

# Ant Colony Optimization Algorithms for Sequence Assembly with Haplotyping

Liang-Tai Wei, Chang-Biau Yang<sup>†</sup>, Hsing-Yen Ann and Yung-Hsing Peng

Department of Computer Science and Engineering

National Sun Yat-sen University, Kaohsiung, Taiwan 80424

<sup>†</sup>cbyang@cse.nsysu.edu.tw

Yow-Ling Shieh

Institute of Biomedical Sciences

National Sun Yat-sen University, Kaohsiung, Taiwan 80424

## Abstract

*In this paper, we study how to reconstruct a pair of chromosomes and haplotypes from a given set of fragments obtained by DNA sequencing. We define a new problem, the chromosome pair assembly (CPA) problem, which also involves the determination of (single nucleotide polymorphism) SNP sites. The goal of the problem is to find a pair of sequences (chromosomes) which have the minimum mismatches with the input fragments and their lengths are minimal. We first transform the problem instance into a directed multigraph, in which a Hamiltonian path represents a possible assembly. Then we apply the ant colony optimization (ACO) approach to optimize the solution. In our algorithm, for a given possible assembly of a pair of chromosomes, we design a dynamic programming method to determine SNP sites, which form two haplotypes. This dynamic programming method also provides the fitness score function to evaluate the goodness of the given possible assembly. We perform our algorithm on some artificial test data. The experimental results show that our results are near the optimal solutions of the test data.*

**Key words:** bioinformatics, haplotype, sequence assembly, ant colony optimization (ACO), dynamic programming

## 1 Introduction

The identity between any two human genomes is more than 99.9%, and a difference in their sequences may occur in every 1000 through 1500

base pairs. These variations in DNA sequences provide the difference among humans. The smallest possible variation is the change at a single nucleotide, called the *single nucleotide polymorphism* (SNP) [14, 15], where each variant is called an *allele*. In human body, only two variants are presented in SNPs, called *biallelic*. Alleles with the more frequency are called *major alleles*, while that with the less frequency are called *minor alleles*.

In diploid organisms, each chromosome consists of two distinct copies. A collection of sequential SNPs found on a single chromosome is defined as a *haplotype* [13, 5]. Given a set of SNPs, each individual possesses two haplotypes, one from the paternal chromosome and the other from the maternal chromosome. If an SNP site has two identical values, it is called a *homozygous* site; otherwise it is called a *heterozygous* site. The conflation of two haplotypes is called a *genotype*.

The shotgun sequencing is the process to break up one DNA fragment into smaller fragments. The DNA fragments to be sequenced are first cloned to multiple copies, and then they are randomly cut into random smaller fragments, with some overlapping regions. The goal of the sequence assembly problem is to reconstruct the original two sequences from a given set of fragments obtained by shotgun sequencing. However, in this problem, fragments may also contain SNP sites. Therefore, it is more practical to study how to combine SNP-related issues with the problem of sequence assembly.

Recently, the SNP haplotype assembly problem, whose goal is to reconstruct the two original haplotypes from a given set of DNA fragments, has been noticed and studied[7, 10, 6, 9, 11]. In gen-

eral, previous works for solving this problem are divided into three stages, DNA sequence assembly, polymorphism identification, and haplotyping. In addition, previous results usually passed through these three stages step by step, instead of considering combination of them at one time. For example, the haplotyping was often neglected when one dealt with the DNA sequence assembly.

In this paper, we define a new problem, the *chromosome pair assembly* (CPA) problem, which discusses the sequence assembly along with haplotyping, as follows.

We first define the matching cost function of two characters  $c_1$  and  $c_2$  as follows:

$$mc(c_1, c_2) = \begin{cases} 0 & \text{if } c_1 = '-' \text{ or } c_2 = '-' \\ \varphi & \text{if } c_1 = c_2 \neq '-' \\ \psi & \text{otherwise,} \end{cases} \quad (1)$$

where ' $-$ ' represent a null character (a gap), and  $\varphi$  and  $\psi$  denote the *match score* and *mismatch score*, respectively.

Given a sequence  $S$ , let  $S[i]$  denote the  $i$ th element in  $S$  and  $|S|$  denote the length of  $S$ . Let  $S$  and  $T$  be two sequences over the alphabet  $\{A, G, C, T, -\}$ , where  $|S| \leq |T|$ . The minimum mismatch cost score between  $S$  and  $T$  can be found by sliding  $S$  on  $T$ . It is formally defined as

$$mcs(S, T) = \min_{0 \leq k \leq |T|-|S|} \sum_{i=1}^{|S|} mc(S[i], T[k+i]). \quad (2)$$

Let  $\Phi = \{F_1, F_2, \dots, F_n\}$  be a set of input fragments. Given a distance threshold  $d$ , a pair of candidate sequences  $S_a$  and  $S_b$  should fulfill the following requirements:

1.  $|S_a| = |S_b|$ .
2. Let  $P = \{i \mid S_a[i] \neq S_b[i], 1 \leq i \leq |S_a|\}$ . The absolute difference of any two elements in  $P$  is greater than or equal to  $d$ .

