Algorithms for Computing and Integrating Physical Maps Using Unique Probes

Mudita Jain [*] Eugene W. Myers [†]

April 3, 1997

Abstract

Current physical mapping projects based on STS-probes involve additional clues such as the fact that some probes are anchored to a known map and that others come from the ends of clones. Because of the disparate combinatorial contributions of these varied data items, it is difficult to design a "tailored" algorithm that incorporates them all. Moreover, it is inevitable that new experiments will provide new kinds of data, making obsolete any such algorithm. We show how to convert the physical mapping problem into a 0/1 linear programming (LP) problem. We further show how one can incorporate additional clues as additional constraints in the LP formulation. We give a simple relaxation of the 0/1 LP problem that solves problems of the same scale as previously reported tailored algorithms to equal or greater optimization levels. We also present a theorem proving that when the data is 100% accurate, then the relaxed and integer solutions coincide. The LP algorithm suffices to solve problems on the order of 80-100 probes — the typical size of the 2- or 3-connected contigs of Arratia et al. We give a heuristic algorithm which attempts to order and link the set of LP-solved contigs. Unlike previous work, this algorithm only links and orders contigs when the join is 90% or more likely to be correct. It is our view that there is no value in computing an optimal solution with respect to some criteria over very noisy data as this optimal solution rarely corresponds to the true solution. The paper involves extensive empirical trials over real and simulated data.

1 Introduction

The STS-probe physical mapping problem involves a collection of *clones* that are long, anonymous segments of a target DNA sequence, and a collection of STS *probes* that are short sequences of DNA whose presence in a clone can be tested. It is assumed that each probe sequence occurs only once in the target DNA. Formally, suppose there are m probes and n clones. The input to the basic physical mapping problem is an $m \times n$ incidence matrix D, where d_{ij} is 1 if probe i hybridizes with clone j and 0 otherwise. The problem is to find an ordering of the probes such that the probes incident to each clone are consecutive in the order. Formally, one seeks a permutation π of the probes/rows that gives the permuted matrix D_π the consecutive ones property [BL76]. If the information in D is perfect than the problem is solvable in linear time [BL76]. However, in practice, the measured data contains errors in the form of *false positives* (i.e. a 1 where there should be a 0), *false negatives* (i.e. a 0 where there should be a 1), and *chimeras* (i.e. two distinct intervals reported as one). Under these more realistic assumptions, the problem becomes one of finding the most-likely permutation π and set of error corrections that give D_π the consecutive ones property. Unfortunately, this problem is NP-complete [Kou77, GKS94].

Let π be a permutation of probes, D_π be the permuted incidence matrix D, and T be the corrected version of D that reflects the error corrections made in D to explain π. Further suppose that one has

[*] Dept. of Computer Science, University of Arizona, Tucson, AZ 85721 (e-mail: jainm@cs.arizona.edu). Fully supported by NLM grant LM-04960

[†] Dept. of Computer Science, University of Arizona, Tucson, AZ 85721 (e-mail: gene@cs.arizona.edu). Partially supported by NLM grant LM-04960

estimated that false negatives occur at fixed rate ϵ (typically 10-20%), false positives at rate δ (typically 5-10%), and chimeras at rate χ (typically 1-5%). Alizadeh et al. [AKWZ95] showed that minimizing the objective function:

$$C \cdot fc(D_\pi) + P \cdot fp(D_\pi) + N \cdot fn(D_\pi) \tag{1}$$

maximizes the likelihood, $Pr(T|D_\pi)$, of T given D_π, where $fc(D_\pi)$ is the number of chimera corrections, $fp(D_\pi)$ is the number of false positive corrections, $fn(D_\pi)$ is the number of false negative corrections, and the constants C, P, and N are the log likelihood ratios: $C = -\ln \frac{\chi}{1-\chi}$, $P = -\ln \frac{\epsilon}{1-\delta}$, and $N = -\ln \frac{\delta}{1-\epsilon}$. Thus the phyiscal mapping problem in the presence of false positives and negatives, and chimeras is to find π *and* T that minimizes (1).

While there has been considerable work on the version of the problem just presented, the accuracy of the data that has been generated has not been sufficient to arrive at the biologically correct solution. Because of this, the experimentalists have gone on to map/order a subset of *anchored probes* using other technologies such as FISH-mapping, and to further select *end-probes* from the ends of a subset of the clones in the dataset. While this data only makes the problem more complex and less tractable from a theoretical point of view, in practice, it considerably limits the solution space.

Suppose $A = < a_1, a_2, a_3, \cdots a_{|A|} >$ is the list of the anchored probes in their known order. There can be errors in these maps. If anchored probes are mislocated at rate α (typically 1%), then an easy extension of the analysis of [AKWZ95] reveals that it suffices to add the term $M \cdot fa(\pi)$ to the objective (1) above, where $fa(\pi)$ is the number of mislocated anchored probes with respect to π, and the constant M is the log-likelihood ratio: $M = -\ln \frac{\alpha}{1-\alpha}$. More generally, if one knows that probe p_i is to the left of p_j with probability γ, then any order placing p_i to the right of p_j should be penalized an extra $-\ln \frac{\gamma}{1-\gamma}$.

In this paper we consider the physical mapping problem including end-probe and anchored-probe information. We first give a 0/1 LP formulation and associated relaxation that models all the data and errors discussed above. What is appealing here is that the LP framework is so generic that when and if new forms of data become available, one is likely to be able to rapidly incorporate them. In the second part, we then give a heuristic algorithm that specifically takes advantage of these additional data items to create the best possible global ordering of the 2- or 3-connected contigs of Arratia et al. [ALTW91].

