CIE 5015 Operations Research Lecture 8
Solving Integer Programming Problems II

陳柏華 Albert Chen

National Taiwan University (NTU)

Department of Civil Engineering

November 22^{th}, 2019

Outline

Outline

Integer Linear Program & It's Relaxed Problem

Integer Linear Program

$$\max 17x_1 + 12x_2$$

s.t.

$$10x_1 + 7x_2 \leq 40$$

$$x_1 + x_2 \leq 5$$

$$x_1, x_2 \geq 0$$

$$x_1, x_2 \in int$$

Relaxed Linear Program

$$\max 17x_1 + 12x_2$$

s.t.

$$10x_1 + 7x_2 \leq 40$$

$$x_1 + x_2 \leq 5$$

$$x_1, x_2 \geq 0$$

$$\cancel{x_1, x_2 \in int}$$

Robert J. Vanderbei, ORF, Princeton

Facts on IP and Its Relaxed LP

- An integer linear program is a LP further constrained by the integrality restrictions.
- In a maximization problem, the value of the objective function, at the LP optimum, will always be an upper bound on the optimal IP objective.
- Any integer feasible point is always a lower bound on the optimal IP objective value.

(Bradley, Hax, and Magnanti, AMP, 1977)

Outline

1. Review: Solving Integer Programming Problems
2. The Lagrangian Relaxation
3. The Bound Provided by Lagrangian Relaxation
4. The Facility Location Problem
5. Example of Lagrangian Relaxation for the K-Madoids Problem
6. Python, Gurobi, and Lagrangian Relaxation

Lagrangian Relaxation (拉氏釋限/拉氏放鬆)

- (Mixed) Integer Programming (M)IP problems could be hard to solve.
- Lagrangian relaxation is based upon the observation that many difficult integer programming problems can be modeled as a relatively easy problem complicated by a set of side/hard constraints.
- A Lagrangian problem converts the problem with the hard constraints replaced with a penalty term in the objective function.
- The penalty term (associated to dual variables) is related to the amount of violation of the hard constraints.

Lagrangian Relaxation

The Lagrangian problem is easy to solve and provides (for a maximization problem) an upper bound on the optimal value of the original problem.

It can be used in place of a linear programming relaxation to provide bounds in a *Branch and Bound* algorithm.

Original Integer Programming Problem

Problem (P):

very hard to solve

$Z = \max cx$

s. t.

$Ax \leq b$

$Dx \leq e$

$x \geq 0$, and integer.

where x is $n \times 1$, b is $m \times 1$, and e is $k \times 1$. We assume that the constraints of (P) have been partitioned into the two sets $Ax \leq b$ and $Dx \leq e$ so that (P) is relatively easy to solve if the constraint set $Ax \leq b$ is removed.

Changing the Objective Function

We first define an m vector of non-negative multipliers u and add $u(b - Ax)$ to the objective function of (P) to obtain:

$$\max cx + u(b - Ax)$$

s. t.

$$Ax \leq b$$

$$Dx \leq e$$

$x \geq 0$, and integer.

The optimal value of this problem for u fixed at a nonnegative value is an upper bound on Z because we have added a nonnegative term to the objective function.

Start here

Lagrangian Relaxation

We create the Lagrangian problem by removing the constraints $Ax \leq b$ to obtain:

$$Z_D(u) = \max cx + u(b - Ax)$$

s. t.

$$Dx \leq e$$

$x \geq 0$, and integer.

Since removing the constraints $Ax \leq b$ cannot decrease the optimal value, $Z_D(u)$ is also an upper bound on Z. Moreover, by assumption the Lagrangian problem is relatively easy to solve.

Lagrangian Relaxation

There are three major questions in designing a Lagrangian-Relaxation-based approach:

1. Which constraints should be relaxed? The relaxation should make the problem significantly easier to solve.
2. How to compute good multipliers u? There is a choice between a general purpose procedure called the *subgradient method* and "smarter" methods which may be better but highly problem specific.
3. How to deduce a good, feasible solution to the original problem, given a solution to the relaxed problem? Problem specific.

