Optimizing Multi-Way Number Partitioning

Cheng Qiu

November 29, 2024

Abstract

The NP-Complete Number Partitioning Problem divides a given multiset of positive integers into subsets where the difference between the sum of each partition is minimized. There are effective approximation algorithms published in recent literature for this problem such as the Greedy heuristic and the KK Heuristic that finds sub-optimal solutions. In this paper, I propose a new approximation partitioning algorithm base on the Greedy Heuristic, the Entropy Approximation Algorithm (EAA). In most of the test cases performed, EAA consistently proved to be more effective and as accurate as the other approximation algorithms in its run-time and accuracy.

1 Introduction

The Number Partitioning Problem is a well-studied combinatorial optimization problem with many approximations and exact solutions. It is part of a class of computational problems called NP-Complete Problems [2]. This paper will mainly focus on a specific variant of the number partitioning problem, the Multi-Way Partitioning Problem.

The Number Partitioning Problem is defined as the following: a multiset S of cardinality N and a positive number of partitions K. The problem is to divide S into K partitions that minimize the sum difference between each subset. For example, given the multiset 5,5,5,4,4,3,3,1 and $K = 3$, an optimal solution to this number partitioning problem is 5,5 and 5,4,1 and 4,3,3. Notice that all three partitions have the sum of 10.

Some of the effective approaches to number partitioning problems include Greedy Number Partitioning and Karmarkar–Karp Heuristic (KK Heuristic), which is also known as the largest differencing method. Both of these solutions are approximation algorithms that focus on finding sub-optimal solutions (solutions where the sum difference between each subset is not minimized).

Real-world applications of the number partitioning problem includes multiprocessing scheduling, voting manipulation, and many others. In multiprocessing scheduling, the input set corresponds to the run-time of a set of jobs and the number of partitions K corresponds to the K number of machines. The goal of the multiprocessing scheduling is to distribute each job to each partition so that the time it takes to complete that set of jobs is minimized [1, 6]. Coming up with more effective partitioning algorithms will be able to provide more optimized solutions to solve problems such as allocation of resources.

This paper will describe in-depth some of the approximation algorithms used to find sub-optimal solutions and relevant problems to the Number Partitioning Problem in Section 2. Section 3 will propose a more efficient algorithm for solving the number partitioning problem. This section will also introduce the experimental metrics used to test the new algorithms against current algorithms and also detail the experimental results used to compare the different algorithms.

2 Prior Works

This section will discuss a closely related problem to the number partitioning problem. Due to the NP-Complete status of the Number Partitioning Problem, the exact algorithms for the Number Partitioning Problem will take exponential time and a significant amount of computational resources to find optimal solutions for multisets with higher cardinality. For this reason, this paper will focus primarily on describing and exploring approximation algorithms used to find sub-optimal solutions.

2.1 Subset-Sum Problem

Given a multiset of integer S with cardinality N and a target sum of T, the subset sum problem seeks to find subsets of S that have the sum closest to T and do not exceed T [7]. For example, given a multiset of 2,2,3,4,5,6,7 and $T = 15$, one possible optimal solution could be 6,5,2,2 which totals up to 15. This problem answers the question, is it possible to construct a subset from S that has a sum that is close or equal to T? Finding solutions to the two-way partitioning problem is equivalent to finding solutions to the subset sum problem with the target sum of $Sum(S)/K$ [6]. Many of the algorithms used to solve the subset sum problem and two way partitioning problem are used to solve the multi-way number partitioning problem.

2.2 Greedy Heuristic

The Greedy approach first sorts the multiset of cardinality N in descending order, and then proceeds to place the next unassigned number into the partition with the lowest sum until all numbers have been assigned. The Greedy algorithm does not always return an optimal solution to the given multiset S; therefore, the greedy heuristic is an approximation algorithm to the Number Partitioning Problem. This algorithm has the time complexity of $\mathcal{O}(\log n)$ to sort the numbers in descending order and $\mathcal{O}(n)$ to assign each number to each subset, giving a total time complexity of $\mathcal{O}(n \log n)$ [3].

