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 $p_1, \ldots, p_k$
  - $a = p_1^{n_1} \ldots p_k^{n_k}, \quad b! = p_1^{m_1} \ldots p_k^{m_k}$
- Then we only need to check whether
  - $n_i \le m_i \ , \ \forall i \in \{1, 2, \ldots, k\}$
- 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
  - $f(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3$
  - $g(x) = t_0 + t_1 x + t_2 x^2 + t_3 x^3$
- 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
  - $\frac{d(f-g)}{dx} = (b_1 - t_1) + 2(b_2 - t_2)x + (b_3 - t_3)x^2 = 0, \quad (0 \le x \le 1)$
- 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

f_1	0	1	2	0	1
f_2	2	1	2	1	0
x_1	+1	+0	x_3		
x_2	+2	+2	x_4		

  - $x_1 + 2x_2 + 2x_4 = 0$
  - $s.t. \begin{cases} x_1 + x_2 = n_{diff} \\ x_3 + x_4 = n_{same} \end{cases}$
  - $result = \sum_{x_4 \in \{0,1,\ldots,n_{same}\}} 2^{x_1} (q-2)^{x_2} 1^{x_3} (q-1)^{x_4} \binom{n_{same}}{x_3} \binom{n_{diff}}{x_2}$
  - Be care for overflowing

- $\binom{n}{k} = \frac{n(n-1)\ldots(n-k+1)}{k(k-1)\ldots 1} = \frac{n}{1}\cdot\frac{n-1}{2}\cdots\frac{n-k+1}{k}$

D: Hamming Ellipses (1)

Benelux Algorithm Programming Contest 2016

- Task: Count the number of length-n strings over q symbols where $\mathrm{hammingdist}(p, f_1) + \mathrm{hammingdist}(p, f_2) = D$.

$f_1 = 0\,1\,2\,0\,1$, $f_2 = 2\,1\,2\,1\,0$, $p = 1\,0\,0\,0\,2$

- 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 $w = \mathrm{hammingdist}(f_1, f_2)$
- For all k_2, k_3, k_4 such that $k_2 \le n - w$ and $k_3 + k_4 = w$ and $2k_2 + 2k_3 + k_4 = D$, 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`

Solutions — BAPC Preliminaries 2016 — September 24, 2016 6 / 18

- Method 2: Dynamic Programming
  - $dist(pos, d) = \begin{cases} dist(pos-1, d) + (q-1)\cdot dist(pos-1, d-2), & if\ agree \\ 2dist(pos-1, d-1) + (q-2)\cdot dist(pos-1, d-2), & if\ disagree \end{cases}$

D: Hamming Ellipses (2)

Benelux Algorithm Programming Contest 2016

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

Solutions — BAPC Preliminaries 2016 — September 24, 2016 7 / 18

- Reference
  - https://2016.bapc.eu/media/filer_public/2016/09/30/bapc2016-preliminaries-solutions.pdf

Problem D: Linear Recurrences

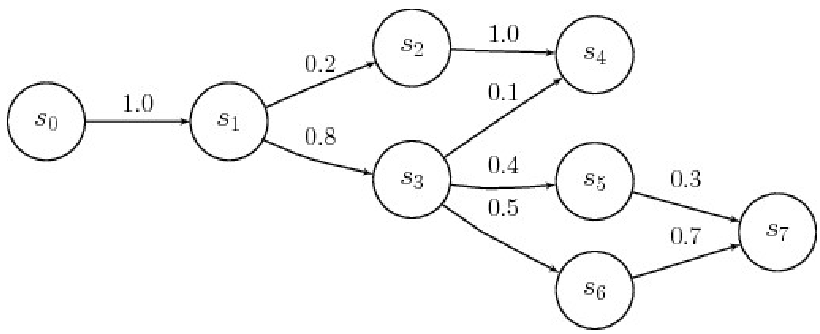
- Linear Recursion

- $f(n) = a_1 \cdot f(n-1) + a_2 \cdot f(n-2) + \ldots + a_k \cdot f(n-k)$
- $M \begin{bmatrix} f(n-1) \\ \vdots \\ f(n-k) \end{bmatrix} = \begin{bmatrix} f(n) \\ \vdots \\ f(n-k+1) \end{bmatrix}$
- $MA_n = A_n + 1 \implies A_n = M^n A_0$
- We can compute M^n with $\Theta(log(n))$ by divide and conqure
- $M^n = M^{n/2} M^{n/2}$

