# Problem Analysis Session

SWERC Judges

SWERC 2015

| Problem | Est. # ACC | # ACC $<$ 4h |
|---|---|---|
| A - Promotions | 27 | 12 |
| B - Black Vienna | 8 | 4 |
| C - Canvas Painting | 12 | 12 |
| D - Dice Cup | 47 | 52 |
| E - Wooden Signs | 15 | 16 |
| F - Landscaping | 3 | 2 |
| G - Game of Cards | 3 | 7 |
| H - Sheldon Numbers | 25 | 26 |
| I - Text Processor | 1 | 0 |
| J - St. John's Festival | 5 | 11 |

# H - Sheldon Numbers

## Problem

Given two positive integers $A$ and $B$, compute the number of integers in the range $[A, B]$, whose binary representation has the form $1^n(0^m1^n)^k$ or $1^n(0^m1^n)^k0^m$.

## Classification

- Categories: Number Theory
- Difficulty: Easy

# H - Sheldon Numbers

## Sample Solution

- Testing if each integer is a Sheldon Number is not feasible given the limits on $A, B$
- However, we can easily generate Sheldon Numbers $\leq$ *limit* bits:

  for $n \leftarrow 1$ to *limit* // number of ones
      for $m \leftarrow 0$ to $limit - n$ // number of zeros
          $k \leftarrow 0$ // repetitions
          while $n + k \times (n + m) \leq limit$
              generate $1^n(0^m1^n)^k$
              $k \leftarrow k + 1$

  // and similar for numbers of the form $1^n(0^m1^n)^k0^m$
- A few thousands under 63 bits: we can just generate all of them
- Need a collection to remove duplicates (e.g. Array/List/TreeSet/...) and count elements in the range $[A, B]$

# A - Promotions

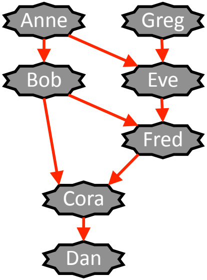
## Problem

Given a directed acyclic graph where each edge $(x, y)$ means that employee $y$ may be promoted only if employee $x$ is promoted, and a number $N$ of promotions (which is an endpoint of $[A, B]$):

- How many employees will certainly be promoted?
- How many employees have no possibility of being promoted?

## Classification

- Categories: Graph Theory
- Difficulty: Easy

## Sample Solution

- DAG with $V$ nodes and $E$ edges; $N$ promotions.
- Employee $x$ has no possibility of being promoted if the number of predecessors of $x$ (excluding $x$) is at least $N$:

$$\mathrm{Pred}(x) \geq N.$$

- Employee $x$ will certainly be promoted if the number of successors of $x$ (excluding $x$) is at least $V - N$:

$$\mathrm{Suc}(x) \geq V - N.$$

- For every employee $x$, compute $\mathrm{Pred}(x)$ and $\mathrm{Suc}(x)$.
- Any $\mathcal{O}(V^2 + VE)$ time algorithm (BFS/DFS/TS) was accepted.

# C - Canvas Painting

## Problem

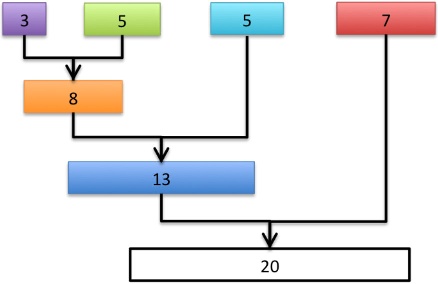
Given the sizes of the canvasses Samuel bought, find the minimum amount of ink the machine needs to spend in order to have all canvasses with different colors.

## Classification

- Categories: Greedy, Huffman Coding
- Difficulty: Medium

## Sample Solution

- Lets think backwards: merge blocks of distinct colors into one.




- Consider that the canvasses sizes are the weight of each node. The problem becomes equivalent to building the Huffman tree to minimize $\sum_{i=1}^{N} size_i \times length(c_i)$

**Sample Solution**

- We can build the Huffman tree in $\mathcal{O}(N \log N)$
- Insert the given sizes in a priority queue
- While we have more than one item
  - Pop the two smallest items $A$ and $B$
  - Add $A + B$ to the result
  - Push $A + B$ into the priority queue

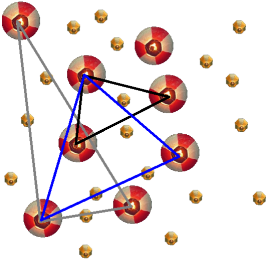
# J - Saint John Festival

## Problem

Given two sets of points $\mathcal{A}$ and $\mathcal{B}$ in the **plane**, how many points in $\mathcal{B}$ are in the interior or on the boundary of triangles defined by any 3 points in $\mathcal{A}$.

