



Longest biased interval and longest non-negative sum interval

Lloyd Allison

School of Computer Science and Software Engineering, Monash University, Clayton, Victoria, Australia 3800

Received on October 16, 2002; revised on November 17, 2002; accepted on January 22, 2003

ABSTRACT

Summary: Described is an algorithm to find the longest interval having at least a specified minimum bias in a sequence of characters (bases, amino acids), e.g. 'at least 0.95 (A+T)-rich'. It is based on an algorithm to find the longest interval having a non-negative sum in a sequence of positive and negative numbers. In practice, it runs in linear time; this can be guaranteed if the bias is rational.

Availability: Java code of the algorithm can be found at <http://www.csse.monash.edu.au/~lloyd/tildeProgLang/Java2/Biased/>

Contact: lloyd@bruce.cs.monash.edu.au

Supplementary information: Examples of applications to *Plasmodium falciparum* genomic DNA can be found at the above URL.

INTRODUCTION

Regions of low information content in biological sequences can be of considerable interest, for example in connection with the centromeres of *Plasmodium falciparum* 'Eleven of the 14 chromosomes contained a single region of 2–3 kb with extremely high (A+T) content (0.97)' Gardner *et al.* (2002).

Here, an algorithm is described to find the *longest biased interval* (LBI), having a specified minimum bias, in a sequence of characters. It relies on an algorithm which, given a sequence of numbers, finds the *longest non-negative sum interval*.

The problem considered here is related to the *maximum sum interval* (MSI) problem, that is given a sequence of numbers, $x[0 \dots n - 1]$, find an interval, $x[i \dots j]$, such that the sum of the numbers in the interval is as large as possible. Bentley (1986) attributes the 2D version of that problem to U.Grenander and the solution to the 1D problem to J.Kadane. The MSI problem is solved by considering *cumulative sums* in the sequence from $x[0]$ to $x[k]$ (solid line, Fig. 1). An MSI corresponds to a maximum increase in sums for $x[0 \dots i - 1]$ to $x[0 \dots j]$ where $i \leq j + 1$. Below, a related idea is used to solve the new problem.

NON-NEGATIVE SUM INTERVALS

Given a sequence of numbers, $x[0 \dots n - 1]$, the *longest non-negative sum interval* (LNNSI) problem is to find an interval, $x[i \dots j]$, that (a) sums to a non-negative total and (b) is as long as possible. It too can be solved by considering cumulative sums from $x[0]$ to $x[k]$. But in this case, any *candidate interval* corresponds to a non-decrease in sums for $x[0 \dots i - 1]$ to $x[0 \dots j]$ where $i \leq j + 1$.

The cumulative sum is calculated in a left to right scan of the sequence. Auxiliary arrays are used to store: (a) first occurrences of new minima in the successive cumulative sums and (b) the positions of those first occurrences (dotted line, Fig. 1); the dashed line in the figure indicates the last point at which the sum is at or above each value. At each scan position, j , the algorithm considers the best *candidate interval*, $x[i \dots j]$, over i in $0 \dots j + 1$. The cumulative sum for $x[0 \dots i - 1]$ must be no more than that for $x[0 \dots j]$, in fact i must be the smallest such index; imagine j advancing along the solid line and i tracking back and forth along the dotted line in the figure while staying at or just below j 's height. The best out of all such candidate intervals, $x[i \dots j]$, is remembered during the scan to yield the solution.

BIASED INTERVALS

Given a sequence of characters, $s[0 \dots n - 1]$, a selection of preferred characters, and a minimum proportion (bias), f , of preferred characters, the *longest biased interval* (LBI) problem is to find an interval, $s[i \dots j]$, such that: (a) the interval contains at least the specified proportion of preferred characters; and (b) the interval is as long as possible.

Preferred characters are 'good' and are given a positive score, $1 - f$. Other characters of 'bad' and are given a negative score, $-f$. Only candidate intervals have non-negative total scores: For an interval of g good and b bad characters, $g/(g + b) \geq f$ is equivalent to $g \geq (g + b) \cdot f$ and to $g \cdot (1 - f) - b \cdot f \geq 0$, so LBI is equivalent to an LNNSI problem.