In other words,  $P$  represents the set of possible SNP sites, and  $d$  represents the polymorphism distance, which is the minimum distance between two neighboring SNP sites.

To combine SNP issues into the sequence assembly problem, our goal is to find out a good solution for the sequence assembly problem, such that this solution also satisfies the restriction on minimum distance between any two neighboring SNPs. Let  $\Gamma$  be the score measuring the solution in the problem, which is defined as

$$\Gamma = \sum_{i=1}^n \min(mcs(F_i, S_a), mcs(F_i, S_b)) + \omega|S_a|, \quad (3)$$

← → 20 →

$F_3$	<b>AACCATC<b>G</b>TCA</b>	<b>Sb, mcs = 1</b>
$F_5$	<b>CATCCTCA</b>	<b>Sb, mcs = 0</b>
$F_2$	<b>ACCC<b>A</b>ACATCG<b>I</b>C</b>	<b>Sa, mcs = 3</b>
$F_4$	<b>CTAC<b>CA</b>AC</b>	<b>Sa, mcs = 2</b>
$F_1$	<b>AGCT<b>IA</b>ACCTACC</b>	<b>Sb, mcs = 4</b>
<b>▼ ▼ ▼</b>		
<b>Sa</b>	<b>AACCAC<b>C</b>CTAACAT<b>C</b>CTACC</b>	
<b>Sb</b>	<b>AACC<b>AT</b>C<b>CT</b>CAC<b>A</b>AC<b>CT</b>ACC</b>	

Figure 1: An example for the CPA problem with the input set of fragments  $\{F_1, F_2, F_3, F_4, F_5\}$ ,  $d = 4$  and  $\omega = 0.5$ . Here  $\Gamma$  is calculated by  $1 + 0 + 3 + 2 + 4 + 0.5 \times 20 = 20$ .

where  $\omega$  is the weight relative to the length of the output sequences. The *chromosome pair assembly* (CPA) problem is to find out a pair of candidate sequences  $S_a$  and  $S_b$  such that  $\Gamma$  is minimized. Owing to the SNP distance restriction of the two candidate sequences, one can easily see that the CPA problem demands for both good assembly and valid SNP sites. Figure 1 illustrates an example that shows a solution for the CPA problem with the input set of fragments  $\{F_1, F_2, F_3, F_4, F_5\}$ .

The rest of this paper is organized as follows. In Section 2, we present the overview of our algorithm for solving CPA. Sections 3 through 5 are devoted to the detailed explanation of our algorithm. Section 6 shows the experimental results of our method on some artificial test data, which is near the optimal solutions. Finally, the conclusion will be given in Section 7.

## 2 The Overview of Our Algorithm for CPA

The main idea of our algorithm is to encode the input fragments by a directed multigraph, in which each Hamiltonian path represents a candidate solution (a pair of chromosome sequences) in the CPA problem. Our algorithm for solving CPA, named *chromosome pair determination* (CPD), applies the ant colony optimization (ACO) [3, 2, 4, 16, 17, 1], approach to minimize the  $\Gamma$  value of the solution for CPA. In the ACO approach, a procedure of solution finding is executed iteratively until the difference of two consecutive solutions converges to a predefined threshold. Our

CPD algorithm for solving the CPA problem is given as follows.

**Algorithm:** Chromosome Pair Determination (CPD).

**Input:** A set of fragments.

**Output:** A pair of chromosomes, which is a nearly optimal solution of the CPA problem.

**Step 1.** Find a proper assembly for every two input fragments.

**Step 2.** Each ant determines the suitable permutation for all input fragments and guesses the gender (either paternal or maternal) of each input fragment.

**Step 3.** Each ant generates the *meta-chromosome* and the *suspected polymorphism matrix*.

**Step 4.** Each ant performs the polymorphism determining algorithm.

**Step 5.** Each ant compute the fitness score of the solution obtained by itself.

**Step 6.** If the best solution obtained by the ants in this generation reaches the termination condition, then stop; otherwise, go to Step 2.

Figure 2 shows an example for the CPD algorithm, where the detailed explanation will be given in the following sections.