2 An Integer Linear Programming Model

Consider first the problem where one is given the $m \times n$ hybridization matrix $D = [d_{pc}]$ and the only phenomenon to be modeled are false positives and negatives. Two sets of 0/1 variables are required for the LP formulation. There are mn t-variables representing a proposed selection of "true" probe-clone incidences:

$$t_{pc} = \begin{cases} 1 & \text{if probe } p \text{ actually is incident to clone } c. \\ 0 & \text{otherwise.} \end{cases}$$

There is a one-to-one correspondence between the t_{pc} variables and the observed hybridization matrix $D(P, C)$:

$$d_{pc} = \begin{cases} 1 & \text{if probe } p \text{ is incident to clone } c. \\ 0 & \text{otherwise.} \end{cases}$$

Then a false positive is represented by the clause $(d_{pc} = 1 \wedge t_{pc} = 0)$, which is translated into the linear equation $d_{pc}(1 - t_{pc}) = 1$. A false negative is similarly represented by the equation $t_{pc}(1 - d_{pc}) = 1$. There is a second set of m^2 x-variables representing a proposed ordering of the probes:

$$x_{pq} = \begin{cases} 1 & \text{if probe } p \text{ is to the left of } q. \\ 0 & \text{otherwise.} \end{cases}$$

Directly translating equation (1) in terms of these variables, the LP objective is to minimize the sum:

$$\sum_{c,p} P \cdot d_{pc} \cdot (1 - t_{pc}) + N \cdot (1 - d_{pc}) \cdot t_{pc} \tag{2}$$

which is a weighted combination of the t-variables (all other items are constants). There are two groups of constraints, one set insuring that the x-variables encode a permutation, and the second insuring that the t-variables encode interval graph incidences with respect to the x-permutation.

Permutation Constraints:

- $x_{pq} \in \{0, 1\}$.
- $x_{pq} + x_{qp} = 1$. These insure antisymmetry. Informally, probe p is either to the left of probe q, or it is to the right of q.
- $x_{pq} + x_{qr} - x_{pr} \leq 1$. These insure transitivity. Informally, if probe p is to the left of probe q, and q is in turn to the left of probe r, then p is also to the left of r.

Betweeness Constraints:

- $t_{pc} \in \{0, 1\}$.
- $(1 - x_{pq}) + (1 - x_{qr}) + (1 - t_{pc}) + t_{qc} + (1 - t_{rc}) \geq 1$. These insure interval graph incidences.

Informally, the last constraint asserts that if probe q is *between* probes p and r, and probes p and r are both incident to clone c, then probe q has to be incident to clone c also. Note that there are $O(m^2)$ antisymmetry inequalities, $O(m^3)$ transitivity inequalities, and $O(m^3 n)$ betweeness inequalities. It is interesting to note that the constraints on x-variables are those of the much-studied *linear-ordering problem* [Rei85] and the betweeness constraints are closely related to the *betweenness problem* [Opa79].

Modeling Chimeras: Chimerism requires the addition of n χ-variables indicating whether a given clone is chimeric or not:

$$\chi_c = \begin{cases} 1 & \text{if clone } c \text{ actually is chimeric.} \\ 0 & \text{otherwise.} \end{cases}$$

One must further add the term $\sum_c C\chi_c$ to the objective (2) of the LP problem, constrain the variables $\chi_c \in \{0, 1\}$ to be 0/1, and augment the betweeness constraint as follows:

- $(1 - x_{pq}) + (1 - x_{qr}) + (1 - t_{pc}) + t_{qc} + (1 - t_{rc}) + \chi_c \geq 1$.

Informally, if probe q is between probes p and r, and probes p and r are both incident to clone c, then either probe q is also incident to clone c, or clone c is a chimera. Notice that this constraint is problematic in that once a clone is designated a chimera, there is no way to constrain the probes incident on it as being two or more contiguous blocks of probes. However, if the data is not adversarial, the cost of designating a clone as chimeric, and the costs of false negatives and false positives prevent the arbitrary assignment of probes to clones.

Modeling Anchored Probes: For each anchor $a_i \in A$, one must add a variable mis_{a_i} that indicates whether or not the probe is mislocated in the order of A:

$$mis_{a_i} = \begin{cases} 1 & \text{if anchor } a_i \text{ is mislocated.} \\ 0 & \text{otherwise.} \end{cases}$$

One then adds the term $\sum_{a_i} G mis_{a_i}$ to the objective (2), and adds the constraints:

- $x_{a_i a_j} + mis_{a_i} + mis_{a_j} \geq 1$ for all $i < j$.

Informally, either anchor a_i is to the left of anchor a_j, or at least one of a_i or a_j is misplaced. Including anchors into the data adds $|A|$ variables and $O(|A|^2)$ constraints. Note that $|A|$ is generally much smaller than m.

In general, any partial ordering information between probes may be incorporated. For example, if a probe p is known to be to the left of probe q, we can just set x_{pq} to one. If probe p is to the left of probe

q with some probability γ, this fact is easily incorporated into the objective function by adding the term $-\ln \frac{\gamma}{1-\gamma} * x_{pq}$.

Modeling End-Probes: Modeling end-clone information does not require new variables. For a clone c where probes p_e and q_e have been extracted from both ends of the clone, one adds the $(m-2)$ constraints:

- $x_{p_e r} + x_{q_e r} + t_{rc} \leq 2$.

Informally, the constraints insures that either probe r is between the two end probes p_e, q_e, or it is not incident to clone c. If only one end-probe, say q_e is known for clone c, then one adds the $O(m^2)$ constraints:

- $x_{pq_e} + x_{q_e r} + t_{pc} + t_{rc} \leq 3$.

Informally, the constraints insure that if q_e is between p and r, then one of p or r is not incident to c.

