Journal of Applied Physics and Engineering Vol.1, No.1 (2016)10-16

A Study of DNA Fragment Assembly Algorithms

Satyanarayana Reddy Beeram [#1] , Dr. Edara Srinivasa Reddy [*2]

[#]*Associate Professor, Dept. of CSE, Kallam Haranadhareddy Institute of Technology, Guntur, Andhra Pradesh, India.*

[*]*Professor & Dean of Faculty, Dept. of CSE, Acharya Nagarjuna University, Guntur, Andhra Pradesh, India.*

[1] snreddy.beeram@gmail.com

***Abstract*—DNA fragment assembly is a technique that attempts to reconstruct the original DNA sequence from a large number of fragments, each several hundred base-pairs long. The DNA fragment assembly is needed because current technology, such as gel electrophoresis, cannot directly and accurately sequence DNA molecules longer than 1000 bases. However, most genomes are much longer. Originally, the assembly of short fragments was done by hand, which is not only inefficient, but also error-prone. Hence, a lot of effort has been put into finding techniques to automate the shotgun sequence assembly (finding overlap fragments and then assembling back to original DNA) . The general outline of most assembly algorithms is first to create a set of candidate overlaps by examining all pairs, followed by forming an approximate layout of fragments, and finally creating a consensus sequence. All existing methods rely on heuristics, since the fragment assembly problem is NP-hard. The algorithms developed to solve this problem till now are based on genetic algorithms, greedy algorithms, pattern matching algorithms, particle swam optimization and nature inspired algorithms such as ant colony optimization, artificial bee optimization technique, cuckoo search algorithm. Among these algorithms, nature inspired algorithms seems to be the better algorithms for solving DNA** fragment assembly problem.

I. Introduction:

Structure of DNA: DNA (Deoxy Ribo Nucleic Acid) is a polymer. A polymer is a molecule made up of repeating subunits. The repeating subunits in DNA are called nucleotides. Each nucleotide is composed of 3 parts: a 5-carbon sugar, a phosphate group and a nitrogen base. Nucleotides join together to form two long chains, one on each side of the molecule, with the phosphate group and sugar molecules alternating to form the sides or backbone of the DNA molecule. The sugar molecule of each nucleotide bonds with the nitrogen base of the same nucleotide. And, the nitrogen base of one nucleotide bonds with a nitrogen base of another nucleotide on the opposite side of the molecule, which forms the rungs of the ladder. The result is a structure that looks like a twisted ladder as shown in Figure 2.

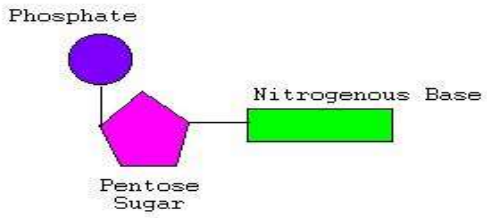

Fig1. A Nucleotide

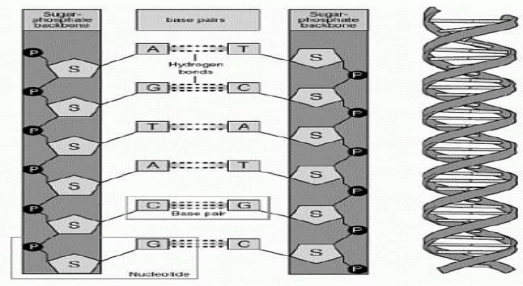

Fig2. A DNA Molecule

Nucleotide – each nucleotide has three parts(see Fig 1.): a 5-carbon sugar, a phosphate group and a nitrogen base. In DNA, there are four possible nucleotides because there are

- four different nitrogen bases.
- 5-carbon sugar (Pentose) – Deoxyribose (sugar's name)
- Phosphate group – is composed of one atom of phosphorous surrounded by four oxygen atoms.

Nitrogen base –

- Adenine(A)
- Thymine (T)
- Cytosine (C)
- Guanine (G)

Selected paper: International Conference On Computing (NECICC-2k15)

JAPE

DNA encodes hereditary information in a chemical language. All cells store their genetic information in the base sequence of DNA. The genotype is determined by the sequence of bases.DNA is the genetic material of living things.DNA is located within the nucleus of all cells apart from red blood cells.DNA is a long chemical sequence and this sequence contains the information needed for that living thing to develop, survive and pass its genetic information on to the next generation.The DNA chemical sequence differs between individuals. The pattern of this sequence which is made up of A,T,C,G is called the genotype.

