Parallel Computation for Chromosome Reconstruction on a Cluster of Workstations

S.M. Bhandarkar S.A. Machaka S.S. Shete J. Arnold
Department of Computer Science Department of Statistics Department of Genetics
The University of Georgia
Athens, Georgia 30602, USA

Abstract

Reconstructing a physical map of a chromosome from a genomic library presents a central computational problem in genetics. Physical map reconstruction in the presence of errors is a problem of high computational complexity which provides the motivation for parallel computing. Parallelization strategies for a maximum likelihood estimation-based approach to physical map reconstruction are presented. The estimation procedure entails gradient descent search for determining the optimal spacings between probes for a given probe ordering. The optimal probe ordering is determined using a stochastic optimization algorithm. A two-tier parallelization strategy is proposed wherein the gradient descent search is parallelized at the lower level and the stochastic optimization algorithm is simultaneously parallelized at the higher level. Implementation and experimental results on a distributed-memory multiprocessor cluster running the Parallel Virtual Machine (PVM) environment are presented.

1 Introduction

Generation of entire chromosomal maps is a central problem in genetics. Chromosomal maps fall into two broad categories - *genetic maps* and *physical maps*. Genetic maps are typically of low resolution (1–10 million base pairs (Mb)) and represent an ordering of genetic markers along a chromosome where the distance between two genetic markers is inversely proportional to their recombination frequency. A physical map is an ordering of distinguishable (i.e., sequenced) DNA fragments called *clones* or *contigs* by their position along the entire chromosome where the clones may or may not contain genetic markers. A physical map has a much higher resolution (10–100 thousand base pairs (Kb)) than a genetic map of the same chromosome. While genetic maps enable a scientist to narrow the search for genes to a particular chromosomal region, it is a physical map that ultimately allows the recovery and molecular manipulation of genes of interest.

The physical mapping protocol essentially determines the nature of clonal data and the probe selection procedure. The physical mapping protocol used in this project is the one based on *sampling without replacement* [3]. Under this protocol, a maximal set $\mathcal{P}$ of non-overlapping equal-length clones from a library is selected as the probe set. The remaining clones $\mathcal{C}$ in the library are hybridized to the probe set resulting in a digital hybridization signature for each clone. The clone-probe overlap pattern is represented by a binary hybridization matrix H where $H_{ij} = 1$ if the ith clone hybridizes to the jth probe and $H_{ij} = 0$ otherwise. If the probes in $\mathcal{P}$ are ordered with respect to their position along a chromosome, then by selecting from H a common overlapping clone for each pair of adjacent probes, a minimal set of clones and probes that covers the entire chromosome (i.e., a minimal tiling) can be obtained. The minimal tiling in conjunction with the sequencing of each individual clone/probe in the tiling and a sequence assembly procedure that determines the overlaps between successive sequenced clones/probes in the tiling [9] can then be used to reconstruct the DNA sequence of the entire chromosome. In reality, H could be expected to contain false positives and false negatives. H_{ij} would be a false positive if $H_{ij} = 1$ when in fact $H_{ij} = 0$. Conversely, H_{ij} would be a false negative if $H_{ij} = 0$ when in fact $H_{ij} = 1$. In this paper, we confine ourselves to errors in the form of false positives and false negatives.

In this paper we describe a maximum likelihood (ML) estimator proposed in [10, 13] which determines the ordering of probes in the probe set $\mathcal{P}$ and also the inter-probe spacings under a probabilistic model of hybridization errors consisting of false positives and false negatives. The estimation procedure involves a combination of discrete and continuous optimization

where determining the probe ordering entails discrete (i.e., combinatorial) optimization whereas determining the inter-probe spacings for a particular probe ordering entails continuous optimization. We propose a two-tier parallelization strategy for efficient implementation of the above estimator. The upper-level comprises of parallel discrete optimization using simulated annealing or microcanonical annealing whereas the lower-level comprises of parallel conjugate gradient descent. The resulting parallel algorithms are implemented on a distributed-memory multiprocessor cluster using the Parallel Virtual Machine (PVM) environment [5, 14]. Convergence, speedup and scalability characteristics of the parallel algorithms are analyzed and discussed.

2 Mathematical Formulation of the ML Estimator

The ML estimator reconstructs the ordering of probes in the probe set $\mathcal{P}$ and the inter-probe spacings under a probabilistic model of hybridization errors consisting of false positives and false negatives. The probe ordering problem can be formally stated as follows. Given a set $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$ of n probes and a set $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ of k clones generated using the sampling-without-replacement protocol and the $k \times n$ clone-probe hybridization matrix H containing both false positives and false negatives with predefined probabilities, reconstruct the correct ordering $\Pi = (\pi_1, \pi_2, \ldots, \pi_n)$ of the probes and also the correct spacing $Y = (Y_1, Y_2, \ldots, Y_n)$ between the probes. The ordering Π is a permutation of $(1, \ldots, n)$ that gives the labels (indices) of the probes in left-to-right order across the chromosome. In the inter-probe spacing vector Y, Y_1 denotes the space between the left end of the first probe P_{π_1} and the left end of the chromosome, and Y_i the spacing between the right end of probe $P_{\pi_{i-1}}$ and and the left end of probe P_{π_i} (where $2 \leq i \leq n$). The spacing between the right end of probe P_{π_n} and the right end of the chromosome is given by $Y_{n+1} = N - nM - \sum_{i=1}^{n} Y_i$ where N is length of the chromosome and M is the length of each clone/probe. Recall that our protocol requires that all probes and clones be of the same length.