As for LNNSI, the sequence (of characters) is scanned from left to right, computing cumulative sums of *scores*,

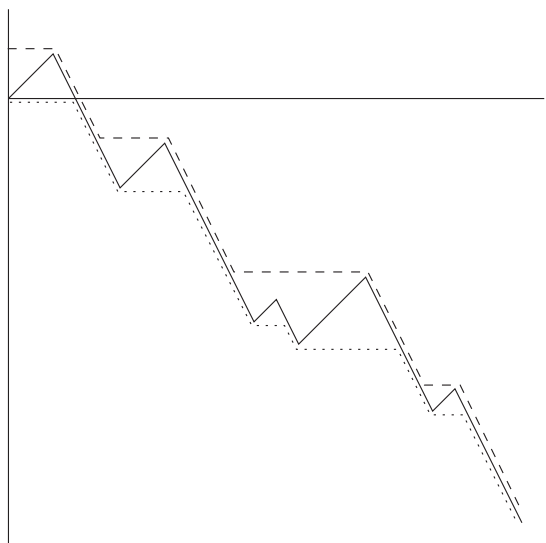


Fig. 1. Cumulative sums: solid line: cumulative sum; dotted line: first below (new minima); dashed line: last above.

and candidate intervals, $s[i \dots j]$. The algorithm runs in linear-time in practice because when j is increased to $j + 1$, i almost always changes by only a small amount under reasonable conditions: If the sum to $j + 1$ increases, then i may ‘retreat’ a few steps to an earlier and higher minimum (dotted line, Fig. 1). If the sum to $j + 1$ decreases, then i may ‘advance’ by a few steps to a later and lower minimum. For pathological sequences and ‘suitable’ choices of f , the algorithm could in principle run more slowly. In practice, a scan of chromosome-14 (3 mb) of *P.falciparum* takes less than 1 s: Java code, just-in-time compiler and a 900 MHz processor. It is thus practical to scan even very long sequences for many different values of bias, f .

There is an alternative coding of the algorithm which is guaranteed to run in linear-time *if* the minimum bias, f , is a rational number: Integer scores, $+p$ and $-q$ say, can then be used and the locations of new minima stored in an array indexed by value. The first occurrence of a given cumulative-sum value can then be found using random access, and the start position of the candidate interval adjusted in $O(1)$ -time. Worst-case space-complexity is linear but the multiplier is the larger of p and q .

Ruzzo and Tompa (1999) gave a linear-time algorithm to find *all* maximum sum intervals (MSIs) of a sequence, i.e. every interval that cannot be extended without decreasing its sum. For our problem, it is possible to have two overlapping biased sequences, yet neither can be extended without falling below the minimum bias: e.g. Consider (A)-rich, bias $f = 0.6$ and sequence ‘ATAT...ATAT’; any instance of ‘ATATA’ has three As out of five characters and satisfies the condition but no extension does. The definition of a *local*, rather than a global, bias condition is therefore problematic. However a divide and conquer approach would usually find regions of interest in practice: Find a global LBI, and repeat recursively on the sections of the sequence to its left and its right — $O(n \cdot \log n)$ -time on average if the splits are often reasonably even, $O(n^2)$ -time at worst if the splits are almost always unbalanced.

REFERENCES

- Bentley, J. (1986) *Programming Pearls*. Addison-Wesley, Reading, MA.
- Gardner, M.J., Hall, N., Fung, E., White, O., Berriman, M., Hyman, R.W., Carlton, J.M., Pain, A., Nelson, K.E., Bowman, S. *et al.* (2002) Genome sequence of the human malaria parasite *Plasmodium falciparum*. *Nature*, **419**, 498–511.
- Ruzzo, W.L. and Tompa, M. (1999) A linear time algorithm for finding all maximal scoring sequences. *7th Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*. Heidelberg, Germany, pp. 234–241.