2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

A Hopfield Neural Network Based Algorithm for Haplotype Assembly from Low-quality Data

Xiao Chen, Qinke Peng, Libin Han and Xiao Wang

***Abstract*—The objective of the haplotype assembly problem is to conclude a pair of haplotypes from a set of aligned single nucleotide polymorphism (SNP) fragments from a single individual. Errors in the SNP fragments, which are inevitable in the real-world application, severely increase the difficulty of the problem. As a result, most methods could not get accurate haplotypes on the data with high error rate. In this paper, we introduce a Hopfield neural network based method, named HNHap, to solve the haplotype assembly problem. Hopfield neural network is a very promising and effective approach to solve the combinatorial optimization problem. The stochastic optimal competitive Hopfield network model that has the mechanism to escape from the local optimum is a great improvement for the original model. Thus we map the haplotype assembly problem onto the stochastic optimal competitive Hopfield network model, in which a group of neurons correspond to an SNP fragment and the states of neurons denote the classification of the fragment. We also design a proper energy function based on the minimum error correction model for the haplotype assembly problem. We compare HNHap with other algorithms and the experiment results show that HNHap is an effective method to solve the haplotype assembly problem, especially on data with high error rate.**

I. Introduction

THE research of Human Genome Project shows that people are almost identical at the DNA level. There should be associations between human diseases and genetic variations [1]. Consequently, the study of genetic variations among individuals has been an active research area in the recent years. Single nucleotide polymorphisms (SNPs) are the main form of human genetic variations, which play an important role in association studies, gene disease diagnoses, etc. [2].

Humans are diploid organisms, i.e. the chromosomes come in two copies: one comes from mother and the other from father. An SNP is a specific base alteration in the chromosome, and the sequence of SNPs in a certain chromosome is called a haplotype. Therefore, there are two copies of haplotypes in human genome. Great efforts have been taken to get the haplotypes because haplotypes contain more information than individual SNP [3]. However, sequencing the two haplotypes directly is very difficult and costly for the current sequencing technologies. So computational methods to get haplotypes now receive more and more attentions.

There are two main classes of computational methods: haplotype inference and haplotype assembly. Haplotype inference is to obtain haplotypes from population genotype data. Various methods have been used to solve the haplotype inference problem [4]-[6]. An alternative way to obtain the haplotypes for an individual is haplotype assembly. Given a set of aligned overlapping fragments which are of arbitrary length and contain errors from a single individual, the haplotype assembly problem is to obtain the pair of haplotypes from them.

The haplotype assembly problem, also named the Single Individual Haplotyping problem, was first introduced by Lancia *et al.* [7]. Various optimization models have been proposed to solve this problem, such as minimum fragment removal (MFR), minimum SNP removal (MSR) [7], minimum error correction (MEC) [8], and maximum fragments cut (MFC) [9]. Among them, MEC is the most complex model; however, it is the most commonly used in practice. Wang *et al.* designed a genetic algorithm based on MEC model. However, the accuracy of the genetic algorithm degrades [10]. In order to improve the accuracy, a clustering algorithm based on two distance functions, named 2d-mec, was proposed [11]. Levy *et al.* introduced a greedy heuristic algorithm for the noise-free data [12]. Bansal *et al.* developed a method, named HapCUT, to minimize the MEC score of the reconstructed haplotypes by iteratively computing max-cuts in graphs derived from the sequenced fragments [13]. Recently, Wang et al. proposed a genetic algorithm based method that is equipped with a well-designed fitness function [14]. Besides, some other heuristic algorithms have also been proposed for the problem [15]-[18].

Although these algorithms perform well in the error-free or low error rate cases, but they perform worse as the error rate of fragments increases. Advanced personalized medicine is one of the goals of current research, and new genetic diagnostic methods are critical for it. Thus, there is an increasing demand for the portable sequencing equipment that can be used widely. However, the portable sequencing equipment is likely to produce low-quality data without the high-tech laboratory environment [18]. Therefore, it is a crucial and challenging task to reconstruct haplotypes from low-quality data. The haplotype assembly problem is a combinatorial optimization problem when using MEC model and Hopfield neural network [19] is a very promising and effective approach to combinatorial optimization problems. A Hopfield neural network based on MFR model is proposed by Xu *et al* [20]. In this paper, we introduce a Hopfield neural network based method named HNHap, to solve the haplotype

Xiao Chen, Qinke Peng, Libin Han and Xiao Wang are with Systems Engineering Institute, School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, 710049, China.

Qinke Peng is the corresponding author (phone: +8602982667964; e-mail: qkpeng@mail.xjtu.edu.cn).

