Journal of Computer Science and Cybernetics, V.33, N.2 (2017), 119–130
DOI 10.15625/1813-9663/33/2/10753

TAXONOMIC ASSIGNMENT FOR LARGE-SCALE METAGENOMIC DATA ON HIGH-PERFOMANCE SYSTEMS

LE VAN VINH[1], TRAN VAN HOAI[2], DUONG NGOC HIEU[2], BUI XUAN GIANG[2], TRAN VAN LANG[3,4]

[1]*Faculty of Information Technology, HCMC University of Technology and Education*
[2]*Faculty of Computer Science and Engineering, Bach Khoa University*
[3]*Institute of Applied Mechanics and Informatics, VAST*
[4]*Lac Hong University*
vinhlv@fit.hcmute.edu.vn

Abstract. Metagenomics is a powerful approach to study environment samples which do not require the isolation and cultivation of individual organisms. One of the essential tasks in a metagenomic project is to identify the origin of reads, referred to as taxonomic assignment. Due to the fact that each metagenomic project has to analyze large-scale datasets, the metatenomic assignment is computationally intensive. This study proposes a parallel algorithm for the taxonomic assignment problem, called SeMetaPL, which aims to deal with the computational challenge. The proposed algorithm is evaluated with both simulated and real datasets on a high performance computing system. Experimental results demonstrate that the algorithm is able to achieve good performance and utilize resources of the system efficiently. The software implementing the algorithm and all test datasets can be downloaded at http://it.hcmute.edu.vn/bioinfo/metapro/SeMetaPL.html.

Keywords. DNA sequences, homology search, metagenomics, parallel algorithm, taxonomic assignment

1. INTRODUCTION

Metagenomics is the study of the genomic content derived directly from complex microbial environment, instead of from culture in laboratories. The discipline offers opportunities to discover microbial communities, and thus brings benefits in many fields, e.g., biotechnology, agriculture, earth sciences [5]. Earlier metagenomic projects usually take many costs to get genomic information directly from microbial samples due to the limit of traditional sequencing technologies (e.g., Sanger sequencing). Fortunately, the next-generation sequencing (NGS) techniques, e.g., 454 pyrosequencing, Illumina Genome Analyzer, AB SOLiD [13], are able to process a large amount of biological data with small costs, and make metagenomic projects feasible. However, it also poses computational challenges for the analysis of metagenomic reads [9, 15].

The taxonomic assignment is an important task in a metagenomic project. The task aims to group reads into bins and determines phylogenetic relationships between the reads and known taxa. Taxonomic assignment algorithms can be roughly classified into composition-based methods and homology-based methods. Composition-based methods (e.g.,TACOA [3], AKE [8]) classify reads by extracting genomic signatures (e.g., oligonucleotide frequencies, GC-content) from themselves.

Although these methods are fast, they are difficult to analyze short reads [10]. Recent taxonomic assignment methods (MEGAN [7], CARMA3 [4], MetaCluster-TA [18]) are mainly based on the homology feature. Blast [1] is one of the commonly-used tools to extract homology information between sequences. Those algorithms are demonstrated to work well with both short and long reads. However, a remaining challenge of the methods is that they are computationally expensive [9].

In previous works, we proposed a semi-supervised taxonomic assignment method for metagenomic reads, so-called SeMeta [17]. It consists of two steps, and utilizes both composition and homology features. In the first step, the method applies a clustering step and chooses representatives of clusters. The second step performs homology search task by Blast algorithm to find the relation with known species in reference databases. SeMeta is able to reduce much computational time comparing with other homology-only based algorithms. However, It still requires much computational time. For instance, SeMeta spends 187.67 hours to analyze a dataset of 428674 reads belonging to 10 genomes [17] from the NCBI (National Center for Biotechnology Information) database. This raises the needs of using high-performance computing techniques to boost classification performance.

