Solving Multidimensional Knapsack Problem – Generating Problem Data

Shih-Fen Cheng
School of Information Systems
Singapore Management University
sfcheng@smu.edu.sg

July 31, 2009

1 Background

In this mini project, we will look at the knapsack problem, one of the most important and well studied "hard optimization problems". Defined very generally, knapsack problem deals with the selection of items from an available pool, and the goal is to achieve highest possible value while observing the capacity limit.

The mathematical formulation of a general multi-dimensional knapsack problem is as follows:

$$\begin{aligned} \max \quad & v_1x_1 + v_2x_2 + \ldots + v_nx_n \\ s.t. \quad & \\ & w_{11}x_1 + w_{12}x_2 + \ldots + w_{1n}x_n \leq b_1 \\ & w_{21}x_1 + w_{22}x_2 + \ldots + w_{2n}x_n \leq b_2 \\ & \ldots \\ & w_{m1}x_1 + w_{m2}x_2 + \ldots + w_{mn}x_n \leq b_m \\ & 0 \leq x_i \leq u_i, \;\; x_i \text{ integer}, i = 1, 2, \ldots, n. \end{aligned} \tag{1}$$

In the above formulation, n stands for the number of tasks, m stands for the number of resources, v_i stands for the value one could obtain in completing task i, b_j stands for the capacity limit on resource j, w_{ji} stands for the consumption of resource j for task i. x_i is an integral variable, indicating how many pieces of task i to include. u_i is the upper bound on x_i. For simplicity, we assume that all problem constants (v_i, w_{ij}, b_i) are integers. We also assume that u_i could be dropped (x_i would be unbounded as a result).

2 Input Data Format

Your program should be designed to take a problem data file as input, with the following format:

```
n
m
v1 w11 w21 ... wm1
v2 w12 w22 ... wm2
...
vn w1n w2n ... wmn
b1 b2 ... bm
```

You would note that u_i is not generated. You could simply assume that x_i is unbounded. A random instance generator is available at:
`http://www.mysmu.edu/faculty/sfcheng/projects/mkp/`.

The random instance generator is constructed following the description of Bertsimas and Demir (2002), and the usage is as follows:

```
usage: genPref2 <M> <N> <A> <C> <mode> <tightness> <tag>
```

`M` and `N` are numbers of resources and tasks respectively. `A` and `C` stand for the upper bound for the resource requirement and task value respectively. `mode` is either 0 or 1, indicating whether the resource requirements and capacities are correlated or not (correlated problems will be much harder). `tightness` is a real number greater than 1, indicating how tight the knapsack constraints are, i.e., $b_j = \texttt{tightness} \cdot \sum_{i=1}^{N} w_{ji}$.

For example, if you would like to generate a single dimensional knapsack problem with 10 tasks, you can use the following command:

```
usage: genPref2 1 10 20 50 1 1.5 test1
```

The problem data generated will be stored in `test1.pref`.

3 Program Output

The output of the program should be the value of x_i's, one line for each x_i.

References

Dimitris Bertsimas and Ramazan Demir. An approximate dynamic programming approach to multidimensional knapsack problems. *Management Science*, 48(4):550–565, 2002.