This work was supported by the National Natural Science Foundation of China under Grant 61173111.

assembly problem based on the MEC model. We map the problem onto the stochastic optimal competitive Hopfield network model (SOCHN) [21] and design an energy function based on MEC model. The SOCHN has the mechanism to escape from local optimum, so it can find better solutions. We evaluate the performance of HNHap using Geraci's data set and compare HNHap with the seven selected algorithms in Geraci's study [22].

The rest of the article is organized as follows. In Section 2, we introduce the haplotype problem formulation and describe our algorithm based on the SOCHN. In Section 3, we show the performance of our algorithm compared with other methods. Finally, we conclude in Section 4.

II. METHODS

A. Problem Formulation

Assume that there are a set of SNP fragments obtained from a pair of chromosomes. These fragments are stored in an SNP matrix $S=(s_{ij})_{m\times n}$, where m is the number of SNP fragments and n is the length of the corresponding haplotypes. In the matrix S, each row and column corresponds to an SNP fragment and an SNP site, respectively. Each entry $s_{ij}\in\{'a','c','g','t','-'\}$, in which '–' is a gap that is uncovered by the fragments or a missing allele, and other four characters represent four types of nucleotides. Because the fragments are all much shorter than n, the uncovered parts of the fragments aligned against the corresponding haplotypes are denoted by gaps.

The haplotypes of this matrix can be represented by a pair of nucleotide strings $H(S)=(h_1,h_2)$, and the length of each string is n. For a specific position i of h_1 and h_2, if it is a homozygous site, two nucleotides in the ith column of h_1 and h_2 are the same; otherwise, they are complementary. The haplotype assembly problem is to divide the rows of the matrix into two disjoint subsets and each subset determines a haplotype.

The distance between two fragments $X=(x_1,x_2,\cdots,x_n)$ and $Y=(y_1,y_2,\cdots,y_n)$, is defined as:

$$\tilde{D}(X,Y)=\sum_{i=1}^{n}\tilde{d}(x_i,y_i) \tag{1}$$

$$\text{where } \tilde{d}(x,y)=\begin{cases}1, & \text{if } x\neq'-',y\neq'-',x\neq y.\\ 0, & \text{otherwise.}\end{cases} \tag{2}$$

The distance between a fragment and a haplotype is defined in the same way. If $\tilde{D}(X,Y)>0$, two fragments X and Y are called conflict, otherwise they are called compatible. Two fragments that conflict either are not from the same chromosome copy or contain errors. The matrix S is called feasible when the rows of S can be divided into two disjoint subsets and fragments in each subset are pairwise compatible. Otherwise, it is infeasible [10]. If the fragments in the matrix S are error-free, S is feasible.

The MEC model is defined as follows: Given an SNP matrix S, correct a minimum number of elements to make the resulting matrix feasible.

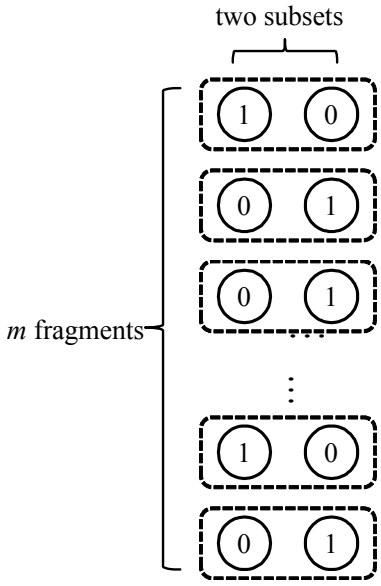

Fig. 1 The mapping of the haplotype assembly problem onto the SOCHN.

B. Hopfield Neural Network

Hopfield neural network is a fully connected feedback neural network, and the energy function of the network corresponds to its state. Thus, the process of finding the minimum value of energy function is translated into the state evolution process of the network towards equilibrium state. Hopfield neural network has been widely applied to associative memory and optimization calculation. Many improvements of Hopfield neural network are also proposed to solve various optimization problems. Galán-Marín *et al.* proposed a discrete Hopfield model termed the optimal competitive Hopfield model (OCHN) [23], [24]. The OCHN maximizes the descent of energy function by updating the groups of neurons.

