Massachusetts Institute of Technology — Handout 28
6.042J/18.062J: Mathematics for Computer Science — April 1, 1994
Professors Charles E. Leiserson and Tom Leighton

Problem Set 6 Solutions

Problem W6-13. Balls and urns.

(a) $3 \cdot 3 \cdot 3 \cdot 3 = 81$.

(b) We can put either all 4 balls in one urn, 3 in one, 2 in one and 2 in another, or 2 in one and 1 in each of the other two. The balls are distinguishable, so there are $\binom{4}{3}$ ways of putting 3 in one. There are $\binom{4}{2}$ ways of putting 2 in one and 1 in each of the other two. There are $\binom{4}{2}/2$ of putting 2 in one and 2 in another. So the total number of ways is

$$1 + \binom{4}{3} + \frac{1}{2}\binom{4}{2} + \binom{4}{2} = 14.$$

(c) Create 6 slots and choose 2 to contain bars leaving r_1, r_2 and r_3 consecutive slots with stars with $r_1 + r_2 + r_3 = 4$. This corresponds to putting r_i balls in urn i. So the answer is $\binom{6}{2} = 15$.

(d) This is the number of ways to partition a multiset $\{1,1,1,1\}$ of 4 indistinguishable objects into 3 (possibly empty) subsets. We can either put $\{1,1,1,1,\}, \varnothing, \varnothing$, or $\{1,1,1\}, \{1\}, \varnothing$, or $\{1,1\}, \{1,1\}, \varnothing$, or $\{1,1\}, \{1\}, \{1\}$. The ordering in which we list the subsets doesn't matter because the urns are indistinguishable. So the answer is 4.

Problem W6-14. Derangements.

(a) There are 9 ways: 4123, 4312, 4321, 3142, 3412, 3421, 2341, 2413, 2143.

(b) $D_n = n! - |A_1 \cup A_2 \cup \cdots \cup A_n|$.

(c) $D_n = n! - (\alpha_1 - \alpha_2 + \alpha_3 - \cdots + (-1)^{n+1}\alpha_n)$.

(d) $\alpha_r = \binom{n}{r}(n-r)! = \frac{n!}{r!}$ since α_r is the sum over the r-subsets of $[n]$ of the number of permutations that leave that subset of r elements fixed—the other $n-r$ elements may be permuted in any way. The formula for D_n follows immediately from part (c).

Problem W6-15. Catalan numbers.

(a) $\binom{2n}{n}$.

(b) The total number of walks is $\binom{2n}{n}$. The number of walks that never go below the x-axis is exactly $C(n)$—an up-step corresponds to a left-parenthesis and a down-step corresponds to a right-parenthesis. It is not hard to see that the well-formed strings of parentheses are exactly those that have at least as many right-parentheses as left-parentheses in any prefix.

(c) We set up a bijection between the set of walks from $(0,0)$ to $(2n,0)$ that go below the x-axis and the set of all walks from $(0,-2)$ to $(2n,0)$. As suggested by the figure, a walk from the first set is mapped to a walk from the second set by finding the first place at which it goes below the x-axis (hits $y=-1$) and reflecting the portion of the walk thus far around the line $y=-1$. This gives a walk starting at $(0,-2)$ and ending at $(2n,0)$. Conversely, any walk starting at $(0,-2)$ is mapped to a walk starting at $(0,0)$ by reflecting the portion of it before the first time it hits $y=-1$. Because these two functions are clearly the inverses of each other, the functions form a bijection between the two sets, which therefore must have equal cardinality.

(d) Choose $n-1$ of the $2n$ steps to go up: $\binom{2n}{n-1}$.
Thus,

$$C(n)=\binom{2n}{n}-\binom{2n}{n-1}=\binom{2n}{n}-\frac{n}{n+1}\binom{2n}{n}=\frac{1}{n+1}\binom{2n}{n}.$$

(e) It is not difficult to see that both $B(n)$ and $C(n)$ satisfy the given recurrence: The number of binary trees on n (unlabelled, indistinguishable) nodes is the number of trees with a left subtree of k nodes and a right subtree of $n-k-1$ nodes, for $0\le k\le n-1$; there are $B(k)$ trees on k nodes and each of these may be paired with any of $B(n-k-1)$ trees on $n-k-1$ nodes. There is a slight subtlety in counting the number of well-formed strings recursively. A correct way of counting them is the number of well-formed strings w of length k times the number of well-formed strings $(\,v\,)$ where v is of length $n-k-1$, for $0\le k\le n-1$. Notice v is in parentheses; it is incorrect to simply multiply the number of strings w of length k times the number of strings v of length $n-k$, $1\le k\le n-1$ since, for instance, that counts the string $()()\ldots()$ $n-1$ times.

A bijection f from binary trees to well-formed strings may be defined inductively: if a tree T has nonempty left subtree L and nonempty right subtree R, f maps T to $(\,f(L)\,)\,f(R)$. If L is empty, f maps T to $(\,)\,f(R)$. If R is empty[1], f maps T to $(\,f(L)\,)$. If both are empty (T is one node), f maps T to $(\,)$. To show that f is a bijection, we need only exhibit a g such that f and g are inverses of each other. First, g maps $(\,)$ to a single node. Next, g maps $(\,w\,)$ to a tree with no right child and with $g(w)$ as the left child. Finally, g maps $(\,w\,)\,v$ to the tree with $g(w)$ as the left subtree and $g(v)$ as the right subtree.[2] This takes care of any string of the form $x\,y$, since $x\,y$ can be written as $x'\,y'$ where x' is $(\,w\,)$. Here, w may be the empty string, in which case we must define $g(w)$ to be the empty tree (no nodes). By construction, $g=f^{-1}$ and $f=g^{-1}$, and so f is a bijection.

[1] Note that mapping T to $f(L)\,(\,)$ here would be wrong because then different trees—for instance, the tree with only left children and the tree with only right children—would get mapped to the same string.

[2] Note we had to be careful here about making sure the string is parsed uniquely—we cannot just talk about g mapping $w\,v$ since there may be several ways to partition the string—for example, $()()()$.