The Improvement of the Parallel Algorithm for Randomization-based Enrichment Analysis

[1]Grishchenko M.V., [1, 2]Yakimenko A.A., [1, 2]Khairetdinov M.S., [2]Lazareva A.V.
[1]Institute Computational Mathematics and Mathematical Geophysics SB RAS,
[2]Novosibirsk State Technical University,
Novosibirsk, Russia
mikhail.grishch@gmail.com

***Abstract*— This work is motivated by algorithm that is used to analyze the multidimensional genetic data. This algorithm, called permutation test, is widely used as part of gene set enrichment analysis method. In this paper, the permutation test algorithm is considered. The dependence of the resampling test algorithm performance on the input data is studied. Several ways to improve the permutation test algorithm are proposed. The most effective way consisting in the reduction of number of iterations of the algorithm is implemented.**

Keywords — resampling, randomization, permutation test, GSEA.

I. Introduction

Processing of genetic data for the analysis genetic determination of traits is very important problem for modern biology. The problems of formalization of inheritance models and testing multiple genetic hypotheses on specific empirical material occupy an important place. The resampling methods based on randomization approach are used for solving these problems [1,2]. By using these methods, a set of genetic characteristics reflecting the simultaneous contribution of many factors in the formation of a single biological trait are tested. Resampling methods allow analyzing multiple hypotheses simultaneously without correction of the statistical significance level. Furthermore, they are more effective than other statistical methods such as t-test, ANOVA etc., since they do not need any information about the data distribution law in the general population, but investigate sample data in various combinations, as if considering them from different angles. However, these methods require much computational resources.

We developed a parallel randomization-based enrichment analysis [3] system for searching statistically significant overrepresented traits of genes under various external or internal conditions. This system based on a permutation test algorithm [4]. We used graphic processor units (GPUs) to increase the performance of the permutation test. However, the range of requirements to this software increases. It is necessary to find ways to increase the performance of this system. One of such ways is the improvement of the permutation test algorithm by the iterations number minimization. This improvement is the aim of this paper.

This work was supported by the Russian foundation for basic research (Grant No. 16-37-00240)1

II. Background

Permutation test algorithm allows us to calculate the p-value simultaneously for all characteristics of the gene sequence. The common idea of organization of the permutation test is as follows:

1) A list of genes associated with list of the properties of genes and measured values of genes are determined

2) The value of the statistic is calculated from the empirical material

$$G_0(x_1,x_2,x_3,\ldots,x_n) \quad (1)$$

where x – the measured quantitative characterization of the gene (expression level, evolution speed, etc.);

3) The random permutation of quantitative characteristics of genes between samples and measured values is performed;

4) For the same samples with randomly permuted variables $x_1,x_2,x_3,\ldots,x_n$ we calculate the value of the same statistic

$$G_u(x'_1,x'_2,x'_3,\ldots,x'_n) \quad (2)$$

which a reflective sample after permutation, and the subscript U is the number of permutations;

5) permutation procedures 3 and calculations of the statistic 4 are repeated U times;

6) the error probability is determined in rejecting the zero hypothesis (p-value) as a proportion of the values of G_u exceeding G_0.

The process of computing p-value is an iterative, in which the values of the computed statistics gradually converge to the stable value of the neighborhood of a certain value p*. This process is shown on the Fig. 1. We executed permutation test with 150000 iterations. As it can be seen from the graph, steady-state value p* is reached after about 70000 iterations and after that p-value is changed only in the neighborhood of the p*. Thus, the last 80000 iterations are redundant and calculations may be stopped after 70000 iterations were done. Based on this result, the following questions question arises: "How many iterations should be performed to obtain reliable results?"

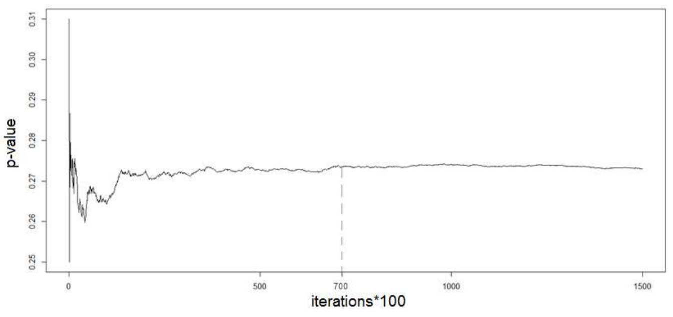

Fig. 1. The p-value convergence process.

Therefore, the main problem of this work is to determine minimal number of iterations of the permutation test algorithm depending on the input data.

III. Methods

As mentioned above, it is not beside the purpose to take a very large number of iterations, since the algorithm may work "in idle". Thus, we consider the mean value of iterations to be need to achieve the steady-state p-value. As shown on the Fig.2, the mean number of iterations place between 27500-28500 iterations and in most cases it does not depend on the input data.

