Schwartz-Zippel lemma による hash の解析

noshi91

2020 年 6 月 6 日

1 概要

Schwartz-Zippel lemma に基づく様々な hash や乱択アルゴリズムの解析を行う。

2 Schwartz–Zippel lemma

Schwartz-Zippel lemma を用いると、多項式にランダムな値を代入したときに 0 になる確率を上から押さえることが出来る。

定理 1. (Schwartz-Zippel lemma) F を体, Q を 0 でない F 上の d 次の多変数多項式、S を F の有限部分集合とする。Q の各変数に S から一様ランダムかつ独立に選んだ値を代入すると、それが 0 になる確率は $d|S|^{-1}$ 以下である。

特に競技プログラミングで頻出の状況に限定するならば、以下のようになる。

系 1. p を素数, Q を 0 でない $\mathbb{F}_p$ 上の d 次の多変数多項式とする。P の各変数に $\mathbb{F}_p$ から一様ランダムかつ独立に選んだ値を代入すると、それが 0 になる確率は dp^{-1} 以下である。

3 rolling hash

rolling hash は素数 p, 基数 b, 数列 S に対して以下のように定義される。

$$\mathrm{hash}(S) \coloneqq \left(\sum_{0 \leq i < |S|} S_i b^i \right) \bmod p$$

ただし S の各要素は $[0, p)$ に含まれる整数とする。

この hash が衝突する、すなわち異なる数列 S, T について $\mathrm{hash}(S) = \mathrm{hash}(T)$ となる確率を考える。利便のため、$n \coloneqq \max(|S|, |T|)$ とする。着目することは、$\mathrm{hash}(S)$ が $\mathbb{F}_p$ 上の $|S| - 1$ 次の 1 変数多項式に b を代入した形になっている事である。すなわち、多項式

$$R \coloneqq \sum_{0 \leq i < n} (S_i - T_i) x^i$$

を考えると $Q(b_0, b_1, \ldots) = 0 \Leftrightarrow \mathrm{hash}(S) = \mathrm{hash}(T)$。また、$S \neq T$ ならばある i が存在して $S_i \neq T_i$ である

から、R は 0 でない。よって系 1 を適用することで、b を一様ランダムに選んだ時衝突の確率が np^{-1} 以下になることが言える。

4 一般論

前節の議論を一般化し、集合 S の元に対する hash を設計することを考える。$Q : S \to \mathbb{F}_p[x_0, x_1, \ldots]$ を用いて $\mathrm{hash} : S \to \mathbb{F}_p$ を $\mathrm{hash}(s) \coloneqq Q(s)(b_0, b_1, \ldots)$ と定義すると、系 1 の適用の為には $s, t \in S$ について $s \neq t \Leftrightarrow Q(s) - Q(t) = 0$ が満たされている必要がある。これは Q が単射であれば十分である。

まとめると以下のようになる。

定理 2. 素数 p, 一様ランダムかつ独立な $\mathbb{F}_p$ の値 $b_0, b_1, \ldots$, 単射 $Q : S \to \mathbb{F}_p[x_0, x_1, \ldots]$ について、$f : S \to \mathbb{F}_p$ を $f(s) \coloneqq Q(s)(b_0, b_1, \ldots)$ と定めると

$$\forall s, t \in S, s \neq t \Rightarrow \Pr[f(s) = f(t)] \leq \frac{\max(\deg(Q(s)), \deg(Q(t)))}{p}$$

Proof. $s \neq t$ である $s, t \in S$ を固定する。Q は単射であるから $Q(s) \neq Q(t)$、よって $R \coloneqq Q(s) - Q(t)$ とすれば $R \neq 0$ かつ $\deg(R) \leq \max(\deg(Q(s)), \deg(Q(t)))$。$f(s) = f(t) \Leftrightarrow R(b_0, b_1, \ldots) = 0$ より、系 1 から示される。 □

f が hash として使う関数である。Q は $Q(s)(b_0, b_1, \ldots)$ が効率的に計算可能かつ $\deg(Q(s))$ が小さくなるように選択することが望ましい。