The problem as stated above is ill-posed since the underlying constraints do not imply a unique solution. Hence the problem is formulated as one of determining a probe ordering and the inter-probe spacings that maximize the likelihood of the observed hybridization matrix H given predefined probabilities for false positives and false negatives.

2.1 Mathematical Notation

The mathematical notation used in the formulation of the ML estimator is given below:
N : Length of the chromosome,
M : Length of a clone/probe,
n : Number of probes,
k : Number of clones,
ρ : Probability of false positive,
η : Probability of false negative,
$H = ((h_{i,j}))_{1 \leq i \leq k,\, 1 \leq j \leq n}$: clone-probe hybridization matrix,
where
$h_{i,j} = \begin{cases} 1 & \text{if clone } C_i \text{ hybridizes with probe } P_j \\ 0 & \text{otherwise,} \end{cases}$
H_i : ith row of the hybridization matrix,
$\Pi = (\pi_1, \ldots, \pi_n)$: permutation of $\{1, 2, \ldots, n\}$ which denotes the probe labels in the ordering when scanned from left to right along the chromosome,
$p_i = \sum_{j=1}^{n} h_{i,j}$: number of 1's in H_i,
$P = \sum_{i=1}^{k} p_i$: total number of 1's in H, and
$Y = (Y_1, Y_2, \ldots, Y_n)$: vector of inter-clone spacings where Y_i is the spacing between the right end of $P_{\pi_{i-1}}$ and the left end of P_{π_i} $(2 \leq i \leq n)$, and Y_1 is the spacing between the left end of P_{π_1} and the left end of the chromosome.

2.2 The ML Model

Given a vector of inter-probe spacings $Y = (Y_1, \ldots, Y_n)$, there are 2^{n+1} possible cases to consider depending on whether $0 \leq Y_i \leq M$ or $Y_i > M$ where $0 \leq i \leq n+1$. It can be shown that the 2^{n+1} cases can be analyzed based on the clone–probe overlap pattern [13]. In general, the clone-probe overlap pattern results in three different types of regions namely,
Type 1: The *Both* region $R_B(P_{\pi_j}, P_{\pi_{j+1}})$ between probes P_{π_j} and $P_{\pi_{j+1}}$, for $j = 1, \ldots, n-1$. An intervening clone hybridizes to both probes if its left end falls in this region.
Type 2: The *Only* region $R_O(P_{\pi_j})$ of probe P_{π_j}, for $j = 1, \ldots, n$. A clone will hybridize to P_{π_j} only if its left end falls in this region.
Type 3: The *None* region $R_N(P_{\pi_j})$ after probe P_{π_j}, for $j = 0, \ldots, n$. A clone will hybridize to no probe if its left end falls in this region. Here probe P_{π_0} denotes the beginning of the chromosome.
Let $l(R)$ denote the length of region R. It can be shown that for $j = 1, \ldots, n-1$, $l(R_B(P_{\pi_j}, P_{\pi_{j+1}})) = M - \min(Y_{j+1}, M)$, and for $j = 1, \ldots, n$, $l(R_O(P_{\pi_j})) = \min(Y_j, M) + \min(Y_{j+1}, M)$, and for $j = 0, \ldots, n$,

$l(R_N(P_{\pi_j})) = Y_{j+1} - \min(Y_{j+1}, M)$. We assume that the left ends of the clones are uniformly distributed over the interval $[0, N-M]$. Therefore it can be shown that for $j = 1, \ldots, n-1$, the probability P_{Both} that a randomly chosen clone will fall in the region $R_B(P_{\pi_j}, P_{\pi_{j+1}})$ is given by $P_{Both} = \frac{M - \min(Y_{j+1}, M)}{N-M}$, for $j = 1, \ldots, n$ the probability P_{Only} that a randomly chosen clone will fall in the region $R_O(P_{\pi_j})$ is given by $P_{Only} = \frac{\min(Y_j, M) + \min(Y_{j+1}, M)}{N-M}$, and for $j = 0, \ldots, n$ the probability P_{None} that a randomly chosen clone will fall in the region $R_N(P_{\pi_j})$ is given by $P_{None} = \frac{Y_{j+1} - \min(Y_{j+1}, M)}{N-M}$ [13].

