

1 包除原理

数え上げのテクニックの一つ。集合 A が与えられ、その要素を変数とする述語たち $\mathcal{P} = \{P_1, \dots, P_n\}$ を考える。 $1 \leq i \leq n$ に対して A の部分集合 A_i を $A_i = \{a \in A \mid P_i(a)\}$ で定めるとき、 $\bigcup_{i=1}^n A_i$ の要素数を求めるものである。 $\Lambda_n = \{1, \dots, n\}$ とする。

$$\left| \bigcup_{i=1}^n A_i \right| = \sum_{\emptyset \subset \Lambda \subseteq \Lambda_n} (-1)^{|\Lambda|-1} \cdot \left| \bigcap_{i \in \Lambda} A_i \right|. \quad (1)$$

$$= \sum_{j=1}^n (-1)^{j-1} \cdot \left(\sum_{|\Lambda|=j} \left| \bigcap_{i \in \Lambda} A_i \right| \right). \quad (2)$$

Λ の要素数を固定したときに $\left| \bigcap_{i \in \Lambda} A_i \right|$ の総和を DP などでも求められるなら、式 (2) を用いて計算することができる。また、特に $|\Lambda| = |\Lambda'| \implies \left| \bigcap_{i \in \Lambda} A_i \right| = \left| \bigcap_{i \in \Lambda'} A_i \right|$ であるなら、 $|\Lambda| = j$ であるような Λ の代表元を Λ^j と書くことにして次のように変形できる。

$$\left| \bigcup_{i=1}^n A_i \right| = \sum_{j=1}^n (-1)^{j-1} \cdot {}_n C_j \cdot \left| \bigcap_{i \in \Lambda^j} A_i \right|. \quad (3)$$

式 (1) においては \sum で足される項が $2^n - 1$ 個だったのに対し、式 (2)–(3) では n に減っていてうれしい。

上の議論は、 $\left| \bigcap_{i \in \Lambda} A_i \right|$ を計算するのが容易であることを前提としているが、逆に $\left| \bigcap_{i \in \Lambda} (A \setminus A_i) \right|$ の計算が容易な状況^{*1}で $\bigcap_{i=1}^n A_i$ の要素数を求めたいときには以下のようにするとよい。

$$\bigcap_{i=1}^n A_i = A \setminus \left(\bigcup_{i=1}^n (A \setminus A_i) \right).$$

すなわち、

$$\left| \bigcap_{i=1}^n A_i \right| = |A| - \left| \bigcup_{i=1}^n (A \setminus A_i) \right|. \quad (4)$$

これは、 $A \setminus A_i$ を A_i と置き直すことで、式 (1) の枠組みで求められる。

^{*1} 満たす条件を決め打ちするよりも、満たさない条件を決め打ちした方が楽な場合。

1.1 証明

\mathcal{P} の大きさ n に関する帰納法で示す． $n = 2$ の場合は Venn 図などから示される（省略）． $n < k$ で成り立っていることを仮定する．まず， $n = 2$ の場合の式を用いて次のように変形する．

$$\begin{aligned} \left| \bigcup_{i=1}^k A_i \right| &= \left| \left(\bigcup_{i=1}^{k-1} A_i \right) \cup A_k \right| \\ &= \left| \bigcup_{i=1}^{k-1} A_i \right| + |A_k| - \left| \left(\bigcup_{i=1}^{k-1} A_i \right) \cap A_k \right| \\ &= \left| \bigcup_{i=1}^{k-1} A_i \right| + |A_k| - \left| \bigcup_{i=1}^{k-1} (A_i \cap A_k) \right| \end{aligned}$$