Some metagenomic applications based on high-performance computing techniques are proposed in literature. MrMC-MinH [12] is a map-reduce framework which aims to cluster metagenomic reads. Another taxonomic clustering method for 16S environment datasets, proposed by Yang *et al* [19] also achieves a cloud based implementation by using map-reduce framework. Parallel-META [14] is a high performance computational pipeline for analyzing metagenomic data. It is based on GPU and multi-core-CPU technology to parallelize a homology search process for speeding up computation. Besides, mpiBlast is a parallel algorithm of the Blast tool. It separates a database into different parts and is based on MPI (Message Passing Interface) technology to perform the homology search distributedly.

This work proposes a parallel taxonomic assignment algorithm for metagenomic sequences using MPI technique, called SeMetaPL. The proposed method is an improvement of SeMeta in which its taxonomic assignment step is parallelized to reduce computational time. The algorithm is evaluated on a cloud-based high performance computing system with both simulated and real datasets. Three aspects of virtualized resources of the system considered are memory size, number of CPUs, and number of virtual machines.

In the rest of the paper, Section 2 presents the details of proposed algorithm. Section 3 provides experimental results. Some conclusions are presented in the final section.

2. METHODS

2.1. Classification of metagenomic reads with SeMeta

SeMeta [17] is a semi-supervised taxonomic assignment for classification of metagenomic reads. It combinedly uses both composition and homology features of sequences in the classification process, and works well with short reads of sufficient mutual coverage. The algorithm consists of two major steps (figure 1) as follows.

- *Step 1: Clustering*
  This step separates reads into clusters of closely related organisms basing on composition features (l-mer frequency) and sequence overlapping information. The algorithm then selects

a representative, so-called *a core*, of each cluster. The size of a core is usually smaller than that of the corresponding cluster. Some reads of extremely low-abundance genomes are not clustered in the step, but still considered as a cluster.

- *Step 2: Taxonomic assignment*
  The step firstly performs the homology search between reads in cores of clusters and reference databases using Blast tool. The algorithm measures of the homology locally instead of attempting to align two sequences over entire sequence lengths. It firstly tries to detect the similarity location between sequences, and then inserts gap-free into them. Finally, a substitution matrix is used to compute the similarity degree between sequences.

  After the homology search task is performed, cores of clusters are then assigned into a taxon in phylogenetic tree. Each cluster is labeled with the taxon assigned to its core. In post processing task, clusters having the same label are merged into a larger cluster. Some reads not matching with reference database or assigned at the highest level of the phylogenetic tree are regarded as unassigned reads. Experimental results in [17] show that the step is a bottleneck of SeMeta because it requires much computation time.

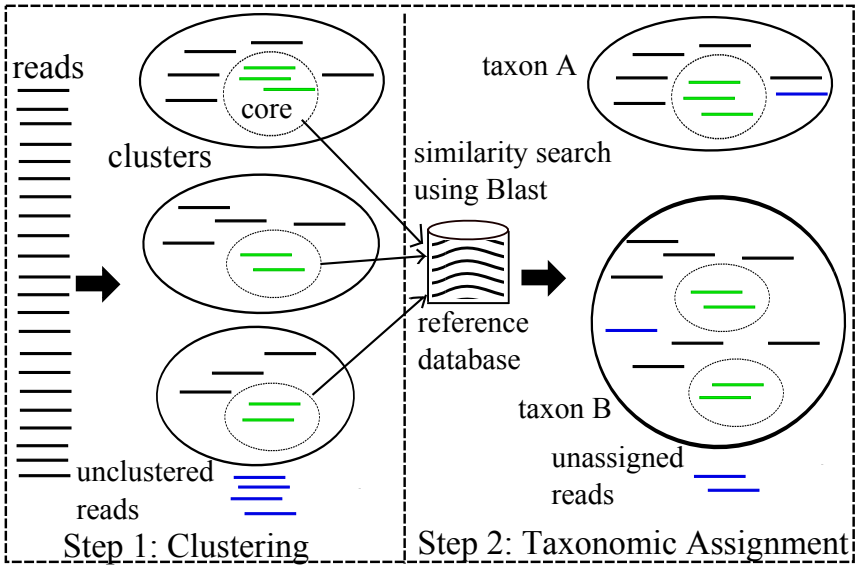

Figure 1. Process of SeMeta using Blast algorithm. Step 1 separates reads into clusters, and builds cluster cores. Step 2 does homology search between the cores and reference sequences, then labels each cluster [17].