Let $O_{i,j}$ be the event that the clone i will fall in the region $R_O(P_{\pi_j})$; $B_{i,j}$ the event that the clone i will fall in the region $R_B(P_{\pi_j}, P_{\pi_{j+1}})$ and $N_{i,j}$ the event that the clone i will fall in the region $R_N(P_{\pi_j})$. Then the conditional probability of observing a clonal signature H_i (i.e., the ith row in H) given a probe ordering Π and an inter-probe spacing vector Y is given by

$$\begin{aligned} P(H_i \mid \Pi, Y) = & \\ & \sum_{j=1}^{n} P(H_i \mid \Pi, Y, O_{i,j}) P(O_{i,j} \mid \Pi, Y) + \\ & \sum_{j=1}^{n-1} P(H_i \mid \Pi, Y, B_{i,j}) P(B_{i,j} \mid \Pi, Y) + \\ & \sum_{j=0}^{n} P(H_i \mid \Pi, Y, N_{i,j}) P(N_{i,j} \mid \Pi, Y) \end{aligned} \quad (1)$$

Given Π, Y and $O_{i,j}$, implies that only $h_{i,\pi_j} = 1$ and all the remaining entries in row H_i should be $= 0$. In other words, $h_{i,\pi_j} \neq 1$ implies a false negative and a 1 in any other column position in the row H_i implies a false positive. That is,

$$h_{i,\pi_j} = \begin{cases} 0 & \text{with probability } \eta \\ 1 & \text{with probability } (1-\eta) \end{cases} \quad (2)$$

and for $k = 1, \ldots, n$ where $k \neq j$

$$h_{i,\pi_k} = \begin{cases} 0 & \text{with probability } (1-\rho) \\ 1 & \text{with probability } \rho. \end{cases} \quad (3)$$

We assume that the false positive and false negative errors at different positions along the clonal signature H_i are independent of each other. Hence $P(H_i \mid \Pi, Y, O_{i,j}) = (1-\eta)^{h_{i,\pi_j}} \cdot \eta^{(1-h_{i,\pi_j})} \cdot \rho^{(p_i - h_{i,\pi_j})} \cdot (1-\rho)^{(n-1)-(p_i - h_{i,\pi_j})}$ Following the same argument we can show that $P(H_i \mid \Pi, Y, B_{i,j}) = (1-\eta)^{(h_{i,\pi_j} + h_{i,\pi_{j+1}})} \cdot \eta^{(2-h_{i,\pi_j} - h_{i,\pi_{j+1}})} \cdot \rho^{(p_i - h_{i,\pi_j} - h_{i,\pi_{j+1}})} \cdot (1-\rho)^{(n-2)-(p_i - h_{i,\pi_j} - h_{i,\pi_{j+1}})}$ and $P(H_i \mid \Pi, Y, N_{i,j}) = \rho^{p_i} \cdot (1-\rho)^{(n-p_i)}$. Hence we get,

$$\begin{aligned} P(H_i \mid \Pi, Y) = & \sum_{j=1}^{n} \Big[(1-\eta)^{h_{i,\pi_j}} \cdot \eta^{(1-h_{i,\pi_j})} \cdot \\ & \rho^{(p_i - h_{i,\pi_j})} \cdot (1-\rho)^{(n-1)-(p_i - h_{i,\pi_j})} \cdot \\ & \frac{\min(Y_j, M) + \min(Y_{j+1}, M)}{N-M} \Big] + \\ & \sum_{j=1}^{n-1} \Big[(1-\eta)^{(h_{i,\pi_j} + h_{i,\pi_{j+1}})} \cdot \eta^{(2-h_{i,\pi_j} - h_{i,\pi_{j+1}})} \cdot \\ & \rho^{(p_i - h_{i,\pi_j} - h_{i,\pi_{j+1}})} \cdot \frac{M - \min(Y_{j+1}, M)}{N-M} \Big] + \\ & \sum_{j=0}^{n} \Big[\rho^{p_i} \cdot (1-\rho)^{(n-p_i)} \cdot \frac{Y_{j+1} - \min(Y_{j+1}, M)}{N-M} \Big] \end{aligned} \quad (4)$$

We assume that the clones $\in \mathcal{C}$ are independently distributed along the chromosome i.e., each row of H is independent of the other rows. Hence $P(H \mid \Pi, Y) = \prod_{i=1}^{k} P(H_i \mid \Pi, Y)$ which gives us

$$\begin{aligned} P(H \mid \Pi, Y) = \prod_{i=1}^{k} C_i \{ R_i - & \\ & \sum_{j=1}^{n+1} (a_{i,\pi_j} - 1)(a_{i,\pi_{j-1}} - 1) min(Y_j, M) \Big\} \end{aligned} \quad (5)$$

where

$$a_{i,j} = \begin{cases} \frac{\eta}{(1-\rho)} & \text{if } h_{i,j} = 0 \text{ and } j = 1, \ldots, n \\ \frac{(1-\eta)}{\rho} & \text{if } h_{i,j} = 1 \text{ and } j = 1, \ldots, n \\ 0 & \text{otherwise,} \end{cases} \quad (6)$$

$C_i = \frac{\rho^{p_i}(1-\rho)^{(n-p_i)}}{N-M}$, and $R_i = N - nM + M \sum_{j=1}^{(n-1)} a_{i,\pi_j} a_{i,\pi_{j+1}}$.

