Chapter 54
Physical Mapping of Chromosomes Using Unique Probes

Farid Alizadeh* Richard M. Karp*† Deborah K. Weisser† Geoffrey Zweig†

1 Introduction

1.1 The Physical Mapping Problem

In this paper we present several combinatorial algorithms for reconstructing a DNA strand given a collection of overlapping fragments of the strand. A human chromosome, which is a DNA molecule of about 10^8 base pairs, is too long to be studied in its entirety and must be broken into fragments or *clones*. Depending on the cloning technology used, the sizes of the clones may be as small as 3,000 base pairs or as large as 2,000,000 base pairs. Information is gathered from the individual clones, and then the DNA is reconstructed by mathematically determining the positions of the clones.

The goal of *physical mapping* is to infer how the clones overlap to form the DNA molecule, given data about each clone. The present paper focuses on the Sequence Tagged Site (STS) mapping strategy, which is widely used for physical mapping within the Human Genome Project and other related molecular biology projects [PSM+91], [MCG+93]. In particular the recent mapping of human chromosome 21 [CRG+92] and human chromosome Y [VFH+92, FVHP92] use this strategy. In the STS approach relatively short substrings called *probes* are extracted from the DNA strand itself, often from the endpoints of clones. Each probe is sufficiently long that it is highly unlikely to occur a second time on the DNA strand; thus it identifies a unique site along the DNA strand. A probe is said to *hybridize* to a clone if it matches a substring in that clone. We present algorithms to determine probe ordering, given data for each clone indicating which probes hybridize to it.

In the absence of errors, the correct orderings can be found very easily using the PQ-tree data structure [BL76] to generate the set of all arrangements of probes consistent with the data. The data is never free from error however, and algorithms are differentiated by their performance in the presence of errors.

The most common type of error is a false negative, where a probe which occurs in a clone fails to hybridize to it. False positives also occur. In addition, in some cloning technologies the data is subject to error due to clone abnormalities, such as *chimeric* clones, which consist of two distinct segments of the DNA strand joined together. Deletions, insertions, and inversions of clone segments also occur.

In this paper we present reconstruction algorithms which are both effective and robust in the presence of errors. Exploiting the fact that each probe occurs at a unique point, we take it as our fundamental problem to determine the left-to-right order of the probes along the DNA strand. Once this order is known the overlap structure of the clones can easily be inferred.

All the central computational problems in this paper are NP-hard. For this reason we have evaluated the effectiveness of our algorithms empirically, using simulated data generated according to a probabilistic model, as well as real data from human chromosome 21.

1.2 A Mathematical Model of the Problem

We model the physical mapping problem geometrically as follows. The DNA molecule being mapped is an interval I on the real line. There are m probes and n clones. Each probe is a point on this interval. The clones are of two types: *normal* and *chimeric*. A normal clone consists of a subinterval of I, and a chimeric clone consists of the union of two disjoint subintervals. A key observation is that the set of probes contained within a normal clone is consecutive in the left-to-right ordering of the probes, and the set of probes contained within a chimeric clone can be partitioned into two subsets, each of which is consecutive in the left-to-right ordering of the probes. Moreover, if a probe was extracted from the end of a normal clone, then it must be either the first or the last probe in the ordering of the probes incident with that clone.

The *underlying data* for the problem is a $m \times n$ matrix $A = (a_{ij})$ with a row for each probe and a column for each clone, where

- $a_{ij} = 2$ if probe i is an end probe for clone j; i.e., it is known to have been extracted from one of the ends of clone j.
- $a_{ij} = 1$ if probe i is contained in clone j but is not an end probe for clone j.

*International Computer Science Institute. Research supported in part by NSF grant no. CDA-9211106.

†Computer Science Division, University of California, Berkeley. Research supported in part by NSF grant no. CCR-9005448.

- $a_{ij} = 0$ otherwise.

The matrix A has the following property: there is a permutation π of the rows (corresponding to placing the probes in their left-to-right order), such that, in each column corresponding to a normal clone, the nonzeroes occur in a single block of consecutive rows, and, in each column corresponding to a chimeric clone, the nonzeroes occur in two blocks of consecutive rows. Moreover, in the case of a normal clone a two may occur only in the first or last row of the block. (see Figure 1.)

The *measured data* $D = (d_{ij})$ differs from A because of the occurrence of false positives and false negatives. For fixed m and n we assume an *error model* defined by three parameters: p, the probability that a clone is chimeric, ϵ, the *false negative rate*, defined as the probability that a 1 in A becomes a 0 in D, and δ, the *false positive rate*, defined as the probability that a 0 in A becomes a 1 in D.

In the *noiseless case*, in which false negatives, false positives and chimeric clones do not occur, the problem is simply to find those permutations π such that, when π is applied to the rows of A, each column has consecutive ones, with twos in flanking positions, if present. Any such permutation is a plausible candidate for the correct left-to-right ordering of the probes.

In the more realistic case where noisy data and chimeric clones are present, we take it as our task to recover the underlying data from the measured data.

2 Algorithmic Approaches

We present several approaches to the problem of recovering the underlying data A from the measured data D.

2.1 A Maximum-Likelihood Method

The goal of the maximum-likelihood method is to find a matrix A which maximizes $p(A|D)$, the conditional probability of underlying data A given the measured data D. In other words, the method tries to select A so as to maximize the probability that the implied chimeric clones and experimental errors actually occurred.

The choice of A involves specifying the following:

- A linear ordering π of the rows;
- A designation of each column as normal or chimeric;
- In each normal column, a designation of one block of rows that are consecutive in π;
- In each chimeric column, a designation of two blocks of rows, each of which is consecutive in π.

The permutation π represents the left-to-right ordering of the probes. The matrix A corresponding to these choices has, in each column, nonzeroes in the positions within the selected block(s), and zeroes elsewhere. In a column corresponding to a normal clone, each two must be in either the first or last position in the block of nonzeroes. This matrix represents the choice of the underlying data.