Lagrangian Relaxation - Example

$$Z = \max 16x_1 + 10x_2 + 4x_4$$

s. t.

$$8x_1 + 2x_2 + x_3 + 4x_4 \leq 10$$

$$x_1 + x_2 \leq 1$$

$$x_3 + x_4 \leq 1$$

$1 \geq x_i \geq 0$, and integer, $\forall i$.

Lagrangian Relaxation - Example

If we dualize the first constraint, we get the Lagrangian Relaxation problem (LRP):

$$Z_D(u) = \max 16x_1 + 10x_2 + 4x_4 + u(10 - 8x_1 - 2x_2 - x_3 - 4x_4)$$

Penalty term, expected to be positive

s. t.

$\cancel{8x_1 + 2x_2 + x_3 + 4x_4 \leq 10}$

$x_1 + x_2 \leq 1$

DV are axis

$x_3 + x_4 \leq 1$

$1 \geq x_i \geq 0$, and integer, $\forall i$.

we have to get the u value by...

- setting u to be a variable ✗
- setting u to be a constant ✓

Lagrangian Relaxation - Example

It is easy to solve this relaxation if the dual variable (also called the Lagrangian Multiplier) u is fixed at some nonnegative value.

but with different u values, there will be different penalties / upper bounds / solutions

we will treat the u value as a variable in another dual problem,

just not in the Lagrangian Relaxation Problem

Lagrangian Relaxation - Example

Different dual variable values produce different bounds to the original problem. We hope to get a tight bound. Ideally, we hope to solve for u in the following dual problem called the Lagrangian Dual.

$$Z_D = \min_u Z_D(u), u \geq 0.$$

because u is the upper bound and we want the tightest upper bound

Think of the dualized constraint as a resource constraint with the right side representing the available supply and the left side the amount of the resource demanded in a particular solution.

u is actually the "shadow price!"

$10 - 8x_1 - 2x_2 - x_3 - 4x_4 = 0$

$$\text{(demand)} \rightarrow 8x_1 + 2x_2 + x_3 + 4x_4 \leq 10 \leftarrow \text{(supply)}$$

when penalty term is 0, this constraint will be strictly bounded (=)

Lagrangian Relaxation - Example

We can then interpret the dual variable u as a "price" charged for the resource. If we can discover a price u^* so that the supply and demand for the resource are equal, we get a tight upper bound.

In our example, the objective function is:

$$\begin{aligned} Z_D(u) = \max\, & 16x_1 + 10x_2 + 4x_4 \\ & + u(10 - 8x_1 - 2x_2 - x_3 - 4x_4) \end{aligned}$$

We hope with the right price u^*, we will have $10 - 8x_1 - 2x_2 - x_3 - 4x_4 = 0$.

Lagrangian Relaxation - Example

Given the u value, we need to solve for the LRP problem, with $Z_D(u)$ as objective function.

but we want to also see if we can lower u to get a tighter upper-bound

We can solve the LRP problem by using Branch and Bound, or by a solver such as CPLEX or Gurobi.

Lagrangian Relaxation - Example

For a better demonstration of our easy example, we choose to use the following method to solve for LRP. But note that actual large scale problems might not be solved so easily as the following method.

Note that if the objective coefficient of any variable is not positive, we can set that variable to 0.

Otherwise, based on the constraints, we choose

(1) either x_1, or x_2 to set to 1, and

(2) either x_3 or x_4 to set to 1, depending on which has the larger objective function coefficient.