2.3 Karmarkar-Karp Heuristic

Karmarkar-Karp Heuristic (KK Heuristic), also known as the Largest Differencing Method, is an approximation algorithm. KK Heuristic first sorts the multiset of cardinality N in descending order. At each step, the algorithm replaces the two largest numbers with their difference (Figure 1 illustrates the differencing process). This process iterates $N - 1$ times until one number remains. The last remaining number represents the total difference between each subset after the partitioning process [4]. To get the actual subsets, methods such as trees and graphs can be used to reconstruct the original subsets. The KK heuristic has the time complexity of $\mathcal{O}(n \log n)$ to sort the numbers in descending order and the time complexity of $\mathcal{O}(n \log n)$ for differencing, giving an overall time complexity $\mathcal{O}(n \log n)$ [3] ,

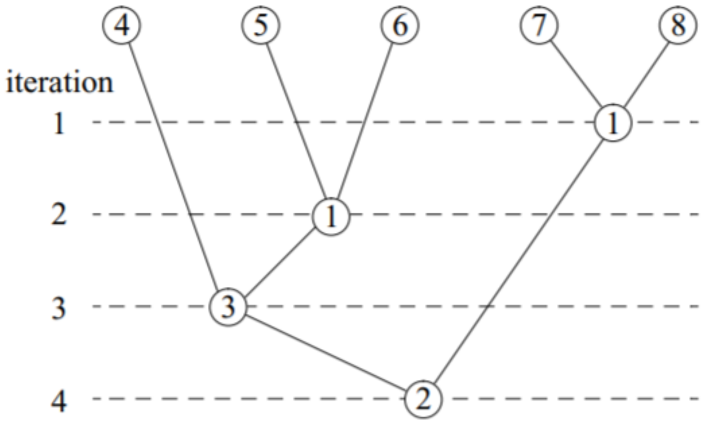

Figure 1: An illustrative diagram of KK Heuristic for two-way partition. For more details, the reader is referred to [8, Fig. 1].

3 Proposed Algorithm: EAA

The Entropy Approximation Algorithm differs from the other algorithms mentioned above in that EAA does not sort multiset S in descending order during initialization. Sorting has the time complexity of $\mathcal{O}(n \log n)$ and omitting the sorting process greatly decreases the run-time of EAA. To compensate for the potential loss in accuracy that sorting the multiset results in, EAA uses entropy, measured by the partition's distance from the optimal sum, to determine the partition that each of the elements goes in to optimize partition solutions.

EAA begins by calculating the optimal sum which is the sum of all the elements in the multiset S divided by the number of partitions K. The optimal sum T is given by the following equation:

$$T = \frac{\sum\limits_{i=0}^{N-1} S[i]}{K} \tag{1}$$

EAA then proceeds to put all of the unassigned elements into each partition according to entropy E defined as the absolute difference of the sum of the partition P_s and the optimal sum T after adding the current element C (the entropy formula is given in equation 2). C is then place in the P_s with the lowest E. In the case when P_s has a sum greater than T, the entropy of putting C into P_s will be scaled higher with weights to discourage the algorithm from selecting that partition.

$$E = |P_s - C \times weight| \tag{2}$$

In addition, for partition that has sum of higher than T, EAA redistribute the values within that specific partition that will bring the sum of the subset closest to T. This enables the optimization of solutions that do not just consider the local optimal solution.

The pseudocode for the proposed algorithm is shown below. The following abbreviations were used:

C : The current number that's being considered
L : The index of the lowest entropy partition
E : The relative distance from T
$MaxInt$: Maximum integer

Algorithm 1: Pseudocode for EAA

```
Initialization: the given multiset $S$; the number of partitions $K$; an
  empty partition array $P$; an empty integer array $D$.
$T \leftarrow$ sum of $S$ divided by $K$ ;
for $i$ in $0, 1, \ldots, K-1$ do
    Append $T$ to $D$;
end
while $S$ is not empty do
    $C \leftarrow pop(S)$;
    $L \leftarrow 0$;
    $E \leftarrow MaxInt$;
    $L, E \leftarrow$ observeEntropy($C$) ;
    Append $C$ to $P[L]$ ;
    $D[L] \leftarrow D[L] - C$ ;
    if $D[L] < 0$ then
        Find integer $n_c$ in $P[L]$ that will bring $D[L]$ closest to 0 and
          remove $n_c$ from $P$ and subtract from $D$;
        $L, E \leftarrow$ observeEntropy($n_c$) ;
        Append $n_c$ to $P[L]$ ;
        $D[L] \leftarrow D[L] - n_c$ ;
    end
end
```

Algorithm 1 shows a general outline of the main features of the EAA.