Let $b(A)$ be the number of columns designated as chimeric, let $c(A)$ be the number of cells containing a 0 in D and a 1 in A, and let $d(A)$ be the number of cells containing a 1 in D and a 0 in A. A brief calculation based on Bayes' Theorem shows that the problem of maximizing $p(A|D)$ is equivalent to maximizing

$$\left(\frac{p}{1-p}\right)^{b(A)} \left(\frac{\epsilon}{1-\delta}\right)^{c(A)} \left(\frac{\delta}{1-\epsilon}\right)^{d(A)}.$$

Letting $K = -\ln \frac{p}{1-p}$, $L = -\ln \frac{\epsilon}{1-\delta}$ and $M = -\ln \frac{\epsilon}{1-\delta}$ we can reexpress the problem as follows: find A to minimize

$$Kb(A) + Lc(A) + Md(A). \tag{2.1}$$

Section 3.1 describes an effective algorithm for the approximate solution of this optimization problem.

2.2 The Hamming Distance TSP

We describe this approach in the case where end clone information is not available, so that the matrix A of underlying data contains only zeroes and ones. The approach can be extended to the case where end clone data is available.

There exists a permutation π of the rows of A which produces a matrix A^π in which each column corresponding to a normal clone contains one block of ones, and each column corresponding to a chimeric clone contains two blocks of ones. Define a *gap* as a block of zeroes in a column, flanked by a one immediately above and a one immediately below. Then the number of gaps in A^π is equal to the number of chimeric clones. Now consider the effect of introducing random errors. A false negative will change a one to a zero, typically splitting a block into two parts, and thus creating a gap. A false positive will change a zero to a one, typically splitting a gap into two gaps. Thus, when the permutation π is applied to the matrix D of measured data, the number of gaps in the resulting matrix D^π tends to be approximately equal to the number of chimeric clones plus the number of false positives and false negatives. This suggests the heuristic principle that a permutation of the rows which minimizes the number of gaps will correspond to a good probe ordering. Minimizing the number of gaps can be cast as a Traveling-Salesman Problem called the Hamming Distance TSP [AKNW91], in which the cities

CLONES

PROBES								
	2	0	0	0	0	0	2	0
	1	1	0	0	0	0	1	0
P	2	1	2	2	0	0	1	0
R	0	2	1	1	1	2	0	0
O	0	0	1	1	1	1	0	0
B	0	0	0	2	2	1	1	0
E	0	0	0	0	0	1	1	2
S	0	0	0	0	0	2	1	1
	0	0	0	0	0	0	2	1
	0	0	0	0	0	0	0	1

The underlying data A in correct order

CLONES

PROBES								
	2	0	0	0	0	0	2	0
	1	1	0	0	0	0	1	0
P	2	0	2	2	0	0	1	0
R	0	2	1	1	1	2	0	1
O	0	0	1	1	1	1	0	0
B	0	0	0	2	2	1	1	0
E	0	0	0	0	0	1	1	2
S	0	0	0	0	0	2	1	1
	0	1	0	0	0	0	2	0
	0	0	0	0	0	0	0	1

The measured data D in correct order

Figure 1: Example of Matrices A and D

are the rows of A together with an additional row of n zeroes, and the distance between two cities is the Hamming distance between the corresponding rows; i.e., the number of positions in which the two rows differ.

The Hamming Distance TSP has the disadvantage that its objective function is insensitive to the false positive rate, the false negative rate and the frequency of chimeric clones; thus it models the choice of an optimal probe ordering less faithfully than the maximum-likelihood approach. Its advantage is that the technology for solving large traveling-salesman problems is far advanced, so that near-optimal solutions are very easily obtained. We use a local search method due to Zweig [Zwe92], based on an operation called Divide-and-Merge, which rapidly and dependably gives near-optimal solutions to instances of the Hamming Distance TSP with around one thousand cities.

We have found that, when the number of ones per column of D is large, near-optimal solutions to the Hamming Distance TSP indeed correspond to near-optimal probe orderings. Moreover, the Hamming Distance TSP has proven to be a highly effective method for screening out false positives, as described in Section 4.1.

2.3 Methods for Obtaining a Good Initial Probe Ordering

The maximum-likelihood approach leads to a complex optimization problem which we attack by a local search method (cf. Section 3.1). The execution time of the local search can be reduced by providing the algorithm with a good initial solution. In addition to using the TSP solution, we have explored two other methods of obtaining such an initial solution: sorting and splitting.

2.3.1 Sorting

A real symmetric matrix (m_{ij}) is called *monotone* if, whenever $i < j < k$, $m_{ij} \geq m_{ik}$. Let b_{ij} be the number of clones that are incident with probes i and j. If probe j lies between probes i and k in the correct probe ordering, then, in the absence of errors every clone that is incident with probes i and k is also incident with probe j. It follows that $b_{ij} \geq b_{ik}$. Equivalently, if π is the correct linear ordering of the probes and we define the matrix B^π by $B^\pi_{ij} = b_{\pi(i),\pi(j)}$ then, in error-free data B^π is monotone.

The sorting method seeks a permutation π such that the matrix B^π is nearly monotone; this permutation is then used as an initial solution for the maximum-likelihood computation. The method computes a sequence of permutations. It passes from one permutation to the next by *pivoting* on some probe i. Given the current permutation σ and the pivot element i, the pivoting operation reorders the probes by moving closer to i those probes for which b_{ij}, the number of clones incident with both i and j, is large. The method is presented in Section 3.2. It has been found to give a good initial ordering, thereby reducing the time required for the maximum-likelihood computation.

2.3.2 Splitting

Let G be a bipartite graph with vertex set $\mathcal{P} \cup \mathcal{C}$, where $\mathcal{P}$ is the set of probes and $\mathcal{C}$ is the set of clones, and with an edge between probe P and clone C if and only if P is incident with C. We assume that G is connected, as otherwise the physical mapping problem decomposes into connected components whose order along the chromosome cannot possibly be ascertained from the data.

