Network Flows

6. Lagrangian Relaxation
6.2 Solving the Lagrangian Multiplier Problem

Fall 2010
Instructor: Dr. Masoud Yaghini
Outline

- Introduction
- Subgradient Optimisation Technique
- Subgradient Optimization and Inequality Constraints
Introduction
Solving the Lagrangian Multiplier Problem

- Consider the *constrained shortest path problem*
- Suppose that now we have a time limitation of $T = 14$ instead of $T = 10$.
- When we relax the time constraint, the *Lagrangian multiplier function* $L(\mu)$ becomes:

\[
L(\mu) = \min\{c_P + \mu(t_P - T) : P \in \mathcal{P}\}.
\]

where, $\mathcal{P}$ is the collection of all directed paths from the source node 1 to the sink node $n$. 
Solving the Lagrangian Multiplier Problem

- For convenience, we refer to the quantity \( c_p + \mu(t_P - T) \) as the *composite cost* of the path \( P \).
- For a specific value of the Lagrangian multiplier \( \mu \), we can solve \( L(\mu) \) by enumerating all the directed paths in \( P \) and choosing the path with the smallest composite cost.
- We can solve the Lagrangian multiplier problem by determining \( L(\mu) \) for all nonnegative values of the Lagrangian multiplier \( \mu \) and choosing the value that achieves \( \max_{\mu \geq 0} L(\mu) \).
Solving the Lagrangian Multiplier Problem

- Path cost and time data with $T = 14$

<table>
<thead>
<tr>
<th>Path $P$</th>
<th>Path cost $c_P$</th>
<th>Path time $t_P$</th>
<th>Composite cost $c_P + \mu (t_P - T)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1–2–4–6</td>
<td>3</td>
<td>18</td>
<td>$3 + 4\mu$</td>
</tr>
<tr>
<td>1–2–5–6</td>
<td>5</td>
<td>15</td>
<td>$5 + \mu$</td>
</tr>
<tr>
<td>1–2–4–5–6</td>
<td>14</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>1–3–2–4–6</td>
<td>13</td>
<td>13</td>
<td>$13 - \mu$</td>
</tr>
<tr>
<td>1–3–2–5–6</td>
<td>15</td>
<td>10</td>
<td>$15 - 4\mu$</td>
</tr>
<tr>
<td>1–3–2–4–5–6</td>
<td>24</td>
<td>9</td>
<td>$24 - 5\mu$</td>
</tr>
<tr>
<td>1–3–4–6</td>
<td>16</td>
<td>17</td>
<td>$16 + 3\mu$</td>
</tr>
<tr>
<td>1–3–4–5–6</td>
<td>27</td>
<td>13</td>
<td>$27 - \mu$</td>
</tr>
<tr>
<td>1–3–5–6</td>
<td>24</td>
<td>8</td>
<td>$24 - 6\mu$</td>
</tr>
</tbody>
</table>
Solving the Lagrangian Multiplier Problem

- The composite cost \( c_P + \mu(t_P - T) \) for any path \( P \) is a linear function of \( \mu \) with an intercept of \( c_P \) and a slope of \( (t_P - T) \).
Solving the Lagrangian Multiplier Problem

- Lagrangian function, $T = 14$
Solving the Lagrangian Multiplier Problem

- To find the *optimal multiplier value* $\mu^*$ of the Lagrangian multiplier problem, we need to find the *highest point* of the *Lagrangian multiplier function* $L(\mu)$.

- Suppose that we consider the polyhedron defined by those points that lie on or below the function $L(\mu)$.

- These are the shaded points in the Figure.

