テーマ１：ストリームアルゴリズム（その２）

本日の講義の目的

- 最頻出回数近似問題
- 最頻出回数近似問題の効率よいストリーミング計算は不可能！
- 最頻出回数近似問題を効率よく行うストリームアルゴリズム？

1.2. 最頻出回数近似問題とそのストリーミング計算困難性

まずは問題を定義する．以下では次のような記法を用いる．

$$
\begin{aligned}
\boldsymbol{a} &= a_1, a_2, ..., a_t, ..., a_m \;=\; a'_1, a'_2, ..., a'_j, ..., a'_n \;=\; \text{与えられる入力列} \\
\boldsymbol{a}_t &= a_1, ..., a_t \;\; \text{時刻 } t \text{ までの列} \\
D &= \{0, ..., N-1\} \;=\; \text{入力要素 } a_i \text{ の定義域} \\
n_{j,t} &= a'_j \text{ が } \boldsymbol{a}_t \text{ に出現する回数} \\
F_{\infty,t} &= \max_{1 \le j \le n} n_{j,t}.
\end{aligned}
$$

この $F_{\infty,t}$ が（時刻 t までにおける）最頻出データの頻出回数であり，これを簡単に<u>最頻出回数</u>と呼ぶ．

ただし，最頻出回数を（非常に限られた領域計算量で）正確に求めるのは難しいように思える．そこで近似を考えよう．一般に目標の値 F に対して，$[(1-\varepsilon)F, F]$ の範囲に入る値を F の<u>ε-近似</u>（正確には，<u>相対誤差 ε-近似</u>）という．ここで最初に考えるのは，次のような近似問題である．

<u>最頻出回数近似問題</u>（相対近似版）
入力： 許容誤差パラメータ ε,　入力列 $\boldsymbol{a} = a_1, a_2, ..., a_t, ...$
仕事： 各 t で $F_{\infty,t}$ の相対誤差 ε-近似を報告する．

実は，近似を目標にしたとしても，さらに，最後のデータ（上の記法では a_m）のときの値 F_∞ $(= F_{\infty,m})$ だけを求めるだけでも，$\mathrm{polylog}\, n$ の領域計算量のストリームアルゴリズムは不可能である．次のような定理を示すことができる（証明は補講のときにでも）．

定理 1.5. 任意の与えられた列 $\boldsymbol{a} = a_1, ..., a_m$ （ただし，$m \le 2N$ で $a_i \in \{0, ..., N-1\}$）に対し，F_∞ の相対誤差 1/3-近似を 0.8 以上の確率で計算するストリームアルゴリズムの領域量は cN 以上である．

1.3. 最頻出回数近似問題の（効率よい）ストリームアルゴリズム

上記のような限界が「証明」されてしまうと苦しい．しかし，どうにもならないだろうか？実は，世の中には，妥当な近似をするアルゴリズムが存在する．しかも最頻出データ候補をすべて列挙することさえできる．今回はそれを紹介しよう．

一見矛盾するような結果となるのは近似の考え方に違いがあるからだ．正解から相対誤差で $1 \pm \varepsilon$ の近似というのは妥当そうに思えるが，正解が小さいときは厳しい．実際，不可能性の証明の中で使う（意地悪な）例は，$F_\infty = 1$ もしくは $F_\infty = 2$ のどちらかになってしまう例なのである．したがって，$\varepsilon = 1/3 - 0.001$ だとしても，$F_\infty = 1$ の場合には近似は高々 $4/3$ 未満であり，$F_\infty = 2$ の場合には近似は少なくとも $4/3$ より上なので，正解を求めていることと同じになってしまうのである．

ここでは見方を変える．絶対誤差（に近い誤差評価）を用いることで「妥当な近似」の定義を変えてみよう．具体的には次のような近似を考えるのである．

定義 1.6. 入力した列 $a_1, a_2, ..., a_t$ 中で，最も出現する a'_j を <u>最頻出データ</u> という．つまり，最頻出データとは $F_{\infty,t}$ 回出現するデータである．それに対して <u>ϵ-絶対誤差内頻出データ</u> とは，$n_{j,t} \geq F_{\infty,t} - \epsilon t$ となるような a'_j である．

