

# Bin Packing

Tom Davis

tomrdavis@earthlink.net

<http://www.geometer.org/mathcircles>

November 29, 2006

## 1 Introduction

Bin packing can refer to lots of different problems. Almost all of them turn out to be harder than they seem. Some of them are simply interesting puzzles and some have very important practical applications. In this article, we'll concentrate mostly on what seems like a very simple version, but we will begin with a few examples of problems that are more puzzle oriented.

The basic idea is to find the best way to pack a bunch of objects (usually of different sizes) into bins (which are usually the same size). We will make a more precise definition later.

Bin packing problems are also sometimes called "knapsack problems".

## 2 Bin Packing Puzzles

Here are a few examples of problems that are not quite the standard bin-packing problem, but are related, and are interesting in themselves.

### 2.1 Packing Grids with Dominoes

Let's begin with a very simple problem that actually has a nice solution. Given a checkerboard of size  $2 \times n$ , in how many ways can it be tiled with  $1 \times 2$  dominoes?

As is almost always the case, the best approach is to gather some data by counting the number of ways it can be done for various sizes of squares.

In Figure 1 shows all possible packings for a  $2 \times n$  rectangle for  $n = 1, 2, 3, 4$  and  $5$ . As you can see, there are, respectively, 1, 2, 3, 5 and 8 solutions. These look a lot like the Fibonacci numbers, and in fact, it's reasonable to say that there is exactly one way to fill a  $2 \times 0$  grid: do nothing, so the numbers would be exactly the Fibonacci numbers: 1, 1, 2, 3, 5, 8, at least for the first 6 of them.

Why would this be the case? The Fibonacci numbers  $F_i$  are defined as follows:

$$\begin{aligned}F_0 &= 1 \\F_1 &= 1 \\F_n &= F_{n-1} + F_{n-2}, \text{ if } n > 1.\end{aligned}$$

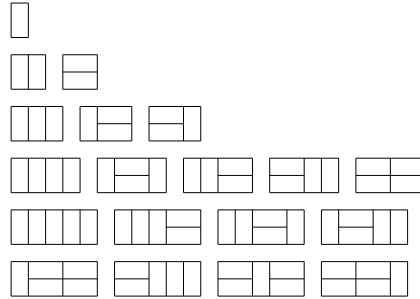


Figure 1: Packing Dominoes in a  $2 \times n$  Rectangle

It's easy to check that the numbers of packings for a  $2 \times n$  grid are 0 and 1 for  $n = 0$  and  $n = 1$ . What if  $n$  is larger than 1? Look at the left-most edge. Either there is a vertical domino (in which case there remains a  $2 \times n - 1$  grid to fill), or there are two parallel horizontal dominoes (in which case there remains a  $2 \times n - 2$  grid to fill). Thus if we think of  $T_i$  as representing the number of tilings with  $n$  dominoes, we have exactly the same definition for the sequence  $T_i$  as we had for the Fibonacci numbers  $F_i$  previously:

$$\begin{aligned} T_0 &= 1 \\ T_1 &= 1 \\ T_n &= T_{n-1} + T_{n-2}, \text{ if } n > 1. \end{aligned}$$

so  $T_i = F_i$  for all  $i$ , and the  $T_i$  must be just the Fibonacci numbers.

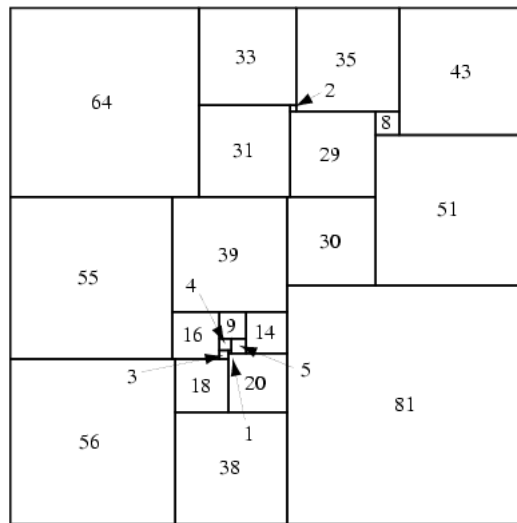
Try to solve the following problem: suppose you want to tile a  $1 \times n$  grid with dominoes, but this time you have two kinds of dominoes:  $1 \times 2$  and  $1 \times 1$ . You can approach the problem in the same way.

A more interesting and difficult problem is that of packing  $n \times m$  grids with  $1 \times 2$  dominoes. Obviously, at least one of  $n$  or  $m$  must be even for this to work. You may find it interesting to consider just the  $3 \times m$  problem, when  $m = 2, 4, 6, \dots$ . Even counting these for small values of  $m$  is a little tricky. Try working out a few on your own; the values for  $m = 2, 4, 6, 8$  and  $10$  appear in Section 4.

## 2.2 Packing Rectangles and Cubes In Two and Three Dimensions

Many puzzles basically require that you find a method to pack a certain number of shapes into another shape. Some easy examples involve packing sets of squares of unequal size into squares or rectangles. These sorts of problems are visually interesting, and today a lot of solutions are known because computers can search for packings.

Figure 2 illustrates a solution to the following problem: Find a set of squares, all of different sizes, that pack together to perfectly fill some larger square. The illustrated solution uses 24 different squares to accomplish the task. The solution is a little disappointing since the tiling squares have somewhat oddball sizes.



24-square perfect square

Figure 2: Packing Squares in a Square

A much nicer problem would be to find a packing of squares with sides of lengths  $1, 2, 3, \dots, n$  that completely fill a larger square. For this to happen, we would need to find integers  $n$  and  $m$  such that:

$$1^2 + 2^2 + 3^2 + \dots + n^2 = m^2.$$

Are there any such solutions?

