Mathematics

Thursday, 12 October 2017 6:50 PM

Problem A: Factovisors

- The question is asking for $a|b$!
- We can represent $a$ and $b$ as product of first $k$ prime numbers $p_1, \ldots, p_k$
  - $a = p_1^{n_1} \cdots p_k^{n_k}$, $b! = p_1^{m_1} \cdots p_k^{m_k}$
- Then we only need to check whether
  - $n_i \leq m_i$, $\forall i \in \{1, 2, \ldots, k\}$
- We first need to factor $a$ out to obtain $n_i$
- For any prime number $p_i$, we can find $m_i$ by
  - $m_i = \left\lfloor \frac{n}{p_i} \right\rfloor + \left\lfloor \frac{n}{p_i^2} \right\rfloor + \ldots$
- We can stop the summing process once $p_i^k > n$

Problem B: Stacking Curvy Blocks

- First define functions $f(x)$ and $g(x)$ to be
  - $f(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3$
  - $g(x) = f_0 + t_1 x + t_2 x^2 + t_3 x^3$
- The question is asking for
  - $max(f - g) - min(f - g)$
- To find the maximum and minimum value of $(f - g)$, we need to take the derivative first
  - $\frac{d(f - g)}{dx} = (b_1 - t_1) + 2(b_2 - t_2)x + (b_3 - t_3)x^2 = 0$, $(0 \leq x \leq 1)$
- Let's denote the solution $x_1, x_2$ (if exists and $x_1, x_2 \in [0, 1]$)
- Then the maximum and minimum value is among
  - $f(0), f(1), f(x_1), f(x_2)$

Problem C: Hamming Ellipses

- $h(f_1, s) + h(f_2, s) = D$
- Method 1: Combination

$$
\begin{array}{cccccc}
 f_1 & 0 & 1 & 2 & 0 & 1 \\
 f_2 & 2 & 1 & 2 & 0 & 1 \\
 x_1 & +1 & +0 & x_3 \\
 x_2 & +2 & +2 & x_4 \\
\end{array}
$$

- $x_1 + 2x_2 + 2x_3 = 0$
- s.t. $\begin{cases} \ x_1 + x_2 = n_{\text{diff}} \\ x_3 + x_4 = n_{\text{same}} \end{cases}$
- $\text{result} = \sum_{i=0}^{n_{\text{same}}} 2^i (q - 2)^i \ 1^i (q - 1)^i \ \left( \binom{n_{\text{same}}}{x_3} \right) \left( \binom{n_{\text{diff}}}{x_2} \right)$
- Be care for overflow
\[ \binom{n}{k} = \frac{n(n-1)(n-k+1)}{k(k-1)...1} = \frac{n-1}{1} \cdot \frac{n-2}{2} \cdot \frac{n-k+1}{k} \]

**D: Hamming Ellipses (1)**

- Task: Count the number of length-\(n\) strings over \(q\) symbols where \(\text{hammingdist}(p, f_1) + \text{hammingdist}(p, f_2) = D\).
  - \(f_1 = 01201\), \(f_2 = 21210\), \(p = 10002\)
  - In positions where \(f_1\) matches \(f_2\), the symbol in \(p\) may
    - \((k_1)\) match \(f_1\) and \(f_2\), or
    - \((k_2)\) differ from both \(f_1\) and \(f_2\) in \((q-1)\) ways.
  - In positions where \(f_1\) differs from \(f_2\), the symbol in \(p\) may
    - \((k_3)\) differ from both \(f_1\) and \(f_2\) in \((q-2)\) ways, or
    - \((k_4)\) differ from either \(f_1\) or \(f_2\) in 2 ways.
  - Calculate \(w = \text{hammingdist}(f_1, f_2)\)
  - For all \(k_2, k_3, k_4\) such that \(k_2 \leq n - w\) and \(k_3 + k_4 = w\) and \(2k_2 + 2k_3 + 4k_4 = D\), count the number of points on the ellipse:
    \[
    (q-1)^{k_2} (q-2)^{k_3} 2^{k_4} \binom{n-w}{k_2} \binom{w}{k_3}
    \]
  - Must be very careful to avoid overflow of int64_t

**D: Hamming Ellipses (2)**

- Task: Count the number of length-\(n\) strings over \(q\) symbols where \(\text{hammingdist}(p, f_1) + \text{hammingdist}(p, f_2) = D\).
  - Alternative solution: dynamic programming over \(D\) and \(n\).
  - Construct a table \(\text{npoint}[k, d] = \text{number of points at distance} \ d, \text{considering only the first} \ k \text{symbols of the strings.}\)
  - If \(f_1\) and \(f_2\) match at position \(k\):
    \[
    \text{npoint}[k, d] = \text{npoint}[k-1, d] + (q-1) \cdot \text{npoint}[k-1, d-2]
    \]
  - If \(f_1\) and \(f_2\) differ at position \(k\):
    \[
    \text{npoint}[k, d] = (q-2) \cdot \text{npoint}[k-1, d-2] + 2 \cdot \text{npoint}[k-1, d-1]
    \]
  - Final answer is \(\text{npoint}[n, D]\)
  - Easier and safe against overflow.