Lagrangian Relaxation - Example

$$Z_D = \min_u Z_D(u), u \geq 0.$$

Observe that, with different u,
we get different upperbounds and solutions

	Lagrangian Solution					Value of Lagrangian solution if feasible
u	x_1	x_2	x_3	x_4	$Z_D(u)$	
0	1	0	0	1	20 best upper bound	not feasible
6	0	0	0	0	60 > 20 1st feasible	0 but not best upperbound
3	0	1	0	0	34 > 20 2nd feasible	10 but not best upperbound
2	0	1	0	0	26	10
1	1	0	0	0	18 best upper bound (lowest)	16 best lower bound (highest)
	1	0	0	1	18	
	0	1	0	0	18	10
	0	1	0	1	18	14
½	1	0	0	1	19	
¾	1	0	0	1	18.5	

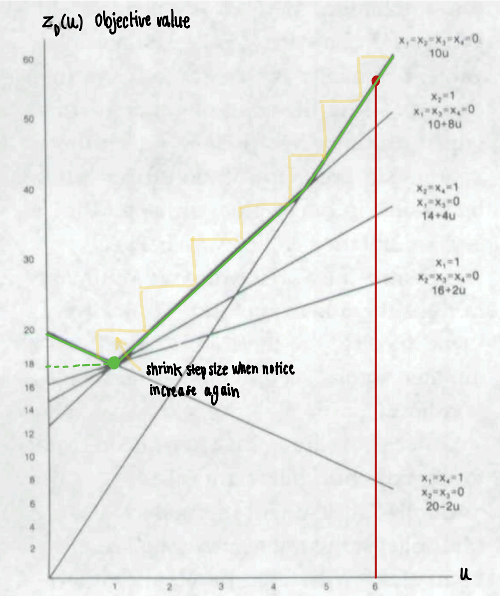
Lagrangian Relaxation - Example

Given a u value, we wish to maximize $Z_D(u)$.

Because $Z_D(u)$ is an upper bound of P, we hope to find the tightest bound.

$$\min Z_D = \min_u Z_D(u) = \min_u \{\max_x cx + u(b - Ax)\}$$

Lagrangian Relaxation - Example



- given u, we are working on a maximization, the variables are x
- when u is changing, it is a minimization problem to find the tightest upperbound. Use step size to find the smallest u

Lagrangian Relaxation - Subgradient Method

By using the Subgradient Method, we hope to repeatedly solve for $Z_D(u)$ with different u values, in hope to seek for the tightest bound of P. The following equation is commonly used to adjust u from iteration k to iteration $k+1$.

Looking for non-negative, because the multiplier (u) has to be non-negative

$$u^{k+1} = \max\{0, u^k - t_k(b - Ax^k)\}$$

where t^k is a scaler step size, and x^k is an optimal solution to $Z_D(u^k)$, the Lagrangian problem with dual variables set to u^k.

Lagrangian Relaxation - Step Size for Subgradient

$u^0 = 0$

$u^1 = 0 - (-2) = 2$

$u^2 = \max\{0, 2\text{-}8\} = 0$

$u^3 = 0 - (-2) = 2$

$u^4 = \max\{0, 2\text{-}8\} = 0$

step size

Table 2: Subgradient method with $t_k = 1$ for all k.

u oscillates between the values 0 and 2.

Lagrangian Relaxation - Step Size for Subgradient

$u^0 = 0$
$u^1 = 0 - (-2) = 2$
$u^2 = \max\{0,\ 2 - \frac{1}{2}(8)\} = 0$
$u^3 = 0 - \frac{1}{4}(-2) = \frac{1}{2}$
$u^4 = \frac{1}{2} - \frac{1}{8}(-2) = \frac{3}{4}$
$u^5 = \frac{3}{4} - 1/16(-2) = \frac{7}{8}$ decreased step sizes
$u^6 = \frac{7}{8} - 1/32(-2) = 15/16$

Table 3: Subgradient method with $t_k = 1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \ldots$

u converges to optimal value of 1.

Lagrangian Relaxation - Step Size for Subgradient

$u^0 = 0$

$u^1 = 2$

$u^2 = \max\{0,\ 2 - 1/3(8)\} = 0$

$u^3 = 0 - 1/9(-2) = 2/9$

$u^4 = 2/9 - 1/27(-2) = .296$

$u^5 = .296 - 1/81(-2) = .321$

$u^6 = .321 - 1{-}243(-2) = .329$

$u^7 = .329 - 1/729(-2) = .332$

Table 4: Subgradient method with t_k=1, 1/3, 1/9, 1/27, 1/81, . . .

u converges to the value of $\frac{1}{3}$.

