テーマ１：ストリームアルゴリズム

本日の講義の目的

- ストリームアルゴリズムとは
- アルゴリズムの書き方，計算量の見積り方
- 欠損要素検出問題に対するストリームアルゴリズム
- 最頻出データ候補列挙問題

1.1. 欠損要素検出問題（ストリーミング計算の導入例として）

次のような問題を考える.（**注意！** この問題では，データの個数 n' が決まっていて，**しかも**データのドメイン D の大きさ n とデータの個数 n' が関連している．これは非常に偏った特異的な例である．一般にはデータのドメイン D の大きさ N はデータ数よりは十分に大きい．また，多くのストリーミング計算では，データには終わりはなく，新たなデータをもらうたびに何らかの答えを求められるような問題が多い.）

欠損要素検出問題

入力： 列 $\boldsymbol{a} = a_1, a_2, ..., a_{n'}$．ただし，

(1) 各要素 $a_i \in D = \{1, ..., N\}$ はすべて異なる，

(2) データの終わりがわかる（つまり $a_{n'}$ が最後のデータだとわかる），

(3) $N = n' + 1$（つまり，D 中の要素が 1 つだけ欠けている）.

注）この問題では N を サイズ を表すパラメータとする.

仕事： $\boldsymbol{a}$ 中の欠けている要素 $b \in D$ を求める.

定理 1.1. 欠損要素検出問題に対する領域計算量 $O(\log N)$ かつ時間計算量 $\mathrm{polylog}\, N$ のストリームアルゴリズムが存在する（正確には構成できる）.

補足１： 計算量

ある問題に対するアルゴリズムの効率を議論するために「計算量」が用いられる．計算量とは，問題例のサイズ（たとえば N）に対して，そのアルゴリズムのサイズ N の入力例に対する（通常は最悪の）計算コストを対応させる関数のことである．この「計算コスト」を厳密に議論するためには，何らかの計算モデル（とくに機械的は計算モデル）が必要である．ここでは，多少曖昧ではあるが，プログラミング言語（それも擬似的なもの）に基づいて議論する．たとえば上の定理の上界を示すには次ページのような擬似アルゴリズムを用いる.

補足２： 問題例のサイズ

計算量（←関数）の引数となるサイズのことを サイズパラメータ などという．ここでは N がサイズパラメータだが，ストリーム計算では一般に，処理の対象となる（これまでに読み込んだ）データ要素（この例では n'）もサイズパラメータとして使用する場合が多い．

補足３： O 記法

以下がコンピュータサイエンスでよく用いられる O 記法の定義．なお，関数 $f(x)$ に対して，f が $\mathrm{polylog}\, n$ である とは，$f(n) = O((\log n)^{O(1)})$ のことである．

定義 1.2. （O 表現：例として 2 変数関数で説明）
自然数上の関数 $f(x, y)$ と $g(x, y)$ に対し，f が $O(g(n, m))$ である（$f(n, m) = O(g(n, m))$ と書く）とは，次が成立することである．

$$\exists c > 0,\ \exists C_1, C_2,\ \forall n \geq C_1, m \geq C_2\ [\ f(n, m) \leq c \cdot g(n, m)\ ].$$

では定理の上界を証拠付けるアルゴリズムを示そう．

Algorithm `FindOneMiss`
  input: $\boldsymbol{a} = a_1, ..., a_{n'}$;
  var.s: t, a_t, N, $\widehat{S}$, b; （注：t, a_t については以下省略）
  init. comp:
    $\widehat{S} = 0$;（注：t の初期設定，インクリメントについては省略）
  comp. at t**:**
    $\widehat{S} = \widehat{S} + a_t$;
  final comp:
    $N = n' + 1$;
    $b = N(N + 1)/2 - \widehat{S}$; output b;
end.