2.2. Proposed algorithm

Due to the limit of SeMeta when processing large-scale datasets, this work proposes a parallelized algorithm, SeMetaPL, which is able to reduce much computational time and utilize resources of high-performance systems efficiently. The method consists of following steps (Figure 2).

- *Step 1: Clustering in single mode*
  This step is performed at server node in single mode as same as the clustering step of SeMeta. List of reads in cores of clusters are selected from input file and delivered to all computer nodes (or put them in shared storage).

- *Step 2: Taxonomic assignment in parallel mode*

  + *Homology searching with mpiBlast*
    The task uses mpiBlast algorithm [2] to determine the similarity degree between reads in cores of clusters and a reference database. It is a parallelization of Blast using MPI (Message Passing Interface) technique. The algorithm attempts to boost the homology search between sequences and a reference database by segmenting the database. mpiBlast allows each node in computing systems only to search on a portion of the database, and thus it helps reducing disk input/output significantly. Furthermore, the segmentation of databases does not generate heavy intercommunication between nodes.

    Let n be the number of computer nodes. The reference database is divided into at least n fragments and stored in shared disks. There are two scenarios of using the fragments. The first scenario is that the database fragments are always stored in a shared storage and computer nodes have to do remote access at runtime. In the second scenario, database fragments are distributed to local storage of each computer node, and accessed locally.

  + *Labeling cores of clusters*
    Let k be the number of clusters generated by step 1. $k/(n-1)$ clusters are labeled at each computer nodes. If $k < n-1$, only k nodes are used to perform labeling clusters. The remaining node is used to label unclustered reads. Algorithm 1 shows activities of master node. It computes ranges of clusters and sends to workers which have to process them. The master then determines labels for unclustered reads. Finally, it receives labeling results of clusters from worker nodes. Each worker receives a range of clusters from the master, and labels the clusters (Algorithm 2). It then send cluster labels to the master.

  + *Post processing*
    This task is done at master node to merge clusters having the same label, and determine unassigned reads.

2.3. Performance metrics

Two metrics *sensitivity* and *precision* are used to evaluate the proposed method. They can be defined as follows (as same as in [11, 17]). Let N be the number of reads, and C be the number of reads assigned by classification algorithms. Assuming that we consider at taxonomic level i, let X_i be the number of reads which are assigned to the correct taxa exactly at or under at the level. The two metrics can be calculated by the following formulations.

$$\textit{sensitivity} \text{ (at level } i) = \frac{X_i}{N},$$

Algorithm 1 Cluster labeling - master

Input: A list of clusters, a list of workers
Output: Labels of clusters

```
for Worker i do
    Compute range of clusters x_i to y_i for worker i
    for Cluster z, x_i ≤ z ≤ y_i do
        Send z to worker i
    end for
end for
Determine labels of unclustered reads
for Worker i do
    for Cluster z, x_i ≤ z ≤ y_i do
        Receive labels of z from worker i
    end for
end for
```

Algorithm 2 Cluster labeling - worker

```
Receive range of cluster x to y from master
for Cluster z, x ≤ z ≤ y do
    Determine label of cluster z
    Send label of z to master
end for
```

