South Pacific Regionals 2016

Editorial

The South Pacific Judging team is comprised of Darcy Best, Mike Cameron-Jones, Malcolm Corney, Tim French, Walter Guttmann, Andrew Haigh, Henning Koehler, Richard Lobb, Kourosh Neshatian, Evgeni Sergeev, Kevin Tran, Max Ward-Graham.

A Anticlockwise Spiral

Author: Malcolm Corney — Complexity: $O(\sqrt{n})$ or $O(1)$

What is the Manhattan distance between two points on a game board?

The main part of this problem is to find the row and column of the two given squares. There are a few ways to find the location, but the key observation needed is that if you start at the centre square and work your way southwest (down-left), the squares are numbered $1, 9, 25, 49, \ldots$. This pattern should jump out at you–the odd perfect squares!

To find the location of a given square, you should first find the 'ring' that the square is on (roughly equal to $\frac{\sqrt{n}}{2}$). From here, you can march around the ring to find the square you are looking for (or use a closed form solution).

Another way to approach problems with spiral shapes is to use the pattern 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, etc. What does this mean? 1 down, 1 right, 2 up, 2 left, 3 down, 3 right, 4 up, 4 left, etc. This will find the places where the spiral changes directions.

B Balloon Warehouse

Author: Andrew Haigh — Complexity: $O(R)$

What are the colours of the balloons in the line after several insertions?

To find the colours of the balloons, we will actually generate the sequence of the first million balloons in the line, then output the desired sequence. Given the colour of the current balloon and the time that it appeared in the line, you can determine the colour of the next balloon. Which one is it? It is the colour of the last instruction that included the colour of the current balloon (assuming that the time of the instruction is after the time of the current balloon). You then recursively ask what is the next colour after that balloon. What about after that? The next colour in line is the second last instruction (so long as the time is after the current time). It is roughly the DFS search order.

```
void generate_sequence(int curr_colour, int curr_time){
  answer.append(curr_colour);
  if answer.length() == 1000000:
    print answer from L to R;
    exit program;

  for all instructions containing curr_colour in reverse:
    if time of instruction > curr_time:
      generate_sequence(new_colour, new_time);
    else
      break;
}
```

C Crazy Rotations

Author: Darcy Best — Complexity: $O(n \log n)$

Which rotation changes the kth least colours?

Obviously, the naive $O(n^2)$ solution is too slow. Instead, we will make 3 polynomials:

$$R(x) = r_0 + r_1 x + r_2 x^2 + \cdots + r_{n-1} x^{n-1}$$
$$B(x) = b_0 + b_1 x + b_2 x^2 + \cdots + b_{n-1} x^{n-1}$$
$$Y(x) = y_0 + y_1 x + y_2 x^2 + \cdots + y_{n-1} x^{n-1}$$

The polynomials are defined so that r_i is 1 if the ith light is red (and 0 otherwise). Similarly, y_i is for yellow and b_i is for blue.

Let's examine this polynomial:

$$R(x) \cdot R(x^{-1}) = \sum_{-n<d<n} \left(x^d \sum_{0 \le i < n} (r_i r_{i-d}) \right)$$

So the x^d term counts the number of lights that were red before and after a shift by d spots (no wraparound). Similarly, $Y(x) \cdot Y(x^{-1})$ and $B(x) \cdot B(x^{-1})$ count the number of yellows and blues, respectively. Thus, the number of lights that do not change on a shift by d spots (with wraparound) is equal to the sum of the dth and $(d-n)$th term in

$$R(x) \cdot R(x^{-1}) + Y(x) \cdot Y(x^{-1}) + B(x) \cdot B(x^{-1}).$$

To make this fast enough, you need to do the polynomial multiplication in $O(n \log n)$. This can be accomplished via Fast Fourier Transform (FFT).

D Dendroctonus

Author: Darcy Best — Complexity: $O(n^3)$ [$O(n^4)$ also accepted]

Is there a circle such that all red points are inside and all blue points are outside?

First of all, if there is only one infected tree, the answer is `No`.

Now, assume there are multiple infected trees. Imagine that you can find a circle. You can morph it in such a way that there are either two or three points that lie on the boundary of the circle (if there are only 2 points, then they can be on the opposite sides of the circle, where the centre is the midpoint between them).

For a slow (but still acceptable) solution, you can try all $\binom{n}{3}$ triples and all $\binom{n}{2}$ pairs of points. For each, there is a unique circle to check (each check takes $O(n)$). In total, this is $O(n^4)$.

A better (and intended) solution is to look at all $\binom{n}{2}$ pairs of points and try to find a circle that goes through those two points. The centre of any circle through these two points must be on the perpendicular bisector of the two points. Each point defines an interval along this line where the centre can be. The intersection of all of the intervals are all of the valid places that the centre of the circle could be. If the interval is empty, then these two points cannot be on the boundary of the circle. In total, this is $O(n^3)$.