2.1 Relaxations

The preceding treatment demonstrates the versatility of the LP framework as one in which to express a physical mapping problem involving a variety of, often partial, constraints. The difficulty is that 0/1 and integer LP problems are NP-complete. However, there has been much work on approximation algorithms based on relaxing the integrality constraints, solving the resulting real-valued LP problem, and then applying rounding heuristics to get back to a feasible integer solution that is often near-optimal [RT87, Shm95]. We chose such a relaxation scheme for the following reasons:

- Linear programming is polynomial time solvable. There exist both, practical algorithms for solving linear programs, and a wide variety of software.
- There is the hypothesis that the linear programming relaxation retains enough of the structure of the original problem to be a useful weak representation [CR96]. Therefore, we hope that the fractional values that result from a solution to the relaxed linear program will provide enough useful information about the structure of the optimal solution.
- Several algorithms exist for rounding the fractional values of the variables, in a fashion that the resulting algorithm provides provably good solutions. That is, there is an approximation bound on the goodness of the solution. See Raghavan and Thompson [RT87] for randomized rounding, and Shmoys [Shm95] for a survey on approximation algorithms for combinatorial optimization problems.

For our first technical result, it is necessary that all noninformative clones and probes be removed from the problem. Specifically, a clone that has one or no probes incident to it does not provide any help in inferring probe orderings and can be removed. Also, a set of probes that are all incident to exactly the same set of clones cannot be distinguished/ordered and may be collapsed to a single probe. The procedure below:

```
repeat
  RemoveSingleIncidenceClones();
  ReduceEquivalentProbes();
until (no change);
```

filters the data set so that every pair of probes is distinguishable and every clone has at least two probes incident to it.

Proposition 1 *After filtering, every pair of probes in the data is distinguishable. That is, for every pair of probes p_i and p_j , there is a clone c such that exactly one of p_i or p_j is incident to it. Clone c also has another probe p_k, $k \neq i, j$ incident to it.*

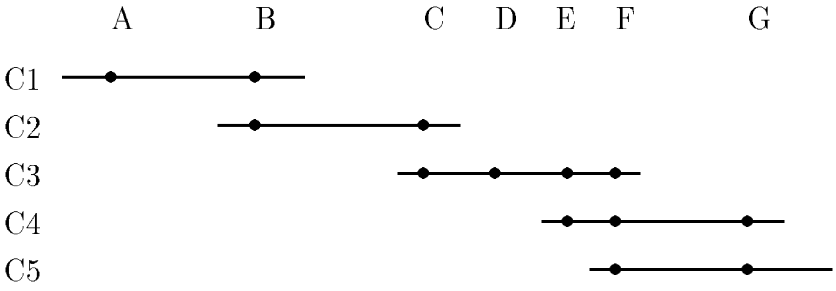

Figure 1: An example layout of clones and probes.

Given a filtered data set, two probes p and q are said to be connected, written $p \sim q$ if there is a clone incident to both probes. The equivalences classes of probes induced by the reflexive and transitive closure, $\sim^\star$, of the relation $\sim$ are called *contigs*. Clearly, there is no possible way to order probes in different contigs. Given a filtered data set, the physical mapping problem is to find the most probable order of each contig.

Consider now, relaxing the 0/1 constraints and letting the variables range over the real numbers between 0 and 1. The immediate difficulty is that an optimal objective value of 0 is obtained by setting all the x-variables to $\frac{1}{2}$ and setting $t_{pc} = d_{pc}$ for all p and c. To avoid this degenerate solution we need to select a pair of probes p and q in each contig and set $x_{pq} = 1$ for each pair. Effectively, one is arbitrarily orienting each contig. The interesting result which we prove in the full paper is as follows:

Theorem 1 *Consider the relaxed LP-formulation of a filtered physical mapping problem involving only false positives and negatives. Further assume that each contig has been oriented with the addition of an $x_{pq} = 1$ constraint. If D has the consecutive ones property then the optimal solution to the relaxed problem is 0/1 with value 0 for the objective.*

In other words, if there are no errors in the data then the relaxed LP solution provides an encoding of an interval graph consistent with D. Appealing to continuity, one then intuits that the variables drift from their 0 or 1 values as the error rates increase from zero.

Proof: Let $l_1, \ldots, l_i, p_j, b_1, \ldots, b_m, p_k, r_1, \ldots, r_n$ be the ordering along the chromosome of a set of connected probes. As is clear from the ordering, in this proof we will assume that p_j is to the left of p_k. In the case that p_j is to the right of p_k, our proof still goes through, except that the solution to the linear program produces the reverse ordering $r_n, \ldots, r_1, p_k, b_m, \ldots, b_1, p_j, l_i, \ldots, l_1$. In figure 1, let $B = p_j$ and $E = p_k$.

We will first show that having set $x_{p_j p_k} = 1$, the values $x_{p_y p_k} = 1$ are forced by our constraint system for all probes p_y that are to the left of probe p_k. By proposition 1, there is a clone c_1 such that it has exactly one of p_j or p_k incident to it — suppose that p_j is incident to c_1. By the same proposition, there is at least one more probe, say p_w, also incident to c_1. In figure 1 $c_1 = C1$, and $A = p_w$. Now consider the constraint

$$x_{p_j p_k} + x_{p_k p_w} + t_{c_1 p_j} + t_{c_1 p_w} - t_{c_1 p_k} \leq 3 \tag{3}$$

It is evident that a solution of zero cost has $t_{pc} = d_{pc}$ for all p, c. In the case of equation 3 $t_{c_1 p_j}, t_{c_1 p_w} = 1$, and $t_{c_1 p_k} = 0$. Also, we are given that $x_{p_j p_k} = 1$. Given these constraints, equation 3 is satisfied only if $x_{p_k p_w} = 0$, which means that $x_{p_w p_k} = 1$. By repeated application of this procedure, $x_{p_y p_k} = 1$ is forced for all probes p_y that are to the left of p_k and share a clone c_i with a probe p_w such that $x_{p_w p_k} = 1$, and p_k is not incident to c_i.