さらに， $n = k - 1$ の場合の式を用いて右辺の各項は次のように変形できる．

$$\begin{aligned} \left| \bigcup_{i=1}^{k-1} A_i \right| &= \sum_{\emptyset \subset \Lambda \subseteq \Lambda_{k-1}} (-1)^{|\Lambda|-1} \cdot \left| \bigcap_{i \in \Lambda} A_i \right| . \\ |A_k| &= \sum_{\Lambda=\{k\}} (-1)^{1-1} \cdot \left| \bigcap_{i \in \Lambda} A_i \right| . \\ - \left| \bigcup_{i=1}^{k-1} (A_i \cap A_k) \right| &= - \sum_{\emptyset \subset \Lambda \subseteq \Lambda_{k-1}} (-1)^{|\Lambda|-1} \cdot \left| \bigcap_{i \in \Lambda} (A_i \cap A_k) \right| \\ &= - \sum_{\emptyset \subset \Lambda \subseteq \Lambda_{k-1}} (-1)^{|\Lambda|-1} \cdot \left| \left(\bigcap_{i \in \Lambda} A_i \right) \cap A_k \right| \\ &= \sum_{\emptyset \subset \Lambda \subseteq \Lambda_{k-1}} (-1)^{|\Lambda \cup \{k\}|-1} \cdot \left| \bigcap_{i \in \Lambda \cup \{k\}} A_i \right| . \end{aligned}$$

第 1 項は A_k を含まない $k - 1$ 個以下の積集合，第 2 項は A_k のみからなる集合，第 3 項は A_k を含む 2 個以上 k 個以下の積集合に関する式になっており，次のようにまとめることができる．

$$\left| \bigcup_{i=1}^k A_i \right| = \sum_{\emptyset \subset \Lambda \subseteq \Lambda_k} (-1)^{|\Lambda|-1} \cdot \left| \bigcap_{i \in \Lambda} A_i \right| .$$

1.2 発展

1.2.1 高速ゼータ変換・高速 Möbius 変換

高速ゼータ変換は，集合 S の部分集合を指数に取る関数 f に対して $g(S) = \sum_{T \subseteq S} f(T)$ を $O(|S| \cdot 2^{|S|})$ 時間で求める．高速 Möbius 変換はその逆変換に相当し， $f(S) = \sum_{T \subseteq S} (-1)^{|S \setminus T|} \cdot g(T)$ を求める．

```
// fast zeta transformation
for (size_type i = 0; i < n; ++i)
    for (size_type j = 0; j < (1_zu << n); ++j)
        if (j >> i & 1_zu) dp[j] += dp[j ^ (1_zu << i)];
```

```
// fast Moebius transformation
for (size_type i = 0; i < n; ++i)
    for (size_type j = 0; j < (1_zu << n); ++j)
        if (j >> i & 1_zu) dp[j] -= dp[j ^ (1_zu << i)];
```

各 j について $dp[j] = f(j)$ で初期化し、ゼータ変換を施すと $dp[j] = g(j)$ になっている。Möbius 変換についても g と f が逆になること以外は同様である。ここで、 j は部分集合を 2 進数でエンコードしたものである。

たとえば、 Λ_n の各部分集合 S に対して $\sum_{T \subseteq S} (-1)^{|T|} \cdot |\bigcap_{i \in T} A_i|$ を考える。これは、 $g(T) = |\bigcap_{i \in T} A_i|$ として高速 Möbius 変換で得られる $f(S)$ を用いて $(-1)^{|S|} \cdot f(S)$ として計算できる。

1.2.2 DP

式 (2) より、 j 個の条件を満たす場合の数を求めることを考える。式 (4) を用いて j 個の条件に違反する場合の数を考えることもできる。

$dp[i][j]$ では、 i 番目までの条件を見て、そのうち j 個の条件を満たす場合の数（を求めるための値^{*2}）として定義する。 $dp[i][j]$ から $dp[i+1][j+1]$ への遷移（ i 番目の条件を満たす）と $dp[i+1][j]$ への遷移（ i 番目の条件を考慮しない）をそれぞれがんばって考える。その後、 $dp[n][j]$ から $\sum_{|\Lambda|=j} |\bigcap_{i \in \Lambda} A_i|$ を求めることで式 (2) を計算できる。

状態数が $O(n^2)$ であり、 $n \leq 5000$ などでは MLE しうるので、一次元配列を使いまわすテクを使うとよい。また、操作回数など別のパラメータが必要になる場合は適宜 DP の次元を必要がある（それはそう）。

1.2.3 約数系包除

書かなきゃにゃん。

1.2.4 その他

Everything on It 的なを書く。

^{*2} 遷移のしやすさと相談するとよさそう。

1.3 補足

1.3.1 暗黙の制約

$n \leq 10^9$, $n \leq 10^{18}$ なら n の持つ素因数の個数はそれぞれ、たかだか 9 個, 15 個なので, n が多少大きくても n の素因数に関する部分集合であれば式 (1) をそのまま計算することができる.