CS40 - Summer 2015

June 25

Victor Amelkin

A note on combinations with repetitions

The purpose of this note is to give you a better understanding of how combinations with repetitions work. Let us deal with a problem of counting how many times a line of code executes:

```
for i = 1 to 20
  for j = 1 to i
    for k = 1 to j
      print("x")   <------ # of times this line executes - ?
```

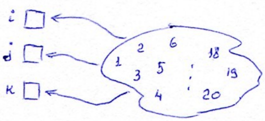
-- a bunch of nested loops is just a way to hide (a set of) constraints imposed upon i, j, k

The first part of the solution is to arrange all the variables:

$$1 \le k \le j \le i \le 20$$

-- these constraints

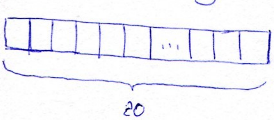
Now, the counting problem has narrowed down to counting the number of choices of k, j, i such that $1 \le k \le j \le i \le 20$. One way of thinking about it is as follows:



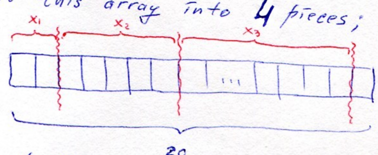
We have a "pool" of numbers from 1 to 20, and we need to fill i, j, k using these numbers. Repetitions are allowed – if i is filled with 5, then this 5 remains in the pool and can be used to fill other variables. The number of ways to fill i, j, k, by definition, is the number of combinations with repetitions $\binom{20+3-1}{3} = \binom{22}{3}$.

The following is a great way to visualize problems of integer constraint satisfaction and partitioning integers:

Another way of thinking about it is to consider the following cell array:



Now, let us cut this array into 4 pieces; we will need to do 3 cuts:



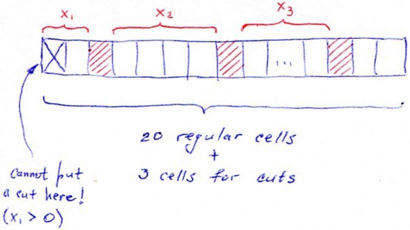
The numbers of cells in the first 3 parts are denoted with x_1, x_2, and x_3. Now, we will define i, j, k:

$$k = x_1,$$
$$j = x_1 + x_2,$$
$$i = x_1 + x_2 + x_3.$$

Note: 1) Cuts can repeat (if the first and the second cuts "coincide", it means $x_2 = 0$ and, hence, $k = j$);

2) Cuts cannot occur at the left border of the array, because $k \geq 1$ (i.e., $k > 0$).

Each such partition of a 20-cell array into 4 pieces corresponds to a legal choice of k, j, i. If we "expand" each of 3 cuts into a cell, then it is easy to count the number of ways to partition the array and, hence, the number of choices for i, j, k.



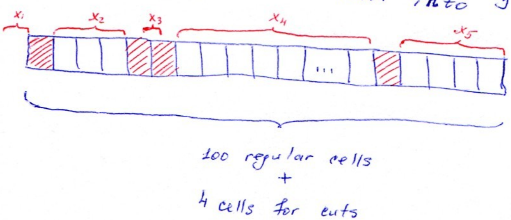
- The number of such partitions is exactly the number of ways to choose 3 "cut-cells" out of 22 available cells: $\binom{22}{3} = \binom{20+3-1}{3}$.

20 -- number of original cells (together, comprising integer 20)
3 -- number of special cells for cuts
-1 -- the first cell cannot be used as special
= how many ways to choose 3 special (cut) cells out of 20 + 3 - 1 available cells?

You can use the same cell-array visualization thechnique to count the number of integer solutions of the equation

$$x_1 + x_2 + x_3 + x_4 + x_5 = 100, \quad x_i \geq 0 \quad (\text{notice, } x_i \text{ can be } = 0)$$

Take 100 cells and partition them into 5 possibly empty parts with "cell-cuts":



(For this particular example, $x_1 = 0$, $x_2 = 3$, $x_3 = 0$, $x_4 = 90$, $x_5 = 4$)

The number of ways to select values for $x_1, \dots, x_5$ is the number of ways to select 4 cut-cells out of 104 available cells (unlike in the problem with loops, the first cell is available for cutting, i.e., x_i can be 0 in this problem): $\binom{104}{4}$.