The Easiest Hard Problem

15-48 EC masters' thesis project

For:	Master Students that know how to program - and are interested. If you've ever done a course in heuristics, it is a pre.
Load:	15 - 48 EC, depending on blocks (see below)
Period (approx.):	From September 2021 onward
Supervisor:	Daan van den Berg
Contact:	daan@yamasan.nl, kawarimasen0010@gmail.com
Introduction video:	https://bit.ly/3DWGFrG

About the project

The *number partition problem* (a special case of the *subset sum problem*) is known as "the easiest hard problem". Given some set, for example S = {1,4,9,12,17,31,41,59}, is it possible to split up the set into two sets of equal value? Or, if not, what is the closest you can get?

It is an NP-hard problem, which means we don't have a fast algorithm to do the job, but its hardness depends not only on the size of the set *n*, but also on *m* the 'typical number of bits for a number in the set'. When *m* is small relative to *n*, meaning the set has lots of small numbers, it is relatively easy, because it has many solutions [1] (also see Figure 1). Therefore, the problem is *weakly NP-hard*.

However, a very recent publication from UvA [2] suggests this is not the end of the story. Even if the number of informational bits remains constant, the *exact distribution* also seems to influence its hardness. We want to know more about this.

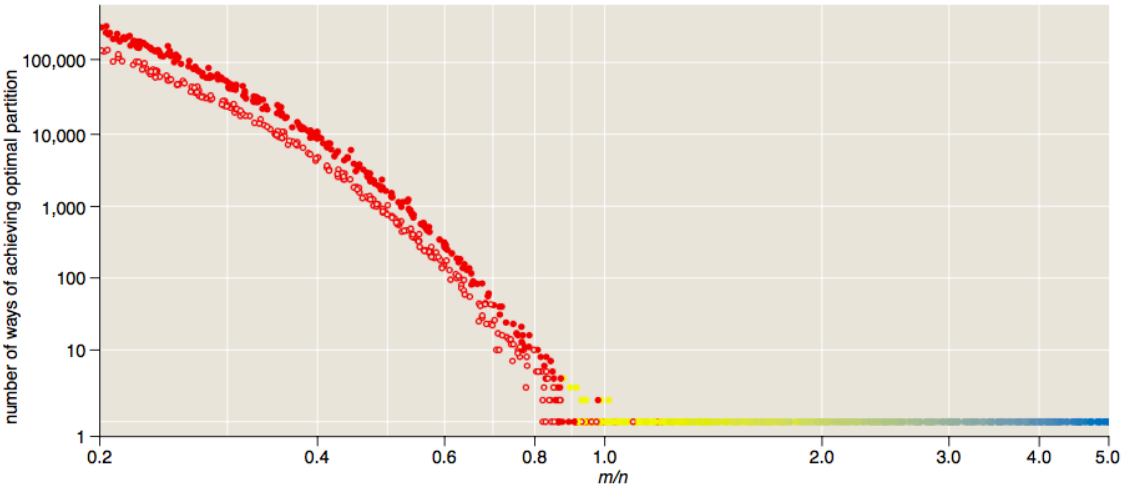

Figure 1 (adapted from Brian Hayes' paper) The number of solutions of the partition problem depens on the numbers in the set n, as well as the number of bits m of a 'typical integer' in the set.

Each block is about 15-18 EC's worth. You can choose blocks to your liking, though there is a strong emphasis on doing Block #1 first. Generally, we can adapt blocks as we go along, but once a block is finished, we're not going back, only forward.

Block #1
Start reading about the partition problem, and about hardness classes. Replicate the study by Van den Berg & Adriaans, and try to make larger templates and instances; 50 integers would be great, 100 would be fantastic.

Block #2
Enrich the experiment by adding more exact algorithms. Kamarkar-Karp would be good, but perhaps there's other options too. Do the hardness results differ per algorithm? Quantify your results.

Block #3
There is a bit of evidence that suggests that for some of these templates, exact solutions, or maybe even closest approximations, break up the dimensionality of the integer line. A structure of broken dimensionality (for instance 1.49 dimensions) is called a *fractal*. First, read about fractals and how to measure them. Then, for a few instances, make an integer line, and for each integer, give the number of solutions. Then measure its fractality. Perhaps we can expand the same S with one integer, and 'recompress' the integer line to [0,1]. We have to discuss how many of these instances, and what sizes, need to be done to complete this block.

Block #4
Measure the fractality for a number of full scale template-instances for various algorithms. And make comparisons. To what extent this is reasonable respective to the work load needs to be discussed, as it involves generating a lot of data.

References

[1] Hayes, Brian. "Computing science: The easiest hard problem." American Scientist 90.2 (2002): 113-117.

[2] Daan Van Den Berg, Pieter Adriaans "Subset Sum and the Distribution of Information." IJCCI 2021: Proceedings of the 13th International Joint Conference on Computational Intelligence (to appear in November 2021; mail me for a preprint)