Briefly, given a probe s, we remove all clones incident to it and all the clones entirely included in them along with those probes which are not on any clone. If G is left with exactly two components then p is a *splitter*. Probe s is called a *splitter* if, when P and all the vertices

within distance two of it in G are deleted, the resulting subgraph has exactly two connected components. In this case we accept that the probes in one of these subgraphs lie to the left of P, and the probes in the other lie to the right of P. When there are no false positives the ordering implied by all the splitters is good, and its use as an initial ordering speeds up the maximum-likelihood computation.

2.4 Screening and Unscreening

We have found that false positives are particularly troublesome for our algorithms. Therefore, we have devised methods of identifying and eliminating them. These methods are not foolproof, and they sometimes create false negatives by changing a correct one to a zero; however, the overall effect of the screening methods is quite favorable.

By changing correct ones to zeroes (i.e., by deleting edges of the connected bipartite graph G that represent true hybridizations) the screening process sometimes breaks G into several components. Without restoring these critical deletions it is not possible to determine the order in which these components occur along the DNA strand. Thus, we use a two stage approach. In the first stage we obtain a good ordering for the probes in each connected component. Then, by inspecting the deleted edges which connect components via probes near their ends, we restore the critical screened out edges that hook the components together. This process is called *unscreening*.

Screening is described in Section 4.1 and unscreening is described in Section 4.2.

2.5 Algorithmic Strategy

We have experimented with a number of strategies for finding a good probe ordering. A very fast method which gives reasonably good solutions is simply to solve the Hamming Distance TSP. The following strategy has proved effective for obtaining better solutions, at the cost of a modest increase in execution time.

- Apply screening to eliminate most of the false positives.
- After screening the data may decompose into several connected components. For each connected component:
  1. Use the sorting heuristic to obtain a reasonably good initial probe ordering.
  2. Starting with this initial ordering, obtain a near-optimal solution to the maximum-likelihood problem.
- Use an unscreening procedure to hook together the probe orderings for the several connected components of the screened data.

Our computational results are summarized in Section 6.

3 Reconstruction Algorithms

3.1 Local Search

Recall that the choice of A involves specifying a linear ordering π of the rows of D, designating each column as normal or chimeric, choosing a block of rows for each normal column, and choosing two blocks of rows in each chimeric column. We shall show that, given π, there is a simple procedure for making the remaining choices optimally. The resulting optimal choice of the underlying data given π will be denoted $A(\pi)$. This yields the following optimization problem: Find π to minimize the objective function $F(\pi)$, defined as $Kb(A(\pi)) + Lc(A(\pi)) + Md(A(\pi))$.

To determine $A(\pi)$ we make two simple calculations for each column, one based on the assumption that the column is normal, and the other, on the assumption that the column is chimeric. The contribution of a column to the final objective function value is then the smaller of the two associated terms. Let C be the number of non-zero entries in the column; in typical cases C is a small constant. If the column is normal, then we need to choose the first and last rows in its block. For an optimal choice, these rows must contain ones or twos in the given column. Thus, there are $\binom{C}{2}$ choices for the block boundaries, and, using an appropriate data structure, the optimal choice can be determined in time $O(C^2)$ when no end probe information is available. If the column is chimeric we must choose the boundaries of two blocks, and we may assume that the first and last row of each block contain ones or twos. Thus there are $\binom{C}{4}$ choices for the block boundaries, but we have devised an algorithm that finds an optimal choice in time $O(C^2)$, again when no end probe information is available.

When one end probe is known, finding the optimal interpretation of a column is still quadratic (due to the possibility of chimeric fragments), but when both end probes are known, the process is linear. When retesting is performed so that false positive and false negative penalties are different for different hybridization events, an added sweep through the column between its extremal non-zero entries is necessary (see Section 5).

The following procedure describes the objective function calculation for a column in the case where no end probe information is available. It also determines block boundaries. The calculation performed when endpoint information is available is similar, with the added constraint that clone fragments must begin and/or end

with known end probes. This constraint reduces the number of block boundaries to consider.

Column Interpretation:

First assume the clone is not chimeric. For every possible pair of end probes, calculate the implied cost and store the best cost so found.

Now assume the clone is chimeric. For every possible starting probe of the first fragment, consider each choice of starting probe of the second fragment:

- Find the best end probe for the first fragment: This is either its current best ending probe, or it is the probe immediately preceding the current starting probe of the second fragment.
- Find the best end probe for the second fragment. When such an end probe is found, store it in an array indexed on the second fragment's starting probe. If a stored value is available, use it; otherwise consider every possible probe between the second fragment's starting probe and the end of the clone as an end probe.
- If the cost with the current best end probes is less than the best cost so far found, note the end probes and the associated cost as the new best interpretation.

We attack the problem of optimizing over permutations by local search, using a move operation which is related to the 2-opt and OR-opt move operations used for the Traveling-Salesman Problem. Given the present permutation π, the move operation selects a "neighboring permutation" as follows. Let D^π be the matrix obtained by reordering the rows of D in accordance with the current permutation; i.e., $D^\pi_{i,j} = D_{\pi(i),j}$. Define a *gap* in column j as an interval $[i..k]$ in column j, such that rows i and k contain ones in column j, and the intervening rows contain zeroes.

Operation Move:

- Select a gap at random in the matrix D^π. Let it be interval $[i..k]$ in column j. The chosen move will make rows i and k adjacent.
- Let u, the *upper weight* of the gap, be defined as the number of ones in column j at or above row i, and let ℓ, the *lower weight* of the gap, be the number of ones in column j at or below row k. Randomly choose either to move row i or to move row k, where the probability of choosing i is $\frac{\ell}{\ell+u}$ and the probability of choosing k is $\frac{u}{\ell+u}$;
- If i has been chosen then either move row i downward past rows $i+1, \cdots, k-1$ or reverse the block of rows $i, i+1, \cdots, k-1$, the two choices being equally likely;
- If k has been chosen then either move row k upward past rows $k-1, \cdots, i+1$ or reverse the block of rows $i+1, i+2, \cdots, k$, the two choices being equally likely.