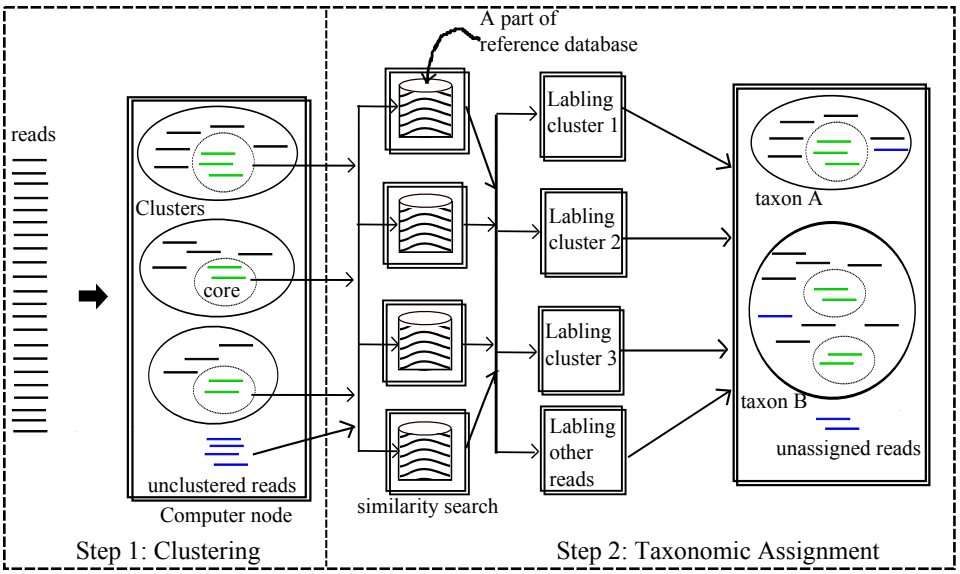

Figure 2. Process of SeMetaPL, using mpiBlast

$$precision \text{ (at level } i) = \frac{X_i}{C}.$$

For example, given a read originating from *Bordetella avium*, when we consider at genus level, a labeling of the read as *Bordetella, Bordetella bronchiseptica* or *Bordetella pertussis* would increase X_i. The metrics are computed at four taxonomic levels: species, genus, family, and order.

2.4. Datasets and reference databases

In order to generate datasets, we download real bacterial genomes from the NCBI (National Center for Biotechnology Information) database. Three simulated datasets are created by ART tool [6] following whole genome shotgun sequencing techniques. The datasets, presented in Table 1, contain single-end reads with the length of 150bp and follows the Illumina error profile. SeMetaPL also is used to classify the Acid Mine Drainage (AMD) dataset [16] - a real metagenome. It consists of 180,713 sequences, downloaded from NCBI trace archive.

Table 1. Simulated datasets

Dataset	Species/Strain	Coverage	No. of reads
ds1	*Borrelia burgdorferi JD1*	15	450
	Methylobacterium extorquens DM4	20	600
ds2	*Marinomonas mediterranea MMB1*	10	270
	Mycobacterium liflandii 128FXT	15	405
	Nitrosopumilus maritimus SCM1	15	405
ds3	*Bordetella avium 197N*	10	250
	Burkholderia xenovorans LB400	10	240
	Methanosarcina mazei Go1	15	375
	Neisseria meningitidis Z2491	15	375

Reference database used for analyzing the real metagenome is entire bacterial RefSeq database (release 69, downloaded from the NCBI database) with approximately 24 GB after formatted by mpiBlast. In case of simulated dataset, because we needs to conduct a lot of running scenarios, it is better to analyze with a smaller database. Thus, a part of the bacterial RefSeq database with approximately 5.3 GB (after formatted) is used. All of species in the tested datasets are contained in the database.

3. EXPERIMENTS RESULTS

3.1. Experimental setup

Experiments for simulated datasets are conducted on a virtualized system hosted on two physical machines. Each machine consists of 12 CPUs, 120G RAM, and 100GB disk storage. The performance of SeMeta is evaluated with different aspects of virtual resources (memory sizes, number of virtual machines, number of processors). The performance of SeMetaPL is compared with SeMeta in cases of using similar virtual resources. Besides, classification qualities of the two algorithms are also considered. SeMeta uses Blast tool (version 2.4) which is downloaded from the NCBI website to do homology search task. SeMetaPL performs the search task by using the latest version of mpiBlast (version 1.6.0). This version of mpiBlast uses Blast 2.2.20.