Now consider the case of all probes b_i to the left of p_k such that

$$\forall c_j (d_{b_i c_j} = 1) \Rightarrow (d_{p_k c_j} = 1) \tag{4}$$

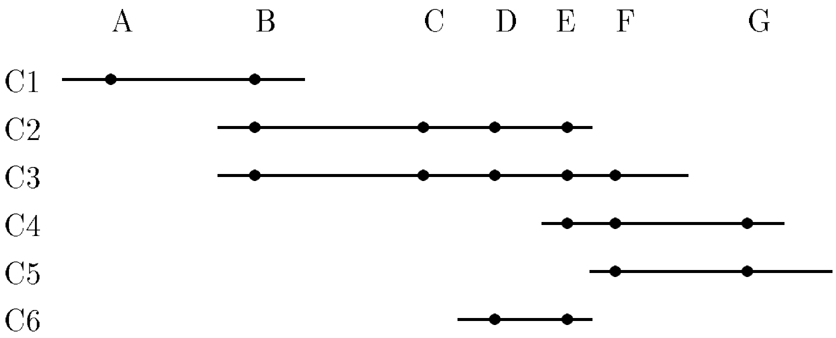

Figure 2: A layout of probes and clones in which there are no probes p_y between $p_j = B$ and $p_k = E$ that are both incident to a clone containing p_k and a clone not containing p_k.

That is, probe p_k is incident to all clones that contain probes b_i. In figure 1 $D = b_i$. We would like to show that $x_{b_ip_k} = 1$ is forced by the constraints. We first consider the case of probe b_m, the probe immediately to the left of p_k in the probe permutation.

Given probes b_m and p_k, we know that there is a clone c_j such that both b_m and p_k are incident to it, and there is a probe l also incident to c_j for which $x_{lp_k} = 1$ is known. That such an l exists is seen by noting that there is a set of clones connecting p_j to p_k. In the set of probes L to the left of p_k and incident to this set of clones, there may be a rightmost probe that is both, incident to a clone with p_k and is also incident to a clone not containing p_k. We have already shown that if there is a clone containing p_j and not p_k, then for all such probes it can be inferred that they are to the left of p_k. In figure 1, $C = l$. In the case that there is no such rightmost probe, p_k is incident to all clones containing probes in L. But then we have a clone with both p_j and p_k incident to it, and $l = p_j$. Figure 2 illustrates such a layout.

As b_m and p_k are distinguishable, there is a set of clones $c_1, \ldots, c_n$ that contain p_k and not b_m. In figure 1 clones $C4$ is such a clone. Consider the longest such clone $c^{'}$. The following two cases may occur.

1. $F(c^{'}) \not\subset F(c_j)$. Informally, the set of probes incident to $c^{'}$ is not entirely contained in the set of probes incident to c_j. This case holds in figure 1, as probe G incident to $C4$ is not incident to clone $C3 = c_j$.

2. $F(c^{'}) \subset F(c_j)$. That is, all probes incident to $c^{'}$ are also incident to c_j.

In case 1, there is a probe r_i incident to $c^{'}$, and not incident to c_j. This leads to the following series of events.

(1.1) $x_{r_il} + x_{lp_k} + t_{c^{'}r_i} + t_{c^{'}p_k} - t_{c^{'}l} \leq 3 \Rightarrow x_{r_il} = 0$
Therefore $x_{lr_i} = 1$.

(1.2) $x_{lr_i} + x_{r_ip_k} + t_{c_jl} + t_{c_jp_k} - t_{c_jr_i} \leq 3 \Rightarrow x_{r_ip_k} = 0$
Therefore $x_{p_kr_i} = 1$.

(1.3) $x_{p_kr_i} + x_{r_ib_m} + t_{c_jp_k} + t_{c_jb_m} - t_{c_jr_i} \leq 3 \Rightarrow x_{r_ib_m} = 0$
Therefore $x_{b_mr_i} = 1$.

(1.4) $x_{p_kb_m} + x_{b_mr_i} + t_{c^{'}p_k} + t_{c^{'}r_i} - t_{c^{'}b_m} \leq 3 \Rightarrow x_{p_kb_m} = 0$
So finally, $x_{b_mp_k} = 1$.

We now consider the case 2. As all probes $r_i \neq p_k$ incident to $c^{'}$ are also incident to c_j, there is a clone $c^{''}$ that distinguishes between p_k and r_i. Note that any clone containing p_k and some r_i has to contain r_1. Therefore in the following, we consider $r_i = r_1$. To distinguish between p_k and r_1, $c^{''}$ may contain one of either r_1 or p_k, but not both.

(2.1) If c'' contains p_k and not r_1, then it perforce contains b_m. In this situation there is no clone that distinguishes between the permutations b_m, p_k, r_1, and r_1, p_k, b_m, and both permutations are valid. Figure 3 illustrates this situation when $D = b_m$, $E = p_k$, $F = r_1$, $C5 = c'$, and $C4 = c''$. Therefore there is no unique valid permutation, a contradiction to our original assumption.

(2.2) If c'' contains r_1 and not p_k, then it contains some other probe r_j to the right of p_k. If r_j is also incident to c_j, there is a clone that has to distinguish between r_j, r_1. Continuing, as we have a finite number of clones, one of two cases will occur. We may eventually find a clone that contains some probe r_y that is not incident to c_j. At this point we can repeatedly apply case 1, to show $x_{b_m p_k} = 1$. If there is no such probe r_y, then we revert to case 2.1, and have no unique valid permutation. Figure 4 illustrates this case, with $C5 = c''$, $G = r_j$, and $H = r_y$.

