Lagrangean Methods
– *bounding through penalty adjustment*

Thomas Stidsen

thst@man.dtu.dk

DTU-Management
Technical University of Denmark

Outline

- Brief introduction
- How to perform Lagrangean relaxation
- Subgradient techniques
- Example: Setcover
- Decomposition techniques and Branch and Bound
- Dual Ascent

Introduction

Lagrangean Relaxation is a technique which has been known for many years:

- Lagrange relaxation is invented by (surprise!) Lagrange in 1797 !
- This technique has been very usefull in conjunction with Branch and Bound methods.
- Since 1970 this has been **the** bounding decomposition technique of choice ...
- ... until the beginning of the 90'ies (branch-and-price)

The Beasley Note

This lecture is based on the (excellent !) Beasley note. The note has a practical approach to the problem:

- Emphasis on examples.
- Only little theory.
- Good practical advices.

All in all: This note is a good place to start if you later need to apply Lagrangean relaxation.

Given a Linear program

Min:

$$cx$$

s.t.:

$$\begin{aligned} Ax &\geq b \\ Bx &\geq d \\ x &\in \{0,1\} \end{aligned}$$

How can we calculate lower bounds ? We can use heuristics to generate upper bounds, but getting (good) lower bounds is often much harder ! The classical approach is to create a *relaxation*.

Requirements for relaxation

The program:

$min\{f(x)|x \in \Gamma \subseteq \mathcal{R}^n\}$

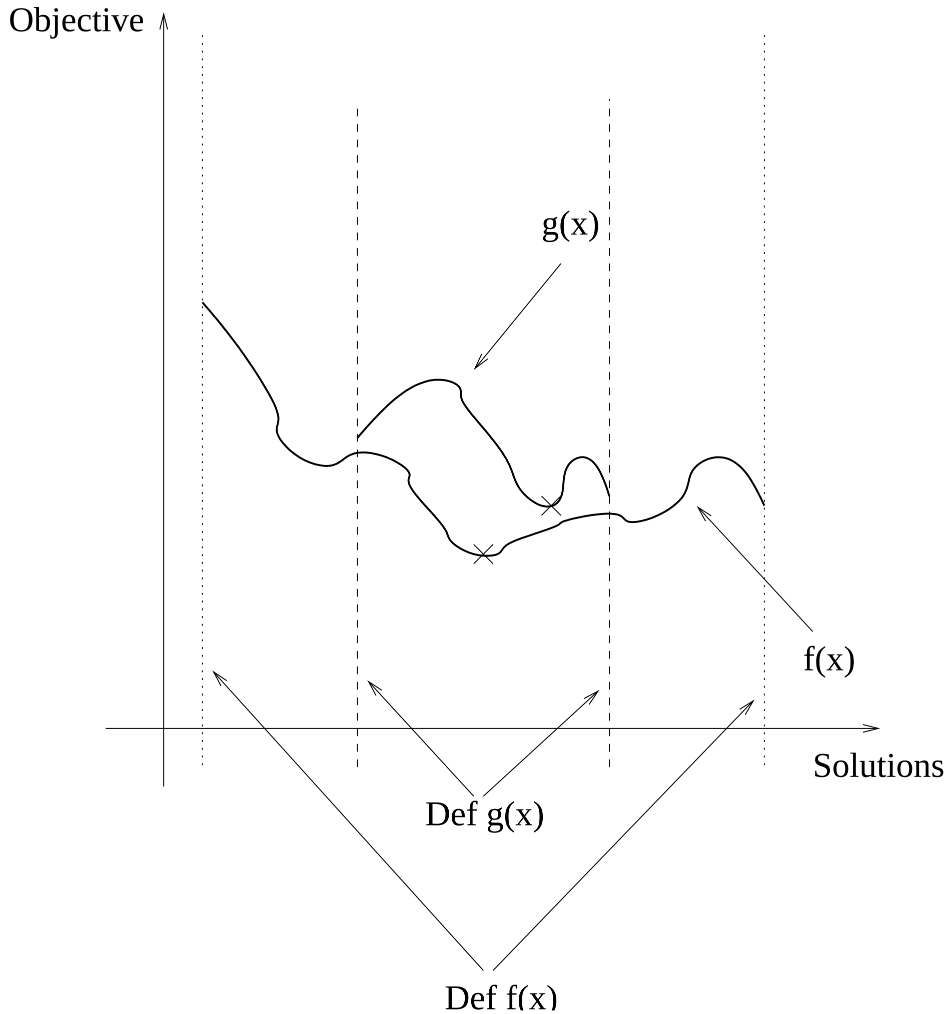
is a relaxation of:

$min\{g(x)|x \in \Gamma' \subseteq \mathcal{R}^n\}$

if:

- $\Gamma' \subseteq \Gamma$
- For $x \in \Gamma' \ : \ f(x) \leq g(x)$

Relaxation Graphically



Example of relaxation: LP

When we perform the LP relaxation:

- $f(x) = g(x)$
- $\Gamma'(= Z) \subseteq \Gamma(= R)$

The classical branch-and-bound algorithm use the LP relaxation. It has the nice feature of being general, i.e. applicable to all MIP models.

A very simple example I

g(x):
Min:

$$5x$$

s.t.:

$$\begin{aligned} x &\geq 3 \\ -x &\geq -10 \\ x &\in R^+ \end{aligned}$$

A very simple example II

f(x): (relaxation of g(x)

Min:

$$5x + \lambda(3 - x)$$

s.t.:

$$\begin{aligned} -x &\geq -10 \\ x &\in R^+ \end{aligned}$$

Relaxation by removal of constraints

Given:
Min:

$$cx$$

s.t.:

$$\begin{aligned} Ax &\geq b \\ Bx &\geq d \\ x &\in \{0, 1\} \end{aligned}$$

What if we instead of relaxing the domain constraints, relax another set of constraints ? (this also goes for integer variables i.e. $x \in Z$

Lagrangean Relaxation

Min:

$$cx + \lambda(b - Ax)$$

s.t.:

$$\begin{array}{rcl} Bx & \geq & d \\ x \in \{0,1\} & & \lambda \in \mathcal{R}^+ \end{array}$$

This is called the Lagrangean Lower Bound Program (LLBP) or the Lagrangean dual program.

Lagrangean Relaxation

First: **IS IT A RELAXATION ?**

- Well the feasible domain has been increased: $\Gamma'(Ax \geq b, Bx \geq d) \subseteq \Gamma(Bx \geq d)$
- Regarding the objective:
  - Inside the original domain: $f(x) = g(x) + \lambda(b - Ax)$ and since we *know* $\lambda(b - Ax) \leq 0 \Rightarrow f(x) =\leq g(x)$
  - Outside, no guarantee, but that is not a problem !

Lagrangean Relaxation

What can this be used for ?

- Primary usage: Bounding ! Because it is a relaxation, the optimal value will bound the optimal value of the real problem !
- Lagrangean heuristics, i.e. generate a “good” solution based on a solution to the relaxed problem.
- Problem reduction, i.e. reduce the original problem based on the solution to the relaxed problem.

Two Problems

Facing a problem we need to decide:

- Which constraints to relax (strategic choice)
- How to find the lagrangean multipliers, (tactical choice)

Which constraints to relax

Which constraints to relax depends on two things:

- Computational effort:
  - Number of Lagrangian multipliers
  - Hardness of problem to solve
- Integrality of relaxed problem: If it is integral, we can only do as good as the straightforward LP relaxation !

The integrality point will be dealt with theoretically next time ! And we will see an example here.

Multiplier adjustment

In Beasley two different types are given:

- Subgradient optimisation
- Multiplier adjustment

Of these, subgradient optimisation is **the** method of choice. This is general method which nearly always works ! Hence, here we will only consider this method. Since the Beasley note more efficient (but much more complicated) adjustment methods has been suggested.

Lagrangean Relaxation

We had:

Min:

$$cx + \lambda(b - Ax)$$

s.t.:

$$\begin{aligned} Bx &\geq d \\ x \in \{0,1\} & \qquad \lambda \in \mathcal{R}^+ \end{aligned}$$

Problem reformulation

Min:

$$\sum_j (c_j \cdot x_j + \sum_i \lambda_i (b_i - a_{ij} \cdot x_{ij}))$$

s.t.:

$$\begin{aligned} Bx &\geq d \\ x_j \in \{0,1\} & \qquad \lambda_i \in \mathcal{R}^+ \end{aligned}$$

Remember we want to obtain the best possible bounding, hence we want to **maximize** the λ bound.