- Then geometrically, we are finding the highest point in a polyhedron defined by the function $L(\mu)$, which is a linear program.
Solving the Lagrangian Multiplier Problem

- Consider the generic optimization model (P), defined

\[ \min \{ cx : Ax = b, \ x \in X \} \]

- and suppose that the set \( X \) is finite.

\[ X = \{ x^1, x^2, \ldots, x^K \} \]

- By relaxing the constraints \( Ax = b \), we obtain the Lagrangian multiplier function:

\[ L(\mu) = \min \{ cx + \mu( Ax - b ) : x \in X \} \]

- By definition:

\[ L(\mu) \leq cx^k + \mu( Ax^k - b ) \quad \text{for all } k = 1, 2, \ldots, K. \]
Solving the Lagrangian Multiplier Problem

- In the space of composite costs and Lagrange multipliers $\mu$, each function $cx^k + \mu(Ax^k - b)$ is a multidimensional "line" called a hyperplane (if $\mu$ is two-dimensional, it is a plane).

- The Lagrangian multiplier function $L(\mu)$ is the lower envelope of the hyperplanes $cx^k + \mu(Ax^k - b)$ for $k = 1, 2, \ldots, K$.

- In the Lagrangian multiplier problem, we wish to determine the highest point on this envelope.
Solving the Lagrangian Multiplier Problem

We can find this the highest point by solving the optimization problem:

Maximize \( w \)

subject to

\[ w \leq cx^k + \mu(\mathcal{A}x^k - b) \quad \text{for all } k = 1, 2, \ldots, K, \]
\[ \mu \text{ unrestricted,} \]
Solving the Lagrangian Multiplier Problem

- Theorem.

  - The Lagrangian multiplier problem $L^* = \max_{\mu} L(\mu)$ with
    \[
    L(\mu) = \min \{ cx^k + \mu(\mathbf{A}x - b) : x \in X \}
    \]
  
  - is equivalent to the linear programming problem
    \[
    L^* = \max \{ w : w \leq cx^k + \mu(\mathbf{A}x^k - b) \text{ for } k = 1, 2, \ldots, K \}.
    \]
Subgradient Optimisation Technique
Subgradient Optimisation Technique

- In solving optimization problems with the nonlinear objective function $f(x)$ of an $n$-dimensional vector $x$, researchers and practitioners often use gradient methods.
Subgradient Optimisation Technique

- Suppose that in solving the *Lagrangian multiplier problem*, we are at a point where the Lagrangian function has a unique solution $\bar{x}$.

  \[ L(\mu) = \min\{cx + \mu(\mathbf{A}x - b) : x \in X \} \]

- Since

  \[ L(\mu) = c\bar{x} + \mu(\mathbf{A}\bar{x} - b) \]

- The solution $\bar{x}$ remains optimal for small changes in the value of $\mu$, the *gradient* at this point is

  \[ \mathbf{A}\bar{x} - b \]
A gradient method would change the value of $\mu$ as follows:

$$\mu \leftarrow \mu + \theta(\mathbf{A} \mathbf{x} - \mathbf{b}).$$

- $\theta$: is a step size (a scalar) that specifies how far we move in the gradient direction.
- If $(\mathbf{A} \mathbf{x} - \mathbf{b})_i = 0$, the solution $\mathbf{x}$ uses up exactly the required units of the $i$th resource, and we hold the Lagrange multiplier (the toll) $\mu_i$ of that resource at its current value;
- If $(\mathbf{A} \mathbf{x} - \mathbf{b})_i < 0$, the solution $\mathbf{x}$ uses up less than the available units of the $i$th resource and we decrease the Lagrange multiplier $\mu_i$ on that resource;
- If $(\mathbf{A} \mathbf{x} - \mathbf{b})_i > 0$, the solution $\mathbf{x}$ uses up more than the available units of the $i$th resource and we increase the Lagrange multiplier $\mu_i$ on that resource.
To solve the Lagrangian multiplier problem, let $\mu^0$ be any initial choice of the Lagrange multiplier; we determine the subsequent values $\mu^k$ for $k = 1, 2, \ldots$, of the Lagrange multipliers as follows:

\[ \mu^{k+1} = \mu^k + \theta_k (Ax^k - b). \]

where $x^k$ is any solution to the Lagrangian subproblem when $\mu = \mu^k$ and $\theta_k$ is the step length at the $k$th iteration.
Subgradient Optimisation Technique