The goal therefore is to determine Π and Y that maximize $P(H \mid \Pi, Y)$ as given in equation (5), that is determine $(\hat{\Pi}, \hat{Y})$ where $(\hat{\Pi}, \hat{Y}) = \arg\max_{(\Pi, Y)} P(H \mid \Pi, Y)$. Alternatively we could consider the negative log-likelihood function $f(\Pi, Y)$ given by $f(\Pi, Y) = -\ln P(H \mid \Pi, Y)$. Since $\ln x$ is a monotonically increasing function of x for all $x > 0$, it follows that $(\hat{\Pi}, \hat{Y}) = \arg\max_{(\Pi, Y)} P(H \mid \Pi, Y) = \arg\min_{(\Pi, Y)} f(\Pi, Y)$.

2.3 Computation of the ML Estimate

Computing the values of $\hat{\Pi}$ and $\hat{Y}$ involves a two stage procedure:

Stage 1: We first determine the optimal spacing $\hat{Y}_\Pi$ for a given probe ordering Π i.e., determine $\hat{Y}_\Pi = (\hat{Y}_1, \ldots, \hat{Y}_n)$ such that for a given Π, $f(\Pi, \hat{Y}_\Pi) = \min_Y f(\Pi, Y) = \min_Y f_\Pi(Y)$. Here the minimum is taken over all feasible solutions Y that satisfy the constraints $Y_i \geq 0$; $i = 1, \ldots, n$ and $\sum_{i=1}^{n} Y_i \leq N - nM$.
Stage 2: We determine $\hat{\Pi}$ for which, $f(\hat{\Pi}, \hat{Y}_{\hat{\Pi}}) = \min_\Pi f(\Pi, \hat{Y}_\Pi) = \min_\Pi f_{\hat{Y}_\Pi}(\Pi)$. Here the minimum is taken over all Π where Π is a permutation of $\{1, \ldots, n\}$. The resulting values of $\hat{\Pi}$ and $\hat{Y}_{\hat{\Pi}}$ are termed the ML estimates (MLEs) of the true probe ordering and the inter-probe spacings, respectively.

2.3.1 Computation of $\hat{Y}_\Pi$

It can be shown that $f_\Pi(Y)$ is convex in $\mathcal{D}$ and therefore possesses a unique local minimum which is also a global minimum [13]. Consequently this minimum can be reached using continuous local search-based techniques such as the steepest descent search [7]. The steepest descent search is a simple iterative procedure which consists of three steps: (i) Determine the initial value of Y, (ii) Compute the downhill gradient at Y and (iii) Update the current value of Y using the computed value of the downhill gradient. Steps (ii) and (iii) are repeated until the gradient vanishes, or in practice, until the gradient magnitude is less than a prespecified threshold. The local downhill gradient is given by $-\nabla f(\Pi, \hat{Y}) = -(\frac{\partial f(\Pi,Y)}{\partial Y_1}, \ldots, \frac{\partial f(\Pi,Y)}{\partial Y_n}) \mid_{Y=\hat{Y}} = (U_1, \ldots, U_n) \mid_{Y=\hat{Y}} = U \mid_{Y=\hat{Y}}$. The current value of $\hat{Y} = \hat{Y}_{old}$ is updated by moving along the downhill gradient direction U. The new value of $\hat{Y} = \hat{Y}_{new}$ is given by $\hat{Y}_{new} = \hat{Y}_{old} + sU$. The problem, therefore, is to find an optimal value of s, say s^* such that $f(\Pi, \hat{Y} + s^*U) = \min_s f(\Pi, \hat{Y} + sU)$. Having obtained the value of s^*, then the new inter-probe spacings are given by $\hat{Y}_{new} = \hat{Y}_{old} + s^*U$.

To determine an optimal value of $s = s^*$ we exploit the convexity of $f_\Pi(Y)$ which implies that the local optimum for s is also a global optimum. Using the constraints that the spacings are non-negative, the clones and probes are of fixed length and the total length of the chromosome is fixed, we compute the upper and lower bounds on the values of s and use the bisection method to find the optimal value of $s = s^*$. If any of the boundary conditions (represented as hyperplanes) on the Y_i's for $i = 1, \ldots, n$ are violated, the gradient vector U is projected onto the admissible region which is represented as the intersection of the k hyperplanes corresponding to the k violated constraints. The minimization procedure then proceeds along the projected gradient direction U_{proj} instead of U. In the limiting case when $k = n$, the minimization procedure has reached an extremal vertex of the admissible region and $U_{proj} = 0$. In this case, the extremal vertex is the desired minimum within the admissible region. Thus the minimization procedure is halted when U vanishes or when an extremal vertex is reached (i.e., U_{proj} vanishes) depending on which situation is encountered first.