In case of real metagenome, a system with higher computing resources is used. It consists of 9 virtual machines with 200G RAM, and 5TB shared disk storage.

3.2. Results

3.2.1. Effects of the numbers of processors on running time

In order to measure the performance of SeMetaPL on multiple processor machines, this work generates 7 virtual machines with numbers of processors of 1, 3, 5, 7, 9, and 11, respectively. Other resources of the machines are similar. The number of processes running concurrently on each machines is set by 15. It is noted that, memory of each machine is enough for running all processes at the same time. Those machines are also used to run SeMeta algorithm for the same datasets (ds1, ds2, and ds3).

Line chart in figure 3 presents average running time of SeMetaPL and SeMeta for the three datasets. It can be seen that using multiple processors is able to boost the performance of SeMetaPL. For instances, the case of using five processors is approximately six times faster than the one of using one processor. When the number of processors used increases from 5 to 11, running time of SeMeta slightly decreases.

SeMeta runs at single mode, thus it does not utilize the advantages of multiple processor machines. The performance of SeMeta still keeps stable with the increase of the number of processors. When the number of processors is higher than 3, SeMetaPL achieves much better performance compared with SeMeta. In case of 1 or 3 processors, SeMetaPL requires similar or higher running time than SeMeta. It can be understood because SeMetaPL spends time for scheduling tasks and exchanging jobs between processes. Besides, because there are many unshared-memory processes running concurrently, SeMetaPL consumes larger amount of memory than SeMeta.

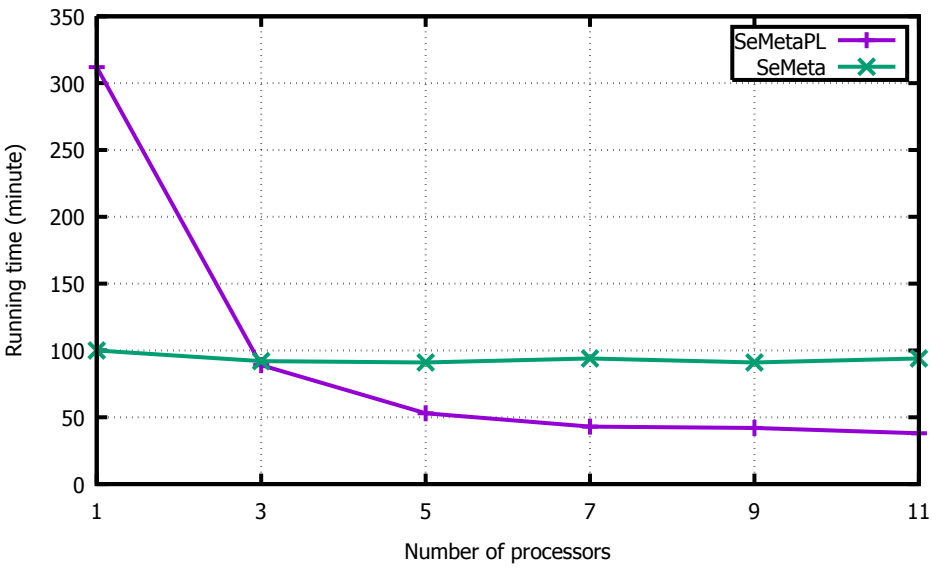

Figure 3. The performance of SeMetaPL and SeMeta with different numbers of processors

3.2.2. Effects of the number of virtual machines and memory sizes on running time

This experiment considers the strength of SeMetaPL when running on a cluster of machines. Twenty virtual machines are used in the experiment. Each machine consists of one processor. Two cases of memory sizes are considered. The first case tests on 10 machines having memory size of 3GB, while the second one tests on 10 machines with 6GB memory size.