The case for forcing $x_{b_i p_k} = 1$ follows similarly. Finally, if all clones that contain p_j also contain p_k, it remains to be shown that for all probes l_i to the left of p_j, $x_{l_i p_k} = 1$ is forced. Letting p_j be immediately to the left of p_k, and l_i be immediately to the left of p_j, an argument exactly the same as above can be made to show that $x_{l_i p_k} = 1$ is forced. Hence we have shown that for all probes p_y to the left of p_k the constraint system forces $x_{p_y p_k} = 1$. Using symmetry it can be shown that $x_{p_j p_w} = 1$ is forced for all probes p_w to the right of probe p_j. For all probes r_i to the right of p_k, we can show in a manner similar to the above that $x_{p_k r_i} = 1$ is forced by the variables $x_{p_j r_i} = 1$. Also for all probes l_i to the left of p_j, $x_{l_i p_j} = 1$ is forced by the variables $x_{l_i p_k} = 1$.

All that remains to be shown now is that an integer value is be forced on every variable $x_{p_a p_b}$ for arbitrary probes p_a, p_b. This is fairly easy now — we have shown that integer values have been forced upon all the variables $x_{p_k p_a}, x_{p_a p_k}, x_{p_a p_j}, x_{p_j p_a}$. Of these variables, we start with any one whose value is 1. Using that variable, in a manner exactly as above, an integer value is forced upon the variable $x_{p_a p_b}$. Theorem 1 ▮

For a current real project, m is four thousand and n is six thousand. Our LP formulation requires $O(mn + m^2)$ variables and $O(m^3 n)$ constraints, for a total of thirty-six million variables and thirty-two trillion constraints. Borrowing several ideas from cutting plane and branch-and-cut algorithms as described in Jünger et al. [JRT95], and Padberg and Grötschel [PG85], we reduce the number of variables and constraints by keeping only the tightest inequalities/facets of the polyhedral solution space. For a proper treatment of solving physical mapping formulations to optimality using branch and cut approaches, see Christof et al. [CJK+97]. In this paper, we find that the simple expedient of eliminating a large number of easily identified constraints and heuristically rounding gives competitive results while retaining the flexibility of not having to redesign the algorithm with introduction of each new constraint type.

Reducing Interval Graph Constraints: Minimizing the objective function involves setting the value of the t-variables as nearly identical to their corresponding D-matrix values as possible. Thus the betweeness constraint $x_{pq} + x_{qr} + t_{pc} + t_{rc} - t_{qc} \leq 3$ tends to constrain the values of its variables only when $d_{pc} = d_{rc} = 1$

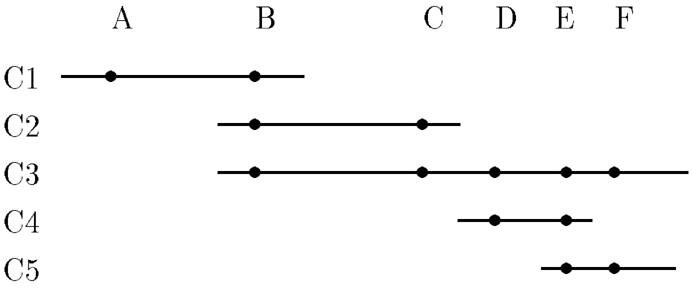

Figure 3: A layout of probes and clones that conforms to case (2.1). Both probe permutations (D, E, F) and (F, E, D) are valid.

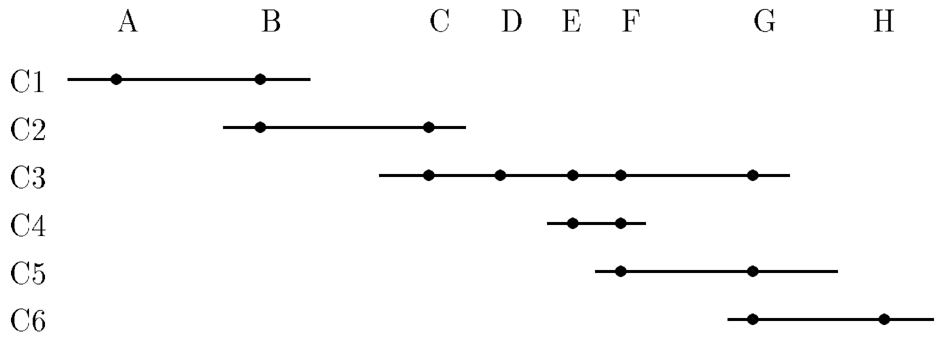

Figure 4: A layout of probes and clones that conforms to case (2.2). Clone $C6$ and probe H is used to infer backwards to $x_{DE} = 1$.

and $d_{qc} = 0$. So we retain only those betweeness constraints for which the D-matrix values are as just stated. Typically, the average number of probes incident to a clone w is a small constant (e.g., 2 to 5), so that the number of constraints remaining is $O(w^2 m)$.

Removing Transitivity Constraints: In the case of input data that contains no errors, one observes in the proof of Theorem 1 that the anti-symmetry and the betweenness constraints are the only ones necessary to obtain a valid permutation. The transitivity constraints are non-constraining and only gradually become so as the percentage of false positives increases. Thus we remove them, and later show another method of coping with false positives.

This culling of constraints leaves us with a real-valued LP system with $O(m^2 + w^2 n)$ constraints and $O(m(m+n))$ variables. While this does not permit us to solve the problem over an entire data set, it does allow us to solve problems where m is 50-100 — adequate for the 2- or 3-connected clusters of the global algorithm of the final section. The solution to the relaxed system gives fractional values. Our next task is to use these to arrive at a probe ordering that is hopefully near to optimal.