Lagrangian Relaxation - Step Size for Subgradient

unknown variable

value of minimization

value of feasible solution

$$t_k = \frac{\lambda_k (Z_D(u^k) - Z^*)}{\sum_{i=1}^{m} (b_i - \sum_{j=1}^{n} a_{ij} x_j^k)^2}$$

when value oscillates

Rule of thumb
to prevent infeasible value
unless we know a better value

where λ_k is chosen to be a scalar between 0 and 2, initiated at 2. It is reduced by a factor of 2 whenever $Z_D(u^k)$ has failed to decrease in a specific number of iterations. Z^* is initially set to 0 and later updated by a solution from $Z_D(u^k)$ that is also feasible to P.

Unless we obtain a u^k for which $Z_D(u^k) = Z^*$, there is no way of proving optimality in the subgradient method. It is usually terminated with a iteration limit, or solution gap.

better way to terminate

Lagrangian Relaxation

1. Identify sets of hard constraints

Once the constraints are removed then it is easier to solve.

2. Form the Lagrangian Relaxation Problem
3. Solve the Lagrangian Dual with the Subgradient method
   - 3.1 given u_k, solve Lagrangian Relaxation Problem to get x^k
   - 3.2 reduce λ with a factor (if needed)
   - 3.3 update t_k
   - 3.4 update u_k to u_{k+1}
   - 3.5 termination, or continue back to 3.1.
4. Further actions to get to feasible/optimal solution, if needed.

Reference Documents

- M. L. Fisher (1985) "An Applications Oriented Guide to Lagrangian," *Interfaces*, 15(2), 10-21. (main reference)
- M. L. Fisher (2004) "The Lagrangian Relaxation Method for Solving Integer Programming Problems," *Management Science*, 50(12), 1861-1871. (more advanced material well written)
- A. M. Geoffrion (1974) "Lagrangian Relaxation for Integer Programming," *Mathematical Programming Study*, 2, 82-114. (advanced material)

Outline

1. Review: Solving Integer Programming Problems
2. The Lagrangian Relaxation
3. The Bound Provided by Lagrangian Relaxation
4. The Facility Location Problem
5. Example of Lagrangian Relaxation for the K-Madoids Problem
6. Python, Gurobi, and Lagrangian Relaxation

Optimality Gap

We define the optimality gap to be:

$$gap = \frac{\Delta Z}{Z^*} * 100\%$$

where Z^* is the best known feasible solution, and $Z_D(u)$ the Lagrangian bound.

$\Delta Z = Z_D(u) - Z^*$ for maximization problem, and
$\Delta Z = Z^* - Z_D(u)$ for minimization problem.

Lagrangian Relaxation Bound of LR

We have obtained through the application of Lagrangian Relaxation and the Subgradient Method a feasible solution with a value of 16 and an upper bound on the optimal value of 18.

At this point, we could stop and be content with a feasible solution proven to be within 12.5% of optimality, or we could adjust the solution of the example to optimality by using methods such as *Branch and Bound*, with bounds provided by our Lagrangian Relaxation.

Lagrangian Relaxation Bound of LR

How well does the bound of Lagrangian Relaxation compare with that of the Linear Program relaxation of (P)?

Lagrangian Relaxation Bound of LR

$$Z_D = \min_u Z_D(u) = \min_u \{\max_x cx + u(b - Ax)\}$$

s. t.

$Dx \le e$

$u \ge 0$

$x \ge 0$, and integer.

Lagrangian Relaxation Bound of LR

The original problem will be bounded by the upperbound from LP relaxation of Lagrangian problem

$$Z_D \le Z_D^{LP} = \min_u \{\max_x cx + u(b - Ax)\}$$

s. t.

$$Dx \le e$$

$$u \ge 0$$

$x \ge 0$, ~~and integer~~.