Figure 4 shows results of the experiment. Line chart in the figure presents that the performance of SeMetaPL is proportional to the number of virtual machines. The increase of the number of machines from 2 to 5 helps reducing running time significantly. When the number of machines increases from 5 to 10, running time of SeMetaPL decreases moderately. It can be explained that disk input and output costs required rise when the number of machines increases, and thus it reduces the performance of the application.

The results also demonstrate that there is an effect of memory size on the performance of SeMetaPL. In the first case, machines have less memory size, and thus spend more running time than those of the second case for all tests.

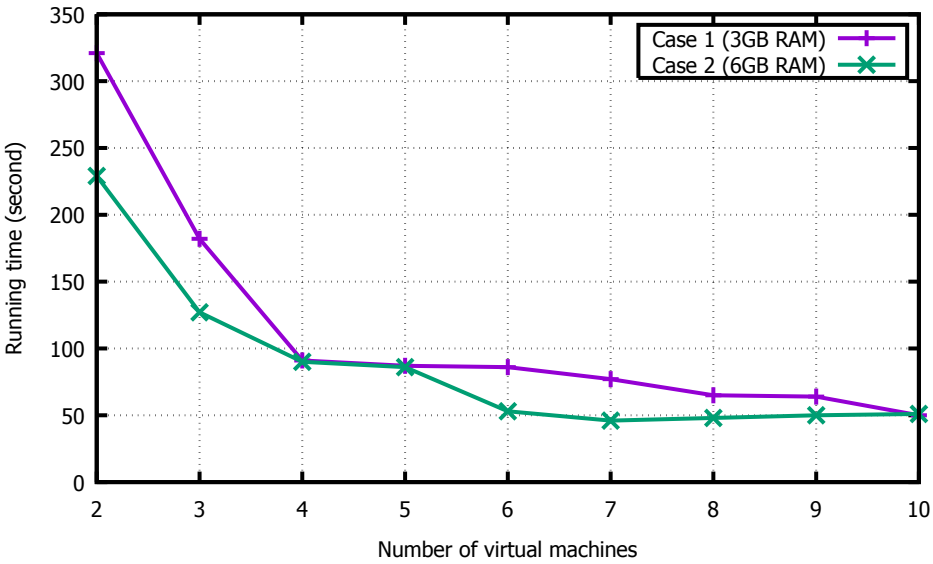

Figure 4. The performance of SeMetaPL with different numbers of virtual machines, with cases of using 3GB RAM and 6GB RAM.

3.2.3. Classification quality

The classification qualities of SeMetaPL and SeMeta are also computed for three dataset ds1, ds2, and ds3. Table 2 presents the precision and sensitivity of the two methods. It can be seen from the table that, SeMetaPL and SeMeta return the same results for most of the test cases. The results can be understood because the classification technique in SeMetaPL is as same as the one in SeMeta. There are some different results at species and genus levels. The difference is due to that the mpiBlast algorithm used in SeMetaPL is derived from a Blast algorithm having different version with the

one in SeMeta. Because the blast tool used in SeMeta has better quality in determining similarity degrees between sequences comparing with the one in SeMetaPL (from BLAST+ Release Notes, NCBI website), the proposed algorithm returns lower sensitivity and precision values compared with SeMeta at species level.

In addition, both methods identify labels for clusters instead of individual reads. Thus, if one of them fails to predict a label of a cluster at a specific level, their precision and sensitivity values will much lower than those of the remaining one. For instance, SeMetaPL gets 56.95% sensitivity and 57.12% precision higher than SeMeta for dataset ds1 at genus level. Conversely, SeMeta achieves 23.06% sensitivity and 50.64% precision higher than those of SeMetaPL at species level for dataset ds3. At higher levels (family and order level), two algorithms achieve the same both sensitivity and precision values for all cases.

3.2.4. Results on AMD dataset

A previous study in [16] recovered that the AMD dataset contains several dominant species. Among the species, *Leptospirillum sp. Group II*, *Leptospirillum sp. Group III* belong to bacteria, and three other species belong to archaea.