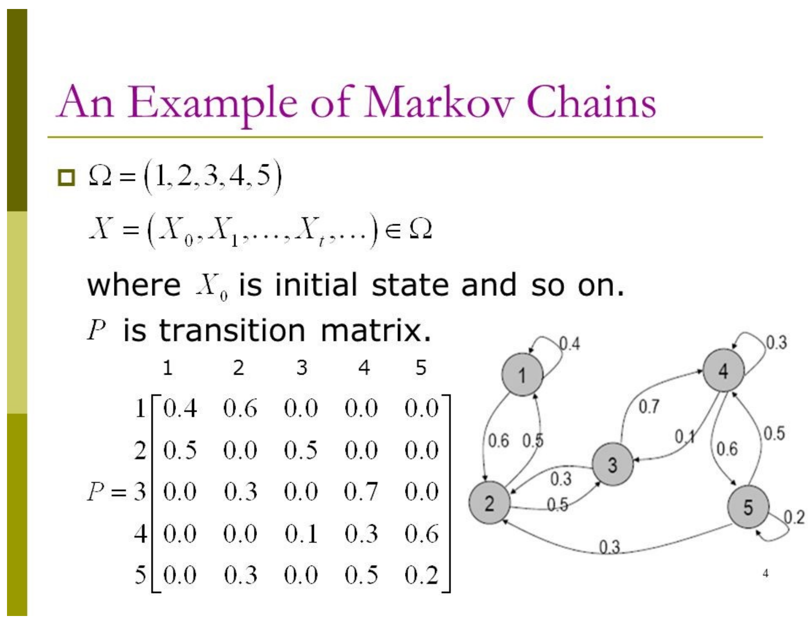
- Reference
  - https://en.wikipedia.org/wiki/Recurrence_relation

Problem E: First Orchard

- Markov Chain



- Matrix representation



- Reference
  - https://en.wikipedia.org/wiki/Markov_chain

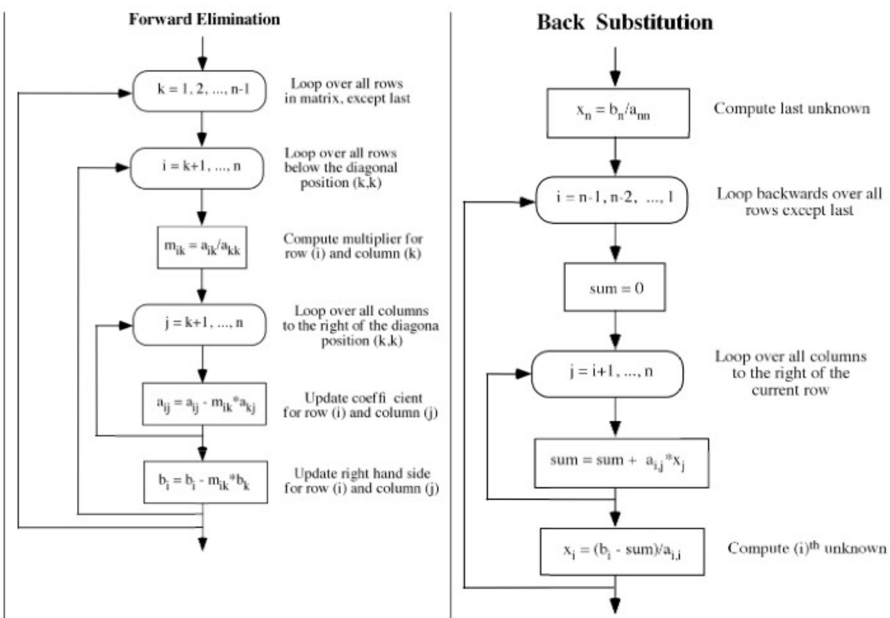
Problem F: Primonimo

- 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:
  - $x_{i,j} + \sum_{\substack{k \in \{1 \ldots m\} \\ k \neq i}} x_{k,j} + \sum_{\substack{k \in \{1 \ldots n\} \\ k \neq j}} x_{i,k} + a_{i,j} \equiv 0 \left(mod\ p \right)$
- We can use Gaussian elimination to solve the system of equations above

System of equations	Row operations	Augmented matrix

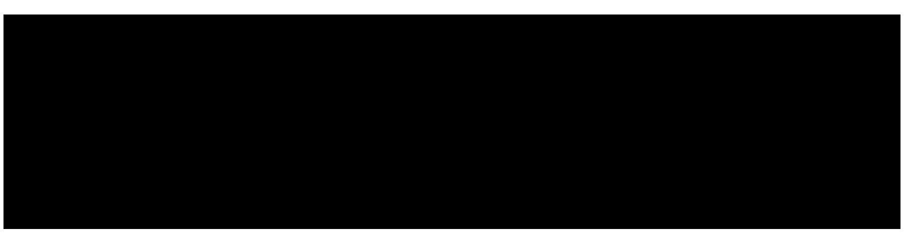
System of equations	Row operations	Augmented matrix
$2x + y - z = 8$ $-3x - y + 2z = -11$ $-2x + y + 2z = -3$		$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ -3 & -1 & 2 & -11 \\ -2 & 1 & 2 & -3 \end{array}\right]$
$2x + y - z = 8$ $\frac{1}{2}y + \frac{1}{2}z = 1$ $2y + z = 5$	$L_2 + \frac{3}{2}L_1 \to L_2$ $L_3 + L_1 \to L_3$	$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ 0 & 1/2 & 1/2 & 1 \\ 0 & 2 & 1 & 5 \end{array}\right]$
$2x + y - z = 8$ $\frac{1}{2}y + \frac{1}{2}z = 1$ $-z = 1$	$L_3 + -4L_2 \to L_3$	$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ 0 & 1/2 & 1/2 & 1 \\ 0 & 0 & -1 & 1 \end{array}\right]$
The matrix is now in echelon form (also called triangular form)		
$2x + y = 7$ $\frac{1}{2}y = \frac{3}{2}$ $-z = 1$	$L_2 + \frac{1}{2}L_3 \to L_2$ $L_1 - L_3 \to L_1$	$\left[\begin{array}{ccc\|c} 2 & 1 & 0 & 7 \\ 0 & 1/2 & 0 & 3/2 \\ 0 & 0 & -1 & 1 \end{array}\right]$
$2x + y = 7$ $y = 3$ $z = -1$	$2L_2 \to L_2$ $-L_3 \to L_3$	$\left[\begin{array}{ccc\|c} 2 & 1 & 0 & 7 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -1 \end{array}\right]$
$x = 2$ $y = 3$ $z = -1$	$L_1 - L_2 \to L_1$ $\frac{1}{2}L_1 \to L_1$	$\left[\begin{array}{ccc\|c} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -1 \end{array}\right]$