A. DNA Sequencing:

DNA sequencing is the process of determining the precise order of nucleotides within a DNA molecule. It includes any method or technology that is used to determine the order of the four bases—adenine, guanine, cytosine, and thymine—in a strand of DNA. The advent of rapid DNA sequencing methods has greatly accelerated biological and medical research and discovery.

Knowledge of DNA sequences has become indispensable for basic biological research, and in numerous applied fields such as diagnostic, biotechnology, forensic biology, and biological systematics [1][3]. The rapid speed of sequencing attained with modern DNA sequencing technology has been instrumental in the sequencing of complete DNA sequences, orgenomes of numerous types and species of life, including the human genome and other complete DNA sequences of many animal, plant, and microbial species.

Uses of DNA Sequencing:

DNA sequencing may be used to determine the sequence of individual genes, larger genetic regions (i.e. clusters of genes or operons), full chromosomes or entire genomes. Sequencing provides the order of individual nucleotides in DNA or RNA (commonly represented as A, C, G, T, and U) isolated from cells of animals, plants, bacteria or virtually any other source of genetic information. This is useful for:

- Molecular biology - studying the genome itself, how proteins are made, what proteins are made, identifying new genes and associations with diseases and phenotypes, and identifying potential drug targets
- Evolutionary biology - studying how different organisms are related and how they evolved
- Metagenomics - Identifying species present in a body of water, sewage, dirt, debris filtred from the air, or swab samples of organisms. Helpful in ecology, epidemiology, microbiome research, and other fields.

Less-precise information is produced by non-sequencing techniques like DNA fingerprinting. This information may be easier to obtain and is useful for:

- Detect the presence of known genes for medical purposes (see genetic testing)
- Forensic identification
- Parental testing

B. DNA Fragment Assembly Problem:

Biologists have developed very clever laboratory methods for sequencing DNA. Unfortunately these methods only work on short DNA fragments, on the order of hundreds of units. A complete strand of DNA, however, is made of millions of bases. Biologists do know how to chop DNA strands up into short fragments, of the size that their sequencing machines can handle. (Strands which are too short or too long are not sequenced at all.)

There are some major problems with this. First, when scientists chop up a strand of DNA, they are not able to control exactly where the cuts are made; instead, the DNA is chopped into pieces at random locations in the string. Luckily, though, the fragments are usually small enough to be read by the sequencing machine. The second difficulty is that there is no way to keep track of how the resulting fragments are ordered in the target strand. So, it will end up with the sequences of hundreds of thousands of DNA fragments, but no way to piece them together.

Therefore, algorithms are needed to solve this fragment assembly problem.

II. DNA Fragment Assembly Algorithms:

The general outline of most assembly algorithms is first to create a set of candidate overlaps by examining all pairs, followed by forming an approximate layout of fragments, and finally creating a consensus sequence. All existing methods rely on heuristics, since the fragment assembly problem is NP-hard. More specifically, assembling DNA fragments is divided into three distinct phases [4]:

a) Overlap Phase - Finding the overlapping fragments. This phase consists in finding the best or longest match between the suffix of one sequence and the prefix of another. We

Selected paper: International Conference On Computing (NECICC-2k15)

compare all possible pairs of fragments to determine their similarity.

Usually, the dynamic programming algorithm is used in this step to find semiglobal alignments.

b) Layout Phase - Finding the order of fragments based on computed similarity scores. This is the most difficult step because it is hard to determine true overlaps. After the order is determined, the progressive alignment algorithm is applied to combine all the pairwise alignments obtained in the overlap phase. c) Consensus Phase - Deriving the DNA sequence from the layout. The most common technique used in this phase is to apply the majority rule in building the consensus. Other methods exist for finding the consensus, such as the use of probabilistic scores in PHRAP, a tool used for DNA assembly . The DNA fragment assembly problem is NP-hard, therefore, it is not possible to find an exact algorithm that solves this problem and runs in polynomial time (unless P = NP). The complexity of the problem increases even further due to the following factors.