We have observed that, in cases where the only errors are false negatives, the move operator appears to have the following property: if the move operator is applied repeatedly, then after a long enough period of time the successive permutations oscillate around a near-optimal permutation; i.e., there is a near-optimal permutation τ such that, for each probe i, the long-run average position of probe i approaches $\tau(i)$. In the case of noiseless data, random iteration of the move operation tends to converge to an ordering with the consecutive ones property. We have no theoretical explanation for these phenomena, (see Figure 10.)

Since the above move is defined only in terms of gaps, and makes no reference to known end probes, it sometimes happens that its application fails to place twos in extremal positions. Therefore we occasionally apply another move, in which a non-extremal two is chosen at random, and the row in which it lies is swapped with another row chosen at random in which there is a one in the same column as the selected two. Typically this move is used ten percent of the time.

We have found that simulated annealing, using the move operators defined above, is an effective method of obtaining near optimal solutions. We use a conventional cooling schedule and a conventional stopping rule [Laa88]. In its relation to simulated annealing, the neighborhood structure we use is significant for several reasons. First it is important to note that moves are not in general reversible; for example, when a gap is eliminated, there is no guarantee it can be recreated. This is important because the theoretical justifications of simulated annealing are predicated on a symmetrical transition matrix. The non-reversible moves that we use have the advantage, however, that in the absence of false positives and chimerics, convergence is much faster than it might otherwise be; excellent solutions are often found early on despite the fact that almost all the moves that are proposed are accepted. To take advantage of this we terminate the search when the best solution found is not replaced within a certain number of moves. Computational experience with the simulated annealing algorithm is reported in Section 6.

3.2 Sorting

We have devised a simple heuristic procedure based on the concept of a monotone matrix for finding a good initial ordering of the probes in the presence of

false negatives; the procedure finds the correct order of probes in the noiseless case. We first study the procedure in the noiseless case, and then discuss its extension to the case where false negatives are present.

We require some definitions. A square symmetric matrix $B = (b_{ij})$ is called *monotone* if, for all i and j with $i \leq j$, $b_{ij} \leq b_{i,j+1}$. A 0-1 matrix A has the *consecutive ones property for columns* if there exists a permutation matrix Q such that, in each column of QA, the ones form a consecutive block.

THEOREM 3.1. *Let A be a 0-1 matrix which has the consecutive ones property for columns. Let Q be a permutation of the rows of A. Then the following are equivalent:*

1. *In each column of QA, the ones form a consecutive block.*
2. *QAA^TQ^T is monotone.*

Proof. Let $D = QA$ such that in each column of D the ones appear in a consecutive block. If $P = DD^T$ is not monotone, then for some $i < j < k$, $P_{ij} > P_{ik}$. Thus row i of D has more ones in common with row k than with row j, and there must be at least one column with a one in rows i and k and zero in row j; but then ones in that column do not appear in a connective block, contrary to the assumption.

Now let D be a 0-1 matrix such that DD^T is monotone, and yet in some column, say c, not all ones appear in one block. We show that D does not have the consecutive ones property for columns. Let $i < j < k$, $D_{ic} = D_{kc} = 1$, and $D_{jc} = 0$. Since row i has more ones in common with row j than with row k (or at least as many) there must be at least one column with ones in rows i and j and zero in row k. Similarly, there must be one column with ones in rows k and j and a zero in row i. Thus we have shown that D contains the following matrix, or a permutation of its columns, as a submatrix:

$$F := \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

But no matrix containing F as a submatrix can have the consecutive ones property, as required.

The connection between monotone matrices and the consecutive ones property suggests a procedure for finding the correct ordering of the rows: find a permutation matrix Q such that QAA^TQ^T is monotone and sort the probes according to Q; i.e., if $Q_{ij} = 1$ then probe j occurs in the ith position of the ordering. The following discussion is based on the assumption that no clone is included in another, i.e. the clones induce a *proper interleaving*. Note that the noninclusion assumption implies that both D and D^T have the consecutive ones property for columns. In fact one can show a stronger result: for each row i let x_i and y_i respectively be the first and the last columns with a one in row i. Similarly, for each column j let u_j and l_j be respectively the first and the last rows with a one in column j. Then it is possible to order both rows and columns such that for a pair of columns i and k, if $i < k$ then $u_i \leq u_j$ and $l_i \leq l_k$, and similarly for a pair of rows s and t, if $s < t$ then $x_s \leq x_t$ and $y_s \leq y_t$. Let us call such an order of rows and columns *the canonical order.*

Let P_i be the set of clones incident to probe p_i. Observe that in the canonical ordering $x_i - y_i + 1$ is the number of clones n P_i, and, for $j > i$, $|P_i \cap P_j| = x_j - y_i + 1$. For a positive integer k the *k-neighborhood* $\Delta_k(P)$ of probe p is the set of all probes p_i such that $|P \cap P_i| = k$.

LEMMA 3.1. *For each probe p and positive integer k, $\Delta_k(P)$ may be decomposed into (possibly empty) chains $P_{i_1} \subset \cdots \subset P_{i_r}$ and $P_{j_1} \subset \cdots \subset P_{j_s}$ such that, in every correct ordering, the two chains occur on opposite sides of p, and, in each chain, the elements occur consecutively in order of increasing cardinality, with the minimal element closest to p.*

Proof. Let P_i and P_j, $i < j$, be the sets of clones incident on probes p_i and p_j, respectively. Furthermore, assume that both P_i and P_j are in $\Delta_k(P)$ and neither is a subset of the other. Then we must have, $x_i < x_j$ and $y_i < y_j$. Now if both P_i and P_j occur to the left of P, we must have

$$x - y_i + 1 = |P \cap P_i| = |P \cap P_j| = x - y_j + 1$$

which implies $y_i = y_j$. Similarly, if they both occur to the right of P we will have $x_i = x_j$, and in both cases we have a contradiction. So p_i and p_j are in opposite sides of P.

