CHUNG, Youngseog (丁映硕)

2016403003

Project and Work Description

My part in this project comprised of writing the simulated annealing algorithm part (in files “saoptimalboxpacker.h” and “saoptimalboxpacker.cpp”).

Overview

The simulated annealing algorithm implemented in this project efficiently and effectively outputs a new layout of the boxes repeatedly, that will be tested for better total encasing volume. The simulated annealing algorithm is implemented as a member function of the class *SAOptimalBoxPacker* and utilizes a temperature variable that effectively limits the number of iterations and runtime, and also an acceptance probability function that is influenced by the temperature and that decides whether an inferior case will be accepted in order to search for a globally efficient case.

The *SAOptimalConfig* class contains essential parameters for the *SAOptimalBoxPacker* class. It is through the *SAOptimalConfig* class that the user can decide the minimum and maximum temperatures (*T* and *T_min*), as well as the increment size of the temperature (*alpha*) and the constant used in the acceptance probability function (*gamma*).

The *SAOptimalBoxPacker* takes in the configurations specified through the *SAOptimalConfig* class and its main functionality is the *packBoxes* function, where the simulated annealing algorithm is implemented.

packBoxes function

The outer for loop with iterator integer *t* is for extra iteration purposes, and at the same time modifies the starting temperature of each iteration so that the starting temperature is steadily decreased, but is floored at a minimal level, i.e. the number of iterations carried out at each temperature decreases but has a minimum number of sufficient iterations.

At each temperature, 100 or 200 perturbations of the boxes are carried out, and in each case a *BoxPerturber* instance *tPerturber* is created. *tPerturber* creates a neighboring new layout of the boxes. The neighboring aspect is enforced by the perturbation index, which is correlated with the temperature at each iteration. If the new random layout case is more efficient than the old layout, or if the acceptance probability of the case is bigger than a randomly generated

number between 0 and 1, than the new layout is accepted and the current best scenarios (cost and perturber) are replaced by the new cost and perturber (variables *tCost* and *tPerturber*).

This process is terminated as soon as the temperature *T* reaches the minimal temperature *T_min* as it is repeatedly incremented by **alpha*. After each iteration through the temperature range, if the *bestCost* found is better than the previously found *bestCost* that was stored in *returnCost*, *returnCost* is replaced with the *bestCost* and same goes for the *returnPerturber* as well.

Ultimately, the *returnPerturber* and *returnCost* are parameters that will be returned or saved through this function.

Reasoning

The point in having simulated annealing is its efficiency in finding a good solution in a given unit of time. Of course, the optimal solution would be to iterate through the whole solution space, which is O(n!$\frac{3^{3n}}{2^{2n}n^{1.5}}$) big with n modules, but the sheer runtime will be incalculable with many boxes and the process will run seemingly forever.

Therefore, a random sampling procedure is introduced to collect data on random points through the solution space, but to systemize this process, neighboring cases of a random sample will be tested to find a local optimum. However, because the local optimum may not be the global optimum, an acceptance probability that depends on the temperature of each iteration and the difference in cost is utilized to probabilistically adopt an inferior case to search the premises of the new inferior case for another local optimum. These comparisons of these local optimums take place and in the given time range defined by the starting and minimum temperature, the best among the local optimums is outputted. This method has been proven to output much more efficient results in the same runtime compared with other search algorithms, and thus has been utilized in this project as well.