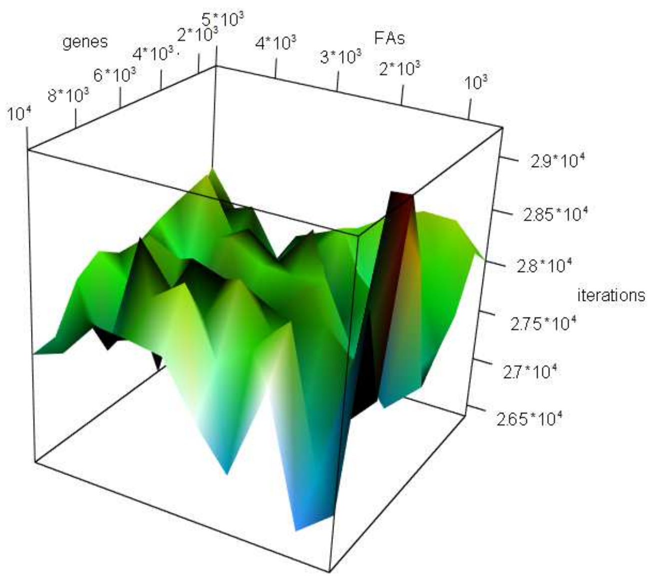

Fig. 2. The mean number of iterations, which is necessary to achieve p-value.

In the common case, we could set the number of iterations at 28000 iterations, but then not all p-values will reach their optimal values and. Besides, as can be seen from the graph, there are cases, for which 28000 iterations are not enough. These factors force us to look for other ways to optimize this algorithm.

IV. Results

This other way is to use the maximum number of iterations necessary to achieve p-value convergence. In other words, the algorithm stops when all the sought p-value reaches its steady-state values.

Fig. 3 shows the number of iterations of the permutation algorithm, under which all p-values reach these optimal values. By analyzing this graph, it can be concluded that the number of iterations depends on the number of measured genes characteristics (Functional Annotations (FA) Gene Ontology), but does not depend on the amount of genes.

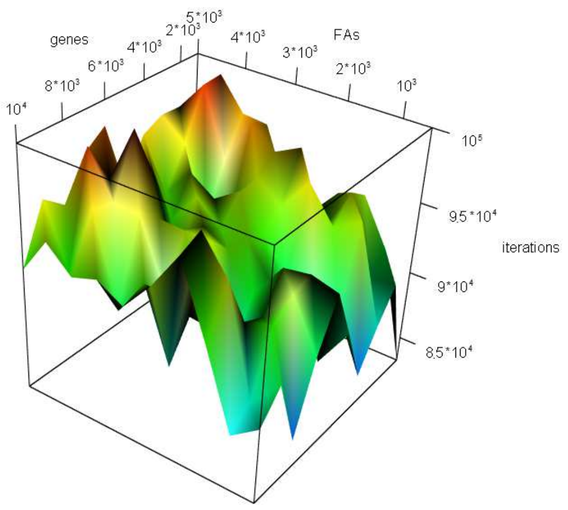

Fig. 3. The maximum number of iterations, which is necessary to achieve p-value.

It would be possible to calculate a function of the relationship between numbers of Functional annotations and genes. However, as can be seen on the graph, this function could not trivial and the calculation of the optimal number of iterations based on the input data is not possible. An alternative approach is to calculate intermediate p-values in runtime and stop algorithm execution when all p-values stop changing.

In addition, we investigated the effect of the number of random permutations per iteration on the convergence of p-value. Fig. 4 shows, that as the number of permutations increases, the p* is enriched faster.

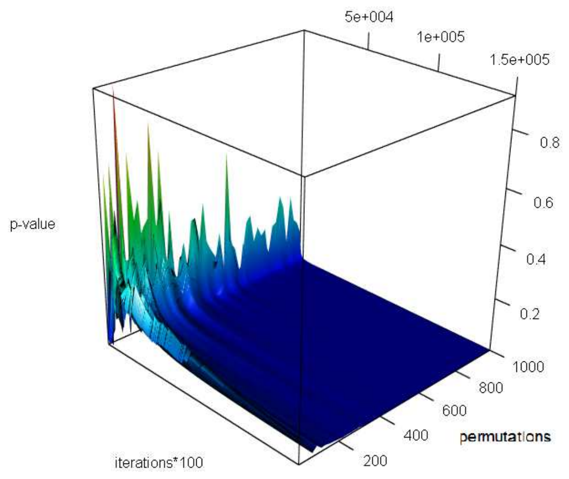

Fig. 4. The convergence of p-value depending on the number of permutations per iteration

It follows that in order to reduce the number of iterations, it is necessary to make the number of permutations equal to number of genes. We implement this feature with the help of Fisher–Yates shuffle algorithm [5].

V. Conclusion

The ways for reduction number of iterations of the permutation test algorithm for genetic data analysis on a hybrid supercomputer have been investigated. The measurements of the iteration numbers depending on number of genes and properties of genes show that the most effective way for reducing number of iterations is the calculation of the intermediate p-values during the program execution. In addition, we replace algorithm of random permutations to Fisher-Yates shuffle algorithm.

References

[1] B. Efron, “Nontraditional methods of statistical analysis,” Moscow: Finansy i statistika, 1988. 263 p.

[2] M. Ashburner, et al., “Gene ontology: tool for the unification of biology,” in The Gene Ontology Consortium. Nat Genet. 2000. 25(1). P. 25-29.

[3] Subramanian,A. et al. (2005) From the Cover: Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proc. Natl Acad. Sci. USA, 102, 15545–15550.

[4] A. A. Yakimenko, K. V. Gunbin, M. S. Khairetdinov, “Search for the Overrepresented Gene Characteristics: The Experience of Implementation of Permutation Tests Using GPU,” in Optoelectronics, Instrumentation and Data Processing. 2014. Vol. 50, No. 1. P. 123–129.

[5] Knuth, Donald E. (1969). Seminumerical algorithms. The Art of Computer Programming. 2. Reading, MA: Addison–Wesley. pp. 139–140. OCLC 85975465.