It is easy to see that if two probes P_i and P_j are both on one side of a given probe P and $|P \cap P_i| > |P \cap P_j|$ then P_i must be closer to P than P_j. Based on the preceding discussion we can now propose the following algorithm:

ALGORITHM *Sortmatrix*
Input: A 0-1 matrix D
Output: A permutation π where $\mathcal{P}^\pi = D_\pi D_\pi^T$ is monotone
Method:

```
Set P = DD^T.
for i = 1, ..., n do
    π ←SORTLEFT(i, P^π)
    π ←SORTRIGHT(i, P^π)
    If i < n and END(i, P) then
        π ←REVERSE(i, P^π)
        π ←SORTRIGHT(i, P^π)
    end if
end for
```

The key operations in *Sortmatrix* are the

SORTLEFT$(i, \mathcal{P})$ and the *SORTRIGHT*$(i, \mathcal{P})$ functions. Briefly, *SORTLEFT* sorts the entries of row i of $\mathcal{P}^\pi$ in nondecreasing order up to the diagonal element. Similarly *SORTRIGHT*$(i, \mathcal{P}^\pi)$ sorts the entries of row i in nonincreasing order from diagonal element to the end. In case of ties the following rule is applied: If $\mathcal{P}^\pi_{ij} = \mathcal{P}^\pi_{ik}$ and $P_j \subset P_k$ then P_j succeeds P_k when applying *SORTLEFT* (i.e. $k, j < i$) and P_j precedes P_k when applying *SORTRIGHT* ($k, j > i$). Otherwise sorting should be stable, and the relative order of P_j and P_k should not change. We refer to operations *SORTLEFT* and *SORTRIGHT* together as *pivoting* on probe P_i. The function *END*$(i, \mathcal{P})$ returns true if the current pivot P_i has empty intersection with all probes ahead of it (i.e. $\mathcal{P}_{ik} = 0$ for all $k > i$), and false otherwise. Finally, REVERSE$(i, \mathcal{P}^\pi)$ reverses the order in $(p_i, \cdots, p_j)$.

It can be shown that in each iteration, *Sortmatrix* maintains the following invariant: If $(p_i, p_{i+1}, \cdots, p_j)$ is the current sequence of pivoted probes, then they must occur (up to reversal) consecutively in the final correct ordering. In particular, the current probe about to be pivoted on will not disturb the order in $(p_i, p_{i+1}, \cdots, p_j)$ sequence up to reversal. Therefore,

THEOREM 3.2. *Let D be the 0-1 incidence matrix of clones and probes from a proper interleaving of clones and without errors. Let π be the permutation found by Sortmatrix. Then $\mathcal{P}^\pi = D_\pi D_\pi^T$ is monotone.*

Although the validity of Algorithm *Sortmatrix* is established only in the case of proper interleavings and noiseless data, we have used it successfully as a heuristic for obtaining a good probe ordering in the presence of low error rates and occasional clone inclusion. We have also observed in our computational experiments that *Sortmatrix* benefits greatly by screening false positives and chimeric clones.

3.3 Splitting

Splitting is a method for finding an approximate probe ordering. It works well in the presence of false negatives, but becomes ineffective when a large number of false positives or chimeric clones are present. Consider the bipartite graph G as described in Section 2.3.2. We assume that this graph is connected, as otherwise the mapping problem decomposes into disconnected components whose order along the chromosome cannot possibly be ascertained from the data.

We say a set of clones $\mathcal{C}$ *covers* a clone C if all of the probes incident on C are also incident on some clone in $\mathcal{C}$. Let $SPAN(s)$ be the set of clones incident on s. Probe s is called a *splitter* if, when all the clones in $SPAN(s)$ and all of the clones covered by $SPAN(s)$ are deleted from G along with all probes left without any incident clones, then the resulting graph is left with exactly two connected components which contain clones. Let the vertex sets of these two components be denoted A and B, and let the vertices in neither component be denoted M.

Then, in the noiseless case M separates A from B; i.e., one of the two possible left-to-right orientations of the DNA strand has the property that every probe or clone in A is strictly to the left of every probe or clone in M, and every probe or clone in B is strictly to the right of every probe or clone in M.

In the case where false negatives occur but false positives and chimeric clones do not, the partitions associated with splitters still provide valuable information about the ordering of the probes. Assume that there is a fixed false negative rate ϵ and that the bipartite graph G of incidences between probes and clones is connected. Let s be a probe which is a splitter with respect to the noiseless data and is not involved in any false negatives. Then s is also a splitter with respect to the noisy data. This implies that splitters will be abundant.

Moreover, the following claim establishes that the connected components A and B associated with a splitter s provide accurate information about the ordering of the probes. Let the length of every clone be bounded above by a constant L. Let q, r and s be probes such that q and r are on the same side of s, with r closer to s, and s is a splitter. Let A and B be the two components associated with s. Then, since ideally A contains the probes on one side of $SPAN(s)$ and B contains the probes on the other side, we would expect q and r to lie together either in A or in B. It can be shown that, conditioning upon the event that s is a splitter, the probability that this fails to happen tends to zero exponentially as a function of the distance from r to s.

The above observations suggest that the following algorithm should give a good probe ordering in the absence of false positives and chimeric clones.

1. Determine which probes are splitters and, for each splitter, determine the components A and B;
2. Identify two sets of probes which frequently occur in opposite components. Label one set "left" and the other "right".
3. Associate with each probe q a *left tally* $\alpha(q)$ giving the number of times it occurs in a component that has more in common with "left" than "right". Generate a *right tally* $\beta(q)$ analogously.
4. Sort the probes in decreasing order of $(\alpha(q) - \beta(q)$, breaking ties in increasing order of $\alpha(q)$.

This procedure gives a good initial ordering, even when there are moderate number of false negatives.

4 Screening and Unscreening

4.1 Screening out False Positives

As mentioned earlier, false positives are particularly troublesome for our algorithms; The speed of Local Search degrades when a large number of false positives are present, and the splitting and sorting algorithms perform satisfactorily in the absence of false positives, but less well when false positives are present. We present two methods that have proved effective for screening out false positives.

