LE08: Combinatorial optimization

Backtracking and branch-and-bound

Rucksack–Backtrack2, a recursive call with $x_{k+1} = 1$, is only executed if the cumulative size of the already selected items plus the size of the $(k+1)$-th item **does not esceed the total capacity** K_0. A bounding function $B(x)$ is used to reduce the search space. For maximization problems, $B(x)$ must represent an **upper bound** for the maximum value $P(x)$ of all feasible solutions in the subtree with root x. A subtree can be pruned if the computed bounding function $B(x)$. The value of the current optimal solution , $optf$, is **less than or equal** to the currently optimal value, since the optimal solution in this subtree cannot be improved. The main improvement of *Rucksack–Backtrack3* over *Rucksack–Backtrack2* is the use of a **bounding function to truncate** subtrees early. The bounding function $B(x)$ for the integer backpack problem is derived by adding the value of the partial solution $f(x)$ to the value of an optimal **rational backpack** for the remaining items. To directly determine an optimal rational backpack, the items must be **listed in descending order** according to the **ratio** f_i/g_i (value per weight) should be sorted.

Travelling Roundabout Problem (TSP)

The traveling salesman problem is computationally demanding because it is classified as **NP-hard**. A traveling salesman is mathematically defined as **as a simple circuit** in K_n containing all vertices (a Hamiltonian circuit). Since the traveling salesman problem is a minimization **problem**, a bounding function $B(x)$ must provide a **lower bound** for the cost of all possible extensions of the partial solution x. The bounding function $B(x)$ for the TSP is composed of the sum of the edges already traversed and the value of the **minimum spanning tree** of the remaining subgraph.

Heuristic algorithms

Heuristic algorithms replace backtracking when combinatorial optimization problems have a **very large seach space**. The **quality** of an approximate solution provided by a heuristic algorithm is generally unpredictable. A **neighborhood**, N, is defined as a **mapping**, $N : X \to P(X)$, that maps each solution $x \in X$ to a set of **neighboring**. For the knapsack problem, two knapsacks are considered neighboring if they differ by exactly one item, a concept known as the **Hamming distance** ($d(x, y) = 1$). In the traveling salesman problem, the neighborhood is typically defined by the **Lin-2-Opt step**.

A feasible solution $x^* \in X$ is considered a **local maximum** if $f(x^*) \geq f(x)$ for all feasible x in the **neighborhood** $N(x^*)$ holds. In the uphill method, for a feasible solution x, a neighboring feasible solution $y \in N(x)$ with a **higher value** is constructed. The uphill method is called the method of **steepest ascent if a feasible solution with maximum value** is always chosen in each neighborhood. The generic algorithm *HeuristicSearch* terminates prematurely if the heuristic H_N **no** neighboring feasible successor y is found (i.e., $y = fail$). Combinatorial optimization problems typically have **a very large number of local optima**.