The OCHN contains p disjoint groups of neurons and each group consists of k neurons. For the ith neuron in the rth group, the input is $u_{ri}(t)$, the output is $v_{ri}(t)$ and the threshold is θ_{ri}, where $v_{ri}(t)\in\{0,1\}$ and t is the discrete time. $w_{ri,lj}$ denotes the connection weight between the ith neuron in the rth group and the jth neuron in the lth group. $w_{ri,ri}$ can be arbitrary values and $w_{ri,lj}=w_{lj,ri}$. The energy function of the OCHN is as follows:

$$\begin{aligned}E(t)=&-1/2\sum_{r=1}^{p}\sum_{i=1}^{k}\sum_{l=1}^{p}\sum_{j=1}^{k}w_{ri,lj}v_{ri}(t)v_{lj}(t)\\&+\sum_{r=1}^{p}\sum_{i=1}^{k}\theta_{ri}v_{ri}(t).\end{aligned} \tag{3}$$

The input of the neuron is computed by the updating rule

$$u_{ri}(t)=\sum_{l=1}^{p}\sum_{j=1}^{k}w_{ri,lj}v_{lj}(t)-\theta_{ri}. \tag{4}$$

At time t, only one group is updated, i.e. the states of neurons in the same group are synchronously updated and the states of neurons in different groups are cyclically updated. Note that, in a specific group, one and only one neuron's output is 1. Let rc be the neuron with the output 1 in the rth group at time t and ra be the candidate neuron in the rth group that will has the output 1 at time $t+1$, then the energy

TABLE I
COMPARISON OF HNHAP WITH OTHER ALGORITHMS ON DATA SET OF $l=100$.

e	c	Baseline	SPH	FAST	2d-mec	HapCUT	MLF	SHR	DGS	HNHap
0.0	3	1.0000	0.9989	0.9998	0.9905	1.0000	0.9730	0.8162	1.0000	0.9799
0.0	5	1.0000	1.0000	0.9996	0.9973	1.0000	0.9922	0.8609	1.0000	0.9936
0.0	8	1.0000	1.0000	1.0000	1.0000	1.0000	0.9970	0.9119	1.0000	0.9992
0.0	10	1.0000	1.0000	1.0000	1.0000	1.0000	0.9984	0.9440	1.0000	0.9977
0.1	3	0.9707	0.8948	0.9128	0.9116	0.9286	0.8891	0.6957	0.9301	**0.9324**
0.1	5	0.9918	0.9675	0.9642	0.9508	0.9204	0.9697	0.7377	0.9851	**0.9867**
0.1	8	0.9972	0.9892	0.9930	0.9835	0.9006	0.9854	0.7584	0.9894	**0.9993**
0.1	10	0.9992	0.9904	0.9981	0.9881	0.8921	0.9951	0.7621	0.9967	**0.9998**
0.2	3	0.8984	0.6230	0.7150	0.7380	0.7822	0.7251	0.6148	0.7250	**0.8328**
0.2	5	0.9444	0.7992	0.7974	0.7931	0.8380	0.8358	0.6546	0.8127	**0.9430**
0.2	8	0.9672	0.8517	0.8807	0.8730	0.8640	0.9176	0.6812	0.8785	**0.9871**
0.2	10	0.9802	0.8654	0.9154	0.8943	0.8710	0.9376	0.6989	0.9175	**0.9935**
0.3	3	0.7886	0.4802	0.6169	0.6233	0.6019	0.6176	0.5571	0.6111	**0.6838**
0.3	5	0.8404	0.6370	0.6391	0.6403	0.6294	0.6528	0.5993	0.6469	**0.8542**
0.3	8	0.8780	0.6666	0.6610	0.6749	0.6726	0.6968	0.6322	0.6634	**0.9362**
0.3	10	0.9030	0.6758	0.6754	0.6779	0.7086	0.7146	0.6321	0.6876	**0.9668**

difference is

$$\Delta E_r(t) = E(t+1) - E(t) = u_{rc}(t) - (u_{ra}(t) - W_{rc,ra}), \tag{5}$$

where $W_{rc,ra} = -1/2(w_{rc,rc} + w_{ra,ra} - 2w_{rc,ra})$. The neuron with the maximum value of $u_{ra}(t) - W_{rc,ra}$ is selected as ra, i.e., the input-output function of the ith neuron in the rth group is

$$v_{ri}(t+1) = \begin{cases} 1, & \text{if } u_{ri}(t) - W_{rc,ri} = \max\limits_{j=1,\ldots,k}\{u_{rj}(t) - W_{rc,rj}\}. \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

In this case, $\Delta E_r(t) \le 0$ is guaranteed and the energy descends maximum at each time.