The Hamming Distance TSP

Besides producing near-optimal probe orderings, the Hamming Distance TSP has proved to be an effective method for screening out false positives. The screening is accomplished by solving the Hamming Distance TSP, applying the resulting permutation to the rows, and then eliminating every isolated one occurring in a column containing a block of at least two ones. On artificially generated data this method typically eliminates about 90 percent of the false positives. However, a false negative is created whenever the method incorrectly identifies a true one as a false positive and changes it to zero. Typically one false negative is created for every three to ten false positives that are eliminated. More detailed computational results are given in Section 6.

Neighborhood Screening

Recall the bipartite graph G described 2.3.2. Define the d-neighborhood of a vertex as the set of all vertices within distance d of that vertex. Neighborhood screening proceeds by repeatedly focusing on a neighborhood of some clone, identifying the likely false positives in that subproblem, and deleting them. The algorithm is as follows:

For $c = 1, 2, \cdots, n$ do

- Select clone c;
- Construct a submatrix D' whose rows and columns, respectively, correspond to the probes and clones in the 3-neighborhood of c;
- Using simulated annealing, obtain a near-optimal solution A' to the problem of finding the most likely matrix of underlying data, given the measured data D';
- Designate as a false positive each cell in any column where D' contains a 1 and A' contains a 0.
- For each designated false positive, set the corresponding cell of D to zero.

This screening method is compared with the Hamming Distance TSP in Section 6.

4.2 Unscreening

A side effect of screening false positives is the introduction of some false negatives. When the coverage is low, this can result in disconnecting an originally connected component into several components. We have developed an algorithm which identifies connected components created by the screening process and reorders them according to the unscreened data (see Figure 2.)

The unscreening algorithm first finds connected components in the screened data which has been previously ordered. Since the data has already been ordered we assume that two nearby rows that do not overlap are in separate components. An ordering of the connected components is found which maximizes the intersection between the end probes of adjacent components.

5 Retesting

We have developed a highly effective approach to recovering the noiseless data through an interactive process in which an algorithm is used to detect anomalies in the data, and these anomalies are then rechecked through further hybridization experiments. At a general step, for each probe i and clone j, some number $t_{ij} \geq 1$ of independent tests will have been conducted, each purporting to measure whether probe i occurs on clone j; let y_{ij} denote the number of such tests with a positive outcome, and let n_{ij} denote the number of such tests with a negative outcome. The problem of finding a maximum likelihood matrix A of noiseless data, given the matrices (y_{ij}) and (n_{ij}) can be cast as a generalization of the minimization problem 2.1 in which $b(A)$ denotes the number of columns designated as chimeric, $c(A)$ denotes the sum of the n_{ij} over all those cells in which A contains a 1, and $d(A)$ denotes the sum of the (y_{ij}) over all those cells in which A contains a zero. Near-optimal solutions to this problem can be found by local search. Once a solution $A = (a_{ij})$ has been found, we identify certain *questionable cells* whose data needs to be confirmed or disconfirmed by further tests. The following is the simplest of several strategies we have tried. Call cell ij *probably positive* if the conditional probability of obtaining the data for that cell, given that $a_{ij} = 1$, is greater than the conditional probability of obtaining the data for that cell, given that $a_{ij} = 0$; i.e., if $\epsilon^{n_{ij}}(1-\epsilon)^{y_{ij}} > (1-\delta)^{n_{ij}}\delta^{y_{ij}}$. Otherwise, the cell is *probably negative*. Then cell ij is questionable if $a_{ij} = 1$ but cell ij is probably negative, or if $a_{ij} = 0$ but cell ij is probably positive. The identification of the underlying data proceeds by repetition of the following algorithm:

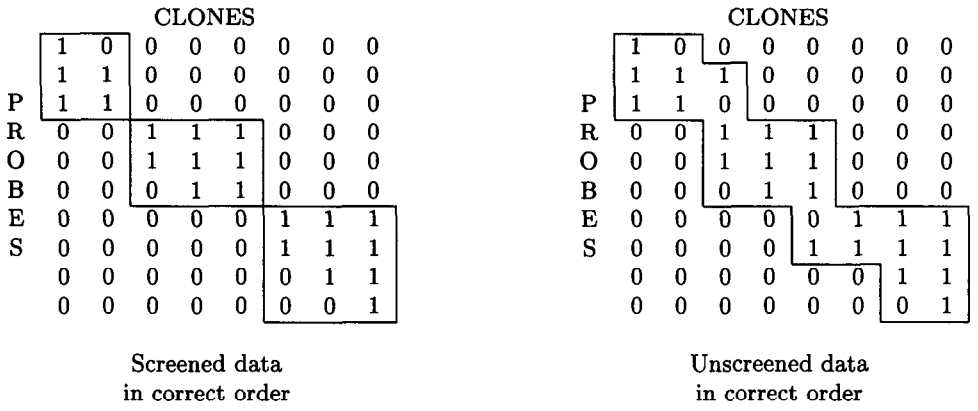

Figure 2: Disjoint connected components created by screening and reconnected by unscreening

- For the given data (y_{ij}) and (n_{ij}), use local search to find a matrix A that (nearly) maximizes $p(A|D)$.

- Identify and retest the questionable cells.

The iteration continues until no questionable cells remain. The analysis presented so far does not incorporate all the information provided by local search. the zero-one value that local search produces for a bit should be weighted along with the hybridization results for that bit in the probability calculations. While a precise weighting of these two types of evidence is difficult to determine from theoretical considerations, we have found a simple procedure that much improves the retesting results. If retesting confirms an error prediction, we reset the corresponding y_{ij} and n_{ij} entries to zero-one values; i.e. all previous hybridization evidence is discarded. This modification has proved useful in eliminating false positives when the false positive rate is much lower than the false negative rate. Without it, many retests are potentially necessary to accumulate enough evidence to deem a bit probably negative.

Using simulated data, only a few iterations are typically required to identify the correct permutation of the probes, and the number of repeated tests is comparable to the number of errors present in the data. Computational results on simulated data are found in Section 6.

6 Computational Results

In this section we present some computational results for the various algorithms. We have worked both with simulated data and data from human chromosome 21 generated by Cohen's group at CEPH [CRG+92].