Lagrangian Relaxation Bound of LR

dualization variable fordualization problem

$$Z_D^{LP} = \min_u \{\min_v ub + ve\}$$

(LP duality)

s. t.

$$c - uA \leq vD$$

$$u \geq 0$$

$$v \geq 0$$

Lagrangian Relaxation Bound of LR

$$Z_D^{LP} = \min_{u,v} ub + ve$$

s. t.

$$uA + vD \geq c$$

$$u \geq 0$$

$$v \geq 0$$

Lagrangian Relaxation Bound of LR

Almost the same as original question

$$Z_D^{LP} = \max_x cx = Z_{LP}$$

(LP duality again)

s. t.

$Ax \leq b$ Decision variables are relaxed

$Dx \leq e$

$x \geq 0$

Lagrangian Relaxation Bound of LR

Pure linear program relaxation

$$Z_D \leq Z_D^{LP} = Z_{LP}$$

Lagrangian relaxation smaller

Lagrangian Relaxation Bound of LR

Observation:

$$Z_D \leq Z_D^{LP} = Z_{LP}$$

Having $Z_D = Z_{LP}$ means the removal of the integrality constraint of x on the Lagrangian problem Z_D has no effect on the solution.

In the Lagrangian problem for the example problem, the optimal values of the variables will be integer whether we require it or not. This implies that we must have $Z_D = Z_{LP}$.

Lagrangian Relaxation - Example

Original Problem:

$$Z = \max 16x_1 + 10x_2 + 4x_4$$

s. t.

$$8x_1 + 2x_2 + x_3 + 4x_4 \leq 10$$

$$x_1 + x_2 \leq 1$$

$$x_3 + x_4 \leq 1$$

$1 \geq x_i \geq 0$, and integer, $\forall i$.

Lagrangian Relaxation - Original Formulation

In the Lagrangian problem for this example, optimal variables will be integer whether we require it or not. This implies that we must have $Z_D = Z_{LP}$.

$$Z_D(u) = \max 16x_1 + 10x_2 + 4x_4$$
$$+ u(10 - 8x_1 - 2x_2 - x_3 - 4x_4)$$

s. t.

$$\cancel{8x_1 + 2x_2 + x_3 + 4x_4 \leq 10}$$

$$x_1 + x_2 \leq 1$$

$$x_3 + x_4 \leq 1$$

$$1 \geq x_i \geq 0, \text{ and integer, } \forall i.$$

With or without the relaxation, the DVs are going to be integers. DVs which are more profitable will be 1, the other will be just 0. (x_1 is more profitable then $x_1 = 1$, $x_2 = 0$)

Lagrangian Relaxation - Alternative Formulation

How about this formulation? $Z_D \leq Z_D^{LP} = Z_{LP}$

$$Z_D(u) = \max 16x_1 + 10x_2 + 4x_4$$
$$+ v_1(1 - x_1 - x_2) + v_2(1 - x_3 - x_4)$$

s. t.

$$8x_1 + 2x_2 + x_3 + 4x_4 \leq 10$$

$\cancel{x_1 + x_2 \leq 1}$ By relaxing these constraints, then it is actually easier to solve.

$\cancel{x_3 + x_4 \leq 1}$

$1 \geq x_i \geq 0$, and integer, $\forall i$.

TRICK ① strict equality constraints will usually make the problem harder to solve. Get rid of them!

② Also each kind of problems will have their methods of solving. Find out by reading literature.

Lagrangian Relaxation - Alternative Formulation

Lagrangian Solution

v_1	v_2	λ_k	x_1	x_2	x_3	x_4	$Z_D(v_1,v_2)$	Z^*
0	0	1	1	1	0	0	26	0
13	0	1	0	0	0	1	17	4
			(feasible with $Z = 4$)					
0	0	1	1	1	0	0	26	4
11	0	1	1	0	0	0	16	16
			(feasible with $Z = 16$)					

perfect bound

Table 5: The subgradient method applied to the improved relaxation.