Because the OCHN is a gradient descent based algorithm, it is easy to fall into the local minima. Wang *et al.* proposed the SOCHN to help the OCHN escape from the local minima by applying the stochastic hill-climbing dynamics. In the SOCHN, the input-output function of the ith neuron in the rth group is modified as follows:

$$u_{ri}^{'}(t) = \alpha(z) \cdot (u_{ri}(t) - W_{rc,ri}), \tag{7}$$

$$v_{ri}(t+1) = \begin{cases} 1, & \text{if } u_{ri}^{'}(t) = \max\limits_{j=1,\ldots,k}\{u_{rj}^{'}(t)\}, \\ 0, & \text{otherwise,} \end{cases} \tag{8}$$

where $\alpha(z) = random(g(z),1)$, $g(z) = 1 - 2e^{-z/\beta}$ and $z = t/p$. β is a parameter that controls the evolution speed of stochastic dynamics. The SOCHN permits energy to increase initially and reverts towards the OCHN finally.

C. The HNHap Method

In this section, we propose a method termed HNHap to solve the haplotype assembly problem, based on the SOCHN. Before we map the problem onto the SOCHN, we first make some preprocess on data.

1) Data Preprocess:

Given an SNP matrix $S = (s_{ij})_{m \times n}$, we calculate a $n \times 2$ matrix *MF*, where $MF(j,1)$, $j = 1, \cdots, n$ is the most frequent nucleotide in the jth column of S and $MF(j,2)$, $j = 1, \cdots, n$ is the second most frequent nucleotide in the jth column of S. Considering the jth column of S, if the frequency of $MF(j,1)$ is greater than 0.8, this column is considered as a homozygous site with the nucleotide $MF(j,1)$. Otherwise, the jth column is considered as a heterozygous site. The threshold is set to 0.8 according to the statistical analysis. Too large threshold leads to an omission of real homozygous sites because of the existence of data errors, and too small threshold gets many false homozygous sites. In the jth column of the matrix S, the nucleotides equal to $MF(j,1)$ are replaced with '1', and those equal to $MF(j,2)$ are replaced with '0'. Other nucleotides if any in the jth column will be replaced with '–'. Then the homozygous sites are removed from the matrix S. After doing this, the SNP matrix S is converted to a smaller binary matrix S_0, which facilitates the following steps. Note that, the homozygous sites will be inserted into the corresponding positions of the final solution at last.

2) SOCHN for The Haplotype Assembly Problem:

A partition P of the matrix S_0 divides the rows of S_0 into two disjoint subsets. Let C_1 be one subset of rows, and the character with maximum frequency at position q among the fragments in C_1 is denoted as $\varepsilon_q^{C_1} \in \{0,1\}$. Thus the consensus string $h_1^{'}$ deduced by C_1 is defined as a string, where the character at position q is $\varepsilon_q^{C_1}$. The corresponding consensus string $h_2^{'}$ of the other subset of fragments is obtained in the same way.

For a partition $P = (C_1, C_2)$ of S_0, the error function can be defined as:

$$ERR(P) = \sum_{i=1}^{2} \sum_{f \in C_i} \tilde{D}(f, h_i^{'}). \tag{9}$$

The error function denotes the number of corrected elements to make the resulting matrix feasible based on the partition P. Therefore, the objective of MEC model for the haplotype assembly problem is equivalent to finding a partition P^* of

TABLE II
COMPARISON OF HNHAP WITH OTHER ALGORITHMS ON DATA SET OF $l = 350$.

e	c	Baseline	SPH	FAST	2d-mec	HapCUT	MLF	SHR	DGS	HNHap
0.0	3	1.0000	0.9993	0.9896	0.9650	1.0000	0.8642	0.8297	0.9998	0.9626
0.0	5	1.0000	1.0000	0.9996	0.9927	1.0000	0.9288	0.8295	1.0000	0.9983
0.0	8	1.0000	1.0000	1.0000	0.9978	1.0000	0.9692	0.8954	1.0000	0.9997
0.0	10	1.0000	1.0000	0.9999	0.9991	1.0000	0.9810	0.8784	1.0000	0.9996
0.1	3	0.9707	0.8188	0.8711	0.8386	0.9299	0.7517	0.6822	0.9260	0.8943
0.1	5	0.9905	0.9592	0.9453	0.9130	0.9134	0.8582	0.7244	0.9785	**0.9832**
0.1	8	0.9972	0.9843	0.9852	0.9641	0.8963	0.9327	0.7416	0.9963	**0.9969**
0.1	10	0.9990	0.9836	0.9948	0.9781	0.8883	0.9616	0.7285	0.9982	**0.9993**
0.2	3	0.8959	0.4392	0.6843	0.6746	0.7709	0.6418	0.5915	0.6914	**0.8015**
0.2	5	0.9430	0.7287	0.7456	0.7284	0.8306	0.7278	0.6318	0.7689	**0.9392**
0.2	8	0.9680	0.8247	0.8529	0.7912	0.8616	0.7985	0.6699	0.8423	**0.9863**
0.2	10	0.9811	0.8555	0.8774	0.8169	0.8672	0.8314	0.6682	0.8784	**0.9910**
0.3	3	0.7826	0.2509	0.5901	0.5927	0.5648	0.5808	0.5476	0.5781	**0.6380**
0.3	5	0.8401	0.5784	0.6021	0.6061	0.5817	0.6063	0.5575	0.6095	**0.8307**
0.3	8	0.8734	0.6294	0.6259	0.6230	0.6206	0.6339	0.6043	0.6285	**0.9165**
0.3	10	0.9026	0.6381	0.6437	0.6340	0.6641	0.6408	0.6189	0.6408	**0.9600**