For the simulated data unit length clones were generated[1] with their midpoints uniformly and independently distributed over an interval of length N. Next probes were chosen from the ends of clones. The probe rate ρ, which is the expected number of probes to be taken from a clone, varied between between 0 and 2. For $\rho \leq 1$, each clone was chosen with probability ρ to contribute a single probe. For $\rho > 1$, each clone contributed at least one probe, and with probability $\rho - 1$ contributed two probes. False negatives and false positives were introduced with rates ϵ and δ as described in Section 1.

The ratio of the number of clones to the length of the chromosome is called the *coverage*. As expected the performance of our algorithms improved as coverage increased.

In all our simulations the correct probe ordering is $[1, 2, \cdots, m]$, and in the following figures the computed permutations are plotted against this identity permutation. In the plots for the chromosome 21 data, the results of our algorithms are plotted against the order published in [CRG+92]. The quality of an ordering can thus be judged by how close it is to the identity permutation or its reverse. Plots with long line segments of ± 45 degrees are more desirable than those with short scattered segments.

Figure 3 presents the results of neighborhood and Hamming distance TSP screening algorithms on simulated data.

Figure 4 presents the number of retests required to obtain internally consistent data using the method described in Section 5.

In each of Figures 5, 6, and 7 we show three sets

[1] Our algorithms (except for the sorting algorithm) work even when the clones have different sizes and in particular some may be completely covered by other clones. We will report on our experiments with variable length clones in the final version of the paper.

false positive rate	false positives screened		Changes causing false negatives	
	Neighborhood	Hamming Distance TSP	Neighborhood	Hamming Distance TSP
0.1%	97%	97%	18%	33%
0.5%	96%	91%	20%	10%
1.0%	97%	91%	25%	6.3%

False negative rate fixed at 10%
200 clones, coverage 5 and probe rate 1.0.

Figure 3: Screening by Neighborhood and Hamming Distance TSP averaged over ten problem instances.

false positive rate	ratio of retests to original errors	ratio of retests to correctly identified errors
0.1%	0.971	1.18
0.5%	1.04	1.16
1.0%	1.03	1.10

200 clones, coverage 5 and probe rate 1.0. False negative rate fixed at 10%

Figure 4: Retesting Table

of plots. First we show the ordering generated by the TSP heuristic. Next we show the ordering generated by the sequence of TSP screening, sorting, local search, and unscreening. Finally we show the results of TSP screening followed by splitting, local search, and post screening.

Figures 7 and 9 demonstrate the performance of the sequence of TSP, sorting, local search, and unscreening for varying false positive rates and varying probe rates.

Figure 10 shows the output of the random search (described at the end of Section 3.1) on data from chromosome 21. The plot shows the average position of each probe over the moves 35,000 to 40,000.

Acknowledgement

We would like to thank Donn Davy of Lawrence Berkeley Laboratories for providing us with the data from human chromosome 21. We also thank Suzanna Lewis, Lee Newberg, Rudiger Reischuk and Ron Shamir for many fruitful discussions.

References

[AKNW91] Farid Alizadeh, Richard M. Karp, Lee A. Newberg, and Deborah K. Weisser. Physical Mapping of Chromosomes: A Combinatorial Problem in Molecular Biology. In *Proc. 4th ACM-SIAM Symposium on Discrete Algorithms*, 1991. (to appear in *Algorithmica*).

[BL76] Kellogg S. Booth and George S. Lueker. Testing for the Consecutive Ones Property, Interval Graphs, and Graph Planarity Using *PQ*–Tree Algorithms. *J. Comp. and Syst. Sci.*, 13:335–379, 1976.

[CRG+92] Ilya Chumakov, Philippe Rigault, Sophie Guillou, Pierre Ougen, Alain Billaut, Ghislaine Guasconi, Patricia Gervy, Isabelle LeGall, Pascal Soularue, Laurent Grinas, Lydie Bougueleret, Christine Bellanné–Chantelot, Bruno Lacroix, Emmanuel Barillot, Philippe Gesnouin, Stuart Pook, Guy Vaysseix, Gerard Frelat, Annette Schmitz, Jean-Luc Sambucy, Assumpcio Bosch, Xavier Estivill, Jean Weissenbach, Alain Vignal, Harold Reithman, David Cox, David Patterson, Katheleen Gardiner, Masahira Hattori, Yoshiyuki Sakaki, Hitoshi Ichikawa, Misao Ohki, Denis Le Paslier, Roland Heilig, Stylianos Antonarakis, and Daniel Cohen. Continuum of overlapping clones spanning the entire human chromosome 21q. *Nature*, 359:380–386, October 1992.

[FVHP92] Simon Foote, Douglas Vollrath, Adrienne Hilton, and David Page. The Human Y Chromosome: Overlapping DNA Clones Spanning the Euchromatic Region. *Science*, pages 60–66, October 1992.

[Laa88] P. Van Laarhoven. Theoretical and Computational Aspects of Simulated Annealing. Centrum voor Wiskunde en Informatica, Amsterdam, 1988.

[MCG+93] Toru Mizukami, William I. Chang, Igor Garkavtsev, Nancy Kaplan, Diane Lombardi, Tomohiro Matsumoto, Osami Niwa, Asako Kounosu, Mitsuhiro Yanagida, Thomas G. Marr, and David Beach. A 13kb Resolution Cosmid Map of the 14 Mb Fission Yeast Genome by Nonrandom Sequence–Tagged Site Mapping. *Cell*, 73:121–132, 1993.

[PSM+91] Michael J. Palazzolo, Stanley A. Sawyer, Christopher H. Martin, David A. Smoller, and Daniel L. Hartl. Optimized strategies for sequence–tagged–site selection in genome mapping. *Proc. Natl. Acad. Sci. USA*, 88:8034–8038, 1991.

[VFH+92] Douglas Vollrath, Simon Foote, Adrienne Hilton, Laura G. Brown, Peggy Beer-Romero, Jonathan S. Bogan, and David C. Page. The Human Y Chromosome: A 43–Interval Map Based on Naturally Occurring Deletions. *Science*, pages 52–59, October 1992.