It is not hard to show that:

$$1^2 + 2^2 + 3^2 + \dots + n^2 = \frac{n(n+1)(2n+1)}{6},$$

so we basically need to see when the quantity  $n(n+1)(2n+1)/6$  is a perfect square. This occurs if  $n = 1$  (which is not interesting) or when  $n = 24$ :

$$1^2 + 2^2 + \dots + 24^2 = 70^2.$$

These are the only two solutions. As far as I know, the best packing achieved so far uses all but the  $7 \times 7$  square, covering 4851 of the 4900 unit square areas, and nobody knows if a better tiling exists. It has been shown that a perfect tiling is impossible. For more information, see:

<http://mathworld.wolfram.com/PerfectSquareDissection.html>

There is also a great chapter on square packing in the book *Ingenuity in Mathematics* by Ross Honsberger.

Somewhat surprisingly, it is easy to show that there is no solution of any type that will allow a packing of a rectangular box completely with more than two perfect cubes,

all of which are different sizes. If it were possible, the bottom of the box as a flat surface would be packed with squares (which is certainly possible), but one of those squares would be the smallest. It's easy to check a couple of cases to see that this smallest cube could not be on the edge of the bottom.

Thus this square would be covered by the smallest cube and that cube would be surrounded by higher cubes, so the top of the smallest cube on the bottom would have to be tiled by smaller cubes. There must be a smallest of these, and the argument can be repeated, showing that any tiling must require an infinite number of cubes.

### 3 The Standard Problem

Standard problem statement: What is the minimum number  $k$  of identical bins, each with capacity  $C$  needed to store a finite collection of indivisible items having weights  $w_1, w_2, \dots, w_n$  such that the sum of the weights of the items in each bin does not exceed  $C$ ?

Here are a couple of "obvious" theorems.

- If any  $w_i > C$ , there is no solution. If all  $w_i \leq C$ , then there is a solution, and that solution requires  $n$  or fewer bins.
- The minimum number of bins is  $(\sum w_i)/C$ .
- Theorem if all the weights are the same?

Example: Bins of size 10; weights: 3, 6, 2, 1, 5, 7, 2, 4, 1, 9.

What are some approaches (possibly idiotic)?

- **Next fit method:** Start by placing  $w_1$  in the first bin. Keep examining items, in order, and if they fit, continue to place in that bin. As soon as an item does not fit, advance to the next bin, and so on.

How bad can this be? How good can it be? Is this completely idiotic?

- **First fit method:** Use the same method as before, except allow reexamination of bins. In other words, for each new item, examine every bin, beginning with the first, to see if the new item can be placed in that bin. Only if it will not fit in any bins should we begin to fill a new bin.

Same questions as above. Can you think of other methods?

- **Best fit method:** same as first fit, but put each new item into a bin that comes closest to filling it.
- **Worst fit method:** put new items into existing bin that leaves the most space available in that bin.

Is this necessarily worse than the other methods?

- **Almost worst fit method:** almost the same as worst fit, but put the item in the second-emptiest bin.<sup>1</sup>

The approaches above all assume that we have to handle the items in the order received; for some applications this makes sense; for others, it doesn't. Sometimes we have all the items ahead of time and can sort the list (or pre-process it in various other ways) and take advantage of knowing all the weights before we begin committing the items to bins.

If we sort the items into decreasing (or, alternatively, increasing) order and then apply the approaches above, which ones are likely to improve and which to get worse?

## 4 Grid Packing with Dominoes

The number of packings of a  $3 \times 2n$  grid for  $n = 1, 2, 3, 4$  and  $5$  are 3, 11, 41, 153 and 573, respectively. For more information, look at sequences *A099390* and *A065072* from *The On-Line Encyclopedia of Integer Sequences*:

<http://www.research.att.com/~njas/sequences/>

## 5 Samples

The following plots show the results of various algorithms applied to the same set of input data.

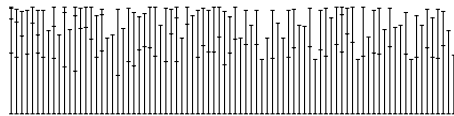


Figure 3: Best Fit

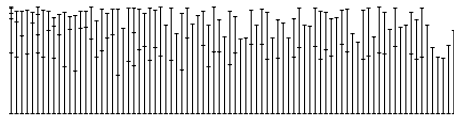


Figure 4: First Fit

---

<sup>1</sup>This method is actually better than worst fit, and can be proven to be so.

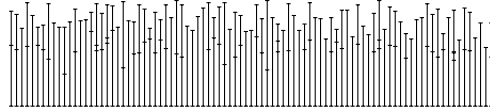


Figure 5: Worst Fit

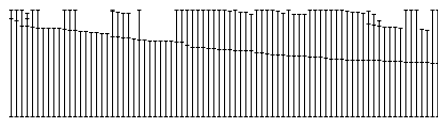


Figure 6: Sort Down, then Best Fit

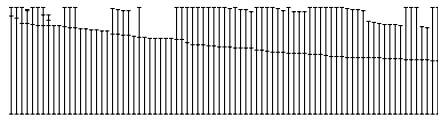


Figure 7: Sort Down, then First Fit

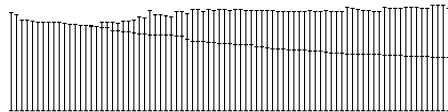


Figure 8: Sort Down, then Worst Fit

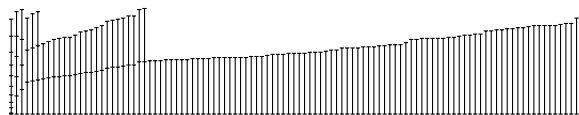


Figure 9: Sort Up, then Worst Fit