

```
Current multipliers:
                    i:   1     2     3     4
                 w[i]: 11.25 11.25 11.25 11.25
Solving knapsack problems:
  Machine #1
     NO items
  Machine #2
     Alternate optimal solutions:
          Solution #1: items 1 2
          Solution #2: items 2 4
  Machine #3
     NO items

*** Dual value is 41.5 ***
(Improvement: 41.5)
# of times jobs are assigned: 1 1 0 0
Subgradient of Dual Objective is 0 0 1 1
Stepsize is 6.9375
```

multipliers for jobs 3 & 4 will be increased

Iteration # 3

```
Current multipliers:
                i:  1     2      3        4
             w[i]: 11.25 11.25 18.1875 18.1875
Solving knapsack problems:
  Machine #1
     items 4
  Machine #2
     items 2 4
  Machine #3
     items 4

*** Dual value is 41.0625 ***
(Improvement: ¯0.4375)
# of times jobs are assigned: 0 1 0 3
Subgradient of Dual Objective is 1 0 1 ¯2
Stepsize is 2.36719
```

lower bound was worsened!

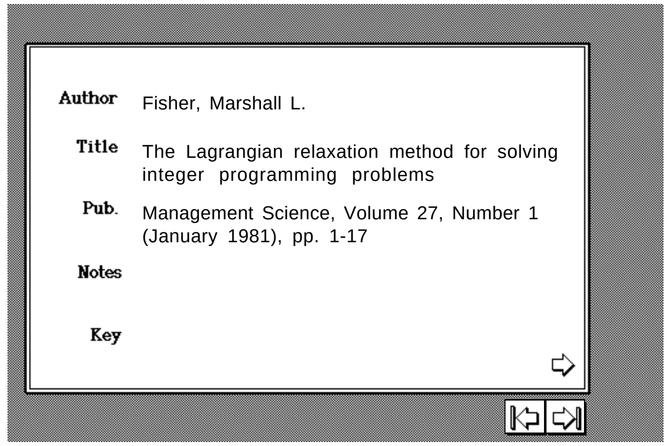
multiplier for job 4 will be decreased, while those for jobs 1 & 3, increased

Iteration # 4

```
Current multipliers:
                    i:   1       2      3       4
                 w[i]: 13.6172 11.25 20.5547 13.4531
Solving knapsack problems:
  Machine #1
     items 3
  Machine #2
     items 2 3
  Machine #3
     items 1 3

*** Dual value is 46.3438 ***
(Improvement: 5.28125)
# of times jobs are assigned: 1 1 3 0
Subgradient of Dual Objective is 0 0 ¯2 1
Stepsize is 2.04844
```

lower bound was improved!



Author Fisher, Marshall L.

Title The Lagrangian relaxation method for solving integer programming problems

Pub. Management Science, Volume 27, Number 1 (January 1981), pp. 1-17

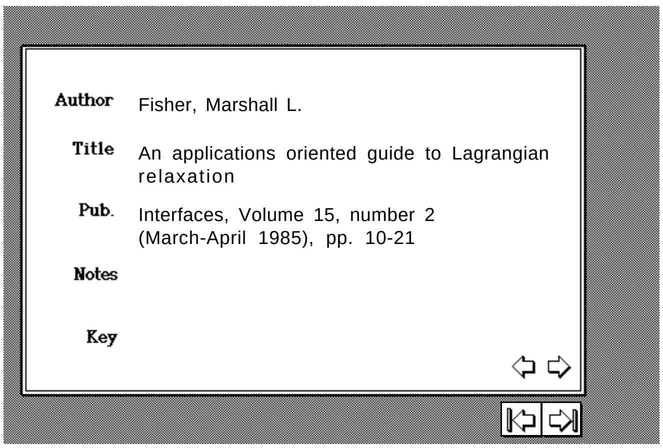
Notes

Key

Iteration # 5

```
Current multipliers:
              i:  1        2     3       4
           w[i]: 13.6172 11.25 16.4578 15.5016
Solving knapsack problems:
  Machine #1
     NO items
  Machine #2
     items 1 4
  Machine #3
     items 1 4

*** Dual value is 43.5891 ***
(Improvement: ¯2.75469)
# of times jobs are assigned: 2 0 0 2
Subgradient of Dual Objective is ¯1 1 1 ¯1
Stepsize is 3.07705
```



Author Fisher, Marshall L.

Title An applications oriented guide to Lagrangian relaxation

Pub. Interfaces, Volume 15, number 2 (March-April 1985), pp. 10-21

Notes

Key