André Gonçalves Pinto
Luis Vieira Relvas
Rodrigo Moisés Baptista Ribeiro

Group_A1_48

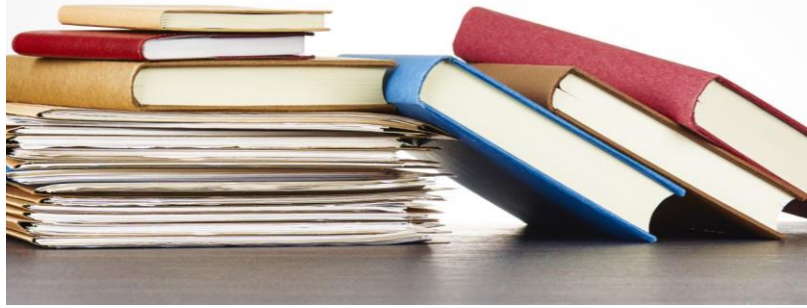
Book Scanning

Introduction

- Given a description of libraries and books available, plan which books to scan from which library to maximize the total score of all scanned books, taking into account that each library needs to be signed up before it can ship books.
- Python
- Hill Climbing Best Neighbor
- Simulated Annealling Random
- Tabu Search

Problem Statements

- L libraries;
- N books;
- T days required for signup;
- M Books that can be shipped per day (after the signup process is done);
- Objective: Maximize the number of books that can be scanned before Time is over;
- Constraints:
  - Each library can only be scanned/signup once;
  - The scan of the books can only be done after the library signup;
  - The number of books that we can ship per day is defined for each library;



State Representation and Initial State

- State Representation
  - [LE,BE,BS,LS,CD]
  - LE -> Libraries Explored : list
  - BE -> Books Explored : dict {key=library_id,value=[books_explored]}
  - BS -> Books Scores : dict {key = book_id, value=score}
  - LS ->Library Signing : list
  - CD -> Current Day : integer

- Initial State
  - [LE,BE,BS,LS,CD]
  - LE = []
  - BE = {}
  - BS = {Book Scores}
  - LS = []
  - CD = total_days

Hill Climbing

- We initially implemented a Hill Climbing algorithm. This algorithm starts with a randomized starting solution and iteratively explores neighboring states, seeking the neighbor that maximizes the total score. Operations available to the algorithm:
    - Swap books explored, add/remove books from the books explored dict, add/remove libraries from the library explored list and add/remove libraries from the library signing list.
- After a comprehensive neighbor search, the algorithm identifies feasible neighbors and selects the one with the highest score to compare with the current solution.
- If a best neighbor is found, it becomes the new current state, and the process repeats.
- If no better neighbors exist, the algorithm, ends.
- Drawbacks:
    - Local Optima : Hill Climbing algorithms are susceptible to becoming trapped in local optima, which are solutions better than their immediate neighbors but not globally optimal.
    - Computational Cost: Evaluating all potential neighbors in each iteration can be computationally expensive, especially for large and complex problems.

Simulated Annealing

- Our Simulated Annealing algorithm aims to find better solutions by strategically accepting worse solutions during its search, increasing the chance of escaping local optima.
- Begins with an initial randomized solution, and calculated the initial temperature, the temperature is used to control the probability of accepting worse solutions.
- Randomly select a subset of operations (same as Hill Climbing) and applies them to the current state, generating multiple potential neighbors. After this, calculates the score of each neighbor and the current state.
- If a neighbor has a better score, it is automatically accepted as the new current state, although if it has a worse score, it might still be accepted based on a calculated probability that depends on the score difference and the current temperature.
- The temperature gradually decreases, reducing the likelihood of accepting worse solution over time.
- The algorithm stops when the temperature reaches a near-zero value or if there hasn’t been an improvement within a specified number of iterations.
- Drawbacks:
    - Initial Temperature Calculation : If the initial neighbors are of poor quality, the algorithm mught start with na inappropriatly high or low temperature;
    - Neighbor Selection : The code chooses the neighbors randomly, so we can’t predict the “path”;

Tabu Search

- This algorithm begins with a randomized initial solution, and an empty list to store recently visited states.
- Starts by applying a set of heuristic functions (similar to the hill climbing) to the current solution and generates a list of potential neighboring states.
- Determines the best neighbor based on their total score and updated the tabu list by adding the next solution and removing the oldest entry if the list exceeds the tabu size.
- Sets the current solution to the next solution and updated the best solution if a better neighbor is found
- This algorithm is going to stp after a maximum number of iterations or if no neighbors are found.
- Drawbacks:
    - Tabu List Size : The fixed size might be too restrictive or too leniente depending on the problem.
    - Heuristic Reliance : The effectiveness of Tabu Search heavily depends on well-designed heuristics. Poor heuristics may lead to limited neighbor quality .

Conclusion

- All three algorithms (Hill Climbing, Simulated Annealing and Tabu Search) currently begin with randomized initial solutions. This means the starting point for exploration is diferente each time, potentially influencing the quality of the final solution. Additionally, like the knapsack problem, the concept of “current day” acts as a constraint – decisions need to fit within the available time.
- The choice of algorithm depends on the trade-off we are willing to make between speed and solution quality. For complex scenarios where getting stuck in local optima is a significant concern, Simulated Annealing or Tabu Search could offer advantages over Hill Climbing.