The subgradient

We define the subgradient:
$G_i = b_i - \sum_j a_{ij} X_j$

If subgradient G_i is positive, decrease λ_i if G_i is negative, increase λ_i

Subgradient Optimisation

The sub-gradient optimisation algorithm is now:

Initialise $\pi \in]0, 2]$
Initialise λ values
repeat
  Solve LLBP given λ values get Z_{LB}, X_j
  Calc. the subgradients $G_i = b_i - \sum_j a_{ij} X_j$
  Calc. step size $T = \frac{\pi(Z_{UB} - Z_{LB})}{\sum_i G_i^2}$
  Update $\lambda_i = max(0, \lambda_i + T G_i)$
until we get bored ...

Example: Setcover

Min:

$$2x_1 + 3x_2 + 4x_3 + 5x_4$$

s.t.:

$$\begin{aligned} x_1 + x_3 &\geq 1 \\ x_1 + x_4 &\geq 1 \\ x_2 + x_3 + x_4 &\geq 1 \\ x_1, x_2, x_3, x_4 &\in \{0, 1\} \end{aligned}$$

Relaxed Setcover

Min:

$$
\begin{aligned}
2x_1 + 3x_2 + 4x_3 + 5x_4 & \\
+ & \ \lambda_1(1 - x_1 - x_3) \\
+ & \ \lambda_2(1 - x_1 - x_4) \\
+ & \ \lambda_3(1 - x_2 - x_3 - x_4)
\end{aligned}
$$

s.t.:

$$
\begin{aligned}
x_1, x_2, x_3, x_4 & \in \{0, 1\} \\
\lambda & \geq 0
\end{aligned}
$$

How can we solve this problem to optimality ???

Optimization Algorithm

The answer is so simple that we are reluctant calling it an optimization algorithm: Choose all x'es with negative coefficients !

What does this tell us about the strength of the relaxation ?

Rewritten: Relaxed Setcover

Min:

$$C_1 x_1 + C_2 x_2 + C_3 x_3 + C_4 x_4 + \lambda_1 + \lambda_2 + \lambda_3$$

s.t.:

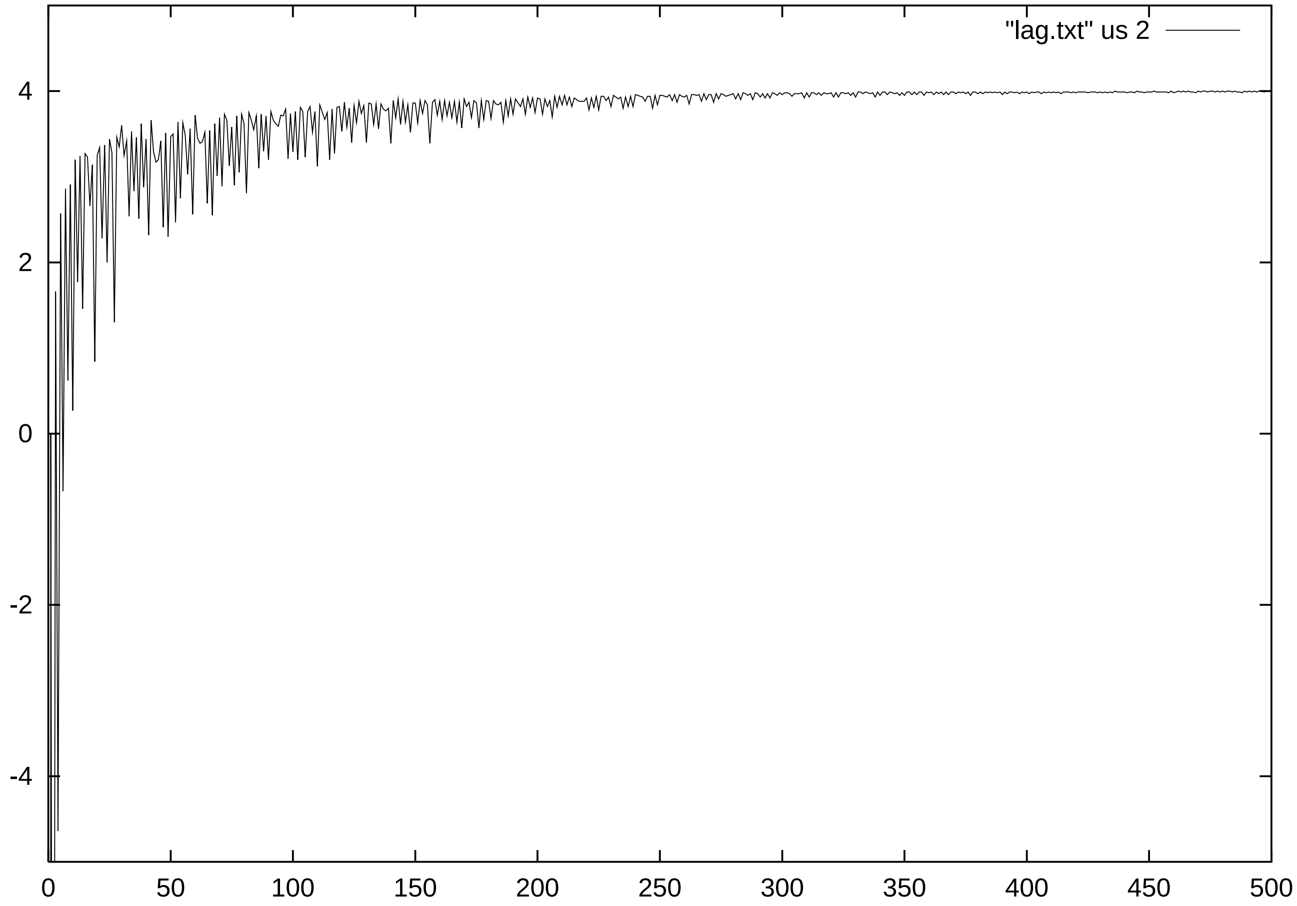
$$\begin{aligned} x_1, x_2, x_3, x_4 &\in \{0, 1\} \\ \lambda &\geq 0 \end{aligned}$$