Lagrangian Relaxation

Lagrangian Relaxation

1. Identify sets of hard constraints
2. Form the Lagrangian Relaxation Problem
3. Solve the Lagrangian Dual with the Subgradient method
   3.1 given u_k, solve Lagrangian Relaxation Problem to get x^k
   3.2 reduce λ with a factor (if needed)
   3.3 update t_k
   3.4 update u_k to u_{k+1}
   3.5 termination, or continue back to 3.1.
4. Further actions to get to feasible/optimal solution, if needed.

Outline

The Facility Location Problem

Given a set of customers I, a set of facilities J, cost c_j for opening a facility $j \in J$, and cost d_{ij} of connecting customer $i \in I$ to facility j, the Uncapacitated Facility Location Problem (UFLP) seeks to open a subset of facilities and assign each customer to exactly one facility at minimum cost.

The Facility Location Problem

cost of connecting customer i to facility j

cost of opening facility

$$\min \sum_{i=1}^{M}\sum_{j=1}^{N} d_{ij}x_{ij} + \sum_{j=1}^{N} c_j y_j$$ facility j open or not

customer i connect or not to facility j

s.t.

$$\sum_{j=1}^{N} x_{ij} = 1, i = 1, \ldots, M$$

$x_{ij} \leq y_j, i = 1, \ldots, M, j = 1, \ldots, N$ if we want to connect customer i to facility j then facility j has to be openned.

$x_{ij} \in \{0, 1\}, i = 1, \ldots, M, j = 1, \ldots, N$ customer i connect or not to facility j

$y_j \in \{0, 1\}, j = 1, \ldots, N$ facility j open or not

The K Center Problem

If we further restrict the problem to have at most only K number of facilities established, the problem becomes the K Center Problem.

The K Center Problem

$$\min \sum_{i=1}^{M}\sum_{j=1}^{N} d_{ij}x_{ij} + \sum_{j=1}^{N} c_j y_j$$

s.t.

$$\sum_{j=1}^{N} x_{ij} = 1, i = 1, ..., M$$

$$x_{ij} \leq y_j, i = 1, ..., M, j = 1, ..., N$$

$$\sum_{j=1}^{N} y_j \leq K$$

$$x_{ij} \in \{0, 1\}, i = 1, ..., M, j = 1, ..., N$$

$$y_j \in \{0, 1\}, j = 1, ..., N$$

The K Medoids Problem

If we have a K Center Problem, but to have the set of candidate centers to be the same as the customers ($I = J$), we get the K Medoids Problem. This problem is used in data mining for grouping/clustering data points for unsupervised classification.

customer set and facility set are the same (same elements) = "K Medoids" problem

"customer j is being served by facility j"

The K Medoids Problem - Strong formulation

$$\min \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij} x_{ij}$$

s.t.

$$\sum_{j=1}^{N} x_{ij} = 1, i = 1, ..., N$$

$$x_{ij} \leq x_{jj}, i, j = 1, ..., N$$

$$\sum_{j=1}^{N} x_{jj} \leq K$$

$$x_{ij} \in \{0, 1\}, i, j = 1, ..., N$$

Better, although more constraints
when the problem is big, big difference
will be seen.

The K Medoids Problem - Weak formulation

$$\min \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij} x_{ij}$$

s.t.

$$\sum_{j=1}^{N} x_{ij} = 1, i = 1, ..., N$$

$$\sum_{i=1}^{N} x_{ij} \leq N x_{jj}, j = 1, ..., N$$

$$\sum_{j=1}^{N} x_{jj} \leq K$$

$$x_{ij} \in \{0, 1\}, i, j = 1, ..., N$$

Outline

The K Medoids Problem

If we have a K Center Problem, but to have the set of candidate centers to be the same as the customers ($I = J$), we get the K Medoids Problem. This problem is used in data mining for grouping/clustering data points for unsupervised classification.