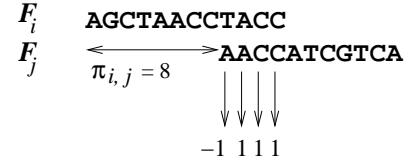
### 3 Representation of a Candidate Solution

We shall describe how to transform the input fragments of the CPA problem to a directed multigraph in which each Hamiltonian path represents a candidate. Note that every solution of CPA corresponds a unique assembly of the input fragments. The *overlapping strength* of the assembly between two fragments  $F_i$  and  $F_j$  can be defined as

$$\sigma(F_i, F_j, \pi_{i,j}) = \sum_{k=1}^{\min(|F_i| - \pi_{i,j}, |F_j|)} m_c(F_i[k + \pi_{i,j}], F_j[k]), \quad (4)$$

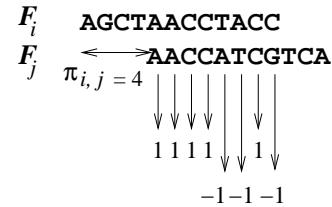
where  $\pi_{i,j}$  denotes the number of shifts between  $F_i$  and  $F_j$ .

Figure 3 shows two examples for the calculation of overlapping strength. It can also be seen that each  $\pi_{i,j}$  corresponds to a unique assembly of



overlapping strength = 2

(a)



overlapping strength = 2

(b)

Figure 3: Examples for calculating overlapping strength of assembly of two fragments with multiple  $\pi_{i,j}$ 's, where  $\varphi = 1$  and  $\psi = -1$ . (a)  $\pi_{i,j} = 8$  and  $\sigma(F_i, F_j, \pi_{i,j}) = 2$ . (b)  $\pi_{i,j} = 4$  and  $\sigma(F_i, F_j, \pi_{i,j}) = 2$ .

$F_i$  and  $F_j$ . Generally, the higher the overlapping strength is, the better the assembly is.

However, one drawback of this measurement is the case of tandem repeats. Once a tandem repeat exists in the suffix of  $F_i$  and the prefix of  $F_j$ , it would be difficult to determine  $\pi_{i,j}$ , since there are more than one assembly for these two fragments. To overcome this problem, one can keep more than one kind of  $\pi_{i,j}$  for every two fragments. Figure 3 shows an example of multiple  $\pi_{i,j}$ 's for computing the overlapping strength of the two fragments. In such a case, one may also put different weights on different  $\pi_{i,j}$ . For ease of understanding, in this paper, for two fragments  $F_i$  and  $F_j$ , we only keep the  $\pi_{i,j}$  that yields the highest overlapping strength.

After  $\pi_{i,j}$  and  $\sigma(F_i, F_j, \pi_{i,j})$  for each pair in  $\Phi$  have been decided, the input fragments can be transformed into a directed multigraph  $G = (V, E)$ , where  $V = \Phi \cup \{\text{sv}\}$ ,  $E = V \times \Phi \times \{\mathbf{m}, \mathbf{p}\}$ , and  $\{\text{sv}\}$  means a common source vertex. Each input fragment is now represented by a vertex in  $V - \{\text{sv}\}$ . For  $u, v \in \Phi$ , there are four directed edges  $(u, v, \mathbf{m})$ ,  $(u, v, \mathbf{p})$ ,  $(v, u, \mathbf{m})$  and  $(v, u, \mathbf{p})$ , whose initial weights are set to  $\sigma(u, v, \pi_{u,v})$ ,

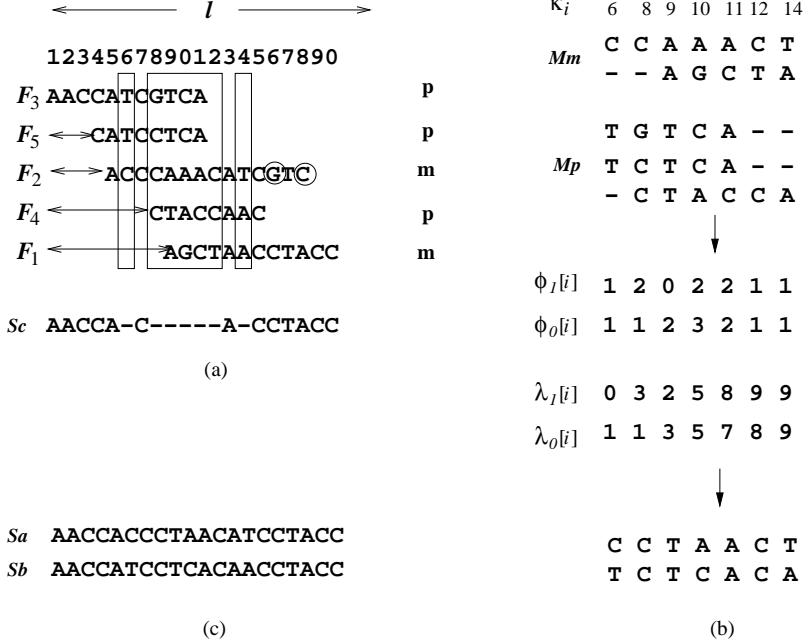


Figure 2: An example for illustrating our algorithm. (a) The suspected polymorphism matrix and the meta-chromosome  $S_c$ . (b) The polymorphism determined by the suspected polymorphism matrix. (c) The pair of output chromosomes with the minimum error correction.