a) Unknown orientation: After the original sequence is cut into many fragments, the orientation is lost. The sequence can be read in either 5' to 3' or 3' to 5'. One does not know which strand should be selected. If one fragment does not have any overlap with another, it is still possible that its reverse complement might have such an overlap.

b) Base call errors: There are three types of base call errors: substitution, insertion, and deletion errors. They occur due to experimental errors in the electrophoresis procedure. Errors affect the detection of fragment overlaps. Hence, the consensus determination requires multiple alignments in high coverage regions.

c) Incomplete coverage: It happens when the algorithm is not able to assemble a given set of fragments into a single contig.

d) Repeated regions: Repeats are sequences that appear two or more times in the target DNA. Repeated regions have caused problems in many genome sequencing projects, and none of the current assembly programs can handle them perfectly [1].

e) Chimeras and contamination: Chimeras arise when two fragments that are not adjacent or overlapping on the target molecule join together into one fragment. Contamination occurs due to the incomplete purification of the fragment from the vector DNA.

Over the past decade a number of fragment assembly packages have been developed, such as Phrap , TIGR assembler, STROLL, CAP3 (Contig Assembly Program), Celera assembler , and EULER [4] . In this work, we study several programs based on different computational methods to tackle the problem. This work can direct the reader who is interested in this kind of work and who would like to know more about algorithms, their design and implementation of these algorithms.

1. DNA Fragment Assembly Using the Genetic Algorithm

The most challenging step in "overlap-layout consensus" DNA fragment assembly is to order the fragments. Since finding the exact order of the fragments is an extremely slow process, heuristic techniques, such as Genetic Algorithm (GA) can be used. Our GA heuristic was inspired by the work of Parsons et. al. [10]. The GA population consists of a set of individuals. Each individual represents one possible alignment. The search space for the fragment assembly problem is the set of all possible solutions in the population. We use the permutation representation with integer number encoding. Permutation representation requires special operators to make sure that we always get legal solutions. In order to maintain a legal solution, the two conditions that must be satisfied are: first, all fragments must be presented in the ordering, and second, no duplicate fragments are allowed in the ordering.

The fitness function measures the quality of the alignment and finds the one that yields the best score. It is applied to each individual and it should guide the genetic algorithm towards the optimal solution. We implemented two fitness functions. Fitness function F1 sums the overlap score for adjacent fragments in a given solution. The goal here consists in finding the permutation of fragments (an individual in the population) that has the highest score [7][11].

The second fitness function, F2, not only sums the overlap score for adjacent fragments, but also sums the overlap score for all other possible pairs.

This fitness function penalizes solutions in which strong overlaps occur between non-adjacent fragments in the layouts. The objective of F2 is to minimize the overlap score. The overlap score in both F1 and F2 is computed using the semi-global dynamic programming algorithm [5]. In genetic algorithms, operators are applied to a population of individuals to create a new population. In the assembler, three such operators are employed: selection, crossover, and mutation. Ranking selection mechanism is used, in which the GA first

Selected paper: International Conference On Computing (NECICC-2k15)

JAPE

sorts the individuals based on their fitness values and then selects the individuals with the best fitness score until the specified population size is reached.

2. DNA Fragment Assembly Using The Greedy Algorithm

The Greedy Algorithm is based on the Best Set of Maximum Weight Contigs Approach [4]. The algorithm considers unknown orientation and missing fragments. The first step of the algorithm is to construct the Best Set of Maximum Weight Contigs (BSC). The complexity of this step is O(n2 l 2), where n is the number of fragments and l is the average length of fragments. The second step of the algorithm is to order the Maximum Weight Contigs (MWC) of BSC based on contig overlaps order. The complexity of this step is O(m2 l 2), where m is the number of MWCs. The Greedy Algorithm computes contig overlaps rather than fragment overlaps. The advantages are twofold: it enables us to take only the true overlaps into account and it gives a better guarantee for finding the orientation of the fragments.

The major steps of Greedy Algorithm are:

A) Construction of BSC: The algorithm takes as input a set of fragments and outputs a best set of contigs stored in a vector template. The linked list template and the vector template are the major data structures used in this step. The linked list is composed of the overlap scores sorted in descending order. Each item contains a pair of fragment IDs and their overlap score. The first fragment is from the forward direction and the second fragment is the reverse complement of another fragment that has overlap length above some threshold with the first fragment. The vector represents a set of contigs. Each Contig object contains a pair of fragments and their overlap weight.