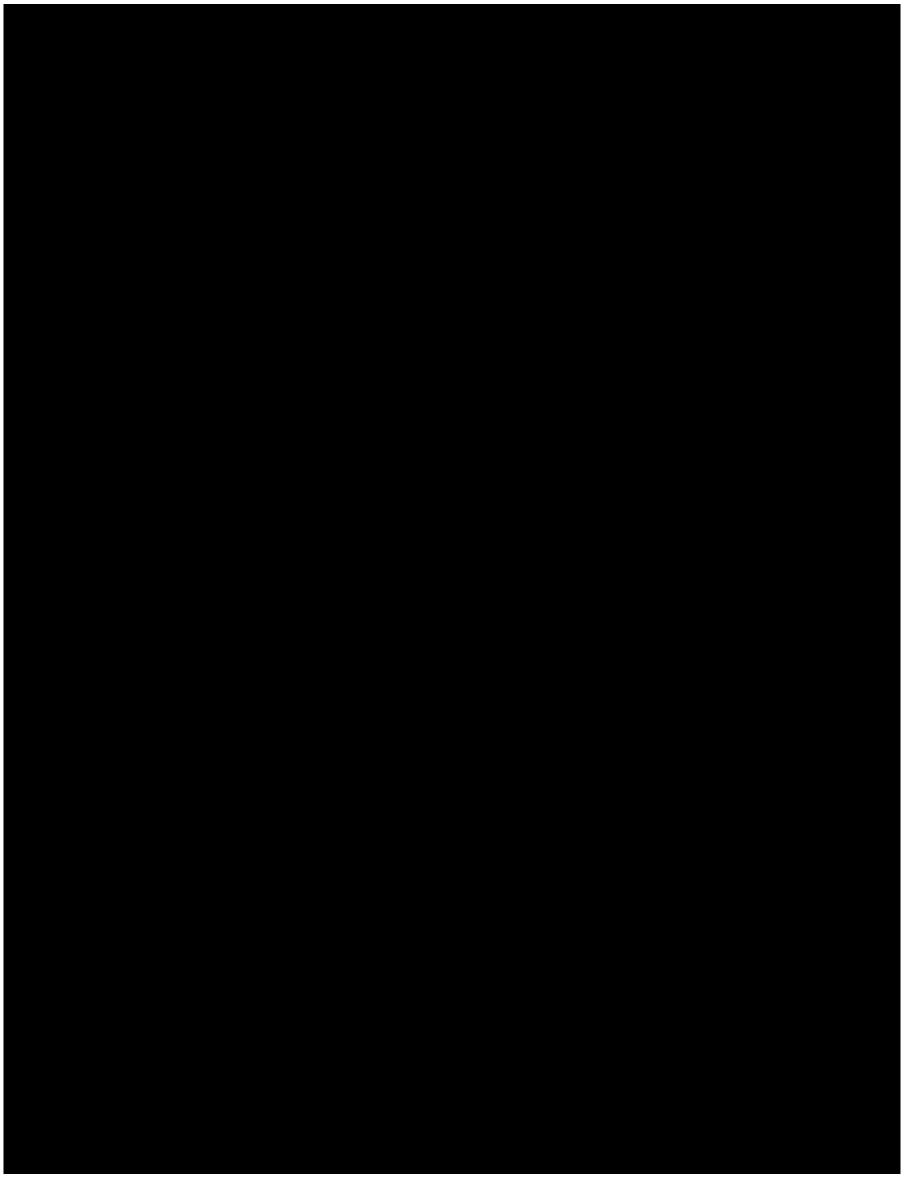
- Flow chart for Gaussian elimination



Flowchart for forward elimination. Flowchart for back substitution.

- Pseudocode





- Modular arithmetic

$$a^{-1} = \frac{m(n+1)+1}{a} \pmod{m}$$

- Reference
  - https://en.wikipedia.org/wiki/Gaussian_elimination
  - https://en.wikipedia.org/wiki/Modular_arithmetic