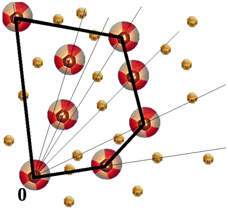
## Classification

- Categories: Geometry
  - Convex hull
  - Point in **convex** polygon
- Difficulty: Medium



**Sample Solution**

Background (from Charatheodory's Theorem): The union of all triangles having vertices in $\mathcal{A}$ is the **convex hull** $\mathcal{CH}(\mathcal{A})$ of $\mathcal{A}$.



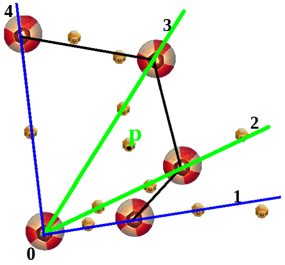

1. Find $\mathcal{CH}(\mathcal{A})$ in $\mathcal{O}(L \log L)$, e.g., by Graham scan, with $L = |\mathcal{A}|$.
   - Do not compute polar angles.
   - Make use of primitive operations based on cross product and inner product: left-turn, right-turn; handle collinearities.
2. Then, for each $p \in \mathcal{B}$, check if $p \in \mathcal{CH}(\mathcal{A})$ efficiently.

# J - Saint John Festival

## Sample Solution

For each $p \in \mathcal{B}$, check if $p \in \mathcal{CH}(\mathcal{A})$ in $\mathcal{O}(\log h)$, where $h$ is the number of vertices of $\mathcal{CH}(\mathcal{A})$.

- $\mathcal{CH}(\mathcal{A})$ partitioned into wedges with apex $p_0$ (the lowest vertex);
- The wedges are already sorted. **Binary Search** for finding $p$.




$p \in$ wedge$(0, 2, 3)$
$(0, 2, p)$ L-turn
$(0, 3, p)$ R-turn
$(2, 3, p)$ L-turn

- $\mathcal{CH}(\mathcal{A})$ is a **convex** polygon.
- Overall time complexity: $\mathcal{O}(L \log L + S \log h)$ or $\mathcal{O}((L + S) \log L)$.
- Robust: L-turn; R-turn; collinear; point in line segment.
- Other partitions of $\mathcal{CH}(\mathcal{A})$, e.g., in two monotone chains.
- Optional filter: $p \in \text{MBBox}(\mathcal{CH}(\mathcal{A}))$?

# D - Dice Cup

## Problem

Given the number of sides of 2 dice, compute the most likely outcomes for the sum of two dice rolls. Assume each die has numbered faces starting at 1 and that each face has equal roll probability.

## Classification

- Categories: Math, Mode
- Difficulty: Easy

# D - Dice Cup

**Sample Solution #1**

- Enumerate possible values for each die: $v_1 \leftarrow 1$ to $N$, $v_2 \leftarrow 1$ to $M$
- Count occurrences of the sums $v_1 + v_2$ in an array (histogram)
- Output all sums with the most occurrences

**Sample Solution #2**

- Observe that the most likely values are just $1 + \min(N, M)$ to $1 + \max(N, M)$

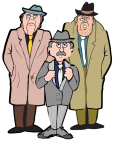
# B - Black Vienna

## Problem

Count sets of three suspects (*Black Vienna*) that are *not* in either of two player's hands using their replies to *investigations*.

## Classification

- Categories: Satisfiability
- Difficulty: Medium



**Base Approach**

- There are only $\binom{26}{3} = 2600$ choices for the BV; we can enumerate them and check all investigations.
- However, there are $2^{26-3}$ assignments of the remaining suspects to players – too much to try out exhaustively.

**Sample Solution**

- For each choice of BV, encode investigations as *boolean XOR clauses*
  - Consider $2 \times 26$ boolean variables $A_1, A_2, B_1, B_2, \ldots Z_1, Z_2$
  - Variable $X_i$ is 1 iff player $i$ holds suspect $X$
  - For each investigation, add constraints (3 possible replies):

| | |
|---|---|
| `AB i 0` | $A_i = 0 \ \wedge \ B_i = 0$ |
| `AB i 2` | $A_i = 1 \ \wedge \ B_i = 1$ |
| `AB i 1` | $A_i \oplus B_i = 1$ |

  - For each suspect $X$ in BV: add constraints $X_1 = 0 \wedge X_2 = 0$
  - For remaining suspects $Y$: add constraints $Y_1 \oplus Y_2 = 1$

# B - Black Vienna

**2 Possibilities:**

1. **Use 2-SAT**:
   - Turn XOR clauses into 2-SAT form: $A \oplus B = (A \vee B) \wedge (\neg A \vee \neg B)$.
   - 2-SAT can be checked in linear time.
2. **Solve XOR satisfiability directly** using *Gaussian elimination* in $O(N^3)$ time.