- To ensure that this method solves the Lagrangian multiplier problem, we need to exercise some care in the choice of the step sizes $\theta$.
  - If we choose them too small, the algorithm would become stuck at the current point and not converge;
  - If we choose the step sizes too large, the iterates $\mu^k$ might overshoot the optimal solution and perhaps even oscillate between two nonoptimal solutions.
**Subgradient Optimisation Technique**

- **Newton's method**
  - It is an important variant of the subgradient optimization procedure
  - Suppose that
    \[ L(\mu^k) = cx^k + \mu^k(Ax^k - b) \]
    - that is, \( x^k \) solves the Lagrangian subproblem when \( \mu = \mu^k \)
  - We assume that \( x^k \) continues to solve the Lagrangian subproblem as we vary \( \mu \); or, stated in another way, we make a linear approximation
    \[ r(\mu) = cx^k + \mu(Ax^k - b) \] to \( L(\mu) \)
Subgradient Optimisation Technique

- **Newton's method (cont.)**
  - Suppose that we know the optimal value $L^*$ of the Lagrangian multiplier problem (which we do not).
  - Then we might move in the subgradient direction until the value of the linear approximation exactly equals $L^*$. 
Subgradient Optimisation Technique

- The constrained shortest path example, $T=14$
Subgradient Optimisation Technique

The constrained shortest path example

- The figure shows an example of Newton’s method when applied to the constrained shortest path example, starting with $\mu^k = 0$.

- At this point, the path $P = 1-2-4-6$ solves the Lagrangian subproblem and $A x - b$ equals $t_p - T = 18 - 14 = 4$.

- Since $L^* = 7$ and the path $P$ has a cost $c_p = 3$, in accordance with this linear approximation, or Newton's method, we would approximate

$$L(\mu) \text{ by } r(\mu) = 3 + 4\mu,$$

set

$$3 + 4\mu = 7$$

and define the new value of $\mu$ as

$$\mu^{k+1} = (7 - 3)/4 = 1$$
Subgradient Optimisation Technique

• In general, we set the step length \( \theta_k \) so that

\[
r(\mu^{k+1}) = cx^k + \mu^{k+1}(Ax^k - b) = L^*,
\]

• since,

\[
\mu^{k+1} = \mu^k + \theta_k(Ax^k - b),
\]

• then,

\[
r(\mu^{k+1}) = cx^k + [\mu^k + \theta_k(Ax^k - b)](Ax^k - b) = L^*.
\]
Subgradient Optimisation Technique

- recalling that
  \[ L(\mu^k) = cx^k + \mu(AX^k - b) \]
- and letting the Euclidean norm of the vector \( y \):
  \[ \| y \| = (\sum_j y_j^2)^{1/2} \]
- we can solve for the *step length* and find that
  \[ \theta_k = \frac{L^* - L(\mu^k)}{\| AX^k - b \|^2}. \]
Subgradient Optimisation Technique

- Since we do not know the optimal objective function value $L^*$ of the Lagrangian multiplier problem,
  - practitioners of Lagrangian relaxation often use the following popular heuristic for selecting the step length:

$$
\theta_k = \frac{\lambda_k[\text{UB} - L(\mu^k)]}{\|Ax^k - b\|^2}.
$$

- $\text{UB}$: is an upper bound on the optimal objective function value $z^*$ of the problem ($P$), and so an upper bound on $L^*$ as well
- $\lambda_k$: is a scalar chosen (strictly) between 0 and 2.
Subgradient Optimisation Technique

The heuristic procedure:

- Initially, the upper bound is the objective function value of any known feasible solution to the problem (P).
- As the algorithm proceeds, if it generates a better (i.e., lower cost) feasible solution, it uses the objective function value of this solution in place of the upper bound UB.
- Usually, practitioners choose the scalars $\lambda_k$ by starting with $\lambda_k = 2$ and then reducing $\lambda_k$ whenever the best Lagrangian objective function value found so far has failed to increase in a specified number of iterations.
- Since this version of the algorithm has no convenient stopping criteria, practitioners usually terminate it after it has performed a specified number of iterations.
we might note that the subgradient optimization procedure is not the only way to solve the Lagrangian multiplier problem:

- practitioners have used a number of other heuristics, including methods known as *multiplier ascent methods*.
Subgradient Optimization and Inequality Constraints
If we apply Lagrangian relaxation to a problem with constraints $Ax - b$ stated in inequality form instead of the equality constraints, the Lagrange multipliers $\mu$ are constrained to be nonnegative.

The update formula

$$\mu^{k+1} = \mu^k + \theta_k (Ax^k - b)$$

might cause one or more of the components $\mu_i$ of $\mu$ to become negative.
Subgradient Optimization and Inequality Constraints

- To avoid this possibility, we modify the update formula as follows:

\[ \mu^{k+1} = [\mu^k + \theta_k(\mathcal{A}x^k - b)]^+ \]

- where, the notation \([y]^+\) denotes the "positive part" of the vector \(y\); that is, the \(i\)th component of \([y]^+\) equals the maximum of 0 and \(y_i\).

- Stated in another way, if the update formula

\[ \mu^{k+1} = \mu^k + \theta_k(\mathcal{A}x^k - b) \]

- would cause the \(i\)th component of \(\mu_i\) to be negative, then we simply set the value of this component to be zero.
Subgradient Optimization and Inequality Constraints

- We then implement all the other steps of the subgradient procedure exactly the same as for problems with equality constraints.
  - i.e., the choice of the step size $\eta$ at each step and
  - the solution of the Lagrangian subproblems

- For problems with both equality and inequality constraints, we use a mixture of the equality and inequality versions of the algorithm
Subgradient Optimization and Inequality Constraints

- The constrained shortest path example:
  - We start to solve our constrained shortest path problem at \( \mu^0 = 0 \) with \( \lambda^0 = 0.8 \) and with UB = 24, the cost corresponding to the shortest path 1-3-5-6.
  - Suppose that we choose to reduce the scalar \( \lambda_k \) by a factor of 2 whenever three successive iterations at a given value of \( \lambda_k \) have not improved on the best Lagrangian objective function value \( L(\mu) \).
  - The solution \( x^0 \) to the Lagrangian subproblem with \( \mu = 0 \) corresponds to the path \( P = 1\text{-}2\text{-}4\text{-}6 \), the Lagrangian subproblem has an objective function value of \( L(0) = 3 \), and the subgradient \( A x^0 - b \) at \( \mu = 0 \) is \( (t_p - 14) = 18 - 14 = 4 \).
Subgradient Optimization and Inequality Constraints

- bold lines denote the shortest path with Lagrange multiplier $\mu = 0$
So at the first step, we choose

$$\theta_k = \frac{\lambda_k [\text{UB} - L(\mu^k)]}{\| Ax^k - b \|^2}.$$  

$$\theta_0 = 0.8(24 - 3)/16 = 1.05,$$

$$\mu^{k+1} = [\mu^k + \theta_k (Ax^k - b)]^+$$

$$\mu^1 = [0 + 1.05(4)]^+ = 4.2.$$
For $\mu^1 = 4.2$, the path $P = 1-3-2-5-6$ solves the Lagrangian subproblem;

Therefore,

$$L(\mu) = \min\{c_P + \mu(t_P - T) : P \in \mathcal{P}\}.$$  

$$L(4.2) = 15 + 4.2(10) - 4.2(14) = 15 - 16.8 = -1.8,$$

And $Ax^1 - b$ equals $(t_p - 14) = 10 - 14 = -4.$

Since the path 1-3-2-5-6 is feasible, and its cost of 15 is less than $UB$, we change $UB$ to value 15.

