Introduction to Optimization, DIKU 2005/06
Tuesday, January 10
David Pisinger

Program of the day:
- Surrogate relaxation
- Subgradient optimization for Lagrange multipliers
- A clue on solving LP problems without Simplex
- Applications: Manpower planning

Relaxation
In a branch-and-bound algorithm we find upper bounds by relaxing the problem

Relaxation (Wolsey sec. 2.1)
\[
\max \{ cx : x \in S \} \quad (IP) \\
\max \{ f(x) : x \in T \} \quad (RP)
\]

RP is a relaxation of IP if
- \( S \subseteq T \)
- \( f(x) \geq cx \) for all \( x \in S \)

Which constraints should be relaxed?
- Quality of bound (tightness of relaxation)
- Remaining problem can be solved efficiently
- Proper multipliers can be found efficiently
- Constraints difficult to formulate mathematically
- Constraints which are too expensive to write up

Overview
Different relaxations
- LP-relaxation
- Deleting constraint
- Lagrange relaxation
- Surrogate relaxation
- Semidefinite relaxation

Relaxations are often used in combination.

Hierarchy
- Tighter
  - Best surrogate relaxation
  - Best Lagrangian relaxation
  - LP relaxation

3

Surrogate relaxation, example
\[
\begin{align*}
\text{maximize} & \quad 4x_1 + x_2 \\
\text{subject to} & \quad 3x_1 - x_2 \leq 6 \\
& \quad x_2 \leq 3 \\
& \quad 5x_1 + 2x_2 \leq 18 \\
& \quad x_1, x_2 \geq 0, \text{ integer}
\end{align*}
\]

IP solution \((x_1, x_2) = (2, 3)\) with \(z_{IP} = 11\)
LP solution \((x_1, x_2) = \left(\frac{30}{11}, \frac{24}{11}\right)\) with \(z_{LP} = \frac{144}{11} = 13.1\)

First and third constraint complicating, surrogate relax using multipliers \(\lambda_1 = 2\), and \(\lambda_3 = 1\)
\[
\begin{align*}
\text{maximize} & \quad 4x_1 + x_2 \\
\text{subject to} & \quad 11x_1 - x_2 \leq 3 \\
& \quad x_1, x_2 \geq 0, \text{ integer}
\end{align*}
\]

Solution \((x_1, x_2) = (2, 3)\) with \(z_{SR} = 4 \cdot 2 + 3 = 11\)
Upper bound

4
Surrogate relaxation

Integer Programming Problem

\[
\begin{align*}
\text{maximize} & \quad cx \\
\text{subject to} & \quad Ax \leq b \\
& \quad Dx \leq d \\
& \quad x_j \in \mathbb{Z}_+, \quad j = 1, \ldots, n
\end{align*}
\]

Surrogate relax \(Dx \leq d\), using multipliers \(\lambda \geq 0\), i.e. add together constraints using weights \(\lambda\)

\[
\begin{align*}
\text{maximize} & \quad z_{SR}(\lambda) = cx \\
\text{subject to} & \quad Ax \leq b \\
& \quad \lambda Dx \leq \lambda d \\
& \quad x_j \in \mathbb{Z}_+, \quad j = 1, \ldots, n
\end{align*}
\]

**Proposition 1** Optimal solution to relaxed problem gives upper bound on original problem

**Proof** show that relaxation multiplier \(\lambda_i\) is “weighting” of constraint

If \(\lambda_i\) large \(\Rightarrow\) constraint satisfied (weakening other constraints)

If \(\lambda_i = 0\) \(\Rightarrow\) drop constraint

---

Surrogate relaxation

Surrogate relaxation (as well as all other relaxations) has the following property

\[
\begin{align*}
\text{maximize} & \quad z_{SR}(\lambda) = cx \\
\text{subject to} & \quad Ax \leq b \\
& \quad \lambda Dx \leq \lambda d \\
& \quad x_j \in \mathbb{Z}_+, \quad j = 1, \ldots, n
\end{align*}
\]

If solution to \(z_{SR}\) is feasible to original problem, then

\(z = \bar{z} = z_{SR}\)

hence problem solved to optimality

Surrogate relaxation (example)

If we surrogate relax all constraints of a BIP, then we obtain a Knapsack Problem.

\[
\begin{align*}
\text{maximize} & \quad \sum_{j=1}^{n} p_j x_j \\
\text{subject to} & \quad \sum_{j=1}^{n} w_j x_j \leq c \\
& \quad x_j \in \{0, 1\}, \quad j = 1, \ldots, n
\end{align*}
\]

---

Tightness of relaxation

\[
\begin{align*}
\text{maximize} & \quad cx \\
\text{subject to} & \quad Ax \leq b \\
& \quad Dx \leq d \\
& \quad x_j \in \mathbb{Z}_+, \quad j = 1, \ldots, n
\end{align*}
\]

\[
\max \left\{ cx : x \in \text{conv}(Ax \leq b, Dx \leq d, x \in \mathbb{Z}_+) \right\}
\]