B) Ordering of Contigs: The time complexity of this step is O(m2 l 2), where m is the number of contigs and l is the average length of the fragments. The algorithm takes as input the best set of contigs (output of the first step) and outputs a list representing the contigs ordering. The algorithm makes use of a vector representing the contigs information (the output of the first step).

3. DNA Fragment Assembly using Structured Pattern Matching:

The Structured Pattern Matching Algorithm is based on a technique called hybridization fingerprinting that is usually used by biologists to deduce the overlap information among DNA clones from biological probes. DNA clones are exact copies of a particular part of a genome and are much longer than fragments [8]. To tackle the DNA fragment assembly problem, This algorithm divides the task into three phases. The first phase is called probe matching. Instead of using biological probes, short probes (e.g. 12 bps) are randomly selected from each fragment. Then exact pattern matching is used in determining the relative positions (i.e. probes occurrences) of the input fragments, including their reverse complements. Thus, each fragment is represented as an ordered set of probes and associated interprobe distances rather than a sequence of nucleotides. The second phase is called overlap map construction. It constructs a detailed map to show how fragments are ordered and how they align. The algorithm first determine how fragments overlap based on the probes occurrences obtained from the previous phase. Contigs consisting of a set of fragments are then constructed in a greedy fashion, guided by a heuristic measure of fragment alignments. However, it does not solely rely on scoring pairwise fragment alignments; instead, each fragment is dynamically scored against each contig. Comparing a fragment to a contig exploits the multiple coverage characteristics of shotgun sequencing data. The third phase is called sequence determination. It is relatively straightforward since all the information we need is available from the second phase. The time complexity of the Structured Pattern Matching algorithm is approximately linear in the length of the target sequence. The efficiency is due to the compact encoding representation of fragments.

The three phases of the algorithm are:

1) Probe matching to identify probe occurrences in input fragments. It takes as input a set of DNA fragments and each fragment's reverse complement. The output of this phase consists of fragments represented as ordered set of probe occurrences and not of nucleotides (as with other traditional algorithms).

2) Overlap map construction: Once the probes are selected and detected, the Structured Pattern Matching algorithm greedily constructs an overlap map based on the patterns of probe occurrences information obtained in the previous step.

The following steps describe the overview of the overlap map construction:

Step 1: Construct a pairwise overlap table that records overlap lengths for all possible pairs.

Step 2: Select the fragment pair with the highest score and construct an initial contig containing just these two fragments.

Selected paper: International Conference On Computing (NECICC-2k15)

Step 3: The new contig is added to the set of contigs and rescored against all remaining fragments.

Step 4: The process continues, adding the bestscoring fragment to the contig and updating the modified contig's scores against all remaining fragments. If no fragment exhibits a significant overlap score, the process terminates, and a new contig is constructed.

3) Sequence determination. Once the overlap map is completed, a consensus sequence is generated using all the available information. The time complexity is linear. From the overlap map, the order of the fragments is known in each contig and it is also known that their relative left positions occur in the map, which makes the sequence determination straightforward.

4. DNA Fragment Assembly using the Clustering Heuristic Algorithm

The traditional three steps: overlap, layout, and consensus, are used in this algorithm. We use the semiglobal alignment algorithm to find all possible pairwise overlaps. When the overlaps are determined, we use a greedy heuristic in the layout phase to find the multiple sequence alignment among a set of fragments. We take the pair of fragments with highest overlap as the starting point. The layout is constructed by successively adding the fragment that has the highest overlap with the assembled fragments. This algorithm takes the unknown orientation into account [14]. The idea is based on the clustering concept. It means that the fragment that is newly added into the alignment has the best overlap with either the last fragment or with the first fragment in the current alignment. Each fragment is progressively added into the existing alignment until no fragment is left.

In what follows, the main parts of the Clustering Heuristic Algorithm are described.

Step 1: Construct a score table for all possible pairs of fragments considering forward directions and reverse complements.

Step 2: Sort the score in descending order and insert all FragmentPair objects (a pair whose score is above some threshold) into a linked list. Each FragmentPair node contains a pair of fragment IDs and overlap score. If the score is positive, it indicates that both fragments are from the same strand. A negative score means that the two fragments are from different strands.