Therefore,

$$\theta_1 = 0.8(15 + 1.8)/16 = 0.84,$$

$$\mu^2 = [4.2 + 0.84(-4)]^+ = 0.84.$$
## Subgradient Optimization and Inequality Constraints

<table>
<thead>
<tr>
<th>k</th>
<th>$\mu^k$</th>
<th>$t_p - T$</th>
<th>$L(\mu^k)$</th>
<th>$\lambda_k$</th>
<th>$\theta_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.0000</td>
<td>4</td>
<td>3.0000</td>
<td>0.80000</td>
<td>1.0500</td>
</tr>
<tr>
<td>1</td>
<td>4.2000</td>
<td>-4</td>
<td>-1.8000</td>
<td>0.80000</td>
<td>0.8400</td>
</tr>
<tr>
<td>2</td>
<td>0.8400</td>
<td>4</td>
<td>6.3600</td>
<td>0.80000</td>
<td>0.4320</td>
</tr>
<tr>
<td>3</td>
<td>2.5680</td>
<td>-4</td>
<td>4.7280</td>
<td>0.80000</td>
<td>0.5136</td>
</tr>
<tr>
<td>4</td>
<td>0.5136</td>
<td>4</td>
<td>5.0544</td>
<td>0.80000</td>
<td>0.4973</td>
</tr>
<tr>
<td>5</td>
<td>2.5027</td>
<td>-4</td>
<td>4.9891</td>
<td>0.40000</td>
<td>0.2503</td>
</tr>
<tr>
<td>6</td>
<td>1.5016</td>
<td>1</td>
<td>6.5016</td>
<td>0.40000</td>
<td>3.3993</td>
</tr>
<tr>
<td>7</td>
<td>4.9010</td>
<td>-6</td>
<td>-5.4059</td>
<td>0.40000</td>
<td>0.2267</td>
</tr>
<tr>
<td>8</td>
<td>3.5406</td>
<td>-4</td>
<td>0.8376</td>
<td>0.40000</td>
<td>0.3541</td>
</tr>
<tr>
<td>9</td>
<td>2.1244</td>
<td>-4</td>
<td>6.5026</td>
<td>0.40000</td>
<td>0.2124</td>
</tr>
<tr>
<td>10</td>
<td>1.2746</td>
<td>1</td>
<td>6.2746</td>
<td>0.40000</td>
<td>3.4902</td>
</tr>
<tr>
<td>11</td>
<td>4.7648</td>
<td>-6</td>
<td>-4.5886</td>
<td>0.40000</td>
<td>0.2177</td>
</tr>
<tr>
<td>12</td>
<td>3.4589</td>
<td>-4</td>
<td>1.1646</td>
<td>0.20000</td>
<td>0.1729</td>
</tr>
<tr>
<td>13</td>
<td>2.7671</td>
<td>-4</td>
<td>3.9316</td>
<td>0.20000</td>
<td>0.1384</td>
</tr>
<tr>
<td>14</td>
<td>2.2137</td>
<td>-4</td>
<td>6.1453</td>
<td>0.20000</td>
<td>0.1107</td>
</tr>
<tr>
<td>15</td>
<td>1.7709</td>
<td>1</td>
<td>6.7709</td>
<td>0.20000</td>
<td>1.6458</td>
</tr>
</tbody>
</table>
## Subgradient Optimization and Inequality Constraints