# E - Wooden Signs

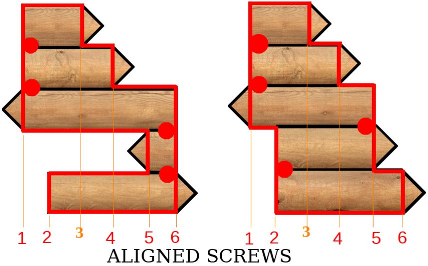
## Problem

Count the number of wooden signs that can be encoded by a given permutation $\pi$ of $1..(N+1)$ according to some crafting rules.

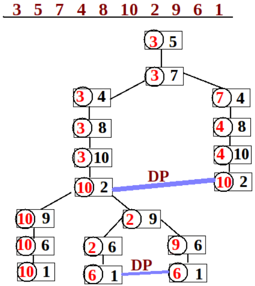
Equivalently, how many **row convex permutominoes** are encoded by $\pi$?

## Classification

- Categories: Dynamic Programming
- Difficulty: Medium



## Sample Solution




| $(\pi_j \mid \pi_k, \pi_{k+1} \ldots)$, with $k > j$ | Case |
|---|---|
| $\pi_j > \pi_k > \pi_{k+1}$ | 1 |
| $\pi_j < \pi_k < \pi_{k+1}$ | 1 |
| $\pi_j > \pi_k < \pi_{k+1}, \pi_j > \pi_{k+1}$ | 2 |
| $\pi_j < \pi_k > \pi_{k+1}, \pi_j < \pi_{k+1}$ | 2 |
| $\pi_j > \pi_k < \pi_{k+1}, \pi_j < \pi_{k+1}$ | 3 |
| $\pi_j < \pi_k > \pi_{k+1}, \pi_j > \pi_{k+1}$ | 3 |

1. $C(j,k) = C(j,k+1); \quad C(j,N+1) = 1$
2. $C(j,k) = C(j,k+1) + C(k,k+1)$
3. $C(j,k) = C(k,k+1)$

- Visit search tree in DFS (with memoization): $\mathcal{O}(N^2)$ time.
- Space $\Theta(N^2)$ accepted but $\Theta(N)$ optimal: keep $C(j,k)$ as $C[j]$.

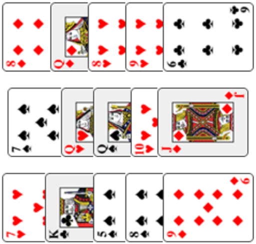
# G - Game of Cards

## Problem

Given the description of the piles and the maximum number of cards Alice and Bob can remove, can Alice win the game if she is the first to play?

## Classification

- Categories: Game Theory
- Difficulty: Medium-Hard



**Sample Solution**

- There are evident similarities between this game and **Nim**.
- Lets imagine that players could always remove any number of cards from any pile:
  - Then a player is in a **losing** position if the **xor** of the sizes of the piles equals 0.
- **Why?** Let $X = n_1$ xor $n_2$ xor ... xor $n_p$
  - An empty board has $X = 0$ and is a **losing** position.
  - If $X \neq 0$, then the active player can make $X = 0$ by changing the biggest pile.
  - If $X = 0$, then the board will always have $X \neq 0$ after this move.
- Therefore, $X = 0$ **has to be a losing position**.

**Our goal:** given our restricted moves, what is each pile's equivalent size in the game of **Nim**?

- Calculating **Grundy Numbers (aka Nimbers):**
  - A pile with no valid play has a Grundy Number $= 0$.
  - A position has value of $k$ if we can move to positions with Grundy Numbers $\{0, ..., k-1\}$.
- We can find all Grundy Numbers and if Alice can win in $\mathcal{O}(P \times N \times K)$, i.e. trying every move once.
- By the way, every *impartial game* is equivalent to a Nimber (Sprague-Grundy theorem).

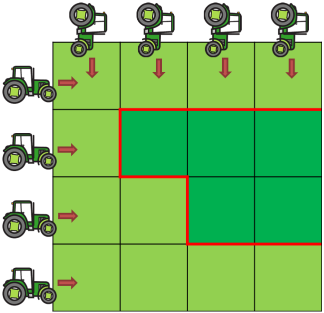
# F - Landscaping

## Problem

Given a matrix representing a field, the price to change the height of a cell and the price to pay for adjacent cells at different levels, find the minimum amount that you will have to pay.

## Classification

- Categories: Graphs - Minimum Cut
- Difficulty: Hard



# F - Landscaping

- Trying every modification is prohibitive: $\mathcal{O}(2^{NM})$.
- **Insight:** We are trying to separate the low and the high cells using a simple cost structure.

### Solution

We can model our matrix as a graph and find a minimum cut.