補足４： 領域計算量の考え方

ストリームアルゴリズムでは領域計算量が中心的な課題となる．そのため，時間計算量は少々曖昧でもよいが，領域計算量については厳密に議論できた方がよい．我々が「ストリームアルゴリズム」と言ったときには，上記のようなアルゴリズムを考えることにする．その際，使用できる変数は，その問題で考えているサイズを表すパラメータ（たとえば n と N）に対して，とくに断りのない限り $O(\max\{\log n, \log N\})$ ビットのデータまでは格納できることにする．多くの場合，その変数を何個使用するかで領域計算コストを議論することにする．さらに正確に議論したい場合には，入力データを表すために必要な変数（この例では t, a_t, n）以外を 作業用変数 と呼び，作業用変数に格納されている総ビット数で議論する．

定理 1.3. 欠損要素検出問題に対する（作業用変数が定数個の）ストリームアルゴリズムの領域計算量は $\Omega(\log N)$ である．

定義 1.4. （Ω 表現：例として 1 変数関数で説明）
自然数上の関数 $f(x)$ と $g(x)$ に対し，<u>f が $\Omega(g(n))$ である</u>（$f(n) = \Omega(g(n))$ と書く）とは，次が成立することである．

$$\exists c > 0,\ \exists C,\ \forall n \geq C\ [\ f(n) \geq c \cdot g(n)\ ].$$

補足： どうしても $\log n$ 領域が必要となってしまう，という主張には，もっと弱い下界の表現でも構わない．たとえば，次のような条件でもよい．これを以下では「$f(n) = \Omega^{\infty}(g(n))$」と書くことにする．

$$\exists c > 0,\ \forall n,\ \exists n' > n,\ [\ f(n') \geq c \cdot g(n')\ ].$$

証明： 作業用が 1 つしか使わないストリームアルゴリズムを考え，その領域計算量が $\lfloor \log N \rfloor$ 以上であることを示す.（定数個の作業用変数の場合は，もっと厳密な計算モデルの話が必要となるが，ここでは省略する.）

また簡単のため $N = 2^h$ の場合を考える．それには n' 個のデータを読み込んだ後の計算を考えればよい．そのときの作業用変数の値がつねに $\log N - 1\ (= h - 1)$ ビット以下だとして矛盾を導く．ビット数が高々 $h - 1$ の範囲で作業用変数の取りうる値は高々[1]

$$2^{h-1} + 2^{h-2} + \cdots + 2^1 + 2^0 \ = \ 2^{h-1+1} - 1 \ = \ N - 1$$

通りである．そうすると鳩ノ巣原理より，ある $i, j, 1 \leq i < j \leq N$ に対して，i が欠けている（同様に j が欠けている）ある入力列 $\boldsymbol{a}_i$（同様に $\boldsymbol{a}_j$）に対しては，$\boldsymbol{a}_i$ と $\boldsymbol{a}_j$ を読み込んだ後の状態はまったく同じになる．しがたがって，アルゴリズムはどちらかに対して誤った答えを出すことになる．矛盾． □

お楽しみ問題（テーマ１：ストリームアルゴリズム）

1.1. 今回は，ちょうど 1 つ，要素が欠けている欠損問題を考えたが，要素が 2 つ欠けている欠損の場合はどうだろうか？ちょうど 2 つの欠損を見つけるメモリ効率の良いストリームアルゴリズムは作れるだろうか？

1.2. 上の下界の証明は（議論を簡単にするために）変数の数を１つに限ったが，定数個であれば同様の議論はできる．一般化を考えてみよう．また，変数の個数が定数個ではなく，入力データ数の増加とともに増やしてもよい場合はどうだろうか？

---

[1]授業後，この見積もりは大きすぎるのでは？という質問がありました．確かに，$h - 1$ が変数の格納できるデータのビット数だとすると，その取りうる値は 2^{h-1} しかないので，上の $2^h - 1$ の見積もりは過大評価ではある．ただ，過大に評価してもダメだ，ということを示すには問題はない．実は，複数個の変数まで考える場合には，このような評価をしていた方がやりやすい．