つまり，$F_{\infty,t}$ の <u>絶対誤差 ϵ-近似</u> となるような出現回数のデータが，ϵ-誤差内頻出データである．相対誤差と絶対誤差の差は

$$\begin{aligned} \epsilon\text{-絶対誤差内頻出} &\iff F_{\infty,t} - \epsilon t \text{ 回以上出現} \\ \varepsilon\text{-相対誤差内頻出} &\iff (1-\varepsilon)F_{\infty,t} \text{ 回以上出現} \end{aligned}$$

である．

このように，最頻出回数近似問題を，この絶対近似版で見直してみると，$F_{\infty,t}$ を ϵt の誤差範囲で求めるだけでなく，その範囲の出現回数のデータをすべて列挙するストリームアルゴリズムを作ることができる．つまり，最頻出データ候補の列挙アルゴリズムである．逆に言えば，その中で最も多く出現しているデータの近似出現回数 $\widehat{F}_{\infty,t}$（これも同時に求まる）を用いると，

$$\widehat{F}_{\infty,t} \ \leq \ F_{\infty,t} \ \leq \ \widehat{F}_{\infty,t} + \epsilon t$$

と推定することができる．

アルゴリズムの記述の前に，いくつか用語を用意しておく．時刻 t までの入力列 $\boldsymbol{a}_t$ を考える．与えられた $\epsilon' > 0$ に対して，$w = \lceil 1/\epsilon' \rceil$ とし，連続する w 個のデータをブロックと呼ぶ．より正確には，データ $a_1, ..., a_w$ が第 1 ブロック，$a_{w+1}, ..., a_{2w}$ が第 2 ブロック,... となる．総ブロック数は $\lceil t/w \rceil \leq t/(1/\epsilon') + 1 = t\epsilon' + 1$ である．

各データ a に対して，(a, f, b) の三つ組をデータ情報と呼ぶ．ここで a はデータそのもの，f は（数え始めてからの）出現回数，b は（数え始めてからの）ブロック経過数，つまり現在までに何個のブロックを通ってきたかの回数である．

Algorithm Frequent_Item_Select

input: $\epsilon > 0$; $\boldsymbol{a} = a_1, ..., a_t, ..., a_m$;

var.s: S: データ情報の集合;

init. comp:

$S = \emptyset$; $\epsilon' = \epsilon/2$;

comp. at t**:**

a_t に対するデータ情報が S に無い場合には $(a_t, 1, 1)$ を S に入れる;

一方，$S \in (a_t, f, b)$ の場合には，$(a_t, f+1, b)$ に更新する;

t がブロックの最後（つまり $t = kw$）のとき {

S 中のデータ情報 (a, f, b) のうち，$f \leq b$ となるものを消去;

残りのデータ情報 (a, f, b) を $(a, f, b+1)$ に更新する;

}

S 中のデータ情報のうち，f が最大のものを $\widehat{F}_{\infty,t}$ とする;

$\widehat{S} = \{\, a : (a, f, b) \in S \wedge f \geq \widehat{F}_{\infty,t} - t\epsilon' \,\}$;

$\widehat{F}_{\infty,t}$ と $\widehat{S}$ を出力する;

end.

入力列 $a_1, ..., a_t$ 中に出てくる任意のデータ a' を考える．その出現回数を n' とする．一方，a' が最初に出現してから，S に連続して保持されるブロック数を u_1，そして u_1 ブロック終了時の a' のデータ情報中の f の値を f_1，次に連続して保持されるブロック数を u_2，最後の f の値を f_2,... とする．時刻 t はブロックの終わりではないが，a_t の処理後の f の値を便宜上 f_{k+1} とする（保持されていないときは $f_{k+1} = 0$）．

このとき，$f_1 + \cdots + f_{k+1} = n'$ である．また，誤差は $f_1 + \cdots + f_k$ であり，$f_i \leq u_i$ である．しかも $\sum u_i \leq k - 1 \leq t\epsilon'$ であることから，次の補題が導ける．

補題 1.7. $f_{k+1} \;\geq\; n' - t\epsilon' \;=\; n' - (t\epsilon/2)$.