Strong formulation

$$\min \sum_{i=1}^{N}\sum_{j=1}^{N} d_{ij}x_{ij}$$

s.t.

$$\sum_{j=1}^{N} x_{ij} = 1, i = 1, ..., N$$

$$x_{ij} \leq x_{jj}, i, j = 1, ..., N$$

$$\sum_{j=1}^{N} x_{jj} \leq K$$

$$x_{ij} \in \{0, 1\}, i, j = 1, ..., N$$

Strong formulation, Hard constraint set?

$$\min \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij} x_{ij}$$

s.t.

$$\sum_{j=1}^{N} x_{ij} = 1, i = 1, ..., N$$

$$x_{ij} \leq x_{jj}, i, j = 1, ..., N$$

$$\sum_{j=1}^{N} x_{jj} \leq K$$

$$x_{ij} \in \{0, 1\}, i, j = 1, ..., N$$

Strong formulation, Hard constraint set?

$$\min \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij} x_{ij}$$

s.t.

$$\sum_{j=1}^{N} x_{ij} = 1, i = 1, ..., N$$

$$x_{ij} \leq x_{jj}, i, j = 1, ..., N$$

$$\sum_{j=1}^{N} x_{jj} \leq K$$

$$x_{ij} \in \{0, 1\}, i, j = 1, ..., N$$

Make sure symbols are consistent

Strong formulation, Dualize hard constraint set.

$$z_D(u) = \min \sum_{i=1}^{N}\sum_{j=1}^{N} d_{ij}x_{ij} - \sum_{i=1}^{N} u_i\{1 - \sum_{j=1}^{N} x_{ij}\}$$

Original objective

Since it is a minimization problem, we want our Lagrangian Relaxation Problem to go down

s.t.

$$x_{ij} \leq x_{jj}, i, j = 1, ..., N$$

$$\sum_{j=1}^{N} x_{jj} \leq K$$

$$x_{ij} \in \{0, 1\}, i, j = 1, ..., N$$

Note the $-$ sign is decided arbitrarily. What follows is based on this sign and needs to be consistent.

Strong formulation, Dualize hard constraint set.

$$z_D(u) = \min \sum_{i=1}^{N}\sum_{j=1}^{N}\{d_{ij} + u_i\}x_{ij} - \sum_{i=1}^{N} u_i$$

s.t.

$$x_{ij} \leq x_{jj}, i, j = 1, ..., N$$

$$\sum_{j=1}^{N} x_{jj} \leq K$$

$$x_{ij} \in \{0, 1\}, i, j = 1, ..., N$$

Note the $-$ sign is decided arbitrarily. What follows is based on this sign and needs to be consistent.

Strong formulation, Dualize hard constraint set.

The Lagrangian dual is the following problem:

$$z_D = \max_u z_D(u)$$

since original problem is a min problem
we have to max our penalty

Note that: the Lagrangian problem $z_D(u)$ is a lower bound of the original minimization problem. We wish to find the tightest bound (in this case the maximized lower bound).

Subgradient Method

Updating the u vector.

$$u^{k+1} = u^k + t_k(1 - \sum_{j=1}^{N} x_{ij}^k)$$

where t^k is a scaler step size, and x^k is an optimal solution to LR_u^k, the Lagrangian problem with dual variables set to u^k.

Note the $+$ sign is decided arbitrarily. What follows is based on this sign and needs to be consistent.

Deciding Step Size for Subgradient

obj function

$$z_D^k(u) = \min \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij} x_{ij}^k - \sum_{i=1}^{N} u_i^k \{1 - \sum_{j=1}^{N} x_{ij}^k\}$$

want to be positive

update u value

$$u^{k+1} = u^k + t_k (1 - \sum_{j=1}^{N} x_{ij}^k)$$

if $1 - \sum_{j=1}^{N} x_{ij}^k > 0$

$\rightarrow$ we try to make $u_i^{k+1} < 0$

$\rightarrow$ we want $t_k < 0$

if $1 - \sum_{j=1}^{N} x_{ij}^k < 0$

if $\rightarrow$ we try to make $u_i^{k+1} > 0$

then $\rightarrow$ we want $t_k < 0$ for $(1 - \sum_{j=1}^{N} x_{ij}^k)$ to be positive

Deciding Step Size for Subgradient

$$t_k = \frac{\lambda_k(Z_D(u^k) - Z^*)}{\sum_{i=1}^{N}(1 - \sum_{j=1}^{N} x_{ij}^k)^2}$$

where λ_k is chosen to be a scalar between 0 and 2, initiated at 2. It is reduced by a factor whenever $Z_D(u^k)$ has failed to increase in a specific number of iterations.