5 多重集合に対する hash

台集合 S と重複度 $m : S \to \mathbb{F}_p$ の組に対する hash を考える。$Q(S, m) \coloneqq \sum_{s \in S} m(s) x_s$ と定義すれば定理 2 を適用して hash 関数が得られ、衝突の確率は p^{-1} 以下。

6 集合に対する hash

本節は zobrist hashing [1] と同様の手法で集合に対する hash を定義する。これは前節の hash と類似している。

$b_s \in \mathbb{F}_{2^w}$ を各 s について独立かつ一様ランダムに選択し、集合 S に対する $\mathrm{hash} : S \to \mathbb{F}_{2^w}$ を以下のように定義する。

$$\mathrm{hash}(S) \coloneqq \sum_{s \in S} b_s$$

$\mathbb{F}_{2^w}$ 上の加法は整数の xor と同じであるから、効率的に計算できる。定理 2 と微妙に状況が異なるが、全く同様の手順で定理 1 を適用して衝突の確率が 2^{-w} 以下であると分かる。

この hash の特長として、hash が集合同士の対称差を保存することが挙げられる。すなわち $\mathrm{hash}(S) + \mathrm{hash}(T) = \mathrm{hash}(S \oplus T)$。

7 2D rolling hash

rolling hash と同様にして、2 次元のデータに対する hash も作成することが出来る。

$$Q(A) \coloneqq \sum_{0 \le i < n} \sum_{0 \le j < m} A_{i,j} x^i y^j$$

と定義し、定理 2 を適用すればよい。衝突の確率は $(n+m)p^{-1}$ 以下である。

8 行列積の検算

本節は Freivalds' algorithm [2] と類似したアルゴリズムを説明する。

$\mathbb{F}_p$ 上の $n \times n$ 行列 A, B, C があり、$AB = C$ かどうかを判定する問題を考える。結論から言えば、ランダムな n 次元ベクトル v を用いて $ABv = Cv$ を判定すればよい。等しくない場合確実に $AB \neq C$ であり、等しければ高い確率で $AB = C$ であることが期待される。

$D \coloneqq AB - C$ とすると $ABv = Cv \Leftrightarrow Dv = \mathbf{0}$。$v$ の各要素 v_i を変数と見ると、もし $D \neq O$ ならばある i が存在して $(Dv)_i \neq 0$ である。そこで 1 次の n 変数多項式となる $(Dv)_i$ に対して系 1 を適用することで、誤って True と判定する確率は p^{-1} 以下という結果を得る。

9 順序無し根付き木の hash

本節は [3] の内容と概ね同一である。

まず、根付き木から多項式への写像 Q を以下のように定める。

$$Q(t) = \prod_i (x_d + Q(t_i)) \quad (d \text{ は } t \text{ の深さ}, t_i \text{は } t \text{ の子})$$

補題 3. Q は単射である

Proof. $Q(t)$ から t が一意に定まることを示す。

1. $Q(t)$ が変数を含まない場合
   $Q(t) = 1$ であり t は単一のノードからなる。
2. $Q(t)$ が変数を含む場合
   d を x_d が $Q(t)$ に含まれるような最大の d とする。t は深さ d の木であり、t_i を t の根の子とすれば $Q(t) = \prod_i (x_d + Q(t_i))$ と表現できる。この因数分解は一意であるから、各 $Q(t_i)$ に再帰的に補題 3 を適用する。

□

よって定理 2 から hash 関数を得る。衝突の確率は np^{-1} 以下である (n は t の頂点数)。

参考文献

[1] Zobrist hashing – Wikipedia

[2] Freivalds' algorithm – Wikipedia
[3] 根付き木のハッシュ – あなたは嘘つきですかと聞かれたら「YES」と答えるブログ