2.2 Probe Ordering

We interpret the values of the variables $x_{p_i p_j}$ as being the weight assigned by the constraint system to the probe p_i being to the left of probe p_j. Now consider the graph $G = (V, E)$, such that the set of probes is the set of vertices V. There are directed edge between each pair of vertices, and the weight of an edge (p_i, p_j) is the value of $x_{p_i p_j}$ as computed by the linear program. In the case that all $x_{p_i p_j}$ have integer values, a topological sort of all the vertices in G yields an optimal probe ordering. In the case that these values are fractional, we appeal to the objective of the linear-ordering problem [Rei85], to design a greedy heuristic that seeks a permutation π of the probes maximizing $\sum_{\pi(j)<\pi(k)} x_{\pi(j)\pi(k)}$. We build the order from left to right, picking the next probe in the ordering at each step. Let π_k be the probe permutation computed thus far, and let U be the set of probes not yet placed in the permutation. For each vertex $p \in U$, compute the sum $\sum_{q \in U - \{p\}} x_{pq}$. Greedily pick the probe p for which this sum is maximal, remove it from U and let $\pi_{k+1} = \pi_k \cdot p$. If there is a tie for the maximum, then for each tied probe p, solve the relaxed LP formulation where the relevant x-variables are set to reflect the assumption that $\pi_k \cdot p$ is the ordering of the $k+1$ leftmost probes. Resolve the tie by picking a probe for which the value of its associated LP objective is minimal. We call the ordering that results the *basic LP order*. Figre 5 gives the pseudocode for the heuristic.

The heuristic above produces a probe ordering that generally consists of blocks of probes that are correctly ordered, separated by break points where the adjacent probes have few clones in common. These breaks are generally due to poor choices made by the tie-breaking strategy of the heuristic above. Proceeding formally, suppose the basic LP order is $< p_1, p_2, \cdots p_m >$. A point i is a *break point* if $|N(p_i) \cap N(p_{i+1})| < \tau$

```
$U \leftarrow \{p_1, \cdots, p_M\}$
$\pi \leftarrow \emptyset$
while $U \neq \emptyset$ do
    maxout = 0
    for $i \leftarrow 1$ to $M$ do
    if $(p_i \in U)$ then
        outgoing[$p_i$] $\leftarrow \sum_{p_j \in U} x_{p_i p_j}$
        if (outgoing[$p_i$] > maxout) then
            next-vertex $\leftarrow \{p_i\}$
            maxout $\leftarrow$ outgoing[$p_i$]
        else if (outgoing[$p_i$] = maxout) then
            next-vertex $\leftarrow$ next-vertex $\cup \{p_i\}$
    if (|next-vertex| = 1)
        $\pi_i \leftarrow$ next-vertex
    else
        new-objective-val $\leftarrow \infty$
        for each $p_i \in$ next-vertex
            objective $\leftarrow$ solve-linear-program($\pi \cup p_i$)
            if (objective < new-objective-val)
                next-vertex$'$ $\leftarrow p_i$
        $\pi_i \leftarrow$ next-vertex$'$
    $U \leftarrow U - \pi_i$
```

Figure 5: The probe ordering heuristic derived from the integer linear programming formulation. solve-linear-program is a procedure that generates and solves the constraints described in the linear program formulation of physical mapping for the probes in U.

where $N(p)$ is the *neighborhood* of clones incident to p and threshold τ is chosen so that one obtains 10-30 break points. A *block* is a segment of probes between two consecutive break points. We then proceed to translocate and/or invert blocks to other break point sites, accepting such a move if it reduces the cost of the objective function. This algorithm is essentially the 2-OPT local search algorithm used by Alizadeh et al. [AKWZ95], except that our moves are non-random. In all of our trials, this swapping algorithm converged to a near-optimal solution after 5 to 10 moves. We call the ordering that results the *swap-refined LP order*.

2.3 Computational Results

In this section we present examples of the basic and swap-refined LP orders that result from these heuristics. We present results over both simulated data and the real mapping data set generated at Lawrence Berkeley Labs for Drosophila.

Simulated Data Characteristics: The left end points of the clones were generated by a Poisson process at a rate which produced an average clone coverage c. We varied c between 5 and 7. Clone lengths were chosen randomly and uniformly between a pre-specified interval [*minlen*, *maxlen*]. Each clone was specified chimeric with probability $\chi = 5\%$. The probes were generated by a Poison process at a rate which produced an average of w probes incident to each clone. We used $w = 3$ or 4 for our experiments. A probe was extracted from the end-point of a clone with a 30% probability, and a probe was designated an anchor with a probability of 20%. For clone fingerprints, we varied the false negative rate, ϵ, between 10% and 20%, and the false positive rate, δ, between 5% and 10%. The data sets used typically had between 5000 and 8000 clones, and between 3000 and 4000 probes.

Drosophila Data Characteristics: The data from the Drosophila mapping project has a clone coverage of 5, and a probe coverage of 2. The false negative rate is estimated at around 20%, the false positive rate as somewhere between 4% and 10%, and the chimera rate is practically zero. A little more than 50% of the probes have been extracted from the ends of clones, and 41% of the probes are anchored to a FISH map. The data consisted of 7446 clones and 3120 probes.

All of the data sets above were first clustered into 2-connected contigs as part of the global algorithm described in Section 3.1. These clusters tend to have much lower false positive and chimera rates, generally around 1% each. So in Figures 6-13 we present results of the LP heuristic on clusters of probes and clones with these lower rates as these are the intended context. However, we also present the results of running our algorithms on data that has not been pre-processed and hence contains high rates of chimeras and false positives in Figures 14 and 15. Due to practical considerations such as the number of constraints generated, the data sets for which we ran our algorithms without pre-processing were small, typically containing between a 100 and 150 clones and 30 to 70 probes.

For probes sets of different sizes, Table 1 gives averages for the number of clones associated with the probe set, the number of constraints generated after all relaxations are applied, and a lower bound on the number of constraints that would have been required without the described relaxations.