TABLE III
COMPARISON OF HNHAP WITH OTHER ALGORITHMS ON DATA SET OF $l = 700$.

e	c	Baseline	SPH	FAST	2d-mec	HapCUT	MLF	SHR	DGS	HNHap
0.0	3	1.0000	0.9993	0.9876	0.9461	1.0000	0.7816	0.7815	0.9999	0.8830
0.0	5	1.0000	1.0000	0.9989	0.9760	1.0000	0.8544	0.8324	1.0000	0.9301
0.0	8	1.0000	1.0000	1.0000	0.9917	1.0000	0.9194	0.8682	1.0000	0.9931
0.0	10	1.0000	1.0000	0.9998	0.9966	1.0000	0.9333	0.8983	1.0000	0.9973
0.1	3	0.9712	0.7047	0.8295	0.7860	0.9269	0.6982	0.6679	0.9315	0.8427
0.1	5	0.9913	0.9471	0.9408	0.8805	0.9158	0.8094	0.7158	0.9775	**0.9782**
0.1	8	0.9972	0.9848	0.9859	0.9483	0.8957	0.8632	0.7429	0.9873	**0.9898**
0.1	10	0.9992	0.9861	0.9955	0.9649	0.8892	0.8839	0.7260	0.9966	**0.9996**
0.2	3	0.8978	0.1990	0.6518	0.6468	0.7531	0.6240	0.5913	0.6692	**0.7771**
0.2	5	0.9427	0.6810	0.7118	0.6969	0.8250	0.6820	0.6170	0.7415	**0.9259**
0.2	8	0.9661	0.8006	0.8078	0.7512	0.8562	0.7475	0.6529	0.8177	**0.9762**
0.2	10	0.9798	0.8127	0.8719	0.7780	0.8610	0.7650	0.6748	0.8607	**0.9945**
0.3	3	0.7860	0.0953	0.5814	0.5828	0.5524	0.5701	0.5363	0.5726	**0.6263**
0.3	5	0.8382	0.5232	0.5915	0.5961	0.5553	0.5944	0.5622	0.5946	**0.7762**
0.3	8	0.8748	0.6158	0.6147	0.6126	0.5966	0.6139	0.6113	0.6138	**0.9509**
0.3	10	0.9024	0.6271	0.6165	0.6219	0.6455	0.6248	0.6251	0.6222	**0.9737**

the matrix S_0 such that $ERR(P^*) \le ERR(P)$ for any partition P.

The haplotype assembly problem is to partition m fragments into two subsets. Thus, it can be mapped onto the SOCHN with $m \times 2$ neurons, where there are m groups of neurons and each group consists of two neurons. A group of neurons correspond to a fragment and the states of the neurons in a group determine which subset the fragment belongs to. Fig. 1 illustrates the mapping of the haplotype assembly problem onto the SOCHN. The number on the neuron denotes its output. For example, in Fig. 1 the first fragment is classified into the first subset, and the second and third fragments are classified into the second subsets.

The energy function based on the SOCHN for the haplotype assembly problem is defined as:

$$E(t) = \sum_{i=1}^{2} \sum_{r=1}^{m} v_{ri} \tilde{D}(f_r, h_i^{'}), \tag{10}$$

where f_r is the rth row of the matrix S_0 and $h_i^{'}$ is the consensus string obtained from the ith subset of S_0 according to a partition. Comparing the energy function (10) and the original energy function (3), we can get the connection weights and the thresholds of SOCHN for the haplotype assembly problem. Then they are substituted into the original input updating rule (4) and the input-output function (6). By referring to (7), we get the resulting input updating rule of SOCHN for the haplotype assembly problem as follows:

$$u_{ri}(t) = -2\alpha(z)\tilde{D}(f_r, h_i^{'}). \tag{11}$$

The resulting input-output function of the ith neuron in the rth group in the SOCHN for the haplotype assembly problem is as follows:

$$v_{ri}(t+1) = \begin{cases} 1, & \text{if } u_{ri}(t) = \max\limits_{j=1,\ldots,k} \{u_{rj}(t)\}. \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

In order to avoid the occurrence of empty subsets, a

hill-climbing term is added to (11). Then the final form of the input updating rule is:

$$u_{ri}(t) = -2\alpha(z)\tilde{D}(f_r, h_i') + C \cdot G(i), \quad (13)$$

where C is a big positive constant, and $G(i) = 1$ if the ith subset of S_0 is empty, otherwise, $G(i) = 0$.