Algorithm 2: observeEntropy(C)

```
Input: An element of $S$
Output: The index of lowest entropy partition and the lowest
  entropy value
for $v$ in $0, 1, \ldots, K-1$ do
    $d \leftarrow D[v] - C$ ;
    if $d < 0$ then
        $d \leftarrow d \times$ -1 $\times$ weight ;
    end
    if $d$< $E$ then
        $E \leftarrow d$ ;
        $L \leftarrow v$ ;
    end
end
returns $L$ and $E$
```

The function observeEntropy() returns the lowest entropy value E and the location of the partition with the lowest entropy L. If the partition entropy is negative, it will be negated and append weights to discourage the algorithm from selecting the partition.

4 Experimental Study

This paper will focus on the effectiveness of the proposed algorithm, EAA, on three-way partitioning problems specifically. The experiment was conducted using a computer with 16 GB of RAM and NVIDIA 1060 GPU.

For this experiment, the main metrics used for comparison are run-time, maximum sum ratio, and minimum sum ratio. The run-time is measured by recording the time it takes to complete each algorithm. The maximum sum ratio is calculated by dividing the maximum subset sum of the algorithm by the maximum subset sum of the optimal solution. The minimum sum ratio is calculated by dividing the minimum subset sum of the algorithm by the minimum subset sum of the optimal solution.

The performance of each algorithm is first measured by recording the run-time, the maximum subset sum, and the minimum subset sum for every run, and then averaging the results. Three figures are shown below with each respectively corresponding to run-time, maximum subset sum, and minimum subset sum.

4.1 Experimental Results

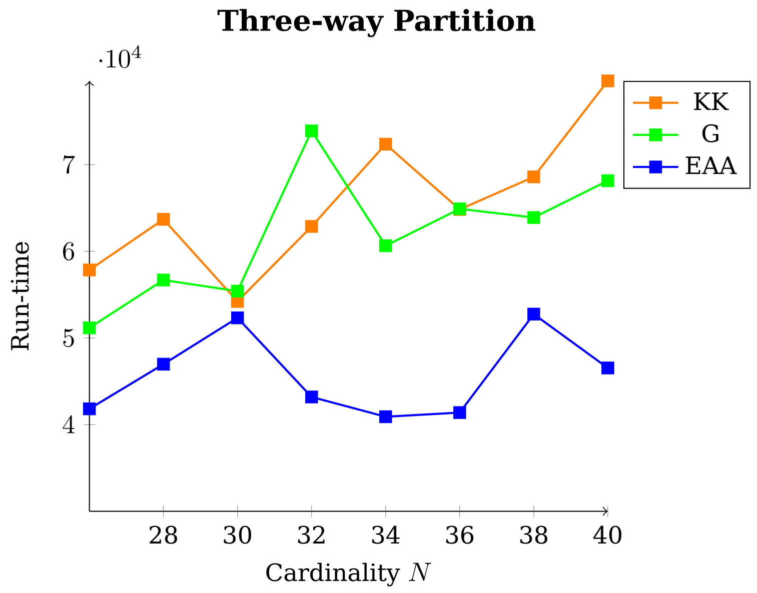

Figure 2: The average run-time in nanoseconds of different algorithms

N	G	KK	EAA
8	1.084	1.084	1
10	1.035	1.035	1.09
12	1.083	1.083	1
14	1.013	1.013	1
28	1	1.03	1
32	1.002	1.006	1
36	1.006	1.006	1.006
40	1	1.003	1.003
44	1	1	1.009

Table 1: Maximum Sum Approximation Ratio

N	G	KK	EAA
8	0.826	0.826	1
10	0.963	0.963	1
12	0.917	0.917	1
14	0.97	0.97	.994
28	1	0.99	1
32	0.994	0.992	1
36	0.995	0.995	0.989
40	0.999	0.996	0.993
44	1	1	0.993

Table 2: Minimum Sum Approximation Ratio

4.2 Discussion

The EAA algorithm attempts to minimize the run-time as well as maintain the maximum sum ratio and minimum sum ratio as close to 1 as possible. The proposed algorithm EAA proved to be the more effective algorithm of the three in partitioning optimal and sub-optimal solutions in the shortest amount of time.

From Figure 2, although the average run-time for all three algorithms fluctuates greatly between different N, the run-time for EAA is generally better than that of KK Heuristic and Greedy Heuristic across the board. For instance at $N = 30$, the run-time for all three algorithms is almost identical, but for $N = 34$, notice that the difference between the run-time is at most twice as fast as Greedy Heuristic and KK Heuristic. Similar to what Korf mention in his paper, the Number Partitioning Problem seems to exhibit an easy-hard-easy transition [9].