$\sigma(u, v, \pi_{u,v})$ ,  $\sigma(v, u, \pi_{v,u})$  and  $\sigma(v, u, \pi_{v,u})$ , respectively. To identify the paternal or maternal sequence, **m** and **p** are used to indicate the *gender* of the fragment represented by the ending vertex of the edge, which is a crucial factor for determining proper SNP sites in the output chromosomes. In addition, **sv** is a pseudo vertex added to the graph, which can be used as the starting vertex of a Hamiltonian path. For each  $u \in \Phi$ , there are two edges  $(\text{sv}, u, \mathbf{m})$  and  $(\text{sv}, u, \mathbf{p})$ , whose initial weights are set to 0.

Figure 4 shows the directed multigraph representing the given set of fragments  $\{F_1, F_2, F_3\}$ . It can be easily seen that a Hamiltonian path in  $G$  corresponds to an assembly of all fragments. For example, suppose  $\Phi = \{F_1, F_2, F_3, F_4, F_5\}$ , the Hamiltonian path  $\langle \text{sv}, \overrightarrow{p}, 3, \overrightarrow{p}, 5, \overrightarrow{m}, 2, \overrightarrow{p}, 4, \overrightarrow{m}, 1 \rangle$  corresponds to the assembly shown in Figure 5. Here we adopt a vector  $H = \langle (g_1, p_1), (g_2, p_2), \dots, (g_n, p_n) \rangle$  to represent a Hamiltonian path. For example, the path  $\langle \text{sv}, \overrightarrow{p}, 3, \overrightarrow{p}, 5, \overrightarrow{m}, 2, \overrightarrow{p}, 4, \overrightarrow{m}, 1 \rangle$  can be represented by  $\langle (\mathbf{p}, 3), (\mathbf{p}, 5), (\mathbf{m}, 2), (\mathbf{p}, 4), (\mathbf{m}, 1) \rangle$ .

When considering multiple  $\pi_{i,j}$ 's of  $F_i$  and  $F_j$ , one can add more edges on any two vertices  $F_i$  and  $F_j$ . For example, in Figure 3, there are two  $\pi_{i,j}$ 's with overlapping strength 2. Hence, one can encode two  $\pi_{i,j}$ 's from  $F_i$  to  $F_j$  by using four out-

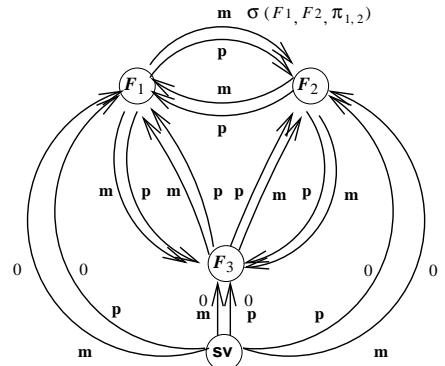


Figure 4: Transforming the input fragments into a directed multigraph with  $\Phi = \{F_1, F_2, F_3\}$ . To simplify this figure, the weights of edges are labeled only on the edge  $(F_1, F_2)$ , which is the overlapping strength of  $F_1$  and  $F_2$ .

$F_3$	AACCATCGTCA	p
$F_5$	CATCCTCA	p
$F_2$	ACCCAAACATCGTC	m
$F_4$	CTACCAAC	p
$F_1$	AGCTAACCTACC	m
	$\pi_{3,5} + \pi_{5,2}$	
	$\pi_{3,5} + \pi_{5,2} + \pi_{2,4}$	
	$\pi_{3,5} + \pi_{5,2} + \pi_{2,4} + \pi_{4,1}$	

Figure 5: An example for the assembly of five fragments, who genders are also determined.

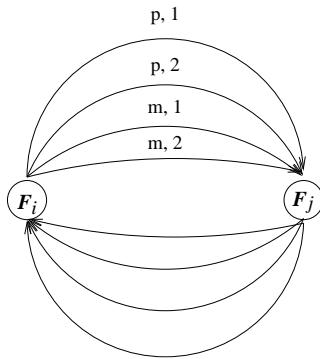


Figure 6: Representation of two  $\pi_{i,j}$ 's by using four outgoing edges, where these two choices on  $F_i$  are labeled as 1 and 2.

going edges, as shown in Figure 6.

## 4 The Fitness Score of a Solution

For each determined assembly, one can construct the *suspected polymorphism matrix* by collecting the suspected columns, which contain two different characters with different genders. The *meta-chromosome* can be easily constructed from non-suspected columns, since all possible SNP sites have been removed. For each non-suspected column, error corrections will be performed if this column contains two or more distinct characters. For any column that needs to be corrected, the answer is determined according to the majority rule. Figure 2(a) shows the process to construct both suspected polymorphism matrix and meta-chromosome, where the blocks denote the suspected matrix and the circles indicate correction need of the characters. After the meta-chromosome and the suspected matrix have been found, the length of the output chromosomes and a partial number of error corrections are both de-

termined.

