Spring 2004

Joe P. Buhler (jpb@msri.org)
Elwyn Berlekamp (berlek@math.berkeley.edu)

Note: These solutions are a work in progress; comments, references, etc. are appreciated. Most references and figures can be found with the problem statements.

Problem 1. 101 ants are placed randomly on a one-meter stick, except that one of them, Alice, is placed in the exact center. Each ant is placed facing a random direction. At a certain moment all of the ants start crawling in the direction they are facing, always traveling at one meter per minute. When an ant meets another ant or reaches the end of the stick, it immediately turns around and continues going in the other direction. What is the probability that after 1 minute Alice is again at the exact center of the stick?

Discussion: Recall the thought experiment for Problem 1, Spring/Fall 2003, and add the requirement that batons bounce back from the ends of the stick. Exactly one minute after starting each baton has bounced back from the end of the stick towards which it was originally headed and has reached the exactly point that is the reflection of its starting point about the center. In particular, Alice's original baton, which started in the center of the stick, is now back in the center, with some ant. But is that Aunt Alice?

The ordering of the ants remains invariant throughout the action-packed minute, so if Alice was the kth ant from the left initially, she is still the kth ant from the left one minute later. Initially, there were $k-1$ batons to the left of Alice's; one minute later, these batons have all been reflected about the center, so there are now exactly $k-1$ batons to the right of Alice's original baton. So at the end Alice cannot have her original baton unless (unless and only unless) there were initially exactly 50 ants on each side of her. The probability of this happening is exactly

$$2^{-100}\binom{100}{50}.$$

Using Stirling's formula, this is very close to $1/(5\sqrt{2\pi})$, or about .0798, nearly 8 per cent.

Problem 2. A white king and a white rook play chess against a black king on a quarter-infinite chessboard consisting of the first quadrant of the Cartesian plane. Initially, the White rook is at the lower left hand square (0,0), the White king is adjacent to it at the square (1,0) on the lower boundary, and the Black king is at (1,2). White moves first. On any move when he is not in check, Black can elect to end the game by cashing out, receiving a payment from White of $\$(x+y)$ if the Black king is on the square (x, y). Assuming correct play, how large a sum can Black earn?

Problem 3. A read-only array (ROM) contains n integers. Find a linear-time algorithm that determines whether the array has a "majority element," and if so, returns that value. An integer x is a majority element in the array if it is in k locations, where $k > n/2$.

Discussion: Repeatedly eliminate pairs of different elements. If any element remains it is the only possible majority element, and a simple linear search can determine whether or not it is. One clean way to perform the first step uses a stack, with operations `push()`, `pop()`, and `stacktop()`:

```
Majority candidate(a)
    for each element c of the array a
        if the stack is empty or c is on top of the stack
            push(c)
        otherwise
            pop()
    if the stack is nonempty
        return top()
```

This makes it clear that the first step takes time $O(n)$, and it is clear that determining whether a specified element is a majority element takes time $O(n)$.

Problem 4. A team of n computer scientists meet and plot strategy in the following game. Your job of course is to devise a strategy for them that maximizes the probability that they will win.

Each member of the team is assigned a unique public number k from 1 to n.

When the contest begins, each contestant in placed in his own private room, with his own primitive computer. No further communications between team members are permitted.

The game show host creates a ROM with n locations. The k-th location has an entry $\pm\pi(k)$, where π is a random permutation of $\{1, \ldots, n\}$, and each of the n signs is chosen randomly.

Each contestant can program his or her computer to access this ROM. The program can examine up to $n/2$ locations the ROM, but no more. Each program is allowed only a small fixed number of scratch locations, but dynamic access to the ROM is allowed (i.e., the "next" location can depend on the values observed in earlier locations).

After examining $n/2$ locations of the ROM (or sooner, if his treasure hunt succeeds) the k-th contestant is required to guess the sign of k in whichever location it happens to occupy.

The team wins if and only if all n members of the team guess correctly. Find a strategy giving your team a significant chance of winning even when the number of players is very large.

Discussion: Consider the graph whose nodes are the integers from 1 to n, with branches from k to $\pi[k]$. The strategy for player k is to look at $a[1]$, $a[2]$, ..., $a[n/2]$, where $a[1] = k$ and $a[j+1] = \pi[|a[j]|]$ for each j. This traces a path in the graph. If node k lies in a cycle of length at most $n/2$, the path traversed by player k will complete the cycle and he will discover the object of his quest.

The team's strategy succeeds if the permutation π is the product of cycles which *all* have length at most $n/2$. The simple version of the strategy fails if and only if the permutation π contains some cycle of longer length. That happens if and only if for some small number s (in the interval $0 \le s < n/2$, there is a cycle of length $c = n - s$. There are $\binom{n}{s}$ ways of picking the elements not in this cycle, and $s!$ permutations among them. The other $m = n - s$ elements must lie in a single cycle, which can happen in $(M-1)!$ ways. So the number of permutations that have a long cycle is

$$\sum_{s=0}^{n/2} \binom{n}{s} (n-s-1)!\, s! = \sum_{S=0}^{n/2} \frac{n!}{n-s}.$$

If a permutation is chosen at random from the set of all $n!$ permutations, the probability of a permutation which causes the team's strategy to fail is

$$p = \sum_{s=0}^{n/2} \frac{1}{n-s}.$$

Note: There is a refinement of this strategy that allows the team to do slightly better: In addition to winning whenever the longest cycle is no greater than $n/2$, they can also win half of the time whenever the longest cycle is one greater. To achieve this, each member records the cumulative product of all of the signs he has seen, and if he fails to find the sign he is looking for, then he guesses that it is equal to the product of all of the signs he has seen. Although this guess is correct only 50% of the time, if the cycle length is $1 + \lfloor n/2 \rfloor$ the guesses of the team members tracing this cycle will either all be correct or all be wrong.

Although this refinement gives a noticeable improvement when n is small, its impact becomes negligible as n becomes large.

For large n, the expression for p asymptotically approaches

$$p = \sum_{k=1+\lfloor n/2 \rfloor}^{n} \frac{1}{k} \approx ln2.$$

So for large n, the team fails only about 69% of the time. With probability over 30%, everyone on the team succeeds in finding the sign bit for which he is looking.