このことから，アルゴリズムの正当性を保証する次の定理を導くことができる．

定理 1.8. アルゴリズムが出力する $\widehat{F}_{\infty,t}$ と $\widehat{S}$ に対して，次のことが成り立つ．

(1) $\widehat{F}_{\infty,t} \geq F_{\infty,t} - t\epsilon' = F_{\infty,t} - (t\epsilon/2)$.

(2) $n' \geq F_{\infty,t} \Rightarrow a'$ は $\widehat{S}$ に含まれる．

(3) $n' < F_{\infty,t} - t\epsilon \Rightarrow a'$ は $\widehat{S}$ に含まれない．

次にアルゴリズムの使用する領域量を評価する．そのためには，S に何個のデータ情報が保持さえるかを見積もれればよい．仮に時刻 t が k 番目のブロックの最後とする．つまり $k = t/w$ である．この k 番目のブロックの終了時（正確にはその直前）の S の要素を考える．その中で k ブロックで新たに保持されたデータの個数を n_1，$k-1$ ブロッ

クから保持されているデータの個数を n_2，以下，$n_3, ..., n_k$ を同様に定義する．明らかに $0 \le n_i \le w$ が成り立つ．また，第 $k-(i-1)$ ブロックから第 k ブロックまで残っているデータの f の値は，$f > i-1$（すなわち $f \ge i$）を満たしている．一方，すべてのデータの総出現回数は高々 $t = kw$．しがって，

$$\sum_{i=1}^{k} i \cdot n_i \ \le \ kw. \tag{1}$$

このような条件下では次の補題が成り立つ．

補題 1.9. 各 $n_1, ..., n_k$ に対し，$0 \le n_i \le w$ が成り立ち，しかも (1) も成立する．このとき

$$\sum_{i=1}^{k} n_i \ \le \ w \sum_{i=1}^{k} \frac{1}{i}$$

ここで $\sum_{i=1}^{k} \frac{1}{i} \le \ln k$ であることを使えば，次の定理は直ちに導かれる．

定理 1.10. S に保持されるデータ情報数は高々次のワード数である．

$$w \ln k \ = \ O\left(\frac{\ln(t\epsilon)}{\epsilon}\right).$$

注意すべき点

このアルゴリズムは以下の論文からの引用である．ただし，この論文では，時刻 t の時点で，与えられた閾値 γ に対して $\gamma \cdot t$ 回以上出現するデータの列挙を目標にしている．これを最頻出回数の近似の話しに（少々強引だが）修正したのが今回の話である．

この絶対近似が意味を持つのは，このように，最頻出データの出現回数が全データ数の定数割合である場合である．最頻出回数がもっと少ない割合の場合には，それに応じた領域計算量が必要になることを忘れてはならない．

[1] G.S. Manku and R. Motwani, Approximate frequency counts over data streams, in *Proc. of the 28th International Conference on Very Large Data Bases* (VLDB'02), 2002.

お楽しみ問題（テーマ１：ストリームアルゴリズム）

1.3. 補題 1.9 を証明してみよう．

↑これはあくまでお楽しみ．**テーマ＃１での課題**は次ページを！

テーマ＃１での課題（〆切：原則として５月１３日（金））

1. この例のように，近似の見方を変えると不可能である（と思われている）問題も可能になる場合がある．そのような例を調べて紹介して欲しい．

採点のレベル（Ｃレベル以上が合格）

Ｃレベル： 問題を明確に示し，近似不可能な（そう思われている）近似基準と，近似可能な近似基準を明確に示し，後者の基準での近似アルゴリズム（の概要）を紹介する．

Ｂレベル： 自明でない不可能性の証明，もしくは自明ではないアルゴリズムの詳細な解析を紹介する．

Ａレベル： 高度な不可能性の証明，もしくは高度なアルゴリズムの詳細な解析を紹介する．あるいは，自分で新たな問題とアルゴリズムを開拓する！

2. ストリームアルゴリズムのその他のおもしろい例を調べ紹介する．

採点のレベル（Ｃレベル以上が合格）

Ｃレベル： 「オンラインアルゴリズムとストリームアルゴリズム（徳山豪著，近代科学社）」で紹介されているものを，自分なりの事例を使って よりわかりやすく 説明する．

Ｂレベル： 最近の文献を調べて，おもしろいアルゴリズムを紹介する．（おもしろさや紹介のレベルに応じて，Ａレベルにもなる．）