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>3.4167</td>
<td>−4</td>
<td></td>
<td>1.3330</td>
<td>0.20000</td>
<td>0.1708</td>
</tr>
<tr>
<td>17</td>
<td>2.7334</td>
<td>−4</td>
<td></td>
<td>4.0664</td>
<td>0.20000</td>
<td>0.1367</td>
</tr>
<tr>
<td>18</td>
<td>2.1867</td>
<td>−4</td>
<td></td>
<td>6.2531</td>
<td>0.10000</td>
<td>0.0547</td>
</tr>
<tr>
<td>19</td>
<td>1.9680</td>
<td>1</td>
<td></td>
<td>6.9680</td>
<td>0.10000</td>
<td>0.8032</td>
</tr>
<tr>
<td>20</td>
<td>2.7712</td>
<td>−4</td>
<td></td>
<td>3.9150</td>
<td>0.10000</td>
<td>0.0693</td>
</tr>
<tr>
<td>21</td>
<td>2.4941</td>
<td>−4</td>
<td></td>
<td>5.0235</td>
<td>0.10000</td>
<td>0.0624</td>
</tr>
<tr>
<td>22</td>
<td>2.2447</td>
<td>−4</td>
<td></td>
<td>6.0212</td>
<td>0.05000</td>
<td>0.0281</td>
</tr>
<tr>
<td>23</td>
<td>2.1325</td>
<td>−4</td>
<td></td>
<td>6.4701</td>
<td>0.05000</td>
<td>0.0267</td>
</tr>
<tr>
<td>24</td>
<td>2.0258</td>
<td>−4</td>
<td></td>
<td>6.8966</td>
<td>0.05000</td>
<td>0.0253</td>
</tr>
<tr>
<td>25</td>
<td>1.9246</td>
<td>1</td>
<td></td>
<td>6.9246</td>
<td>0.00250</td>
<td>0.0202</td>
</tr>
<tr>
<td>26</td>
<td>1.9447</td>
<td>1</td>
<td></td>
<td>6.9447</td>
<td>0.00250</td>
<td>0.0201</td>
</tr>
<tr>
<td>27</td>
<td>1.9649</td>
<td>1</td>
<td></td>
<td>6.9649</td>
<td>0.00250</td>
<td>0.0201</td>
</tr>
<tr>
<td>28</td>
<td>1.9850</td>
<td>1</td>
<td></td>
<td>6.9850</td>
<td>0.00250</td>
<td>0.0200</td>
</tr>
<tr>
<td>29</td>
<td>2.0050</td>
<td>−4</td>
<td></td>
<td>6.9800</td>
<td>0.00250</td>
<td>0.0013</td>
</tr>
<tr>
<td>30</td>
<td>2.0000</td>
<td>−4</td>
<td></td>
<td>7.0000</td>
<td>0.00250</td>
<td>0.0012</td>
</tr>
<tr>
<td>31</td>
<td>1.9950</td>
<td>1</td>
<td></td>
<td>6.9950</td>
<td>0.00250</td>
<td>0.0200</td>
</tr>
<tr>
<td>32</td>
<td>2.0150</td>
<td>−4</td>
<td></td>
<td>6.9400</td>
<td>0.00250</td>
<td>0.0013</td>
</tr>
<tr>
<td>33</td>
<td>2.0100</td>
<td>−4</td>
<td></td>
<td>6.9601</td>
<td>0.00125</td>
<td>0.0006</td>
</tr>
</tbody>
</table>
Subgradient Optimization and Inequality Constraints

- From iterations 2 through 5, the shortest paths alternate between the paths 1-2-4-6 and 1-3-2-5-6.
- At the end of the fifth iteration, the algorithm has not improved upon (increased) the best Lagrangian objective function value of $6.36$ for three iterations, so we reduce $\lambda_k$ by a factor of 2.
- In the next 7 iterations the shortest paths are the paths 1-2-5-6, 1-3-5-6, 1-3-2-5-6, 1-3-2-5-6, 1-2-5-6, 1-3-5-6, and 1-3-2-5-6.
Once again for three consecutive iterations, the algorithm has not improved the best Lagrangian objective function value, so we decrease $\lambda_k$ by a factor of 2 to value 0.2.

From this point on, the algorithm chooses either path $1-3-2-5-6$ or path $1-2-5-6$ as the shortest path at each step.

As we see, the Lagrangian objective function value is converging to the optimal value $L^* = 7$ and the Lagrange multiplier is converging to its optimal value of $\mu^* = 2$. 
The End