3) Determining Haplotypes:

The SOCHN for the haplotype assembly problem outputs the final partition and the corresponding consensus strings. Let $h^*_1(C_1)$ and $h^*_2(C_2)$ denote the two consensus strings. Note that $h^*_1(C_1)$ and $h^*_2(C_2)$ does not contain the homozygous sites deleted at the preprocessing step. Now the homozygous sites are inserted into the consensus strings with the character '1' , such that the binary haplotypes $b\_H = (b\_h_1, b\_h_2)$ with length of n are obtained. The final haplotypes in terms of the original alphabet character $\hat{H} = (\hat{h}_1, \hat{h}_2)$ are determined as: If $b_h_1(q) = $ '1' , $\hat{h}_1(q) = MF(q,1)$; otherwise, $\hat{h}_1(q) = MF(q,2)$. If $b_h_2(q) = $ '1' , $\hat{h}_2(q) = MF(q,1)$; otherwise, $\hat{h}_2(q) = MF(q,2)$.

III. Results

A. Data Set and Measurement Criteria

The data set used to evaluate the proposed method is the one used in Geraci's research [22], which was created from real haplotypes of Phase 1 HapMap data [25]. The Geraci's data set consists of 22 pairs of human autosomes from four different populations and is produced by simulating shotgun sequencing algorithm. There are three parameters for the data set: haplotype length l = 100, 350, 700, error rate e = 0, 0.1, 0.2, 0.3 and coverage rate c = 3, 5, 8, 10. Each triplet of parameters $<l, e, c>$ contains 100 instances, and the results in the following subsection are the average values of 100 instances.

Let $H = (h_1, h_2)$ be the pair of real haplotypes and $\hat{H} = (\hat{h}_1, \hat{h}_2)$ be the pair of haplotypes returned by a haplotype assembly algorithm, where $\hat{h}_1$, $\hat{h}_2$, h_1 and h_2 have the length of n . The performance of the algorithm is measured by reconstruction rate RR , which is defined as:

$$RR_{\hat{H},H} = 1 - \frac{\min(D(h_1,\hat{h}_1) + D(h_2,\hat{h}_2), D(h_1,\hat{h}_2) + D(h_2,\hat{h}_1))}{2 \cdot n}, \quad (14)$$

where $D(X,Y) = \sum_{i=1}^{n} d(x_i, y_i)$, $d(x,y) = \begin{cases} 1, & \text{if } x \neq y \\ 0, & \text{otherwise} \end{cases}$, for two strings $X = (x_1, x_2, \cdots, x_n)$ and $Y = (y_1, y_2, \cdots, y_n)$.

B. Performance Comparison

We conducted several experiments on the same data set to evaluate the best termination criterion for SOCHN. Based on the experiment results, we consider the network to reach an equilibrium state if the energy of the SOCHN for the haplotype assembly problem remains unchanged for 100 iterations. In order to obtain proper evolution speed of the stochastic dynamics and avoid the occurrence of empty subsets, we set $\beta = 60$ and $C = 100$.

To evaluate the performance of our method HNHap, we compare it with the algorithms examined in [22] on the same benchmark. Tables I-III show the results for haplotype length of 100, 350 and 700, respectively. The first two columns in these tables represent the error rate e and the coverage rate c . The third column is the Baseline algorithm that can access the true fragment partition and simply reconstruct haplotypes by majority rule. Note that Baseline is not actually a true haplotype assembly algorithm and it provides a near-optimal solution. The other columns show the results of the seven algorithms which are taken from [22] and the results of our method HNHap. In the last columns of Tables I-III, the values identified in bold are the highest RR among all the algorithms except the baseline and the values in gray are the second highest RR .

TABLE IV
THE AVERAGE *RR* ON ALL THE DATA SETS, ON THE DATA SETS OF $e > 0$ AND ON THE DATA SETS OF $e > 0.1$.