2.3.2 Computation of $\hat{\Pi}$

Determining the optimal clone ordering $\hat{\Pi}$, entails a combinatorial search through the discrete space of all possible permutations of $\{1, \ldots, n\}$. The problem of coming up with such an optimal ordering is isomorphic to the classical NP-complete *Optimal Linear Arrangement* (OLA) problem for which no polynomial-time algorithm for determining the optimal solution is known [4]. One could use a stochastic hill-climbing search algorithm such as simulated annealing (SA) [6] or microcanonical annealing (MCA) [2] both of which are known to be robust in the presence of local optima in the solution space and give near-optimal solutions in average polynomial time.

A single iteration of the SA or MCA algorithm consists of three phases: (i) perturb, (ii) evaluate, and (iii) decide. In the perturb phase, the probe ordering is systematically perturbed by reversing the ordering within a block of probes where the endpoints of the block are chosen at random. In the evaluate phase, $f(\Pi, \hat{Y}_\Pi)$ is computed. In the decide phase, the new probe ordering is accepted and replaces the current probe ordering *probabilistically* using a stochastic decision function. After several iterations at a particular value of temperature or kinetic energy (termed as an annealing step), the stochastic decision function is *annealed* in a manner such that the optimization process resembles a random search in the earlier stages and a greedy local search or a deterministic hill-climbing search in the latter stages. The Metropolis decision function [11] or the Boltzmann decision function [1] are used in SA whereas the kinetic energy value is used in MCA. Both SA and MCA, starting from an initial solution, generate in the limit, an ergodic Markov chain of solution states which asymptotically converges to a stationary Boltzmann distribution [1]. The Boltzmann distribution asymptotically converges to a globally optimal solution when subject to the annealing process [6]. The annealing schedule needed for asymptotic convergence is computationally intensive. This provides the motivation for the parallel computation of the ML estimator. We refer the interested reader to [6] and [2] for a more in-depth treatment of

SA and MCA.

3 Parallel Computation of the ML Estimator

We propose a two-tier parallel computation of the ML estimator corresponding to the two stages of optimization.

Level 1: Parallel computation of the optimal inter-probe spacing $\hat{Y}_{\Pi}$ for a given probe ordering Π that minimizes $f(\Pi, \hat{Y}_{\Pi})$. This entails parallelization of the gradient descent search procedure for constrained optimization in the *continuous* domain.

Level 2: Parallel computation of the optimal probe ordering $\hat{\Pi}$ for which $f(\hat{\Pi}, \hat{Y}_{\hat{\Pi}})$ is minimum. This entails parallelization of the stochastic hill-climbing search procedure (SA or MCA) for optimization in the *discrete* domain.

Both levels of parallel computation were implemented on the Parallel Virtual Machine (PVM) [14] which is based on a distributed-memory message-passing paradigm of parallel computing. We refer the interested reader to [5] for a more detailed description of PVM.

3.1 Parallel Stochastic Hill-Climbing Search

We have formulated and implemented two models of parallel SA (PSA) and parallel MCA (PMCA) algorithms based on the distribution of the Markov chain of solution states on a workstation cluster running PVM. These models incorporate control parallelism with multiple interacting or non-interacting searches of the solution space and are described below:

(i) The Non-Interacting Local Markov chain (NILM) PSA and PMCA algorithms.

(ii) The Periodically Interacting Local Markov chain (PILM) PSA and PMCA algorithms.

In the NILM PSA/PMCA algorithms, each processor runs an independent version of the serial SA/MCA algorithm. Each Markov chain of solution states is local to a given processor. The SA/MCA algorithms run concurrently but asynchronously on each processor. The evaluation function and the decision function are executed concurrently on the solution state within each processor. On termination of the annealing processes on all the processors, the best solution is selected from among all the solutions available on the individual processors. The NILM model is essentially that of multiple independent (i.e., noninteracting) searches.

The PILM PSA/PMCA algorithms are similar to their NILM counterparts except for the fact that just before the temperature parameter or the kinetic energy parameter is updated using the annealing function, the best candidate solution from among those in all the processors is selected and duplicated on all the processors. This focuses the search in the more promising regions of the solution space. The PILM model is essentially that of multiple periodically interacting searches.

In the case of all the above PSA/PMCA algorithms, a master process is used as the overall controlling process. The master process runs on one of the processors and spawns child processes on each processor within the PVM system, broadcasts the data subsets needed by each child process, collects the final results from each child process and terminates the child processes. In the case of the PILM PSA/PMCA algorithms, at each annealing step, the master process collects the results from each child process and broadcasts the best result to all the child processes. On convergence, the master process collects the final results from each of the child processes, selects the best result as the final solution and terminates the child processes.