E Election Frenzy

Author: Darcy Best — Complexity: $O(t + L)$, where L is the total length of the lists.

Can you colour the vertices of a graph with 2 colours in such a way that each vertex is adjacent to a vertex of the other colour?

Note that the only time that it is `Impossible` is when there is an isolated vertex. Otherwise, find a spanning forest of the graph. Any 2-colouring of that spanning forest will be a valid solution (all adjacent vertices in that graph are of the opposite colour). The easiest way (but definitely not the only way) to find a spanning forest is to make a DFS tree (depth first search tree).

Unfortunately, this algorithm is $O(n + m)$ where m is the number of edges in the graph. In this problem, there can be $O(n^2)$ edges, so this is too slow. So instead of keeping a visited array, keep a linked list of all unvisited nodes. When a table lists other tables that it *can* see, do DFS as usual. When a table lists the tables that it *cannot* see, go through the linked list of tables not visited already. If that table is in the list, continue on. If that table is not in the list, then there is an edge in the graph (recursively call DFS).

To see that this is the correct complexity: each node can be removed from the visited linked list just once and when DFS is called on each vertex, we must go through each integer in the the input exactly once.

F False Intelligence

Author: Walter Guttmann — Complexity: $O(1)$ per query + $O(k^2)$, where k is the number of definable functions.

Is each truth table definable using four operations?

Note that there are only 3^9 different possible states. We will build a list of all definable states and have a list of all states that we need to process. First of all, $g(x, y) = x$ and $g(x, y) = y$ are definable. Push these into the list of states we need to process.

While the list of states we need to process is not empty, take a state out (call this function $h(x, y)$). For every state that is already defined (call that function $g(x, y)$), the following eight states are also definable:

$$g(x,y)\ \texttt{AND}\ h(x,y) \quad g(x,y)\ \texttt{OR}\ h(x,y) \quad g(x,y)\ \texttt{EQUALS}\ h(x,y) \quad g(x,y)\ \texttt{IMPLIES}\ h(x,y)$$

$$h(x,y)\ \texttt{AND}\ g(x,y) \quad h(x,y)\ \texttt{OR}\ g(x,y) \quad h(x,y)\ \texttt{EQUALS}\ g(x,y) \quad h(x,y)\ \texttt{IMPLIES}\ g(x,y)$$

For all new states, push them into the list of states and the list of states that we need to process.

There are only 1944 definable functions.

G Graphics Design

Author: Darcy Best — Complexity: $O\left(\sum d_i \log(\max(a,b,c))\right)$

When does each student finish their last subproject?

This problem is a direct simulation problem–there is not much other than implementation.

Here is one way: keep a heap (priority queue) for each of the 8 possible requirements. These heaps contain the next subprojects for each student. The heaps are sorted first by time and then by priority. You then have 3 more heaps, one for each of Camera, Camcorder and Computer indicating when each item is next available.

The next subproject to do is (probably) one of the eight on the top of the subproject heaps. Determine which of the eight should come first. If there are not enough items available for rent in each of them, put the items back into the heap with a later time (the time of the next available item that it needs). If you find a subproject to do, then remove it from the heap as well as all of the items you need and push the next subproject for that student (if applicable) onto the correct heap and update the item heaps.

H Hilbert's Hotel

Author: Tim French — Complexity: $O(\sqrt{n} + \log^3 n)$

How many k^{th} powers are there modulo n?

This is a very tricky problem to do. The code that you write in the end is reasonably short, but the math behind it is quite deep.

H.1 Non-Mathematical Overview

- The answer is multiplicative. If m and n are relatively prime, and you can count the number of k^{th} powers modulo m and the number of k^{th} powers modulo n, then the product of those is the number of k^{th} powers modulo $m \times n$.
- This means that if you can solve for the number of k^{th} powers modulo p^a (where p is a prime), then you can just factor n and solve the problem for the prime-power factors of n.
- Split this problem up into 2 parts:
  - Count the number of units (numbers that are relatively prime to p^a) that are k^{th} powers modulo p^a.
  - Find a recursive relation for the number of non-units that are k^{th} powers modulo p^a.

H.2 Technical Details

Here is a very nice paper which solves the problem for $k = 2$ (that is, the number of squares modulo n)

`http://www.maa.org/sites/default/files/Walter_D22068._Stangl.pdf`

The remainder of this section will follow the paper quite closely to make it easier to follow along. I will assume that you understand everything in that paper from the beginning to the first THEOREM on page 287 (everything past that THEOREM is very specific to $k = 2$ and will not be needed).

For this section, I will be assuming familiarity with basic Group, Ring and Field Theory. Most topics are easily Google-able if you are interested. The most important topics to know: Groups, Rings, Field, units, isomorphisms, Euler-phi function ($\phi(n)$).