[Zwe92] Geoffrey Zweig. A New Neighborhood Structure for the Travelling Salesman Problem. (Submitted), 1992.

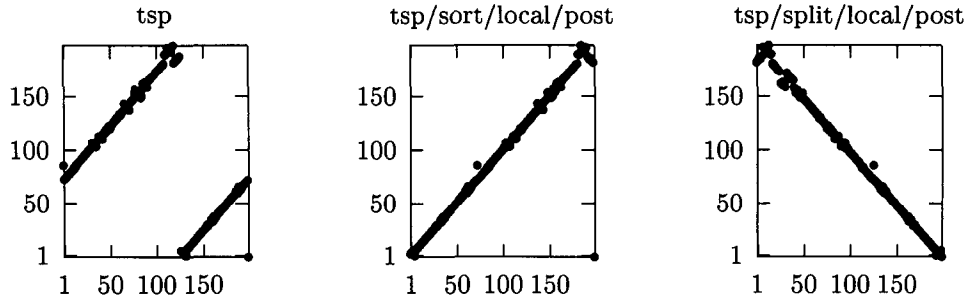

Figure 5: Results of chromosome 21 data.

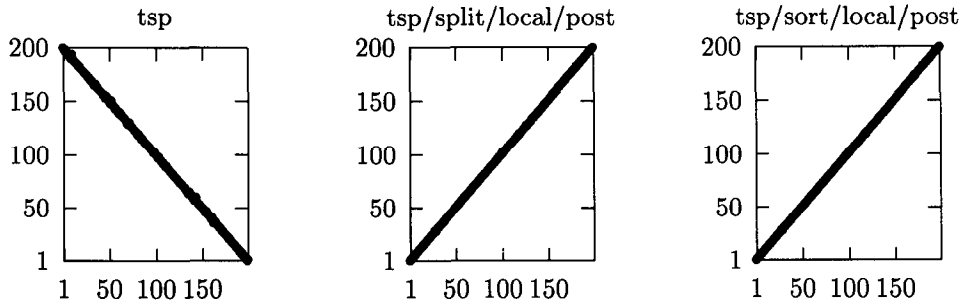

Figure 6: Results on data with coverage 10, probe rate 0.5 false positive rate 0.01 and false negative rate of 0.1.

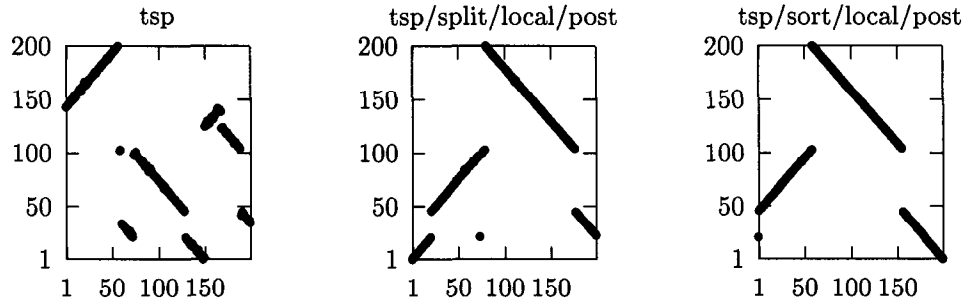

Figure 7: Results on data with coverage 5, probe rate 1.0, false positive rate 0.001 and false negative rate of 0.1.

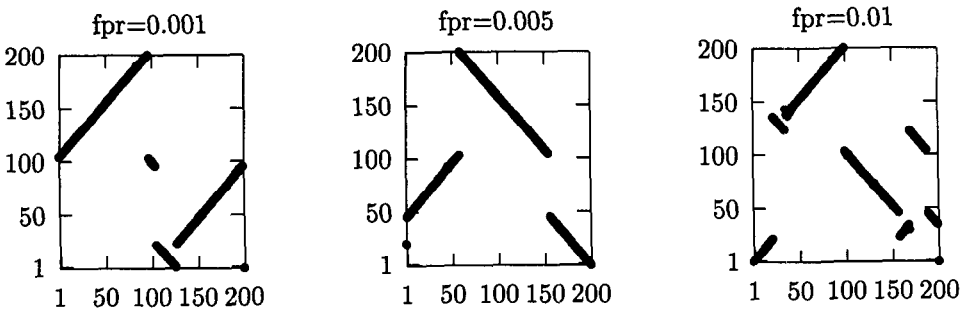

Figure 8: Results of varying false positive rate using tsp/sort/search/post on data with coverage 5, probe rate 1.0 and false negative rate 0.1.

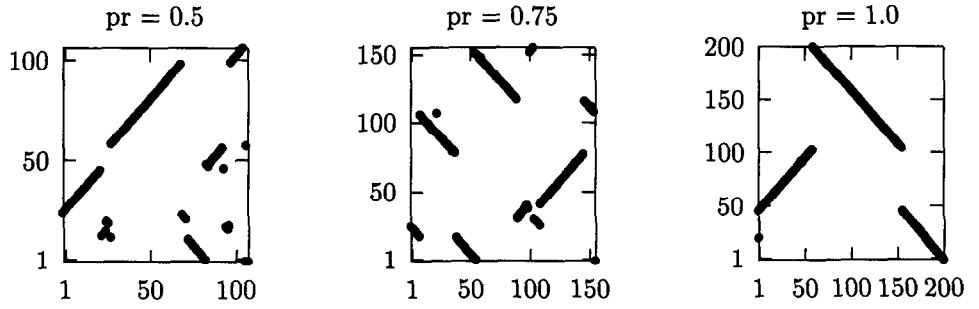

Figure 9: Results of varying probe rate using tsp/sort/search/post on data with coverage 5, false positive rate 0.005 and false negative rate 0.1.

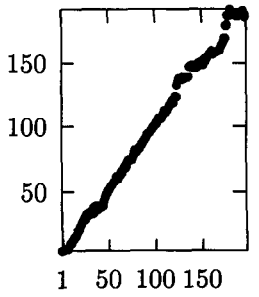

Figure 10: Result of random moves on chromosome 21 data.