Let  $M$  be the suspected polymorphism matrix,  $M[i]$  denote the  $i$ th column of  $M$  and  $\kappa[i]$  denote the index of  $M[i]$  in the original assembly. Let  $S_c$  be the sequence of the meta-chromosome. The algorithm for finding the meta-chromosome and suspected polymorphism matrix of a given Hamiltonian path in the directed multigraph representing the CPA problem is given as follows.

**Algorithm:** Meta-Suspected.

**Input:** A set of fragments  $\Phi$ ,  $\pi_{u,v}$  for each pair in  $\Phi$ , and a vector  $H = \langle (g_1, p_1), (g_2, p_2), \dots, (g_n, p_n) \rangle$ .

**Output:** The meta-chromosome chromosome  $S_c$ , and the suspected polymorphism matrix  $M$ , and a set of column indices  $\{\kappa_1, \kappa_2, \dots\}$ .

**Step 1.** Let  $j = 1$ .

**Step 2.** Determine the assembly for  $\Phi$  corresponding to the path  $H$ . (see Figure 5).

**Step 3.** Let  $C_i$  be  $i$ th column of the assembly.

Starting from the leftmost column to the rightmost one, perform the following:

Case 3.1. Column  $i$  has only one character  $a$ .

Set  $S_c[i] = a$ .

Case 3.2. In one of the two gender groups, there are two or more distinct characters on column  $i$  and there is no character in the other group. Assume  $a$  is the character with the highest frequency.

Set  $S_c[i] = a$ .

Case 3.3. There are two or more distinct characters on column  $i$  and both gender groups are not null on column  $i$ . In this case, column  $i$  is added to  $M$  as follows:

Set  $M[j] = C_i$ ,  $\kappa_j = i$ , and  $j = j + 1$ .

Set  $S_c[i] = '-'$ .

Next, we shall show that the suitable SNP sites with the minimum number of error corrections can be determined by using dynamic programming with the suspected matrix  $M$ . Let  $\theta(C_i, a)$  denote the number of occurrences which character  $a$  appears in column  $C_i$ , and  $\theta(C_i, \Sigma)$  to denote the total number of occurrences which  $\Sigma = \{A, G, T, C\}$  appears in column  $C_i$ . Let  $\phi_1[i]$  and  $\phi_0[i]$  denote the minimum number of error corrections in column  $M[i]$  if  $M[i]$  is an SNP site and is not an

SNP site, respectively. They can be calculated as follows:

$$\begin{cases} \phi_0[i] = \theta(M[i], \Sigma) - \max_{a \in \Sigma} \theta(M[i], a), \\ \phi_1[i] = \theta(M[i], \Sigma) - \\ \max_{a \neq b \in \Sigma} (\theta(M_m[i], a) + \theta(M_p[i], b)), \end{cases} \quad (5)$$

where  $M_m[i]$  and  $M_p[i]$  denote the subcolumns in column  $M[i]$  originally derived from the maternal and paternal fragments, respectively.

Let  $\lambda_0(i)$  and  $\lambda_1(i)$  denote the minimum numbers of accumulated error corrections in  $M[1 : i]$  if column  $M[i]$  is determined to not be and to be an SNP site, respectively, where  $M[1 : i]$  represents columns 1 through  $i$  of  $M$ . Remember the restriction that the distance of any two neighboring polymorphism sites cannot be less than  $d$ . Thus, if any both two columns  $M[i], M[j] \in M$  are determined to be SNP sites, then  $|\kappa[i] - \kappa[j]| \geq d$  must hold. Given a column  $M[i]$ , let  $\epsilon_i$  be the index of leftmost column with  $\epsilon_i \leq i$  and  $\kappa_i - \kappa_{\epsilon_i} \leq d - 1$ . Obviously, if  $\kappa_i - \kappa_{i-1} \geq d$ , then  $\epsilon_i = i$ . The dynamic programming for determining SNP sites is given as follows.

$$\begin{cases} \lambda_0(i) = \min(\lambda_0(i-1), \lambda_1(i-1)) + \phi_0[i], \\ 1 \leq i \leq \text{col}(M), \\ \lambda_1(i) = \min(\lambda_0(\epsilon_i-1), \lambda_1(\epsilon_i-1)) + \phi_1[i] + \\ \sum_{k=\epsilon_i}^{i-1} \phi_0[k], 1 \leq i \leq \text{col}(M), \\ \lambda_0(0) = \lambda_1(0) = 0, \end{cases} \quad (6)$$

where  $\text{col}(M)$  denote the number of columns in  $M$ .