**Problem D: Linear Recurrences**

- Linear Recursion

---

\[ f(n) = a_1 \cdot f(n - 1) + a_2 \cdot f(n - 2) + \ldots + a_k \cdot f(n - k) \]

\[
\begin{bmatrix}
  f(n-1) \\
  \vdots \\
  f(n-k) \\
\end{bmatrix} = 
\begin{bmatrix}
  f(n) \\
  \vdots \\
  f(n-k+1) \\
\end{bmatrix}
\]

\[ M_{Ak} = A_n + 1 \implies A_n = M^a A_0 \]

- We can compute \( M^a \) with \( \Theta(\log(n)) \) by divide and conquer
- \( M^a = M^{n2} M^{n1} \)

**Reference**


**Problem E: First Orchard**

- Markov Chain

![Markov Chain Diagram]

- Matrix representation

**An Example of Markov Chains**

- \( \Omega = \{1, 2, 3, 4, 5\} \)
- \( X = (X_0, X_1, \ldots, X_i, \ldots) \in \Omega \)
- where \( X_0 \) is initial state and so on.
- \( P \) is transition matrix.

\[
\begin{bmatrix}
  0.4 & 0.6 & 0.0 & 0.0 & 0.0 \\
  0.5 & 0.0 & 0.3 & 0.0 & 0.0 \\
  0.0 & 0.3 & 0.0 & 0.7 & 0.0 \\
  0.0 & 0.0 & 0.1 & 0.3 & 0.6 \\
  0.0 & 0.0 & 0.0 & 0.5 & 0.2 \\
\end{bmatrix}
\]

**Reference**

- [https://en.wikipedia.org/wiki/Markov_chain](https://en.wikipedia.org/wiki/Markov_chain)

**Problem F: Primonimo**

- Notice that the order doesn’t matter
- So we only care how many times we push the cells.
- We need to set variables for each cell to indicate the times we need to push the cell
- For any variable, we have the equation below:

\[ x_{ij} + \sum_{k \in \{1, \ldots, m\}} x_{kj} + \sum_{k \in \{1, \ldots, n\}} x_{ik} + a_{ij} \equiv 0 \pmod{p} \]

- We can use Gaussian elimination to solve the system of equations above

![Matrix Representation Diagram]
### System of equations

<table>
<thead>
<tr>
<th>System of equations</th>
<th>Row operations</th>
<th>Augmented matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2x + y - z = 8$</td>
<td></td>
<td>$\begin{bmatrix} 2 &amp; 1 &amp; -1 &amp; 8 \ -3 &amp; -1 &amp; 2 &amp; -11 \ -2 &amp; 1 &amp; 2 &amp; -3 \end{bmatrix}$</td>
</tr>
<tr>
<td>$-3x - y + 2z = -11$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$-2x + y + 2z = -3$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$2x + y - z = 8$</td>
<td>$L_2 + \frac{3}{2}L_1 \rightarrow L_2$</td>
<td>$\begin{bmatrix} 2 &amp; 1 &amp; -1 &amp; 8 \ 0 &amp; 1/2 &amp; 1/2 &amp; 1 \ 0 &amp; 2 &amp; 1 &amp; 5 \end{bmatrix}$</td>
</tr>
<tr>
<td>$\frac{1}{2}y + \frac{1}{2}z = 1$</td>
<td>$L_3 + L_1 \rightarrow L_3$</td>
<td></td>
</tr>
<tr>
<td>$2y + z = 5$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$2x + y - z = 8$</td>
<td>$L_3 + -4L_2 \rightarrow L_3$</td>
<td>$\begin{bmatrix} 2 &amp; 1 &amp; -1 &amp; 8 \ 0 &amp; 1/2 &amp; 1/2 &amp; 1 \ 0 &amp; 0 &amp; -1 &amp; 1 \end{bmatrix}$</td>
</tr>
<tr>
<td>$\frac{1}{2}y + \frac{1}{2}z = 1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$-z = 1$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The matrix is now in echelon form (also called triangular form)

| $2x + y = 7$     | $L_2 + \frac{1}{2}L_3 \rightarrow L_2$ | $\begin{bmatrix} 2 & 1 & 0 & 7 \\ 0 & 1/2 & 0 & 3/2 \\ 0 & 0 & -1 & 1 \end{bmatrix}$ |
| $\frac{1}{2}y = 3$ | $L_1 - L_3 \rightarrow L_1$ |                |
| $-z = 1$          |                |                  |

| $2x + y = 7$     | $2L_2 \rightarrow L_2$ | $\begin{bmatrix} 2 & 1 & 0 & 7 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -1 \end{bmatrix}$ |
| $y = 3$          | $-L_3 \rightarrow L_3$ |                |
| $z = -1$         |                |                  |

| $x = 2$          | $L_1 - L_2 \rightarrow L_1$ | $\begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -1 \end{bmatrix}$ |
| $y = 3$          | $\frac{1}{2}L_1 \rightarrow L_1$ |                |
| $z = -1$         |                |                  |

- Flow chart for Gaussian elimination

**Flowchart for forward elimination.**

**Flowchart for back substitution.**

- Pseudocode
• Modular arithmetic

\[ a^{-1} \equiv \frac{m(m+1)+1}{a} \pmod{m} \]

• Reference
  ○ [https://en.wikipedia.org/wiki/Modular_arithmetic](https://en.wikipedia.org/wiki/Modular_arithmetic)