Heuristics - HW 1

Felipe Coutinho

October 11, 2024

Representation: binary matrix A with $a_{ij} \in \{0, 1\}$, and $i = 1, \ldots, n$ representing number of elements, and $j = 1, \ldots, m$ representing number of subsets. Each column (subset) has a cost $c_j > 0$. The solution s is a list with indices for included columns.

Constructive heuristic: start with empty set, add elements progressively until full coverage

Check if solution is feasible: $\sum_{j \in s} a_{ij} = 1, \forall i$ (quick to compute)

Evaluation functions:

1. At step t, pick a column j^* such that $j^* = \arg\max_j N^{(t)}$, where $N^{(t)}$ is the number of *new* elements that adding j to s would cover.

2. At step t, pick a column j^* such that $j^* = \arg\max_j \frac{N^{(t)}}{c_j}$. Same as previous, but take into account column cost, as we've discussed in the Knapsack problem.

3. Define threshold γ and probability p. At step t, sample $u^{(t)} \sim \text{Uniform}(0, 1)$.

    - If $u^{(t)} > p$, pick a column j^* such that $j^* = \arg\max_j \frac{N^{(t)}}{c_j}$.
    - Otherwise, build set $J = \left\{ j : \frac{N^{(t)}}{c_j} \geq \gamma \times \frac{N^{(t)}}{c_j^*} \right\}$. Then, pick a random sample from J. The idea is to add randomness to try to escape from local optima (this was inspired by Stochastic Gradient Descent, as my background is ML...)

Post-processing: the redundancy elimination (RE) step does not improve approach 1, because by construction *there is no redundant element* in the final solution. However, for approaches 2 and 3, RE improves the answer *basically every time.*

Average deviation from best known solutions: Approach 1 fails miserably, on average x6 larger cost than baseline. Approaches 2 and 3 are *pretty successful*, only 11.1% and 12.5% above optimal *before post-processing.* After RE, they are pushed to 7.5% and 8.2%, respectively. Hence, post-processing *reduces cost* by around 35%.

Conclusions: approach 2 is the best, but perhaps there is still hope for approach 3 if I tinker with the values of (γ, p) enough. One option would be to perform a grid-search on the parameter space, to fine tune the solution.