Lagrange Relaxation, best multipliers \(\lambda \geq 0\)

\[
\begin{align*}
\text{maximize} & \quad z_{LR}(\lambda) = cx - \lambda(Dx - d) \\
\text{subject to} & \quad Ax \leq b \\
& \quad x_j \in \mathbb{Z}_+, \quad j = 1, \ldots, n
\end{align*}
\]

\[
\max \left\{ cx : Dx \leq d, x \in \text{conv}(Ax \leq b, x \in \mathbb{Z}_+) \right\}
\]

Surrogate Relaxation, best multipliers \(\lambda \geq 0\)

\[
\begin{align*}
\text{maximize} & \quad z_{SR}(\lambda) = cx \\
\text{subject to} & \quad Ax \leq b \\
& \quad \lambda Dx \leq \lambda d \\
& \quad x_j \in \mathbb{Z}_+, \quad j = 1, \ldots, n
\end{align*}
\]

\[
\max \left\{ cx : x \in \text{conv}(Ax \leq b, \lambda Dx \leq \lambda d, x \in \mathbb{Z}_+) \right\}
\]

Best surrogate relax. is tighter than best Lagrange relax.
Relaxation strategies

Which constraints should be relaxed

- “the complicating ones”
- remaining problem is polynomially solvable (e.g. min spanning tree, assignment problem, linear programming)
- remaining problem is totally unimodular (e.g. network problems)
- remaining problem is NP-hard but good techniques exist (e.g. knapsack)
- constraints which cannot be expressed in MIP terms (e.g. cutting)
- constraints which are too extensive to express (e.g. subtour elimination in TSP)

Subgradient optimization Lagrange multipliers

(Subsimilar technique can be used for surrogate multipliers)

\[
\text{maximize} \quad cx \\
\text{subject to} \quad Ax \leq b \\
\quad \quad \quad Dx \leq d \\
\quad \quad \quad x_j \in \mathbb{Z}_+, \quad j = 1, \ldots, n
\]

Lagrange Relaxation, multipliers \( \lambda \geq 0 \)

\[
\text{maximize} \quad z_{LR}(\lambda) = cx - \lambda(Dx - d) \\
\text{subject to} \quad Ax \leq b \\
\quad \quad \quad x_j \in \mathbb{Z}_+, \quad j = 1, \ldots, n
\]

Lagrange Dual Problem

\[
z_{LD} = \min_{\lambda \geq 0} z_{LR}(\lambda)
\]

- We do not need best multipliers in B&B algorithm
- Subgradient optimization fast method
- Methods for minimizing a convex, possibly non-differentiable function over a convex and closed set
- Roots in nonlinear programming, Held and Karp (1971)

Subgradient optimization, motivation

Generalization of gradients to non-differentiable functions.

**Definition 1** An \( m \)-vector \( \gamma \) is subgradient of \( f(\lambda) \) at \( \lambda = \lambda \) if

\[
f(\lambda) \geq f(\alpha) + \gamma(\lambda - \lambda)
\]

The inequality says that the hyperplane

\[
y = f(\alpha) + \gamma(\lambda - \lambda)
\]

is tangent to \( y = f(\lambda) \) at \( \lambda = \lambda \) and supports \( f(\lambda) \) from below.
Proposition 2 Given a choice of nonnegative multipliers $\lambda$. If $x'$ is an optimal solution to $z_{LR}(\lambda)$ then
\[ \gamma = d - Dx' \]
is a subgradient of $z_{LR}(\lambda)$ at $\lambda = \overline{\lambda}$.

Proof We wish to prove (1) which in our version is:
\[ \max_{Ax \leq b} (cx - \lambda(Dx - d)) \geq \gamma(\lambda - \overline{\lambda}) \]
where $x'$ is an opt. solution to the right-most subproblem
Inserting $\gamma$ we get:
\[ \max_{Ax \leq b} (cx - \lambda(Dx - d)) \geq (d - Dx')(\lambda - \overline{\lambda}) + (cx' - \overline{\lambda}(Dx' - d)) \]
\[ = cx' - \lambda(Dx' - d) \]
\[ \square \]

Proposition 3 Optimality condition. If the function $f(\lambda)$ is convex and a subgradient $\gamma = 0$ exists in $\lambda = \overline{\lambda}$ then $\overline{\lambda}$ is optimal.

Held and Karp
Initially
\[ \lambda^{(0)} = \{0, \ldots, 0\} \]
compute the new multipliers by recursion
\[ \lambda^{(k+1)}_i := \begin{cases} \lambda^{(k)}_i & \text{if } |\gamma_i| \leq \varepsilon \\ \max(\lambda^{(k)}_i - \theta \gamma_i, 0) & \text{if } |\gamma_i| > \varepsilon \end{cases} \]
where $\gamma$ is subgradient.
The step size $\theta$ is defined by
\[ \theta = \frac{\overline{\lambda} - \bar{z}}{\sum_i \gamma_i} \]
where $\mu$ is an appropriate constant.
E.g. $\mu = 1$ and halved if upper bound not decreased in 20 iterations