Probe Set	Average Clone Set	Average Constraint Set	Original Constraint Set
15	22.11	3090	78,225
20	25.00	5778	208,400
30	37.75	16,727	1,155,362
57	75.00	56,791	14,077,917

Table 1: Given data with clone coverage 5, and probe rate 4, the average number of clones associated with probe sets of particular sizes, as well as the number of constraints generated by our relaxation, and a lower bound on the number of constraints required without relaxations.

Drosophila Data Ordering Plots: The graphs in Figures 6, 7, and 8, are for randomly selected 2-connected contigs from the Drosophila data set. Each plot is of an LP-computed ordering against the optimal ordering according to objective (1). The plot at left is of the basic LP order and the plot at right is of the swap-refined LP order.

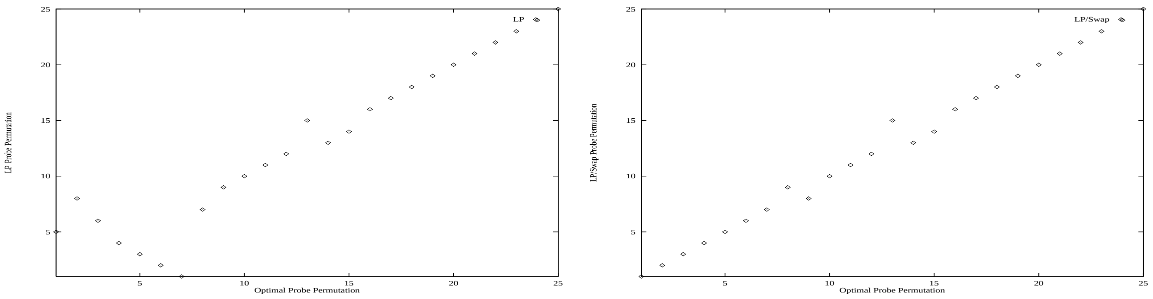

Figure 6: Real data. Clone coverage 5, Probe coverage 2, false negative rate 0.20, false positive rate 0.01, and chimera rate 0.

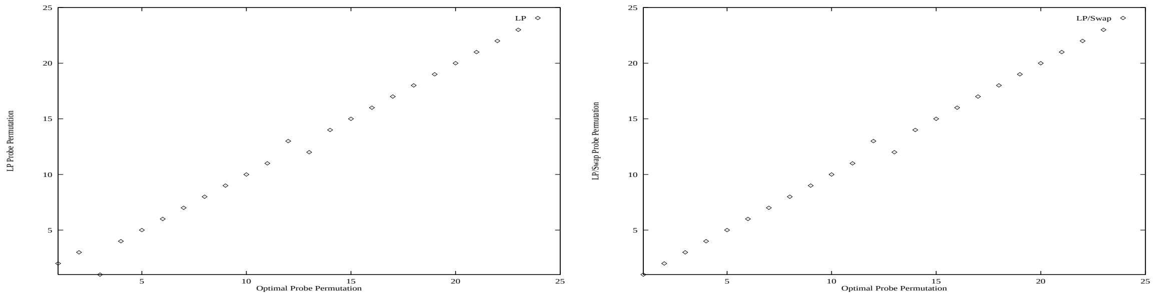

Figure 7: Real data. Clone coverage 5, Probe coverage 2, false negative rate 0.20, false positive rate 0.01, and chimera rate 0.

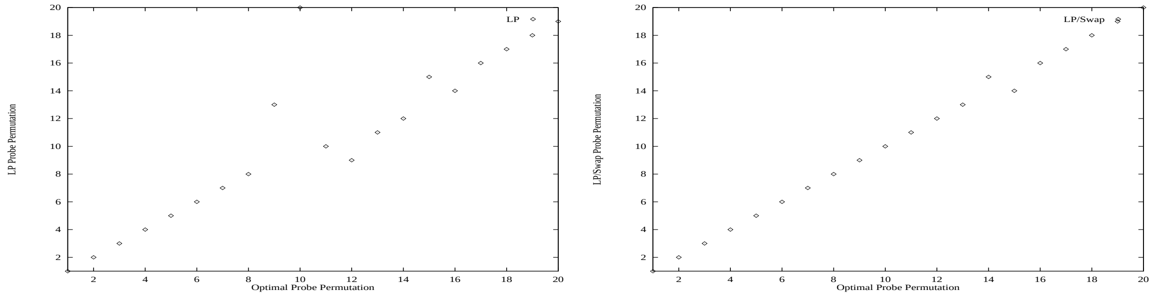

Figure 8: Real data. Clone coverage 5, Probe coverage 2, false negative rate 0.20, false positive rate 0.01, and chimera rate 0.

Simulated Data Ordering Plots: The graphs in Figures 9-13 are for randomly selected 2-connected contigs of the simulated data sets, and Figures 14 and 15 are for two simulated but small, high false positive and chimera rate, data sets. The x-axis in the plots in these figures represents the original permutation, which is known as the data is simulated. In the graph at left the basic LP order is plotted as diamonds, and the optimal ordering according to objective (1) is plotted as plus-signs. The graph at right plots the swap-refined LP order as diamonds along with the optimal ordering as plus-signs.

The basic quality of the computed permutations can be estimated by how closely the plot follows the diagonal or the anti-diagonal. We further computed the fraction of correct probe adjacencies, i.e., we compute the percentage of left and right neighbors computed correctly for all probes. The following observations hold:

- In all cases, neighboring probes in the original or optimal permutations are usually neighboring in the computed permutations. For the Drosophila mapping data, the fraction of correct adjacencies in the swap-refined LP order (where the optimal permutation was taken as the reference) was 0.895. For simulation data of varying characteristics, the fraction of correct adjacencies averaged 0.934.