Therefore, the minimum number of error corrections  $\chi$  in the suspected matrix  $M$  can be obtained by computing

$$\chi = \min(\lambda_0(\text{col}(M)), \lambda_1(\text{col}(M))). \quad (7)$$

Figure 2(b) shows an example for the above computation. Combining  $\chi$  and the length of  $S_c$ , the fitness score function is therefore defined as

$$\Gamma = \chi + \omega |S_c|. \quad (8)$$

By using the tracing back technique used in the longest common subsequence algorithm [8, 18], we can easily obtain the columns selected along with  $\chi$ , which form a pair of output haplotypes.

## 5 The ACO Approach for the Chromosome Pair Assembly Problem

With the fitness score function to measure the goodness of a solution, we can apply the ACO approach [3, 2, 4, 16, 17, 1, 12] to search for globally good solutions of the CPA problem. Since we

desire to find a good Hamiltonian path in the directed multigraph, we can use the similar scheme used to solve the traveling salesperson problem [3]. In other words, in our algorithm, the trail of an ant corresponds to a Hamiltonian path in  $G$ .

Since vertex  $\mathbf{sv}$  has no incoming edge, a Hamiltonian path must start at  $\mathbf{sv}$ . When an ant is at vertex  $i$ , the decision of next vertex  $j$  is based on the following probability function [3]

$$p_{ij}(t) = \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in N_i} [\tau_{il}(t)]^\alpha [\eta_{il}]^\beta}, \quad (9)$$

where  $p_{ij}(t)$  is the probability for an ant on vertex  $i$  to move to vertex  $j$  at time  $t$ ,  $\tau_{ij}(t)$  is the concentration of pheromone on edge  $(i, j)$  at time  $t$ ,  $\eta_{ij}$  is the weight of edge  $(i, j)$ , and  $N_i$  denotes the set of neighboring vertices of vertex  $i$ . Also,  $\alpha$  and  $\beta$  are two parameters to control the impact factor of the pheromone and that of the weight on the edges, respectively. The pheromone will be updated according to Formula [3]

$$\tau_{ij}(t+1) = (1 - \rho)\tau_{ij}(t) + \sum_{k=1}^m \Delta\tau_{ij}^k(t), \quad (10)$$

where  $m$  denotes the number of ants in one generation,  $\Delta\tau_{ij}^k(t)$  represents the concentration of pheromone deposited by ant  $k$  on edge  $(i, j)$ , and  $\rho \in (0, 1]$  is a parameter controlling the evaporation of pheromone. For each ant, we adopt the following policy to deposit pheromone on its visited edges:

1. For a maternal edge  $(F_i, F_j, \mathbf{m})$  visited by an ant  $k$ , the pheromone is updated as

$$\begin{cases} \Delta\tau_{ij}^k \mathbf{m}(t) = \Delta\tau_{ij}^k \mathbf{m}(t-1) + \frac{\mu}{l} + \frac{\mu}{\sqrt{\chi+1}}, \\ \Delta\tau_{ij}^k \mathbf{p}(t) = \Delta\tau_{ij}^k \mathbf{p}(t-1) + \frac{\mu}{l}, \end{cases}$$

where  $\mu$  is a constant and  $\gamma$  is a constant influence factor.

2. For a paternal edge  $(F_i, F_j, \mathbf{p})$  visited by an ant  $k$ , we have

$$\begin{cases} \Delta\tau_{ij}^k \mathbf{p}(t) = \Delta\tau_{ij}^k \mathbf{p}(t-1) + \frac{\mu}{l} + \frac{\mu}{\sqrt{\chi+1}}, \\ \Delta\tau_{ij}^k \mathbf{m}(t) = \Delta\tau_{ij}^k \mathbf{m}(t-1) + \frac{\mu}{l}. \end{cases}$$

Now, we analyze the time complexity of Algorithm CPD as follows. In Step 1, the computation of the overlapping strength of every two input fragments requires  $O(\sum_{1 \leq i < j \leq |\Phi|} |F_i| \cdot |F_j|)$  time. In Step 2, each ant establishes a Hamiltonian path (solution), where each next vertex is selected according to the probability in Formula 9.

Thus, the construction of one Hamiltonian path needs  $O(n^2)$  time, since there are  $O(n)$  vertices and  $O(n^2)$  edges in  $G$ , where  $n = |\Phi|$  denotes the number of input fragments. Also, one can restore each solution into the assembly form and then partition the symbols in each column by using a linear scan, which needs  $O(\sum_{F_i \in \Phi} |F_i|)$  time. In Step 3, each ant performs Algorithm Meta-Suspected for determining the meta-chromosome and suspected matrix, which needs  $O(\sum_{F_i \in \Phi} |F_i|)$  time. In Step 4, each ant determines the SNP sites by using the dynamic programming in Formula 6, which needs only  $(\text{col}(M))$  times. Step 5 needs only constant time to calculate the fitness score of each solution. Therefore, we conclude that the time complexity of Algorithm CPD is  $O((n^2 + nL)gr)$ , where  $L$  denotes the maximum length of the fragments,  $g$  represents the number of ACO generations and  $r$  represents the number of ants per generation.

