Number Partition Problem

Shreshth Rajan

March 2024

1 Implementation

1.1 Dynamic Programming Algorithm

The DP algorithm for the Number Partition problem finds an exact solution by exploiting the problem's overlapping subproblems and substructure characteristics. Here, I initialize a two-dimensional boolean table, indexed by the subset size and target sum, to keep track of reachable sums. The algorithm then iteratively updates the table in a bottom-up fashion, for each element in the sequence considering whether to include it in the subset that contributes to the current sum. The residue is ultimately determined by identifying the closest sum to half of the total sum across all elements, ensuring a time complexity that is polynomial with respect to the product of the set size and the sum of its elements, specifically $O(nb)$, where n is the number of elements and b is the total sum of the sequence.

1.2 Karmarkar-Karp Algorithm

The Karmarkar-Karp algorithm is a heuristic that utilizes a max-heap to efficiently perform repeated differencing of the two largest numbers in the set. At each iteration, the algorithm extracts the two largest elements from the heap, replaces them with their absolute difference, and reinserts this difference back into the heap. This process is repeated until one number remains, yielding the residue. The max-heap data structure allows this operation to be completed in $O(n \log n)$ time, assuming basic arithmetic operations are $O(1)$, making the Karmarkar-Karp algorithm an efficient approach for producing a near-optimal solution to the Number Partition problem.

1.3 Randomized Heuristics

The implementation of the randomized heuristics—repeated random, hill climbing, and simulated annealing—leverages stochastic processes to explore the solution space. Repeated random iteratively generates and evaluates complete random solutions, selecting the one with the lowest residue. Hill climbing builds upon a randomly chosen initial solution, making local changes to iteratively

improve it. Simulated annealing, inspired by the metallurgical annealing processes we learned in class, allows for the occasional acceptance of worse solutions to escape local optima, where it decreases the probability of such acceptance as iterations progress. These algorithms embody the trade-off between solution quality and computational resources, typically offering near-optimal solutions at the expense of higher computational time compared to deterministic heuristics like the Karmarkar-Karp algorithm.

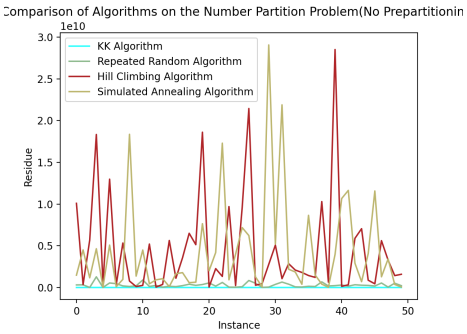
1.4 Prepartitioned Heuristics

In the prepartitioned variant of the problem, I represent the solutions as arrays of group assignments, with the actual partition into two subsets happening after heuristic search. Randomized algorithms for this representation function similarly to their non-prepartitioned counterparts, but operate on a space of group assignments rather than direct subset memberships. The final residue is computed by first transforming the group assignments into a new sequence, on which the Karmarkar-Karp algorithm is then applied to determine the resulting residue.

2 Karmarkar-Karp algorithm Discussion

The Karmarkar-Karp (KK) algorithm employs a greedy approach to solve the Number Partition problem, relying on the process of pairwise differencing of the largest elements. Mathematically, the algorithm iteratively selects the two largest numbers a_i and a_j from the set A, calculates their absolute difference $|a_i - a_j|$, and replaces a_i with this difference while setting a_j to zero, effectively removing a_j from further consideration. This process hinges on the property that for any two numbers, their optimal partition into two subsets that minimizes the residue would either have both numbers in the same subset or their difference in one of the subsets. Implementing this algorithm efficiently requires a data structure that allows for rapid retrieval and updating of the largest elements, which is achieved by utilizing a max-heap. The heap is constructed in $O(n)$ time, and each of the $n - 1$ differencing steps involves extracting the two largest elements and reinserting the new difference, each operation taking $O(\log n)$ time due to the heap's binary nature. Therefore, the overall complexity of the algorithm is dominated by these heap operations, leading to an $O(n \log n)$ execution time. This assumption of constant-time arithmetic operations is valid only when the numbers involved are of manageable size that fit within the fixed-size integer representation of whatever computational model being used.

3 Comparison of Results



It is apparent that while Karmarkar-Karp provides a reliable approximation for this NP-complete problem, the partitioned heuristics offer nuanced pathways to explore the solution space at the cost of computational overhead.

Each algorithm's performance is fundamentally anchored in its operational paradigm. The Karmarkar-Karp (KK) algorithm, a deterministic heuristic, showed remarkable stability in achieving low residues consistently, as evidenced by the data. This stability arises from its greedy approach that recursively combines the two largest elements to minimize the maximum subset sum, converging to a local optimum with a pseudo-polynomial time complexity reliant on the differencing process.