Each child process in the PILM PSA/PMCA algorithm receives the initial parameters from the master process and runs its local version of the SA/MCA algorithm. At the end of each annealing step each child process conveys its result to the master process, receives the best result thus far from the master process and replaces its result with the best result thus far before proceeding with the next annealing step. On convergence each child process conveys its result to the master process. The master and child processes for the NILM PSA/PMCA algorithms are similar to those of their PILM counterparts except for the absence of the periodic interaction at the end of each annealing step.

3.2 Parallel Gradient Descent Search

Steepest descent search and conjugate gradient descent (CGD) search are generally used for unconstrained optimization in the continuous domain. The steepest descent search, in our case, has been adapted to the fact that the solution space of the inter-probe spacings is constrained since $0 \leq Y_i \leq M$ for $i = 1, \ldots, n$. We have used the CGD search instead of the steepest descent search since the former is known to be one of the fastest in the class of gradient descent-based optimization methods [7].

The CGD search is very similar to the steepest descent procedure with the only difference that different

directions are followed while minimizing the objective function. Instead of consistently following the local downhill gradient direction, a set of n mutually orthonormal (i.e., conjugate) direction vectors are generated from the downhill gradient vector where n is the dimensionality of the solution space [12]. Unlike the steepest descent algorithm, the CGD algorithm guarantees convergence to a local minimum within n steps.

Due to its inherent sequential nature, we deemed data parallelism to be appropriate for the parallel CGD algorithm. The Y and U vectors are distributed amongst the different processors and each processor performs the gradient vector computation and updates to the inter-probe spacing vector using its local subvectors Y_{loc} and U_{loc} concurrently with the other processors. Here, $|Y_{loc}| = |Y|/N_{proc}$ and $|U_{loc}| = |U|/N_{proc}$ where N_{proc} is the number of processors in the virtual machine. This scheme entails inter-processor communication and synchronization overhead since the individual subvectors have to be periodically scattered amongst the processors and also periodically gathered to compute a global value for s during the bisection procedure. The parallelization scheme follows the master-child model used for the PSA/PMCA algorithms.

3.3 A Two-tier Parallelization of the ML Estimator

In order to ensure a scalable implementation, two tiers of parallelism were incorporated in the computation of the ML estimator. The finer or lower level of parallelism pertains to the computation of $\hat{Y}$ for a given probe ordering Π using the parallel CGD algorithm for continuous optimization. The coarser or upper level of parallelization pertains to the computation of $\hat{\Pi}$ using a stochastic hill-climbing algorithm for discrete stochastic optimization.

At the coarser level, the user has a choice of using any of the four parallel stochastic hill-climbing algorithms: PILM PSA, NILM PSA, PILM PMCA or NILM PMCA. The parallelization of the CGD algorithm at the finer level is transparent to the coarser level. A parallel CGD algorithm is embedded within each of the stochastic hill-climbing processes. When the parallel CGD procedure is invoked from within the master or child stochastic hill-climbing process, a new set of child CGD processes is spawned on the available processors, whereas the master CGD process runs on the same processor as the stochastic hill-climbing process (master or child). The master and child CGD processes cooperate to evaluate and minimize the value of $f(\Pi, \hat{Y}_{\Pi})$. Once $f(\Pi, \hat{Y}_{\Pi})$ is minimized, the child CGD processes terminate and the corresponding processors are available for future computation. The two-tier parallelism approach can be seen to induce a logical tree-shaped interconnection network on the processors in the PVM system.

Table 1: Specifications of the artificially generated clone-probe hybridization data

Data Set	n	k	N	M	ρ	η
Data Set 1	10	100	180	15	2%	2%
Data Set 2	30	400	680	20	2%	2%

4 Experimental Results

The parallel algorithms were implemented on a dedicated PVM cluster of 200MHz PentiumPro processors running Solaris-x86 and tested with artificially generated clone-probe hybridization data [13]. Two sets of artificial data were used with the specifications outlined in Table 1.

The serial SA and MCA algorithms were implemented with following parameters: the initial value for the temperature or demon energy was chosen to be 0.5, the maximum number of iterations D for each annealing step was chosen to be $100 \cdot n$. The current annealing step was terminated when the maximum number of iterations was reached or when the number of successful perturbations equaled $10 \cdot n$ whichever was encountered first. The temperature or demon energy values were systematically reduced using a geometric annealing schedule with the annealing factor $\alpha = 0.95$. The algorithm was terminated when the number of successful perturbations in any annealing step equaled 0.

