Politechnika Śląska
Wydział Automatyki, Elektroniki i Informatyki

Rozprawa doktorska

Wyznaczanie zbiorów podsłów sekwencji nukleotydowych w danych z sekwencjonowania genomów

mgr inż. Marek Kokot

Promotor: prof. dr hab. inż. Sebastian Deorowicz

Katowice, 2020

Determination of sets of substrings of nucleotide sequences in genome sequencing data

mgr inż. Marek Kokot
Politechnika Śląska
Wydział Automatyki, Elektorniki i Informatyki
Katedra Algorytmiki i Oprogramowania

Supervisor:
prof. dr hab. inż. Sebastian Deorowicz

The dissertation is mainly focused on k-mer counting, which is the first step in many analyses of the genome sequencing data such as *de novo* assembly, repeat detection, fast multiple sequence alignment or metagenomic data classification. The purpose of k-mer counting is to find all k-length substrings in a given sequence (sequences). Additionally, for each such substring also the number of occurrences in the sequence (sequences) should be computed. Thus, the result of k-mer counting is a set of pairs, in which the first element is a k-mer and the second one is the number of its occurrences.

Although the task is simple by the definition it is far from trivial, especially considering data characteristics. First of all, there are errors in the data. A single substitution error may introduce up to k erroneous k-mers. In most cases, such k-mers will occur only once in the input sequence. In the simplest approach for k-mer counting, i.e. using a hash table, such errors will increase memory requirements. In the biggest dataset used in the experiments, containing 736.4 G symbols, the total number of distinct 28-mers is 19.7 G, while the number of distinct 28-mers that occur at least twice is 3.65 G. Thus, k-mer counting algorithm should be memory frugal. Taking into account the size of input data, it is also desirable that the time of computation is as short as possible. The premise that the k-mer counting is not trivial may also be the number of approaches proposed in the literature.

As a starting point of the research described in the dissertation, KMC 1 algorithm was chosen. It is a disk-based k-mer counting algorithm that works in two stages. In the first stage, to each k-mer the bin number is assigned using some non-injective function. Each bin is associated with a file on a hard disk. After all k-mers are stored in appropriate bins, each bin can be further processed independently in a second stage. Sorting is used to count k-mers in a single bin. Sorted identical k-mers are at adjacent positions and can be counted with a single linear scan. An important factor, that affects running time, is the speed of the chosen sorting algorithm. Although sorting is well known, widely examined and many solutions have been proposed so far in the field of algorithmics, the new ones still arise. The reason is that sorting is an intermediate step in many algorithms. Modern hardware architectures offer new ways to improve existing algorithms. Furthermore, workstations or even laptops and mobile devices are equipped with at least several cores. As a consequence, best general-purpose sorting algorithms should effectively use multithreading. Additionally, low-level optimizations may have a high impact on the running time. Examples of low-level optimizations are the realization of conditional branches and effective usage of CPU cache.

In the case of k-mer counting, all records to be sorted are of the same length, so radix sort may be used. As this sorting method is not based on comparisons, it is possible to achieve time complexity

$\mathcal{O}(p)$, where p is the number of records to be sorted. As a part of work two radix-sort-based algorithms were developed: RADULS 1 and RADULS 2. Both are described in detail in the dissertation. The first one was successfully applied in the proposed k-mer counting algorithm.

The function used to assign k-mer to bin in KMC 1 has the feature that two adjacent k-mers form the input sequence with high probability will be assigned to different bins. It may be considered as a disadvantage, as each k-mer must be stored independently. A function for which the probability of assigning the same bin for adjacent k-mers is higher is desirable. In such a case, it is possible to store several k-mers as a single, longer sequence, called super-k-mer. A function based on *minimizers* is an example of a function fulfilling the mentioned criteria. Minimizer is a lexicographically smallest m-mer of a k-mer ($m < k$). The probability that two adjacent k-mers share the same minimizer is high enough to store k-mers in a more compacted form. Minimizers have also some disadvantages. The first of them is the fact that the biggest bin has a relatively large size in comparison with the remaining bins. The second disadvantage is that the number of super-k-mers is relatively large. In a general case, the minimization of both criteria is not easy. In KMC 2 algorithm, that forms a significant part of the dissertation, signatures being the generalization of minimizers were proposed. In the case of signatures, some of m-mers are treated as disallowed and, if present in k-mer, skipped. As it turns out signatures allow better balancing of bin sizes and lowering the total number of super-k-mers. There is another advantage, beyond saving disk space and reducing computation time, of super-k-mers. In the sorting stage, instead of k-mers a little longer records can be used. Such records are denoted as (k, x)-mers. More precisely, (k, x)-mer is a $(k + x')$-mer for $x' \in \{0, 1, ..., x\}$. As a consequence, the total number of records to be sorted is lowered, which reduces memory requirements and time consumption. KMC 2 was further developed, among others, by applying a special procedure for a case of the compressed input and by using RADULS 1 algorithm at the sorting stage. Those and other improvements form the new version of the algorithm — KMC 3, which is also described in detail in the dissertation.

There are applications in which k-mer sets are directly combined. One example of such an application is DIAMUND algorithm. It is designed to detect disease-causing mutations by a direct comparison of genome sequencing data of family trios or tissues. In case of a family trio k-mers are counted for each individual. In the next step from a child (having a genetic disease) all k-mers that are present in any of (healthy) parents are removed. Such applications demonstrate the demand for a tool that is able to perform a variety of operations of k-mers sets. Such a tool, called KMC tools, was proposed and is also described in the dissertation.

The dissertation starts with a short introduction, followed by biological and algorithmic basics (Chapters 1–3). Further, the existing solutions are presented in Chapter 4. Chapter 5 contains a detailed description of the proposed algorithms. In this chapter also time and space complexity analysis of proposed algorithms is presented. The experimental results are shown in Chapter 6. Chapter 7 concludes the dissertation.