Tianyi Li

CS 1501  
Dr.John Aromis

Project 1

Dynamic Programming is a method that can solve the complex optimization problems. It makes problem easy to solve by breaking problem down into a collection of simpler subproblems. Those subproblems can be easily being solved.

The Knapsack Problem is a famous optimization problem. A description of the problem is as follows:

Given a set of pieces cloths, each with a height(h), a width(w) and a value(v), determine the number of each item to include in the collection so that the total height and width is less than or equal to the given limit and the total value is as large as possible. Each cut only can cut horizontal or vertical. What is the best way to cut the cloth with the optimal profit is made and the most amount of cloth is used?

In the project, the original width(ow) and the original height(oh) can only make horizontal or vertical cuts. Moreover, if we make one cut it divide original cloth to two pieces and those two big clothes can make small cuts. For the best profit I want to cut the most profit cut to earn the most money. The cut can perform in follow picture:

Original cloth

First cut horizontal

Cut the following cutting pieces in the cloth



Cut the rest of the in the way



Cut the rest



The rest cut can be with different size though comparing cut different size of patterns, and then return the optimize value, pattern and garment(x,y coordinate)ß

The value of the cloth after cutting is equal to the sum of the values of all the

pieces of cloth cut from it:

Value = d value \* d items + c items \* c value

The first step I took to this problem with force cut with this type of cut is to divide the cloth to half. And I check the three types of the cut:

No cut. There no cut can be the cloth is optimal values and may there are no cloth that left to cut.

Horizontal cut. Cut the cloth horizontal way, the main point of doing this one is cut the cloth into two pieces.

Vertical cut. Cut the cloth vertical way, the cut makes the sum of value of the pieces on the left to right.

The simplest way to solve the knapsack problem is with a dynamic programming algorithm:

Ov = optimal value

Value is the temp value

Optimize (width, height) {

For each pattern if it fit value larger ov,

Set ov = pattern.value

}

Then I approach by horizontally cut:

{ For every horizontally cut

If the value>ov

Do the cut

The original cloth – those cut size(height and weight)

Return ov

}

Then I approach by vertical cut:

{ For every vertical cut

If the value>ov

Do the cut

The original cloth – those cut size(height and weight)

Return ov

}

The basic algorithm is here.

I calculator the a, b, c and d value of which most price of each size.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| a | 2 | 2 | 1 | 1/4=0.25 |
| b | 2 | 6 | 4 | 4/12=0.3 |
| c | 4 | 2 | 3 | 3/8=0.375 |
| d | 5 | 3 | 5 | 5/15=0.3 |

So, the c is the optimal patterns and to approach.

There is one part to use about a JPanel extension that to show the cloth. The main in cloth cutter is make cut, it makes cut in the JPanel, the basic painting part is in cloth. In cloth draws the cloth in the graphic way. Draw cut and draw garment: draw cut is mainly drawing the cut and the draw garment is mainly put the cloth number in the paint.

The Cloth Problem is an actual dynamic example programming that can be used to easily and efficiently present my solution to complex solutions. The simple recursion and dynamic solution of my runtime shows that the dynamic programming solution for cloth cutting programs is due to the need for faster regeneration call completion problems, which also significantly improves memory efficiency. This shows that dynamic programming is not only a feasible method to solve the optimization problem, but also a feasible method to solve the problem.