Directions to use Probabilistic Algorithms for Cardinality for DNA Analysis

Frédéric Giroire

INRIA Rocquencourt,
F-78153 Le Chesnay, France
frederic.giroire@inria.fr

Main Thematics: *Sequence analysis, motifs.*

Technical Fields: *Algorithmics, DNA, data mining.*

Keywords: Probabilistic algorithms, cardinality, DNA correlation, coding regions.

Probabilistic algorithms for cardinality (see for example [1]) allow to estimate the number of *distinct* words of *very large multisets*. Best of them are *very fast* (only few tens of CPU operations per element) and use *constant memory* (standard error of $\frac{c}{\sqrt{M}}$ attained using M units of memory) to be compared with the linear memory used by exact algorithms. Hence they allow to do multiple experiments in few minutes with few KiloBytes on files of several GigaBytes that would be unfeasible with exact counting algorithms. Typically they are used for applications in the area of databases (see [2]) or networking (see [3] or [4]).

Such algorithms are used here to analyze *base correlation* in *human genome*. The correlation is measured by the number of distinct subwords of fixed size k (10 bases for example) in a DNA piece of size N. The idea is that a sequence with few distinct subwords is more corrolated than a sequence of same size with more distinct subwords. Three different angles of study are introduced:

- Are all possible words (4^k subwords of size k) present in the genome or, on the contrary, are a lot of patterns forbidden?
- Is the genome homogeneus or are some areas more corrolated than others? In the late case, is it possible to recognize or have location hints, in a fast and easy way, for regions of different natures such as repetitions, coding or not coding regions?
- What is the arrival rate of distinct subwords in the genome when considered as a sequence read from the 'beginning'? How does it compare to the one of random texts generated by a Bernoulli or Markov source for example?

First results. We realized simulations on a human genome file. We use a probabilistic algorithm for cardinality, MINCOUNT, introduced in [5], and its version for sliding window, SLIDING MINCOUNT (see [6]). It estimates the number of distinct elements of texts with several billion elements with a precision of 2% using a memory of only 12KB. Thanks to its very simple internal loop it is of the order of only 5 times slower than the unix command `wc` that only count the number of words distinct or not. For example, it takes only 12 seconds (respectively 10 minutes) to estimate the number of distinct subwords of size k among the 62 millions (resp. 3 billions) subwords of chromosome 20 (resp. of the human genome). We have first results for our three angles of study:

- *Forbidden patterns?* We study the numbers of distinct subwords of size 1 to 30 in chromosome 20 (62 millions of base pairs) and compare them to 4^k. The first part follows 4^k and all subwords of size from 1 to 11 are present. For size from 12 to 17, not all patterns are present but the growth remains exponential. In the last part, corresponding to lengths from 18 to 30, the growth is very slow (almost constant), from 54 to 59 millions, to compare with the 62 M subwords in chromosome 20. The conclusion is that no patterns seem to be forbidden, nevertheless they don't appear all at the same frequency.

- *More or less corrolated regions.* We made a multi-scale study of the number of distinct words of size 13 ($4^{13} \approx 70$ millions) in the whole 3 billion pairs of bases of human genome. Three scales are introduced corresponding to three cuts of the genome: it was divided in 10, 100 or 1000 pieces of 300, 30 or 3 million bases. We estimate the number of distinct subwords for each of these pieces. To each scale, regions with different correlations clearly appear.
- *Arrival rate of patterns.* We study the arrivals patterns of size 6 to 12 in chromosome 20. The algorithm gives the numbers of distinct subwords seen after x bases, x from 1 to the entire chromosome. These numbers are compared to the theoric expectation for a random text of same size created by a Bernoulli sourced. It can be approximated (see [7]) by $\mathbb{E}[X^{(n)}] \approx \sigma^q \left(1 - \frac{1}{\sigma^q}\right)^{n-q+1} \approx \sigma^q \exp(-\lambda)$. Simulations show that pattern arrival rate is smaller in DNA and so that DNA correlation is stronger and give a measure of the difference.

There is ongoing work for each approach, such as compare zones with specific correlation to kown places of coding regions, and compare the number of distinct subwords with other sources as Markovian ones.

References

[1] P. Flajolet and P. N. Martin, Probabilistic Counting, *Proceedings of the 24th Annual Symposium on Foundations of Computer Science*, IEEE Computer Society Press, 76–82.

[2] J. Considine, F. Li, G. Kollios and J. Byers, Approximate Aggregation Techniques for Sensor Databases, *ICDE '04: Proceedings of the 20th International Conference on Data Engineering*, 2004.

[3] C. Estan, G. Varghese and M. Fisk, Bitmap algorithms for counting active flows on high speed links, *Technical Report CS2003-0738*, 2003.

[4] G. Iannaccone, C. Diot, I. Graham and N. McKeown, Monitoring very high speed links, *ACM SIGCOMM Internet Measurement Workshop*, 2001.

[5] F. Giroire, Order Statistics and Estimating Cardinalities of Massive Datasets, *Proceedings of Discrete Mathematics and Theoretical Computer Science*, 2005.

[6] E. Fusy and F. Giroire, Estimating the Number of Active Flows in a Data Stream over a Sliding Window, *To appear*, 2006.

[7] S. Rahman and E. Rivals Exact and Efficient Computation of the Expected Number of Missing and Common Worlds in Random Texts, *Proceedings of the 11th Symposium on Combinatorial Pattern Matching*, 2000.