Comparison of Material Consumption, Experimental Protocols and Computation Time in DNA Computing

N. Rajaee, K. Hong Ping, A. Lit, D. N. S. A. Salleh, and L. Y. Ng

Abstract—**One of the major constraints in DNA computation is the exponential increase in material consumption and computation time for larger computation size in DNA computing particularly in critical stages such as initial pool generation and extraction during gel electrophoresis. In DNA computation, both the hybridization-ligation method and parallel overlap assembly method can be utilized to generate the initial pool of all possible solutions. In this paper, we discuss and compare the implementation of N × N Boolean matrix multiplication via in vitro implementation between Hybridization-Ligation Method and Parallel Overlap Assembly Method to show that selection of tools and protocols affect the cost effectiveness of a computation in terms of the material consumption, protocol steps and execution time to compute. In general, the the parallel overlap assembly method performs better than hybridization-ligation method in terms of the three parameters mentioned. The calculations are based on approximation of unique sequence strands required for the computation and not actual calculations on the nmol concentration.**

Index Terms—**DNA computing, material consumption, hybridization-ligation method, parallel overlap assembly method.**

I. Introduction

DNA computing holds the promise for a faster and denser computation with its massively parallel computing capabilities. However, there are several difficulties still remain as stumbling blocks which hinder its development as a practical molecular computing. One of which is the amount of DNA required for a computation that increases exponentially with the size of the problem [1]. Current DNA computing strategies are based on enumerating all candidate solutions and then eliminate incorrect DNA by using selection processes. This requires large numbers of starting molecules at each step and each round of selection, usually via initial pool generation and gel electrophoresis [2].

In solving HPP, the seven-node problem was encoded with 20 oligonucleotide strings. Other problems such as maximal clique problems and encoding DNA words were solved with 28 and 108 encoded strings respectively [3]. Going further, a HPP with 23 nodes would start to require a kilogram quantity of DNA and an increase of nodes from 7 to 70 would require 1025 kg of nucleic acids [4]. Methods proposed for solving TSP, clique problem, vertex-cover problem, clique problem and set packing problems all showed exponentially increasing volumes of DNA and linearly increasing time. LaBean et al (2000) proposed that an n1.89n volume, O (n2+m2) time molecular algorithm for the 3-coloring problem and a 1.51n volume, O (n2m2) time molecular algorithm for the independent set problem, where n and m are, subsequently, the number of vertices and the number of edges in the problems resolved [5]. Fu (1997) presented a polynomial time algorithm with 1.497n volume for the 3-SAT problem, a polynomial time algorithm with a 1.345n volume for the 3-coloring problem and a polynomial time algorithm with a 1.229n volume for the independent set [6]. Bunow goes on to estimate that an extension combinatorial database would require nearly 10^{70} nucleotides (by comparison, the universe is estimated to contain roughly 10^{80} subatomic particles) [7].

The second problem with DNA computing is its dependency on the reactions produced by the computation via bio-molecular tools. The DNA computing which relies on wet-lab processes is not an exact process. In many situations, the DNA computer may fail to produce exact, algorithmic results due to the concentration of different species, the environment, the temperature and contamination. Errors can be introduced at any protocol steps of the DNA computation which requires utmost care in its preparation and implementation. Thus, an increase in protocol steps will immediately increase the possibilities for errors. The growing numbers of test tubes involved in the computation cause the whole operation to be labor intensive.

From our proposed algorithm and work, the quantity of initial DNA strands to encode the problem is proportionate to the number of vertices and edges existing in the graph problem representing the matrix multiplication. The number of primers to represent the elements in the product matrix is derived from its total number of row and column indicators whereas the total tubes to represent each element in the product matrix is derived from the total number of primer combinations.

Therefore, for an $(m \times k) \bullet (k \times n)$ matrix multiplication problem, the total number of primers is $m + n$ and total number of tubes is $m \times n$. For a 2×2 product matrix, the total number of primers required is 4 and the total number of tubes is also 4. However, as we have calculated, the number of primers and tubes increases drastically for a larger $N \times N$ computation. For a 10×10 product matrix, the total number of primers required is 20 and the total number of tubes to represent all elements in the product matrix is 100. As the size of the problem increases, the volume of DNA increases exponentially and the number of experimental work becomes tedious and impractical to be considered as a viable technology.

Thus it is necessary to study different strategies to encode

Manuscript received March 10, 2014; revised May 15, 2014. This work was supported in part by Universiti Malaysia Sarawak.

N. Rajaee, K. Hong Ping, A. Lit, D. N. S. A. Salleh, and N. Liang Yew are with Faculty of Engineering, Universiti Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia (e-mail: {rnordiana, hpkismet, lasrani, asdnsdharmiza, ngliangy}@feng.unimas.my).