Step 3: Select the first node (a,b) in the linked list as a starting point to order the rest of the fragments. Set a to be the first fragment and b to be the last fragment in the current layout.

Step 4: Select the next node in the linked list and compare the pair of fragments with the first and the last fragments in the current layout. If the clustering is successful, the fragment ID joins the set. If it needs to be inserted in the front, then reset the first fragment in the layout. If it needs to be appended at the end, then reset the last fragment in the layout. Otherwise, put the node in a temporary sorted linked list. The process continues until the current linked list is traversed. Note that only one possible direction for each fragment can be chosen.

Step 5: When the current linked list is being traversed, a new contig is created. Remove the nodes that contain IDs in the selected fragment set from the temporary linked list. Next, reset the temporary linked list as the current linked list and continue from Step 3 until no more fragments are left. Each contig contains the list of ordered fragment IDs.

5. *Nature inspired algorithms to solve DNA fragment assembly problem:*

A) The Problem of DNA Fragment Assembly Using Ant Colony Optimization: Ant colony optimization (ACO) approach was proposed by Meksangsouy, P. and Chaiyaratana; N. [9] which is an asymmetric ordering representation where a path, co-operatively generated by all ants in the colony represents the search solution. The optimality of the fragment layout obtained International Journal on Bioinformatics & Biosciences (IJBB) Vol.2, No.2, June 2012 48 is then determined from the sum of overlap scores calculated for each pair of consecutive fragments in the layout. Two types of assembly problem are investigated: single-contig and multiple-contig problems. The simulation results indicate that in single-contig problems, the performance of the ant colony system algorithm is approximately the same as that of a nearest neighbor heuristic algorithm. On the other hand, the ant colony system algorithm outperforms the nearest neighbor heuristic algorithm when multiple-contig problems are considered. Zhao Y. and et al. [17] improved sequence alignment method based on the ant colony algorithm. This new method could avoid a local optimum and remove especially the paths scores of great difference by regulating the initial and final positions of ants and by modifying pheromones in different times. Zuwairie Ibrahim and Tri Basuki Kurniawan, [18] in their approach model the DNA sequence design as a path-finding problem, which consists of four nodes, to enable the

Selected paper: International Conference On Computing (NECICC-2k15)

implementation of the ACS and compared their results with other methods such as the genetic algorithm.

B) DNA Fragment Assembly Based on Particle Swarm Optimization:

There are few literatures available which represent solution for DNA Sequence Assembly problem using meta heuristic and nature inspired algorithms. PSO algorithm comes under nature inspired algorithm and it has been proven as an effective optimization technique to solve any optimization problems for optimum result. PSO algorithm can also be used to solve computational biology problem to give better result than the conventional methods. Ravi Vikas and Sanjay [12] proposed a solution for DNA sequence assembly problem using Particle Swarm Optimization (PSO) with Shortest Position Value (SPV) rule. DNA sequence assembly problem is a discrete optimization problem, so there is need of discrete optimization algorithm to solve it. It is a continuous version of PSO with SPV rule to solve the DNA sequence assembly problem. SPV rule transforms continuous version of PSO to discrete version.

C) Cuckoo search algorithm:

Cuckoo search algorithm is one of the most recently defined meta-heuristic algorithms proposed by Yang & Deb (2009, 2010). It has been developed by simulating the intelligent breeding behaviour of cuckoos [15] [16]. It is a population-based search procedure used as an optimization tool, in solving complex optimization problems. Cuckoos lay their eggs in the nests of other host birds with incredible abilities like selecting the recently spawned nests and eliminating existing eggs that enhance hatching probability of their eggs. The host bird takes care of the eggs presuming that the eggs are its own. However, some of host birds are able to combat with this parasitic behaviour of Cuckoos, and throw out the identified alien eggs or build their new nests in new locations. Each egg in a nest represents a solution, and a Cuckoo's egg represents a new solution. When generating a new solution Levy flight is performed [13].

The three ideal rules for CS are described as follows:

(i) Each Cuckoo lays one egg at a time, and deposits it in a randomly chosen nest.

(ii) The best nests with high quality of eggs will carry over to the next generations.

(iii) The number of available host nests is fixed and there is a probability that a host can discover an alien egg. In this case, the host bird can either throw the egg away or abandon the nest to build a completely new nest in a new location.

