Schwartz—Zippel lemma による hash の解析

noshi91

2023 年 12 月 3 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 $d/|S|$ 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 2. 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。Q 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 d/p 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。

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

ただし S の各要素は $[1, p)$ に含まれる整数とする。

この hash が衝突する、すなわち相異なる数列 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

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

を考えると $R(b) = 0 \Leftrightarrow \mathrm{hash}(S) = \mathrm{hash}(T)$。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 2 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 n/p 以下になることが言える。

4 一般化

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。$Q\colon S \to \mathbb{F}_p[x_0, x_1, \ldots]$ を用いて $\mathrm{hash}\colon S \to \mathbb{F}_p$ を $\mathrm{hash}(s) \coloneqq Q(s)(b_0, b_1, \ldots)$ と定義すると、系 2 の適用の為には $s, t \in S$ について $s \neq t \implies Q(s) - Q(t) \neq 0$ 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 3. 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 $Q\colon S \to \mathbb{F}_p[x_0, x_1, \ldots]$ について、$\mathrm{hash}\colon S \to \mathbb{F}_p$ を $\mathrm{hash}(s) \coloneqq Q(s)(b_0, b_1, \ldots)$ と定めると

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

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 $m\colon S \to \mathbb{F}_p$ 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 3 を適用して hash 関数が得られ、衝突の確率は $1/p$ 以下。この hash は $\mathrm{hash}(S) + \mathrm{hash}(T) = \mathrm{hash}(S + T)$ が成り立つため、集合同士の重複度込みの和の hash が効率的に計算できるという特徴がある。

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。

$Q(S) \coloneqq \sum_{s \in S} x_s$ と定義し、これを $\mathbb{F}_{2^w}$ 上で考える。2^w は素数ではないが $\mathbb{F}_{2^w}$ は体であるから定理 3 と同様の事実が成り立ち、得られる hash 関数の衝突の確率は 2^{-w} 以下であると分かる。$\mathbb{F}_{2^w}$ 上の加法は整数の xor と同型であるから、効率的に計算できる。この hash は $\mathrm{hash}(S) + \mathrm{hash}(T) = \mathrm{hash}(S \bigtriangleup T)$ が成り立つため、集合同士の対称差の hash が効率的に計算できるという特徴がある。

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。

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

と定義し、定理 3 を適用すればよい。衝突の確率は $(n + m)/p$ 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 2 を適用することで、誤って True と判定する確率は $1/p$ 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 4. Q は単射である

証明. $Q(t)$ から t が一意に定まることを示す。

(i) $Q(t)$ 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。

(ii) $Q(t)$ 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 4 を適用する。

□

よって定理 3 から hash 関数を得る。衝突の確率は n/p 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