## 6 Experimental Results

To test the performance of our algorithm, we do experiments on some sets of artificial sequences. The artificial sequences in one test set are randomly generated, then randomly cut into fragments, with some overlapping prefix or suffix. The testing data sets are shown in Table 1. Since the pair of chromosomes are artificially generated, the optimal  $\Gamma$  can be estimated by Formula 3, which can be used to compare the  $\Gamma$  value obtained from our ACO algorithm.

The parameters of our ACO algorithm are set as *ant generations* = 150, *number of ants* = 500,  $\alpha = 1$ ,  $\beta = 2$ ,  $\rho = 0.5$ ,  $\gamma = 1$ , and  $\mu = 0.01$ . Also, the errors in shotgun sequencing are simulated by setting the error rate to 0.01. We perform our algorithm 20 times on each data set shown in Table 1, and the experimental results are shown in Table 2. The left part in Table 2 lists the estimated  $\Gamma$  obtained from Formula 3, along with the average  $\Gamma$  obtained from the ACO algorithm. Besides, the right part in Table 2 lists the best solutions obtained from the 20 executions of our ACO algorithm.

Observing the left part in Table 2, one can see that the average  $\Gamma$  values of the solutions obtained by our algorithm are very close to the estimated  $\Gamma$  from Formula 3, the relative differences are all less than 0.04. In addition, the best solutions in the right part shows that our results are even better than the estimated optimal  $\Gamma$ , which implies the hardness of defining the demanded solution.

In our experiments, the error rate of DNA fragment sequencing is the most important factor which affects the accuracy of our algorithm. We consider the relationship between a single character in the pair of original chromosomes and the set of input fragments. If character  $c$  is on position  $i$  of one original chromosome, but no input fragment holds character  $c$  due to sequencing errors, then no method can restore the character  $c$  correctly. Suppose that position  $i$  and the gender of every input fragment can be determined correctly. We have the following facts:

- Suppose position  $i$  is not an SNP site and only one character  $c$  appears on position  $i$  of all input fragments. Character  $c$  can be determined correctly if and only if the character in the fragment is sequenced correctly.
- Suppose position  $i$  is not an SNP site and more than one character appears on position  $i$  of all input fragments. If the error rate of character  $c$  in the input fragments is less than 50%, then the character  $c$  can be determined correctly.
- Suppose position  $i$  is an SNP site, and characters  $a$  and  $b$  appear on position  $i$  of some fragments. If there is no other character  $c$  whose appearance frequency is higher than  $a$  and  $b$ , then  $a$  and  $b$  can be determined correctly.

## 7 Conclusion

In our method, the length of output chromosomes and the number of error corrections are two tradeoff factors when we construct a solution. If one raises the weight  $\omega$  for the chromosome length, the output chromosome tends to be shorter and to contain more mismatch characters, which leads to more error corrections. On the contrary, longer chromosomes usually contain a smaller amount of error corrections. In our experiments, once the determined order of assembly is correct, it is of high tendency that the length of the reconstructed chromosomes is equal to that of the original (demanded) chromosomes. In addition, the DNA fragment sequencing error rate is the most important factor of the accuracy of our method.

Previous models usually handle the assembly and haplotypes independently. Therefore, a good assembly may be invalid when SNP sites are considered. In this paper, we propose the CPA problem, which combines the knowledge of SNP with

Table 1: The artificial data sets. The following abbreviations are used: TS (target sequence), L (length of target sequence), NF (number of fragments), AFL (average length of fragments),  $d$  (distance threshold), ERS (error rate of sequencing), NEC (number of error corrections).

TS	L (bps)	NF	AFL(bps)	$d$	ERS	NEC
Sim1	1200	20	200	200	0.01	38
Sim2	2400	30	300	300	0.01	66
Sim3	3835	42	395	300	0.01	129
Sim4	8428	45	1285	300	0.01	458
Sim5	12580	30	1285	300	0.01	297
Sim6	21280	60	1525	300	0.01	674

Table 2: Results of our algorithm executed on the artificial testing data. The following abbreviations are adopted: TS (target sequence),  $\Gamma'$  (estimated  $\Gamma$  from Formula 3),  $\Gamma''$  (average  $\Gamma$  of 20 executions), MinD (minimal difference), MaxD (maximal difference), AD (average difference), RAD (relative average difference), L (length of sequence), NEC (number of error corrections). Relative average difference is defined as  $\frac{\Gamma'' - \Gamma'}{\Gamma'} \times 100\%$ . Relative weight of length ( $\omega$ ) is set to 0.5.

