Lumber Cutting Sequence Optimization with Simulated Annealing

Youyou Yang

October 18, 2023

1 Introduction

The lumber cutting process optimization problem involves finding an optimal sequence of cutting patterns to minimize the table length required during the process. Given a set of cutting patterns and the lengths of pieces to be obtained from each pattern, the goal is to minimize the maximum table length required at any point in the process.

2 Background on Simulated Annealing

The nature of the problem lends itself to a heuristic optimization approach. A suitable algorithm for this task is Simulated Annealing (SA). Simulated Annealing is a probabilistic optimization algorithm that explores the solution space by accepting both better and worse solutions with a certain probability, which decreases over time. This allows the algorithm to escape local minima and converge to a global minimum with high probability.

The SA algorithm operates as follows:

1. **Initialization**: Start with a random solution and set a high initial temperature.
2. **Iteration**:
   - Generate a neighboring solution by a small random alteration of the current solution.
   - Evaluate the energy difference between the current solution and the neighbor, where energy corresponds to the table length in our case.
   - Accept the neighbor solution with a probability defined by the Metropolis criterion: $P(\Delta E) = e^{-\Delta E/T}$, where ΔE is the energy difference, and T is the temperature.
   - Gradually lower the temperature and repeat the above steps until a termination condition is met.

Mathematically, the acceptance probability of a new solution x_{new} from a current solution x_{current} is given by:

$$P(x_{\text{new}}, x_{\text{current}}) = \exp\left(\frac{f(x_{\text{current}}) - f(x_{\text{new}})}{T}\right) \tag{1}$$

where T is the temperature, and $f(x)$ is the objective function value of solution x. The temperature decreases over time following a cooling schedule, reducing the acceptance probability of worse solutions, guiding the search towards the global optimum.

3 Algorithm Implementation

The implementation comprises two main functions: `simulate_cutting`, which simulates the cutting process given an order of cutting patterns, and `simulated_annealing`, which employs the SA algorithm to find an optimal order of cutting patterns to minimize the table length. The algorithm explores different permutations of the pattern execution order, and the acceptance criteria and cooling schedule follow the standard simulated annealing procedure.

4 Results and Discussion

4.1 Initial Case

The optimization process begins with defining the cutting patterns and their respective lengths. In this case, the cutting patterns are represented as a dictionary, ‘cut_patterns‘, where each key-value pair represents a pattern name and a list of lengths, respectively. Here's the representation of the cutting patterns:

```
cut_patterns = {
    "Pattern 1": [1250, 1250, 3000, 5000],
    "Pattern 2": [1250, 1250, 3000, 3000, 2000],
    "Pattern 3": [2500, 2500, 2500],
    "Pattern 4": [3500, 2500, 5000],
    "Pattern 5": [7500, 3500, 5000],
    "Pattern 6": [7500, 3500, 3000, 1000, 1000]
}
```

Each key in the ‘cut_patterns‘ dictionary represents a unique cutting pattern, and the associated list of values represents the lengths of the pieces obtained from the cutting pattern. The goal is to find an optimal order of executing these cutting patterns to minimize the table length required.

The Simulated Annealing algorithm was employed to optimize the order in which these patterns are executed. The algorithm yielded an optimal order: ['Pattern 6', 'Pattern 5', 'Pattern 4', 'Pattern 1', 'Pattern 3', 'Pattern 2'], with a corresponding minimal table length of 30000 mm.

The minimal table length of 30000 mm is obtained by simulating the cutting process iteratively, adjusting the order of patterns, and calculating the table length at each iteration. This value represents the smallest table length that can accommodate the cutting patterns in the optimized order, ensuring that all pieces are cut with the least amount of waste.

The convergence of the algorithm is illustrated in Figure 1, showing that the optimization primarily occurs within the first 200 iterations.

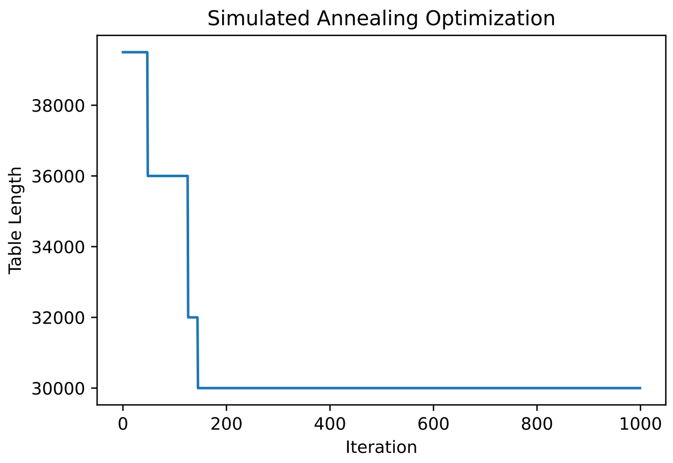

Figure 1: Simulated Annealing Optimization Convergence for initial case. Converge mainly in first 200 iterations.

4.2 Extended Case

The extended case entails a more complex set of cutting patterns, represented as a list of lists, with each inner list containing the lengths of pieces in a pattern. The same Simulated Annealing algorithm was applied to optimize the order of pattern execution. The algorithm converges to an optimal order: [8, 19, 12, 21, 13, 0, 5, 20, 18, 11, 24, 9, 16, 1, 7, 15, 10, 4, 6, 17, 22, 23, 14, 25, 2, 3], with a corresponding minimal table length of 29830 mm within the first 1000 iterations, as depicted in Figure 2.

This minimal table length of 29830 mm is derived through a similar iterative simulation of the cutting process, adjusting the order of patterns, and evaluating the table length at each step. The value signifies the shortest table length required to execute all cutting patterns in the optimized order, minimizing waste.

The convergence in the extended case is illustrated in Figure 2, indicating that the optimization mainly occurs within the first 1000 iterations.

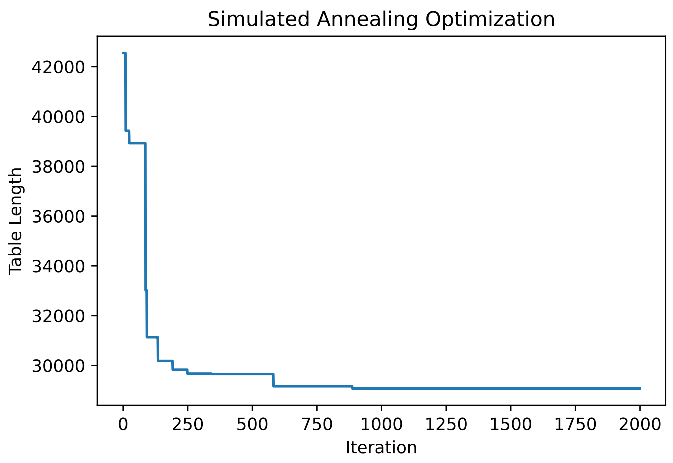

Figure 2: Simulated Annealing Optimization Convergence for extended case. Converge mainly in first 1000 iterations.

5 Conclusion

The simulated annealing algorithm provides a robust method for optimizing the sequence of cutting patterns to minimize the required table length. The visual representation further enhances the understanding of the algorithm's performance over iterations, showcasing the optimization in space utilization crucial for operational efficiency in the lumber cutting process.