$$\begin{aligned} C_1 &= (2 - \lambda_1 - \lambda_2) \\ C_2 &= (3 - \lambda_3) \\ C_3 &= (4 - \lambda_1 - \lambda_3) \\ C_4 &= (5 - \lambda_2 - \lambda_3) \end{aligned}$$

GAMS for Lagrange Rel. for Setcover

```
WHILE( counter < max_it,
  CC(j)= C(j)-SUM(i, A(i,j)*lambda(i));
  x.L(j)=0 + 1\$(CC(j)<0);
  Z_LB = SUM(j, CC(j)*x.L(j)) + SUM(i, lambd
  G(i)=1 - SUM(j,A(i,j)*x.L(j));
  T = pi * (Z_UB-Z_LB)/SUM(i,G(i)*G(i));
  lambda(i)=max(0,lambda(i)+T*G(i));
  counter=counter+1;
  lambda_sum = SUM(i,ABS(lambda(i)));
  put counter, Z_LB, G('1'), lambda('1'), la
);
```

Lower bound



Comments

Comments to the algorithm:

- This is actually quite interesting: The algorithm is very simple, but a good lower bound is found quickly !
- This relied a lot on the very simple LLBP optimization algorithm.
- Usually the LLBP requires much more work ....
- ... but according to Beasley, the subgradient algorithm very often works ...

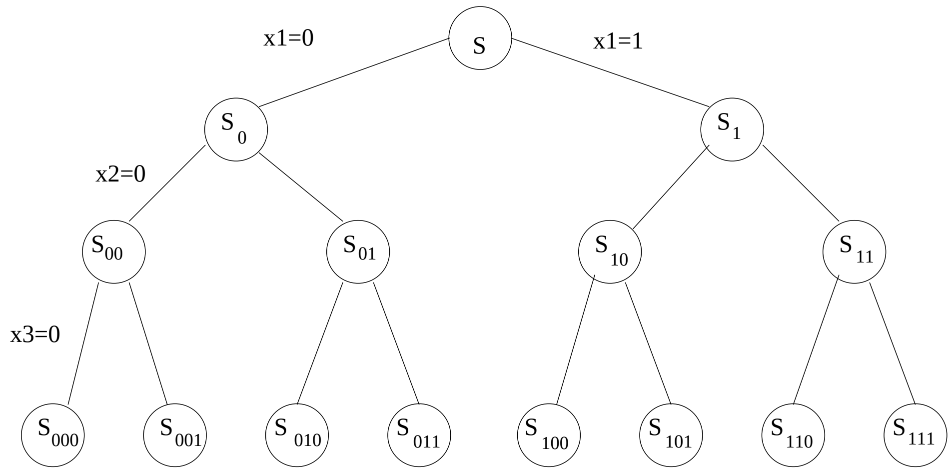
So whats the use ?