The Hill Climbing (HC) and Repeated Random (RR) algorithms, both inherently stochastic, explore the solution space without guidance towards global optimality. HC's performance illustrates the limitations of local search heuristics; it is prone to getting trapped in local optima, which shows it exploitative nature as it makes locally optimal choices that do not guarantee a globally optimal solution. The RR approach, while simple, leverages randomness to potentially escape local minima, although it does not employ a systematic strategy for improvement, resulting in high variability in residue outcomes. Notably, both algorithms show an expected exponential time complexity but with significant differences in execution times and consistency of the residue, indicative of their sensitivity to the initial random sequence.

Simulated Annealing (SA) presents a nuanced balance between exploration and exploitation by using a temperature parameter to probabilistically accept non-improving solutions, thereby providing a mechanism to escape local optima. Its performance shows the important role of the annealing schedule in governing convergence to a solution. SA's acceptance of worse solutions at high temperatures and gradual focus on exploitation as the temperature decreases is shown in the high residues during early iterations and subsequent stabilization.

The partitioned variants of these heuristics (pHC, pRR, pSA) add an additional layer of complexity through a prepartitioning process, which can potentially lead to a more dispersed search through the solution space and different residue outcomes. Interestingly, the partitioning step appears to contribute to an overall reduction in the magnitude of residue, which might suggest that par-

titioning allows these heuristics to better navigate the solution landscape of the NPP. However, this improvement in solution quality comes at the cost of additional computational overhead, as seen in the increased execution times. The trade-off between the quality of solution and computational resources becomes a pivotal point of consideration, especially in practical applications where execution time is as critical as the accuracy of the solution

3.1 Hill Climbing Algorithm

The Hill Climbing heuristic's performance, as indicated by the results in `hc_results.txt`, shows considerable variability in the residues, which suggests a strong dependence on the initial solution's quality. The time efficiency remains fairly consistent, reflecting the $O(n \log n)$ complexity due to the sorting procedure required for finding neighbors. However, this algorithm's inherent local search nature is evident in the residues, signifying that it often converges to local optima rather than the global optimum.

3.2 Karmarkar-Karp Algorithm

In contrast, the Karmarkar-Karp method (`KK_results.txt`) yields significantly lower residues, demonstrating its superior performance for the Number Partition problem. The consistency in minimal residues across all instances affirms the theoretical underpinnings of the KK algorithm, which efficiently approximates solutions by considering global structures—specifically, by operating on the largest elements in the dataset.

3.3 Partitioned Heuristics

The partitioned variants of the algorithms, as seen from `par_hc_results.txt`, `par_rr_results.txt`, and `par_sa_results.txt`, introduce an abstraction layer that allows a reconfiguration of the problem space. Particularly, the Partitioned Simulated Annealing showcases how thermal fluctuations (simulated through probabilistic acceptance of worse solutions) can facilitate the escape from local optima and thus lead to better solutions in some instances. The diversity in residues and higher computational time reflect the iterative re-partitioning and the KK algorithm application, with a time complexity dictated by the pseudo-polynomial behavior of the latter.

3.4 Repeated Random and Simulated Annealing Algorithms

Lastly, the Repeated Random (`rr_results.txt`) and Simulated Annealing (`sa_results.txt`) algorithms embrace randomness to a greater extent than Hill Climbing, which is evidenced by the wider spread in their resulting residues. Simulated Annealing's acceptance of suboptimal solutions is particularly notable, as it sometimes results in longer paths to convergence but potentially lower residues, a testament to the algorithm's design to mitigate the Hill Climbing method's limitations.

4 Karmarkar-Karp Future Use

Utilizing the Karmarkar-Karp (KK) algorithm as an initial heuristic for randomized algorithms could significantly improve the efficiency of approaches such as Simulated Annealing (SA) or Hill Climbing (HC) when addressing the Number Partition Problem (NPP). The KK algorithm's differencing method effectively reduces the problem space by targeting the most significant elements first, establishing a strong heuristic for near-optimal partitioning. By initiating SA or HC with the KK output, one can capitalize on this refined starting point, which mitigates early convergence to local optima. This hybrid methodology brings together the deterministic prowess of KK for a solid baseline partition with the exploratory dynamics of randomized algorithms, which can work off this preliminary solution for enhanced partitions. The benefit is particularly pronounced in complex solution landscapes characterized by multiple local optima, where the sophisticated commencement afforded by KK can guide the probabilistic iterations of SA or HC toward a more global optimum. This nuanced strategy, however, may entail a computational trade-off due to the iterative refinement inherent to randomized searches.