TS	Average of 20 Experiments						Best Solution		
	$\Gamma'$	$\Gamma''$	MinD	MaxD	AD	RAD (%)	L	NEC	$\Gamma$
Sim1	638	638.05	-1	1	0.05	0.008	1200	37	637
Sim2	1266	1278.85	-5	75	12.85	1.015	2400	61	1261
Sim3	2046	2112.55	-4	165	66.55	3.253	3835	125	2042
Sim4	4672	4694.95	-1	90	22.95	0.49	8428	457	4671
Sim5	6587	6770.75	-4	405	183.75	2.79	12580	293	6583
Sim6	11314	11762.85	-4	1016	448.85	3.967	21280	670	11310

the sequence assembly problem. In addition, we also propose an ACO algorithm to overcome the problem. Related algorithms for solving the CPA problem still remains worthy of further devising in the future.

## References

- [1] C. Blum and A. Roli, “Metaheuristics in combinatorial optimization: Overview and conceptual comparison,” *ACM Comput. Surv.*, Vol. 35, No. 3, pp. 268–308, 2003.
- [2] M. Dorigo, G. D. Caro, and L. M. Gambardella, “Ant algorithms for discrete optimization,” *Artificial Life*, Vol. 5, No. 3, pp. 137–172, 1999.
- [3] M. Dorigo and L. M. Gambardella, “Ant colonies for the traveling salesman problem,” *BioSystems*, No. 43, pp. 73–81, 1997.
- [4] M. Dorigo and G. D. Di Caro, *The Ant Colony Optimization Meta-Heuristic*, pp. 11–32. London: McGraw-Hill, 1999.
- [5] S. Gabriel, S. Schaffner, H. Ngyen, J. Moore, J. Roy, B. Blumenstiel, J. Higgins, M. Deflise, A. Lochner, M. Faggart, S. N. Liu-Cordero, C. Rotimi, A. Adeyemo, R. Cooper, R. Ward, E. Lander, M. Daly, and D. Altshuler, “The structure of haplotype blocks in the human genome,” *Science*, Vol. 296, No. 21, pp. 2225–2229, 2002.
- [6] H. J. Greenberg, W. E. Hart, and G. Lancia, “Opportunities for combinatorial optimization in computational biology.,” *INFORMS Journal on Computing*, Vol. 16, No. 3, pp. 211–231, 2004.
- [7] B. V. Halldórsson, V. Bafna, N. Edwards, R. Lippert, S. Yooseph, and S. Istrail, “Combinatorial problems arising in snp and haplotype analysis.,” *DMTCS*, pp. 26–47, 2003.
- [8] D. S. Hirschberg, “A linear space algorithm for computing maximal common subsequence,” *Communications of the ACM*, Vol. 18, No. 6, pp. 341–343, 1975.
- [9] G. Lancia, V. Bafna, S. Istrail, R. Lippert, and R. Schwartz, “Snps problems, complexity, and algorithms,” *ESA ’01: Proceedings of the 9th Annual European Symposium on Algorithms*, London, UK, pp. 182–193, Springer-Verlag, 2001.
- [10] L. Li, J. H. Kim, and M. S. Waterman, “Haplotype reconstruction from snp alignment,” *RECOMB ’03: Proceedings of the seventh annual international conference on Computational molecular biology*, New York, NY, USA, pp. 207–216, ACM Press, 2003.
- [11] R. Lippert, G. L. R. Schwartz, and S. Istrail, “Algorithmic Strategies for the SNPs Haplotype Assembly Problem,” *Briefings in Bioinformatics*, Vol. 3, No. 1, pp. 23–31, 2002.
- [12] W.-T. Liu, C.-B. Yang, S.-H. Shiau, and Y.-L. Shiue, “Primer set selection in multiple pcr experiments,” *Proc. of the IPSI-2005 Amalfi (International Conference on Advances in the Internet, Processing, Systems, and Interdisciplinary Research)*, Feb., 2005.
- [13] M. Olivier, “A haplotype map of the human genome,” *Physiol. Genomics*, Vol. 13, No. 1, pp. 3–9, 2003.
- [14] Riva and I. S. Kohane, “SNPper: retrieval and analysis of human SNPs,” *Bioinformatics*, Vol. 18, No. 12, pp. 1681–1685, 2002.
- [15] B. S. Shastry, “SNP alleles in human disease and evolution,” *Journal of Human Genetics*, Vol. 47, pp. 561–566, 2002.
- [16] T. Stützle and H. Hoos, “ $\mathcal{MAX} - \mathcal{MIN}$  Ant System and local search for the Traveling Salesman Problem,” *Proceedings of the fourth International Conference on Evolutionary Computation (ICEC)* (IEEE, ed.), pp. 308–313, IEEE Press, 1997.
- [17] T. Stützle and H. Hoos, “Max-min ant system,” *Future Gener. Comput. Syst.*, Vol. 16, No. 9, pp. 889–914, 2000.
- [18] C. B. Yang and R. C. T. Lee, “Systolic algorithms for the longest common subsequence problem,” *Journal of the Chinese Institute of Engineers*, Vol. 10, No. 6, pp. 691–699, 1987.