III. Conclusion

The DNA fragment assembly is a very complex problem in computational biology. Since the problem is NP-hard, the optimal solution is extremely difficult to find. Hence, there are many computational techniques that attempt to find good solutions for this problem. In this work, we studied four different algorithms in detail. The algorithms use different techniques to tackle the DNA fragment assembly problem. For smaller data sets, all algorithms got the same result in approximately the same running time. However, the results and the performance vary as the data sets become larger. For larger data sets, i.e., 50 or more fragments, the performance of the algorithms ranking from best to worst is: Structured Pattern Matching Algorithm, Clustering Heuristic Algorithm, Genetic Algorithm, and finally the Greedy Algorithm. Although the algorithms were influenced by techniques and algorithms that are found in the literature, the design and implementations observed in this work vary from the original algorithms. Many features advanced by the authors of the original algorithms were either ignored or completely modified by the design and implementations considered in this work. Therefore, the conclusions made by this work are valid only for our interpretations.

References

[1] Burks C (1994), "DNA sequence assembly, Engineering in Medicine and Biology Magazine", IEEE, Vol. 13, pp. 771- 773.
[2] Chandrasekaran K and Simon S P 2012 Multi-objective scheduling problem: Hybrid approach using fuzzy assisted cuckoo search algorithm. Swarm and Evolutionary compution 5: 1–16
[3] Cooper NG (1994), "The Human Genome Project: Deciphering the Blueprint of Heredity", University Science Books, Mill Valley, CA.
[4] Elloumi M and Kaabi S 1999 Exact and approximation algorithms for the DNA sequence assembly problem. SCI in Biology and Medicine 8
[5] Enrique Alba and Gabriel Luque (2008) , " A Hybrid Genetic Algorithm for the DNA Fragment Assembly Problem", Recent advances in Evolutionary Computation for combinatorial optimization Studies in computational Intelligence, Vol. 153, pp. 101-112, Springer.
[6] E.W. Myers (2000), "Towards simplifying and accurately formulating fragment assembly", Journal of Computational Biology, Vol. 2, pp. 275–290.
[7] Fang, S.C., Wang, Y. and Zhong J (2005), "A Genetic Algorithm Approach to Solving DNA Fragment Assembly Problem", Journal of Computational and Theoretical Nanoscience, Vol. 2, pp. 499- 505.
[8] Kim, S. and Segre, A. M (1999), "AMASS: A structured pattern matching approach to shotgun sequence assembly", Journal of Computational Biology, 6(2), pp. 163-186.
[9] Meksangsouy, P. and Chaiyaratana, N (2003), "DNA fragment assembly using an ant colony system algorithm", Evolutionary Computation, Vol. 3, pp. 1756- 1763.

Selected paper: International Conference On Computing (NECICC-2k15)

[10] Parsons, R.J., Forrest, S. and Burks C (1995), "Genetic algorithms, operators, and DNA fragment assembly", Machine Learning, Vol. 21, pp. 11- 33.
[11] Parsons R.J. and Johnson M.E (1995), "DNA sequence assembly and genetic algorithms- new results and puzzling insights", Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology (ISMB-95), pp. 277- 284.
[12] Ravi, Vikas and Sanjay (2011), "DNA Sequence Assembly using Particle Swarm Optimization", International Journal of Computer Applications Vol.28- No.10, pp. 33-38.
[13] R Indumathy, S Umamaheswari and G Subhasini 2015, "Nature inspired novel cuckoo search algorithm for genome sequence assembly" pp 1-14, Indian Academy of Sciences.
[14] Tammi, M. T. (2003), "The Principles of Shotgun Sequencing and Automated Fragment Assembly", Center for Genomics and Bioinformatics, Karolinska Institute, Stockholm, Sweden.
[15] Yang X-S and Deb S 2009 Cuckoo search via Lévy flights. In Proc. of World Congress on Nature & Biologically Inspired Computing, (NaBIC 2009), IEEE Publications, USA, pp. 210–214
[16] Yang X-S and Deb S 2010 Engineering optimization by Cuckoo search. Int. J. Mathematical Modeling and Numerical Optimization 1: 330–343
[17] Zhao, Y and et al (2008), "An Improved Ant Colony Algorithm for DNA Sequence Alignment", International Symposium on Information Science and Engineering, pp. 683—688.

Selected paper: International Conference On Computing (NECICC-2k15)