Z^* is initially set to a random feasible solution and later updated by better solutions from $Z_D(u^k)$ that are also feasible to P.

Unless we obtain a u^k for which $Z_D(u^k) = Z^*$, there is no way of proving optimality in the subgradient method. It is usually terminated with a iteration limit, and/or gap tolerance.

Outline

1. Review: Solving Integer Programming Problems

2. The Lagrangian Relaxation

3. The Bound Provided by Lagrangian Relaxation

4. The Facility Location Problem

5. Example of Lagrangian Relaxation for the K-Madoids Problem

6. Python, Gurobi, and Lagrangian Relaxation

HW9-1: Use Gurobi to solve MIP: Lagrangian Relaxation

$$Z_D(u) = \max 16x_1 + 10x_2 + 4x_4 + v_1(1 - x_1 - x_2) + v_2(1 - x_3 - x_4)$$

s. t.

$$8x_1 + 2x_2 + x_3 + 4x_4 \leq 10$$

$$\cancel{x_1 + x_2 \leq 1}$$

$$\cancel{x_3 + x_4 \leq 1}$$

$1 \geq x_i \geq 0$, and integer, $\forall i$.

decision variables int OK

Choose these 2 constraints

Dualize them to put in problem

v_1 and v_2 are constants

HW9-1: Use Gurobi to solve MIP: Lagrangian Relaxation

By automation / subgradient method or manual input of the u and v dual variables, can you get the result of 16 (gap $=$ 0%)?

Lagrangian Solution

v_1	v_2	λ_k	x_1	x_2	x_3	x_4	$Z_D(v_1,v_2)$	Z^*
0	0	1	1	1	0	0	26	0
13	0	1	0	0	0	1	17	4
(feasible with $Z = 4$)								
0	0	1	1	1	0	0	26	4
11	0	1	1	0	0	0	16	16
(feasible with $Z = 16$)								

— Should get 16

Table 5: The subgradient method applied to the improved relaxation.

HW9-2: LR for the Strong Formulation of the K Medoids Problem

Implement the LR for the Strong Formulation of the K Medoids Problem. You can make use of the ORkmedoidsLR.py file provided.

What is the best gap you can get with a problem of 300 points? How long does it take?

Code that can adjust points will be given

HW9-3: LR for the Weak Formulation of the K Medoids Problem

Implement the LR for the Weak Formulation of the K Medoids Problem.

What is the best gap you can get with a problem of 300 points? How long does it take?

There is chance the Lagrangian Problem takes longer to solve, and the wait is painful. You can close the terminal/console to stop the solution process.

The K Medoids Problem - Weak formulation

$$\min \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij} x_{ij}$$

s.t.

$$\sum_{j=1}^{N} x_{ij} = 1, i = 1, ..., N$$

$$\sum_{i=1}^{N} x_{ij} \leq N x_{jj}, j = 1, ..., N$$

$$\sum_{j=1}^{N} x_{jj} \leq K$$

$$x_{ij} \in \{0, 1\}, i, j = 1, ..., N$$

What to submit

Please submit a pdf file to discuss your findings in HW9-2 and HW9-3. Zip the pdf with the 3 scrip files for HW9-1, HW9-2 and HW9-3 by Thursday night.

Next Week

1. Midterm II. Lecture 7 & 8 Branch & Bound + Lagrangian.
2. Memoryless and Nonprogrammable calculators are allowed.
3. 1 cheatsheet double page allowed.
   OR applications & alg.
   Wayne L. Winston
   4th edition
4. Close book exam.
5. *email: AlbertChen@ntu.edu.tw*