Example: Manpower planning
One of the most successful applications of OR

| Cover a number of job functions using least possible resources |

Constraints difficult to formulate
- Air-crew scheduling
- Hospital-crew scheduling
- Supermarket-crew scheduling
Model is so general that it can handle
- Assignment of teachers to classes/rooms
- Planning of transportation
Set-covering model
\[ \min \sum_{j=1}^{n} c_j x_j \]
subject to \[ \sum_{j=1}^{n} a_{ij} x_j \geq 1 \]
x$_j$ $\in \{0, 1\}$
where $a_{ij} = 1$ iff job $i$ is covered by job-schedule $j$, and $c_j$ is the cost of job-schedule $j$. 
Example: Manpower planning

\[
\begin{align*}
\text{minimize } & \quad 5x_1 + 4x_2 + 6x_3 + 3x_4 + 7x_5 + 2x_6 + 5x_7 + 3x_8 + 2x_9 + 3x_{10} \\
\text{subject to } & \quad x_1 + x_3 + x_5 + x_8 + x_9 + x_{10} \geq 1 \\
& \quad x_2 + x_3 + x_4 + x_5 + x_7 + x_8 + x_9 + x_{10} \geq 1 \\
& \quad x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_8 + x_9 + x_{10} \geq 1 \\
& \quad x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} \geq 1 \\
& \quad x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} \geq 1 \\
\text{subject to } & \quad x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10} \in \{0, 1\}
\end{align*}
\]

Optimal solution is
\[x_2 = x_5 = x_6 = 1\]

objective value 13

Notice

Minimization problem

Lagrangean dual is a maximization problem

Subgradient \(\gamma = Ax' - b\) where \(b = (1, \ldots, 1)\)

Example: Manpower planning

\[
\begin{align*}
\text{minimize } & \quad \sum_{j=1}^{n} c_j x_j - \sum_{i=1}^{m} \lambda_i \left(\sum_{j=1}^{n} a_{ij} x_j - 1\right) \\
\text{subject to } & \quad x_j \in \{0, 1\}
\end{align*}
\]

which can be reduced to

\[
\begin{align*}
\text{minimize } & \quad \sum_{j=1}^{n} \left(c_j - \sum_{i=1}^{m} \lambda_i a_{ij}\right) x_j + \sum_{i=1}^{m} \lambda_i \\
\text{subject to } & \quad x_j \in \{0, 1\}
\end{align*}
\]

Latter problem is easily solved by inspection

- \(c_j - \sum_{i=1}^{m} \lambda_i a_{ij} < 0\) then \(x_j = 1\)
- \(c_j - \sum_{i=1}^{m} \lambda_i a_{ij} > 0\) then \(x_j = 0\)
- \(c_j - \sum_{i=1}^{m} \lambda_i a_{ij} = 0\) then \(x_j = 1\) or 0

Remaining constraints define the convex hull, hence best choice of \(\lambda\) corresponds to dual variables

Many manpower problems are so large that they cannot be solved by LP solvers

Example: Manpower planning

**Iteration 1**

\[
\begin{align*}
\text{minimize } & \quad x_1 - x_2 + 2x_3 + 3x_4 + 4x_5 - x_6 + 3x_7 - 2x_8 + x_9 - x_{10} + 8 \\
\text{subject to } & \quad x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10} \in \{0, 1\}
\end{align*}
\]

Optimal solution \(x' = (0, 1, 0, 1, 0, 1, 0, 1, 0, 1)\), infeasible

\(\bar{z} = 3, \bar{z} = 40\)

\[\gamma^{(1)} = Ax' - b = (0, 3, 1, 3, 1, 1, -1, 4),\]

\(\theta = 0.19, \lambda^{(2)} = (1.00, 0.42, 0.81, 0.42, 0.81, 0.81, 1.19, 0.22)\)

**Iteration 2**

\[
\begin{align*}
\text{minimize } & \quad 2.56x_1 + 1.34x_2 + 3.36x_3 + 1.56x_4 + 4.39x_5 + 0.56x_6 + 3.78x_7 + 0.14x_8 + 1.19x_9 + 1.14x_{10} + 5.66 \\
\text{subject to } & \quad x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10} \in \{0, 1\}
\end{align*}
\]

Optimal solution \(x' = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0)\), infeasible

\(\bar{z} = 5.66, \bar{z} = 40\)

\[\gamma^{(1)} = Ax' - b = (-1, -1, -1, -1, -1, -1, -1, -1),\]

\(\theta = 0.86, \lambda^{(3)} = (1.86, 1.27, 1.66, 1.27, 1.66, 1.66, 2.05, 1.08)\)
Lagrange Relaxation

Development of $\bar{z}$ and $\bar{\bar{z}}$.

After 30 iterations we have $\bar{z} = \bar{\bar{z}} = 13$

$\lambda = (1.85, 0, 2.77, 0, 1.13, 2.99, 6.11, 0)$

Dual variables

$y = (0, 0, 2, 0, 1, 3, 7, 0)$

Lagrange relaxation and LP

For an LP-problem where we Lagrange relax all constraints

- Dual variables are best choice of Lagrange multipliers
- Lagrange relaxation and LP “relaxation” give same bound

Gives a clue to solve LP-problems without Simplex

- Iterative algorithms
- Polynomial algorithms