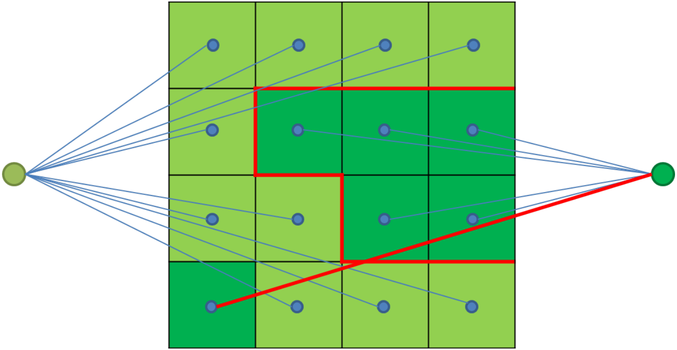
### Max-Flow Min-Cut theorem:

The maximum amount of flow from the source to the sink node is equal to the minimum cut that separates the 2 nodes.

# F - Landscaping

**Graph:**

- Connect every cell to its neighbors using an edge with capacity $A$.
- Connect every low cell to a *super-low* node with capacity $B$.
- Connect every high cell to a *super-high* node with capacity $B$.



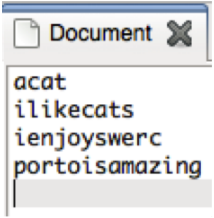
# I - Text Processor

## Problem

Given a string $S$, a fixed width $W$, and $Q$ queries, find the number of distinct substrings for each query interval $[i, i + W - 1]$.

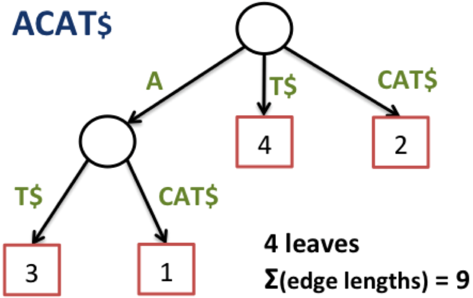
## Classification

- Categories: Strings, Suffix Tree
- Difficulty: Hard



**Key insight**

*The number of distinct substrings of a string S is equal to the length of the edges in its Suffix Tree*



# I - Text Processor

**Base Approach**

- If we build the suffix tree for each query range and count the length of the tree edges, we can answer all queries.
- Problem: building the suffix tree for each query would have a $\mathcal{O}(Q \times W)$ time complexity, which was too slow.

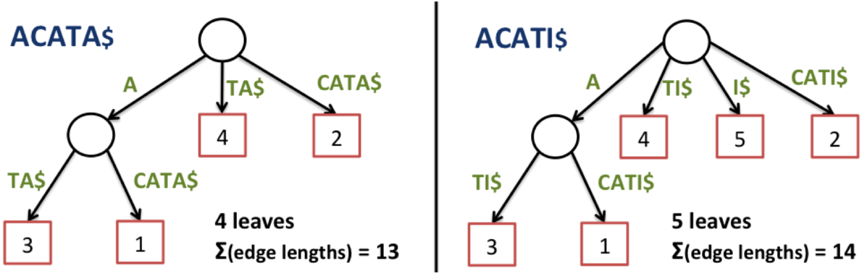
# I - Text Processor

## Sample Solution

- All queries have the same width $W$. We can use a sliding window.
- The difference between queries $i$ and $i+1$ is adding the letter at position $i+W$ and removing the letter at position $i$.
- If we build the Suffix Tree online, e.g. using Ukkonen's algorithm, adding a letter follows from the algorithm itself.
- Removing a letter in this sliding window is equivalent to removing the largest suffix of the tree. This can be done in $\mathcal{O}(1)$ amortized time.
- If we keep the sum of the length of the tree edges while we do these operations, we can answer all queries in $\mathcal{O}(|D|+Q)$ time.

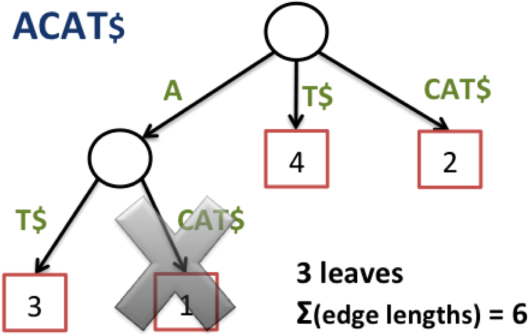
### Adding a letter

- Keep a counter for the number of leaves in the tree.
- Adding a letter increases the number of distinct substrings by the number of leaves in the tree.
  - If adding a letter creates a new leaf, add that to both counters.



**Removing the largest suffix**

- The largest suffix is represented in the oldest leaf.
- Keep a queue of the leaves created. Remove the leaf in the front.
- Delete the edges corresponding to this leaf and update its parents if necessary.



# Questions?