Important Background Information:

These results can be found in the introduction of some Number Theory textbooks:

Theorem 1 (Chinese Remainder Theorem)**.** *If* $\gcd(m,n) = 1$*, then the ring* $\mathbb{Z}_{mn}$ *is isomorphic to the ring* $\mathbb{Z}_m \times \mathbb{Z}_n$*.*

Theorem 2. *Let* p *be an odd prime. The multiplicative group of the units of* $\mathbb{Z}_{p^a}$ *is isomorphic to the additive group* $\mathbb{Z}_{\phi(p^a)}$*.*

Theorem 3. *The multiplicative group of the units of* $\mathbb{Z}_{2^a}$ *(sometimes called* $\mathbb{Z}^*_{2^a}$*) is isomorphic to the additive group* $\mathbb{Z}_{2^{a-2}} \times \mathbb{Z}_2$*.*

Details

If no proof is given below, the corresponding proof from the paper above works with simple changes from squares to k^{th} powers.

Definition 1. *Let* $s_k(n)$ *be the number of* k^{th} *powers modulo* n*. Let* $q_k(n)$ *be the number of units that are* k^{th} *powers modulo* n*.*

Lemma 1. *If* $\gcd(m,n) = 1$*, then*

$$s_k(m \cdot n) = s_k(m) \cdot s_k(n)$$

and

$$q_k(m \cdot n) = q_k(m) \cdot q_k(n).$$

Corollary 1. *If* $p_1^{a_1} \cdot p_2^{a_2} \cdot \dots \cdot p_m^{a_m}$ *is the prime factorization of* n*, then*

$$s_k(n) = s_k(p_1^{a_1}) \cdot s_k(p_2^{a_2}) \cdot \dots \cdot s_k(p_m^{a_m}).$$

From here on out, we will attempt to solve $f_k(p^a)$ and $g_k(p^a)$ for a prime p.

Theorem 4. b *is a* k^{th} *power modulo* p^{a-k} *if and only if* $b \cdot p^k$ *is a* k^{th} *power modulo* p^a*.*

Corollary 2. *For* $a \geq k$*,*

$$s_k(p^a) = q_k(p^a) + s_k(p^{a-k}).$$

Theorem 5. *If* p *is an odd prime, then*

$$q_k(p^a) = \frac{\phi(p^a)}{\gcd(\phi(p^a), k)}.$$

Proof. By Theorem 2, the multiplicative group of units in $\mathbb{Z}_{p^a}$ is isomorphic to the additive group $\mathbb{Z}_{\phi(p^a)}$. This means that instead of counting k^{th} powers in $\mathbb{Z}_{p^a}$, we can instead count the number of multiples of k in $\mathbb{Z}_{\phi(p^a)}$. It is a short exercise to show that the number of multiples of k in $\mathbb{Z}_n$ is $\frac{n}{\gcd(n,k)}$. □

Theorem 6. *If* $p = 2$*, then*

$$q_k(p^a) = \begin{cases} \frac{2^{a-2}}{\gcd(2^{a-2},k)} & \textit{if } a \textit{ is even} \\ \frac{2^{a-2}}{2\gcd(2^{a-2},k)} & \textit{if } a \textit{ is odd.} \end{cases}$$

Proof. By Theorem 3, the multiplicative group of units in $\mathbb{Z}_{2^a}$ is isomorphic to the additive group $\mathbb{Z}_{2^{a-2}} \times \mathbb{Z}_2$. Once again, this means that instead of counting k^{th} powers in $\mathbb{Z}_{p^a}$, we can instead count the number of multiples of k in $\mathbb{Z}_{2^{a-2}} \times \mathbb{Z}_2$. □

I Intuidiff II

Author: Evgeni Sergeev — Complexity: $O(n \log n)$

> How many highlighted characters are there in a string?

This problem is essentially asking you for a weighted longest increasing subsequence. The normal longest increasing subsequence can be modified to account for this. For example: Let `A[x]` be the largest weight subsequence whose last element is `x`. Initially, `A[x]=0` for all `x`. Sweeping from left-to-right,

```
A[x] = max(A[x], length(segment) + max(A[0...x-1])).
```

The `max(A[0...x-1])` and update can be computed in $O(\log n)$ time using a Fenwick Tree (Binary Index Tree).

J Just Terraffic!

Author: Richard Lobb — Complexity: $O(n)$

> How many cars with and without trailers are there on the road?

To solve this problem, you need to use Dynamic Programming. Let `A[i]` be the number of cars with trailers assuming that a car starts at index `i` (or `Ambiguous` or `Impossible`).

Check if the current car could be a 2 wheel or 3 wheel car. It can be a 2 wheel car if: (a) $t_{i+1} - t_i < 2000$, (b) $t_{i+2} - t_{i+1} > 1000$ and (c) `A[i+2]` is not `Ambiguous` or `Impossible`. There are similar checks for a 3 wheel vehicle. Update `A[i]` appropriately being careful about `Ambiguous`.

K Kiwis vs Kangaroos

Author: Darcy Best — Complexity: $O(n)$

> How many characters are in `KANGAROO` or `KIWIBIRD`?

This is a very straightforward problem, no tricks.