	All	e>0	e>0.1
SPH	0.7939	0.7252	0.6208
FAST	0.8409	0.7885	0.7071
2d-mec	0.8256	0.7715	0.6945
HapCUT	0.839	0.7853	0.7239
MLF	0.8036	0.7605	0.6992
SHR	0.7054	0.6532	0.6179
DGS	0.8493	0.7991	0.7114
HNHap	**0.9291**	**0.9129**	**0.8859**

It can be observed in Table I that HNHap achieves the highest RR on all the data sets of $e > 0$. HNHap also gets good RR that is very close to the best result on the data sets of $e = 0$. It is clear that all the other algorithms lose much accuracy on the data sets of $e = 0.2$ and $e = 0.3$. However, our method HNHap achieves much higher RR than other algorithms especially on the data sets of $e = 0.3$. Although the RR of HNHap on the data sets of $c = 3$ drops a little more than that on the data sets of other cover rate as the error rate increases, it is still higher than other algorithms. Table II shows the comparison on data sets of $l = 350$. HNHap gets highest RR on all the data sets of $e > 0$ except only one data set. On the error-free data sets, HNHap also achieves RR that is close to the best one. Similar conclusion can be obtained from the results on the data sets of $l = 700$ in Table 3. It can be seen that the higher the error rate, the more significantly HNHap outperforms others. HNHap is designed to solve the haplotype assembly problem on the higher-error-rate data, so the network structure and parameters of SOCHN is not optimal for the error free data. As a result, HNHap does not perform best on the error free data. As can be seen in Tables I-III, HNHap even outperforms Baseline in some cases. In fact, Baseline is not an upper-bound of the accuracy of all the algorithms. It is included in [22] with simplified '0-1' SNP fragments as input, while HNHap handles raw SNP fragments.

We also compute the average RR on all the data sets, the average RR on the data sets of $e > 0$ and the average RR on the data sets of $e > 0.1$. Table IV shows the comparison of the average RR on different data sets. It can be seen that the highest average RR on all the data sets of other algorithms is 0.8493 that is gotten by DGS, while HNHap gets 0.9291. The

highest average RR for $e > 0$ is 0.7991 that is also given by DGS, while HNHap is 0.9129. The highest average RR for $e > 0.1$ is 0.7239 that is gotten by HapCUT, while HNHap is 0.8859. It is proven that our method is a powerful tool for the haplotype assembly problem, especially in cases with high error rate.

IV. Conclusion

Errors in the fragments makes great barrier to obtain accurate haplotypes for the haplotype assembly algorithms. Few algorithms could get satisfying reconstruction rate in cases with higher error rate. In this study, we map the haplotype assembly problem onto the stochastic optimal competitive Hopfield network model and design an energy function based on the MEC model. Experiment results show that the proposed method HNHap greatly outperforms other methods on data set where the noise in the data is higher. Considering that haplotype assembly from error-free data is trival and there have been many methods that are effective for high-quality data, the slight weakness of HGHap on high-quality data makes no difference. In the coming era of the advanced personalized medicine, the application of the Hopfield neural network in the haplotype assembly problem is meaningful and the proposed method HNHap is effective and promising.

References

[1] J. C. Venter, M. D. Adams, E. W. Myers, P. W. Li, R. J. Mural, G. G. Sutton, H. O. Smith, M. Yandell, C. A. Evans, and R. A. Holt, "The sequence of the human genome," *Science,* vol. 291, pp. 1304-1351, 2001.

[2] M. Wjst, "Target SNP selection in complex disease association studies," *BMC Bioinformatics,* vol. 5, pp. 92, 2004.

[3] J. C. Stephens, J. A. Schneider, D. A. Tanguay, J. Choi, T. Acharya, S. E. Stanley, R. Jiang, C. J. Messer, A. Chew, and J.-H. Han, "Haplotype variation and linkage disequilibrium in 313 human genes," *Science,* vol. 293, pp. 489-493, 2001.

[4] E. Halperin and E. Eskin, "Haplotype reconstruction from genotype data using imperfect phylogeny," *Bioinformatics,* vol. 20, pp. 1842-1849, 2004.

[5] M. Stephens, N. J. Smith, and P. Donnelly, "A new statistical method for haplotype reconstruction from population data," *The American Journal of Human Genetics,* vol. 68, pp. 978-989, 2001.

[6] L. Wang and Y. Xu, "Haplotype inference by maximum parsimony," *Bioinformatics,* vol. 19, pp. 1773-1780, 2003.

[7] G. Lancia, V. Bafna, S. Istrail, R. Lippert, and R. Schwartz, "SNPs Problems, Complexity, and Algorithms," in *Algorithms — ESA 2001*. vol. 2161, F. Heide, Ed., ed: Springer Berlin Heidelberg, 2001, pp. 182-193.

[8] R. Lippert, R. Schwartz, G. Lancia, and S. Istrail, "Algorithmic strategies for the single nucleotide polymorphism haplotype assembly problem," *Briefings in Bioinformatics,* vol. 3, pp. 23-31, 2002.