It takes approximately 606 hours to analyze the dataset. There are approximately 67.32% of the AMD sequences assigned by SeMetaPL. Results of the experiment, presented in table 3, support the previous studies. Our algorithm has detected genus *Leptospirillum* that account for 52.48% of assigned sequences, and other bacterial organisms (47.52%). Although the reference database does not contain two species *Leptospirillum sp. Group III* and *Leptospirillum sp. Group II*, SeMeta identified their genus due to the presence of other species belonging to the taxon in the database. Besides, because the experiment uses bacterial RefSeq database, SeMetaPL could not detect the existence of archaea organisms.

4. CONCLUSIONS

Most of current taxonomic assignment algorithms developed to be used on a single computer are difficult to adapt to the increasing of metagenomic data. In this work, we present a parallel taxonomic assignment algorithm to boost the speed of processing large-scale metagenomic sequences. The main idea of SeMetaPL is to reduce costs of the homology search and labeling task. Comparing with another single-mode algorithm, SeMetaPL could reduce much computational time, while still obtaining similar accuracy results. Besides, our algorithm has proved to work well with a large-scale metagenome, and promises to be a useful tool for real metagenomic projects.

The proposed algorithm could be improved in several ways. Firstly, the implementation of SeMetaPL is based on mpiBlast - an available parallel algorithm. It currently does not get the regular updating of Blast tool. Thus, applying other tools or developing a parallelized homology search tool should be considered. Secondly, SeMetaPL still does not take advantages of multi-core technology which is supported by most of high-performance systems. It motivates us to improve the performance of SeMetaPL in future research direction. Finally, the enhancement of SeMetaPL to improve classification quality by utilizing resources of high-performance systems will also be our concern.

Table 2. The classification quality of SeMetaPL and SeMeta on the datasets at different taxonomic levels

Method		**Species level**	**Genus level**	**Family level**	**Order level**
Dataset ds1					
SeMeta	*Sen.*	**42.76%**	42.76%	99.71%	99.71%
	Pre.	**42.88%**	42.88%	100%	100%
SeMetaPL	*Sen.*	N/A	**99.71%**	99.71%	99.71%
	Pre.	N/A	**100%**	100%	100%
Dataset ds2					
SeMeta	*Sen.*	24.72%	30.24%	61.94%	61.94%
	Pre.	39.91%	30.34%	100%	100%
SeMetaPL	*Sen.*	24.72%	30.24%	61.94%	61.94%
	Pre.	39.91%	30.34%	100%	100%
Dataset ds3					
SeMeta	*Sen.*	**46.69%**	64.84%	64.84%	64.84%
	Pre.	**67.09%**	93.16%	93.16%	93.16%
SeMetaPL	*Sen.*	23.64%	64.84%	64.84%	64.84%
	Pre.	16.45%	93.16%	93.16%	93.16%

N/A= Not Available. The bold values indicate the best results among the algorithms in the aspect of *sensitivity* (*Sen.*) or *precision* (*Pre.*).

Table 3. Results of SeMetaPL on the AMD dataset using bacterial ReqSeq database

Detected organisms	Number of sequences	Ratio
Leptospirillum	63846	52.48%
Other organisms	57815	47.52%

Acknowledgment

This research was funded by HCMC University of Technology and Education, under constract number T2017-26TD. The authors would like to thank Faculty of Computer Science and Engineering, Bach

Khoa University for providing facilities for this study. The applications presented in this paper were tested on the High Performance Computing Center (HPCC) of the faculty.

REFERENCES

[1] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of molecular biology*, vol. 215, no. 3, pp. 403–410, 1990.

[2] A. E. Darling, L. Carey, and W. C. Feng, "The design, implementation, and evaluation of mpiblast," Los Alamos National Laboratory, Tech. Rep., 2003.