Across the multisets of $N \leq 32$ tested on the three algorithms, the EAA outperformed the Greedy Heuristic and KK Heuristic. From Table 1 and Table 2, the Greedy and KK Heuristic performed identically on multisets of lower N value, mainly due to the small N values, with many cases of sub-optimal partition and occasionally optimal partition. In comparison, the EAA was able to return an optimal partition with a few cases of sub-optimal partition for both maximum and minimum sum ratio.

For multisets with $N > 32$ tested, the accuracy of EAA for these multisets in both maximum subset sum ratio and minimum subset sum ratio is lower than both greedy and kk heuristic. Compared to partitions created from multisets with lower cardinality, EAA for multisets with higher cardinality produces sub-optimal solutions more often than optimal. Although EAA is not as accurate in partitioning multisets of higher order cardinality, the partitions produced by EAA are frequently close to the optimal partition, with approximation ratios close to 1.

The EAA outperforms both Greedy and KK Heuristic in run-time due to its ability to partition multiset while maintaining much of the accuracy without sorting. Despite the lost in accuracy of each partition with the increase in the size of the multiset, the EAA generally produces optimal solutions more frequently while maintaining a relatively high accuracy in its partitions

5 Further Study

The proposed algorithm in this paper was applied mainly to the problem of dividing multisets of different cardinality into three equal partitions. The trend of the results for the experiments showed that EAA is more effective than other mentioned algorithms for lower cardinality, but not as effective for higher cardinality. However, EAA is not limited to just three-way partitioning problem. The proposed algorithm can be extended to multi-way partitioning with K greater than 3 to prove that the trend that is shown in this study is also true for four-way partitioning problems and up. Proving that the trends continue can be beneficial in optimizing the cost of equally distributing loads between different machines in real-world situations like multiprocessing scheduling.

6 Conclusion

In this paper, I introduce a new approximation algorithm EAA to solve the multi-way partitioning problem. EAA utilizes the idea of entropy, which is calculated by the sum of all the numbers in the multiset divided by K, to decide whether assigning an unassigned number into a particular partition is the most optimal choice. Unlike the previous algorithms, EAA does not need to sort the multiset in descending order, therefore allowing the algorithm to have significantly lower run-time without losing too much accuracy. This strategy of using entropy proved to be much more consistent in yielding more efficient and accurate solutions than Greedy and KK Heuristic for multisets of lower cardinality and less efficient for multisets of higher cardinality. In general, EAA provides a more effective method in finding sub-optimal partition solutions than other approximation algorithms.

References

[1] J.P.Pedroso and M.Kubo, *Heuristics and exact methods for number partitioning,* European Journal of Operational Research, vol. 202, no. 1, pp. 73–81, 2010.

[2] M.F.Argüello, T.A.Feo, and O.Goldschmidt, *Randomized methods for the number partitioning problem,*

[3] R.E.Korf,*A complete anytime algorithm for number partitioning* Artificial Intelligence, vol. 106, no. 2, pp. 181–203, 1998.

[4] R.E.Korf(2011), *A Hybrid Recursive Multi-Way Number Partitioning Algorithm.* IJCAI International Joint Conference on Artificial Intelligence. 591-596.

[5] S. Boettcher and S. Mertens, *Analysis of the karmarkar-karp differencing algorithm,* The European Physical Journal B, vol. 65, no. 1, pp. 131–140, 2008.

[6] E.L.Schreiber, R.E.Korf, and M.D.Moffitt, *Optimal multi-way NUMBER PARTITIONING,* Journal of the ACM, vol. 65, no. 4, pp. 1–61, 2018.

[7] H.A.Isa, S.Al-Oqaili, and S.Bani-Ahmad, *The subset-sum problem: Revisited with an improved approximated solution,* International Journal of Computer Applications, vol. 114, no. 14, pp. 1–5, 2015.

[8] W. Michiels, J. Korst, E. Aarts, and J. van L.i, *Performance ratios for the karmarkar-karp differencing method,* Electronic Notes in Discrete Mathematics, 23-Apr-2005. [Online]. Available:

https://www.sciencedirect.com/science/article/pii/S1571065304004421. [Accessed: 10-Aug-2021].

[9] Korf, Richard. *Multi-Way Number Partitioning* IJCAI. International Joint Conference on Artificial Intelligence. 538-543.