In the case of the parallel stochastic hill-climbing algorithms the product of N_{proc} and the maximum number of iterations D performed by a processor in a single annealing step was kept constant i.e., $D = (100 \cdot n)/N_{proc}$. This ensured that the overall workload remained constant as the number of processors was varied, thus enabling one to examine the scalability of the speedup and efficiency of the algorithms for a given problem size with increasing number of processors. The other parameters for the parallel stochastic hill-climbing algorithms were identical to those of their serial counterparts. In the NILM PSA/PMCA algorithms, each process was independently terminated

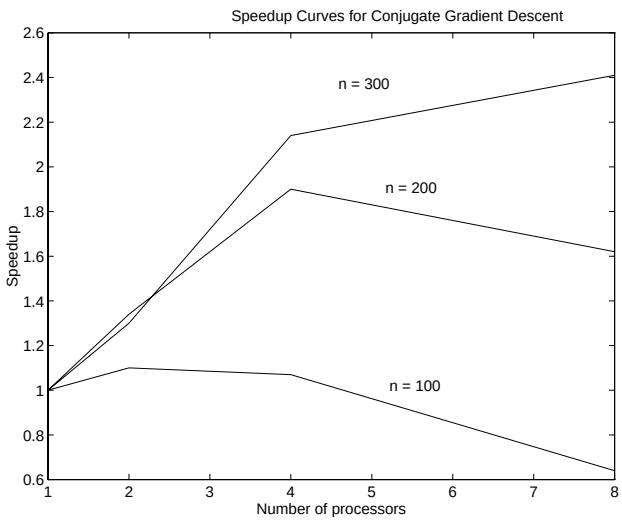

Figure 1: Speedup curves for the parallel CGD algorithm

when the number of successful perturbations in any annealing step for that process equaled 0. In the PILM PSA/PMCA algorithms, each process was terminated when the number of successful perturbations in an annealing step equaled 0 for *all* the processes. This condition was checked during the synchronization phase at the end of each annealing step.

The parallel CGD algorithm was tested on artificial data sets with a varying number of probes $n = 100$, 200 and 300. Figure 1 shows the resulting speedup curves. These results are in conformity with our expectations since the inter-processor communication overhead and synchronization overhead tend to increasingly dominate the overall execution time with increasing N_{proc} values for a given value of n. The payoff in the parallelization of the CGD algorithm is realized only for large values of n (i.e., large problem sizes). This is a natural consequence of the network latency inherent in PVM systems that are comprised of a network of workstations.

In the case of the PSA and PMCA algorithms, we experimented with problem sizes of $n = 10$ and $n = 30$ probes. Since the value of n was small the serial version of the CGD algorithm was used. The speedup curves for $n = 10$ and $n = 30$ are shown in Figure 2. As can be observed, the PSA and PMCA algorithms exhibit consistent and scalable speedup with increasing N_{proc} values. As expected, the speedup scales better with increasing N_{proc} values for larger values of n. The PSA and PMCA algorithms arrived at the correct probe ordering in all cases but for one exception where the reverse probe ordering was obtained. Since

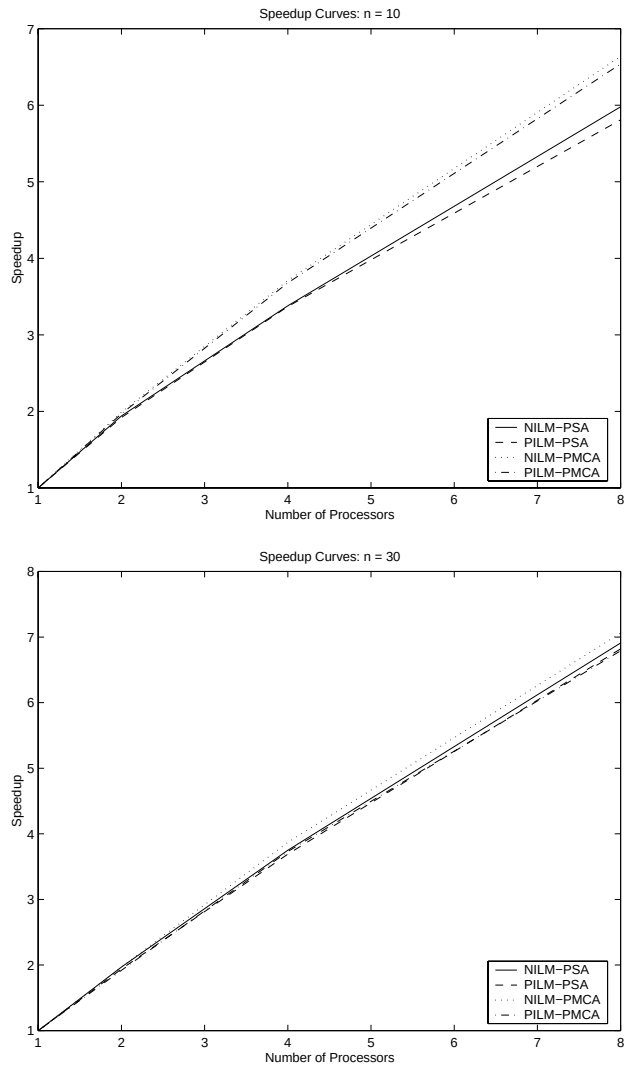

Figure 2: Speedup curves for the parallel stochastic hill-climbing algorithms for $n = 10$ and $n = 30$.

the likelihood function is unique only up to reversal in the probe ordering, the ML estimation procedure is capable of recovering the correct probe ordering only up to reversal.