[3] N. N. Diaz, L. Krause, A. Goesmann, K. Niehaus, and T. W. Nattkemper, "Tacoa–taxonomic classification of environmental genomic fragments using a kernelized nearest neighbor approach," *BMC bioinformatics*, vol. 10, no. 1, p. 56, 2009.

[4] W. Gerlach and J. Stoye, "Taxonomic classification of metagenomic shotgun sequences with carma3," *Nucleic acids research*, vol. 39, no. 14, pp. e91–e91, 2011.

[5] J. Handelsman, *The new science of metagenomics: Revealing the secrets of out microbial planet.* The National Academies Press, 2007.

[6] W. Huang, L. Li, J. R. Myers, and G. T. Marth, "Art: a next-generation sequencing read simulator," *Bioinformatics*, vol. 28, no. 4, pp. 593–594, 2011.

[7] D. H. Huson, S. Mitra, H. J. Ruscheweyh, N. Weber, and S. C. Schuster, "Integrative analysis of environmental sequences using megan4," *Genome research*, vol. 21, no. 9, pp. 1552–1560, 2011.

[8] D. Langenkämper, A. Goesmann, and T. W. Nattkemper, "Ake-the accelerated k-mer exploration web-tool for rapid taxonomic classification and visualization," *BMC bioinformatics*, vol. 15.

[9] S. S. Mande, M. H. Mohammed, and T. S. Ghosh, "Classification of metagenomic sequences: methods and challenges," *Briefings in bioinformatics*, vol. 13, no. 6, pp. 669–681, 2012.

[10] M. H. Mohammed, T. S. Ghosh, N. K. Singh, and S. S. Mande, "Sphinx - an algorithm for taxonomic binning of metagenomic sequences," *Bioinformatics*, vol. 27, no. 1, pp. 22 – 30, January 2011.

[11] R. Ounit, S. Wanamaker, T. J. Close, and S. Lonardi, "Clark: fast and accurate classification of metagenomic and genomic sequences using discriminative k-mers," *BMC genomics*, vol. 16, no. 1, p. 236, 2015.

[12] Z. Rasheed and H. Rangwala, "A map-reduce framework for clustering metagenomes," in *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2013 IEEE 27th International.* IEEE, 2013, pp. 549–558.

[13] J. Shendure and H. Ji, "Next-generation dna sequencing," *Nature biotechnology*, vol. 26, no. 10, pp. 1135–1145, 2008.

[14] X. Su, J. Xu, and K. Ning, "Parallel-meta: efficient metagenomic data analysis based on high-performance computation," *BMC Systems Biology*, vol. 6, no. 1, p. S16, 2012.

[15] H. Teeling and F. O. Glöckner, "Current opportunities and challenges in microbial metagenome analysisa bioinformatic perspective," *Briefings in bioinformatics*, vol. 13, no. 6, pp. 728–742, 2012.

[16] G. W. Tyson, J. Chapman, P. Hugenholtz, E. E. Allen, R. J. Ram, P. M. Richardson, V. V. Solovyev, E. M. Rubin, D. S. Rokhsar, and J. F. Banfield, "Community structure and metabolism through reconstruction of microbial genomes from the environment," *Nature*, vol. 428, no. 6978, pp. 37–43, 2004.

[17] V. Van Le, L. Van Tran, and H. Van Tran, "A novel semi-supervised algorithm for the taxonomic assignment of metagenomic reads," *BMC bioinformatics*, vol. 17, no. 22, 2016.

[18] Y. Wang, H. C. M. Leung, S. M. Yiu, and F. Y. L. Chin, "Metacluster-ta: taxonomic annotation for metagenomic databased on assembly-assisted binning," *BMC Genomics*, vol. 15, 2014.

[19] X. Yang, J. Zola, and S. Aluru, "Large-scale metagenomic sequence clustering on map-reduce clusters," *Journal of bioinformatics and computational biology*, vol. 11, no. 01, p. 1340001, 2013.

Received on September 24, 2017
Revised on December 07, 2017