This is all wery nice, but how can we **solve** our problem ?

- We may be lucky that the lowerbound is also a feasible **and** optimal solution (like integer solutions to LP formulations).
- We may reduce the problem, performing Lagrangean problem reduction, next week.
- We may generate heuristic solutions based on the LLBP, next week.
- We may use LLBP in lower bound calculations for a Branch and Bound algorithm.

In a Branch and Bound Method

Why has Lagrangean relaxation become so important ? Because it is usefull in Branch and Bound methods.



Branching Influence on Lagrangian Bounding

Each branch corresponds to a simple choice: Branch up or branch down. This correspond to choose the value for one of our variables x_i. Hence: If we want to include Lagrangian bounding in a branch and bound algorithm, we need to be able to solve subproblems with these fixings ...

Branching Influence on Lagrangian Bounding II

Given this lower bound on some sub-tree in the branch-and-bound tree, we can (perhaps) perform bounding. Important: Any solution to the Lagrangian problem is a bound, so we can stop at any time (not the case in Branch-and-Price).

Dual Ascent

Another technique considered in the Beasley note is **Dual Ascent.** The idea is very simple: Given a (hard) MIP:

Optimal (min) solution to MIP

$\geq$

Optimal solution to LP

$=$

Optimal solution to DUAL LP

$\geq$

Any solution to DUAL LP

Example in Beasley: Setcover

LP Relaxation:

objective: minimise

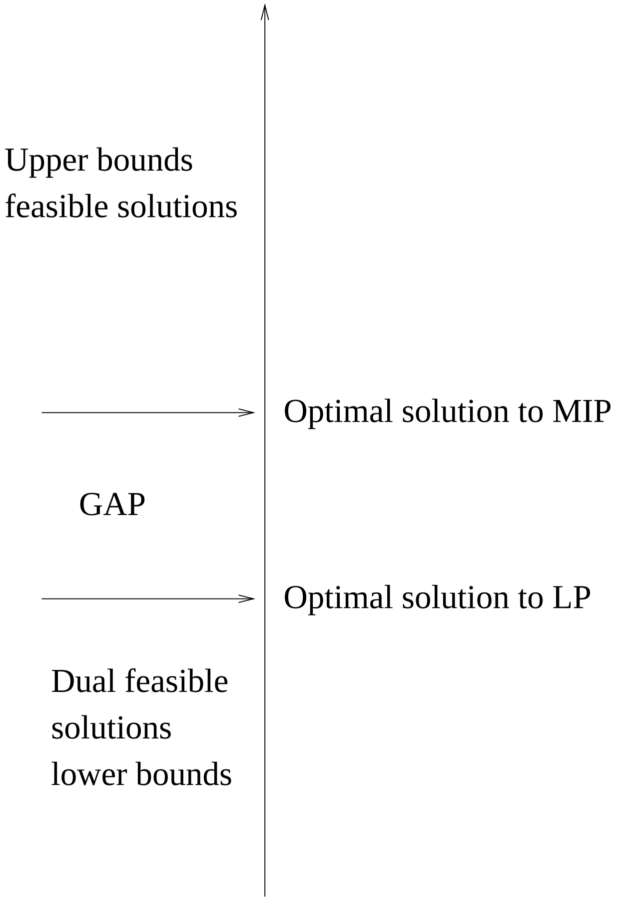
$$\sum_j c_j \cdot x_j$$

s.t.

$$\sum_j a_{ij} \cdot x_j \geq 1 \qquad \forall i$$

$$x_j \geq 0$$

Graphical


Upper bounds
feasible solutions
Optimal solution to MIP
GAP
Optimal solution to LP
Dual feasible
solutions
lower bounds

Dual Setcover

objective: maximize

$$\sum_i u_i$$

s.t.

$$\sum_i u_i \cdot a_{ij} \;\; \leq \;\; c_j \qquad \forall j$$

$$u_i \geq 0$$

Comments to Dual Ascent

- Dual ascent is a simple neat idea ...
- Beasley is not too impressed ...
- Dual ascent critically depends on the efficiency of the heuristic and the size of the GAP