The absolute root mean squared error (RMSE) χ between the true inter-probe spacings Y and the estimated inter-probe spacings $\hat{Y}$ is defined as $\chi = \sqrt{\frac{|Y-\hat{Y}|^2}{n}}$. The RMSE value is typically expressed as a percentage of N (the chromosome length). In our experiments, the percent RMSE value was observed to lie in the range $[0.34\%, 2.63\%]$. The percent RMSE value was also observed to asymptotically approach 0 in the limit $n \rightarrow \infty$, which is in conformity with the statistical theory underlying ML estimation [8].

5 Conclusions and Future Directions

In this paper we presented a ML estimation-based approach to physical map reconstruction under a probabilistic model of hybridization errors consisting of false positives and false negatives. The ML estimate reconstructs the optimal probe ordering and optimal inter-probe spacings when used in conjunction with the sampling-without-replacement experimental protocol. The estimation procedure was shown to entail continuous optimization for determining the optimal inter-probe spacings for a given probe ordering and combinatorial optimization for determining the optimal probe ordering. A two-tier parallelization strategy was proposed wherein the CGD search algorithm for continuous optimization is parallelized at the lower level and the SA or MCA algorithm for combinatorial optimization is simultaneously parallelized at the higher level. Experimental results on a PVM cluster showed that the payoff in data parallelization of the CGD procedure was realized only for large problem sizes. A similar trend was observed in the case of the parallel SA and MCA algorithms.

Future research will investigate extensions of the ML function that also encapsulate errors due to repeat DNA sequences in addition to false positives and false negatives. The current PVM implementation of the ML estimator is targeted towards a homogeneous distributed processing platform such as a network of identical workstations. Future research will explore and address issues that deal with the parallelization of the ML estimator on a heterogeneous distributed processing platform such as a network of workstations that differ in processing speeds.

Acknowledgments: This research was supported in part by an NRICGP grant by the US Department of Agriculture to Dr. Bhandarkar and Dr. Arnold.

References

[1] E.H.L. Aarts and K. Korst, *Simulated Annealing and Boltzman Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*, Wiley, New York, 1989.

[2] M. Creutz, Microcanonical Monte Carlo Simulation, *Physics Review Letters*, Vol. 50, No. 19, pp. 1411–1414, 1983.

[3] Y.X. Fu, W.E. Timberlake and J. Arnold, On the Design of Genome Mapping Experiments using Short Synthetic Oligonucleotides. *Biometrics*, Vol. 48, pp. 337–359, 1992.

[4] M.S. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP–Completeness*, W.H. Freeman, New York, NY, 1979.

[5] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Mancheck, and V. Sunderam, *PVM Parallel Virtual Machine – A User's Guide and Tutorial for Networked Parallel Computing*, MIT Press, Cambridge, MA, 1994.

[6] S. Geman and D. Geman, Stochastic Relaxation, Gibbs Distribution and the Bayesian Restoration of Images, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 6, pp. 721–741, 1984.

[7] M. Hestenes and E. Stiefel, Methods of Conjugate Gradients for Solving Linear Systems. *Journal of Research of the National Bureau of Standards*, Vol. 49, pp. 409–436, 1980.

[8] R.V. Hogg and A.T. Craig, *Introduction to Mathematical Statistics*, Fifth Edition, Prentice Hall, New Jersey, 1995.

[9] J.D. Kececioglu and E.W. Myers, Combinatorial Algorithms for DNA Sequence Assembly, *Algorithmica*, Vol. 13, pp. 7–51, 1995.

[10] J.D. Kececioglu, S.S. Shete and J. Arnold, Reconstructing Distances in Physical Maps of Chromosomes With Nonoverlapping Probes, *Proc. 4th ACM Conf. Comp. Mol. Biol. (RECOMB)*, April 2000, Tokyo, Japan, to appear.

[11] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, Equation of state calculations by fast computing machines, *Jour. Chemical Physics*, Vol. 21, pp. 1087–1092, 1953.

[12] W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, *Numerical Recipes in C*, Cambridge University Press, New York, 1988.

[13] S.S. Shete, *Estimation Problems in Physical Mapping of a Chromosome and in a Branching Process with Immigration*, Ph.D. Dissertation, Department of Statistics, University of Georgia, Athens, Georgia, August 1998.

[14] V. Sunderam, PVM: A Framework for Parallel Distributed Computing. *Concurrency: Practice and Experience*. Vol. 2, No. 2, pp. 315–339, 1990.