[9] J. Duitama, T. Huebsch, G. McEwen, E.-K. Suk, and M. R. Hoehe, "ReFHap: a reliable and fast algorithm for single individual haplotyping," presented at the Proceedings of the First ACM International Conference on Bioinformatics and Computational Biology, Niagara Falls, New York, 2010.

[10] R. S. Wang, L. Y. Wu, Z. P. Li, and X. S. Zhang, "Haplotype reconstruction from SNP fragments by minimum error correction," *Bioinformatics,* vol. 21, pp. 2456-2462, 2005.

[11] Y. Wang, E. Feng, and R. Wang, "A clustering algorithm based on two distance functions for MEC model," *Computational Biology and Chemistry,* vol. 31, pp. 148-150, 2007.

[12] S. Levy, G. Sutton, P. C. Ng, L. Feuk, A. L. Halpern, B. P. Walenz, N. Axelrod, J. Huang, E. F. Kirkness, G. Denisov, Y. Lin, J. R. MacDonald, A. W. C. Pang, M. Shago, T. B. Stockwell, A. Tsiamouri, V. Bafna, V. Bansal, S. A. Kravitz, D. A. Busam, K. Y. Beeson, T. C. McIntosh, K. A. Remington, J. F. Abril, J. Gill, J. Borman, Y.-H. Rogers, M. E. Frazier, S. W. Scherer, R. L. Strausberg, and J. C. Venter, "The Diploid Genome Sequence of an Individual Human," *PLoS Biol,* vol. 5, pp. 254, 2007.

[13] V. Bansal and V. Bafna, "HapCUT: an efficient and accurate algorithm for the haplotype assembly problem," *Bioinformatics,* vol. 24, pp. 153-159, 2008.

[14] T.-C. Wang, J. Taheri, and A. Y. Zomaya, "Using genetic algorithm in reconstructing single individual haplotype with minimum error correction," *Journal of Biomedical Informatics,* vol.45, pp. 922-930, 2012.

[15] A. Panconesi and M. Sozio, "Fast Hare: A Fast Heuristic for Single Individual SNP Haplotype Reconstruction Algorithms in Bioinformatics." vol. 3240, I. Jonassen and J. Kim, Eds., ed: Springer Berlin / Heidelberg, 2004, pp. 266-277.

[16] Y.-Y. Zhao, L.-Y. Wu, J.-H. Zhang, R.-S. Wang, and X.-S. Zhang, "Haplotype assembly from aligned weighted SNP fragments," *Computational Biology and Chemistry,* vol. 29, pp. 281-287, 2005.

[17] Z. Chen, B. Fu, R. Schweller, B. Yang, Z. Zhao, and B. Zhu, "Linear time probabilistic algorithms for the singular haplotype reconstruction problem from SNP fragments," *Journal of Computational Biology,* vol. 15, pp. 535-546, 2008.

[18] L. M. Genovese, F. Geraci, and M. Pellegrini, "SpeedHap: An Accurate Heuristic for the Single Individual SNP Haplotyping Problem with Many Gaps, High Reading Error Rate and Low Coverage," *IEEE/ACM Trans. Comput. Biol. Bioinformatics,* vol. 5, pp. 492-502, 2008.

[19] J. J. Hopfield and D. W. Tank, "Neural computation of decisions in optimization problems," *Biological Cybernetics,* vol. 52, pp. 141-152, 1985 1985.

[20] X. S. Xu, J. Ma, and J. H. Wang, "A Hopfield-Type Neural Network for Haplotype Assembly Problem," in Natural Computation, 2008. ICNC '08. Fourth International Conference on, 2008, pp. 8-12.

[21] J. Wang and Z. Tang, "An improved optimal competitive Hopfield network for bipartite subgraph problems," *Neurocomputing,* vol. 61, pp. 413-419, 2004.

[22] F. Geraci, "A comparison of several algorithms for the single individual SNP haplotyping reconstruction problem," *Bioinformatics,* vol. 26, pp. 2217-2225, 2010.

[23] G. Galan-Marin and J. Munoz-Perez, "Design and analysis of maximum Hopfield networks," *IEEE Transactions on Neural Networks,* vol. 12, pp. 329-339, Mar 2001.

[24] G. Galan-Marin, E. Merida-Casermeiro, and J. Munoz-Perez, "Modelling competitive Hopfield networks for the maximum clique problem," *Computers & Operations Research,* vol. 30, pp. 603-624, Apr 2003.

[25] T. I. H. Consortium, "A haplotype map of the human genome," *Nature,* vol. 437, pp. 1299-1320, 2005.