**Basic Implementation of Branch and Bound for Multi-dimensional Knapsack Problem**

**Yangqi Su Yaying Shi**

The c++ program **MKSBnB** presented implements branch and bound for the multi-dimensional knapsack problem. It takes depth first traversal through the decision tree, and uses a greedy heuristic to calculate the first bound in order to start bounding as quickly as possible.

The program however can **sometimes** only take inputs of up to 100 objects, which takes around 1 second to compute, input sizes larger than 100 become computationally inefficient. Reasons for this could be as follows:

1. First heuristic solution found does not have a tight enough bound, thus too much space is searched during the process. (THIS COULD BE THE MAIN PROBLEM, since looking at estimation show that the first bound generated is much lower than the estimations, thus the traversal stack builds up very quickly).
2. Estimation of bound takes too long due to recalculation, perhaps the implementation could be sped up by traversing through the tree and changing only one value of an old estimation to generate a new estimation.
3. Better traversal method such as choosing higher estimation routes would be better.

Glpsol can take inputs up to one million and solve very quickly, given that weights and sizes are not large integers.

The problem can be generated with the python script:

python MKSLPgenerator.py number\_of\_samples weight\_constraint size\_constraint