- For both simulated and real data, the computed permutations are very close to the optimal permutations. For the Drosophila mapping data, the average ratio of the score of the swap-refined LP permutation and the score of the optimal permutation was 1.47. For simulated data of various characteristics, this average ratio was 1.46. To give additional perspective on this ratio, Table 2 presents the score of the swap-refined order, the score of the optimal/original permutation, the ratio of these two scores, the number of inversions (i) and translocations (t) required to edit the computed permutation into the reference one, and the extra false negatives (fn), false positives (fp), and mislocated anchors

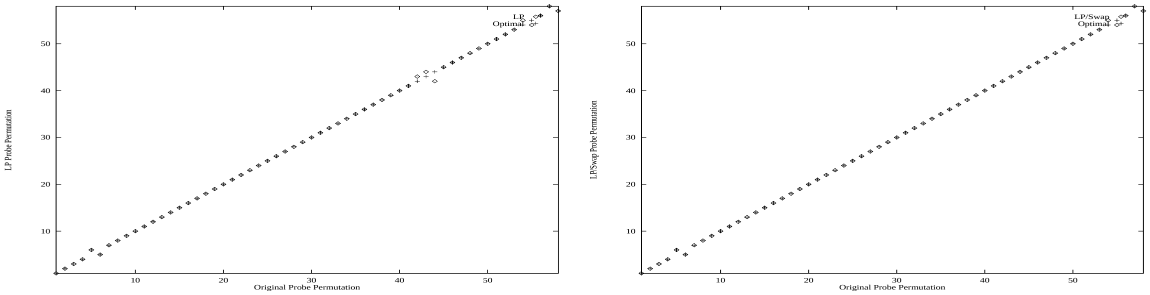

Figure 9: Simulated data. Clone coverage 7.5, Probe coverage 4, false negative rate 0.20, false positive rate 0.01, and chimera rate 0.01.

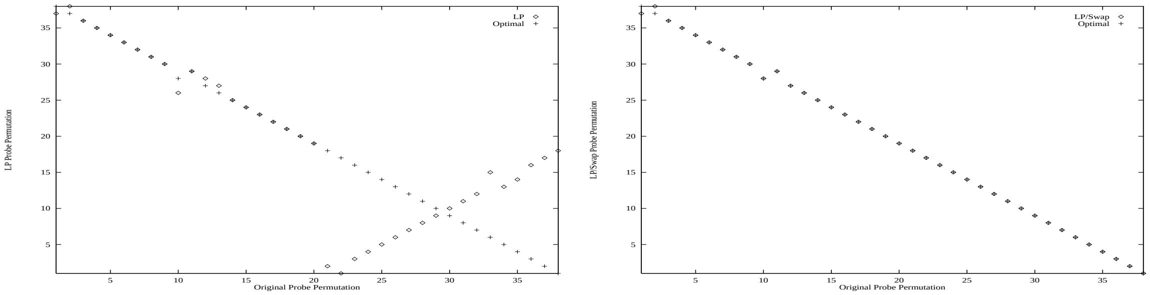

Figure 10: Simulated data. Clone coverage 7.5, Probe coverage 4, false negative rate 0.20, false positive rate 0.01, and chimera rate 0.01.

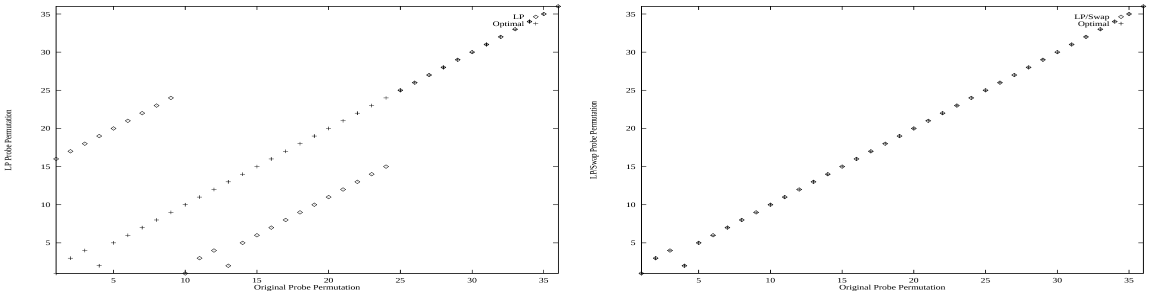

Figure 11: Simulated data. Clone coverage 7.5, Probe coverage 4, false negative rate 0.20, false positive rate 0.01, and chimera rate 0.01.

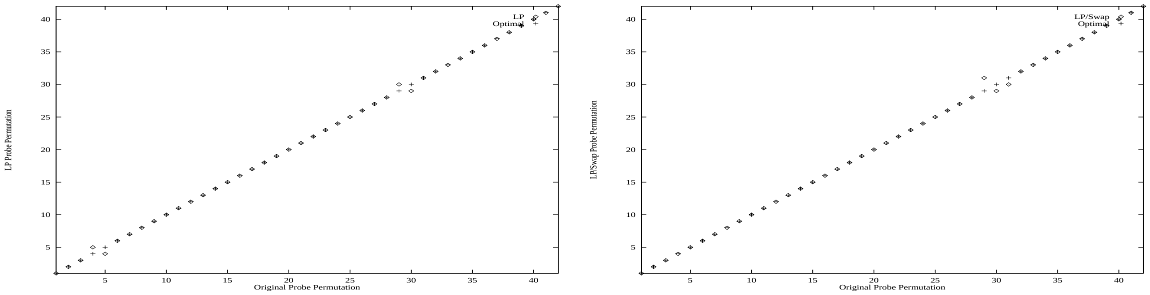

Figure 12: Simulated data. Clone coverage 5, Probe coverage 4, false negative rate 0.20, false positive rate 0.01, and chimera rate 0.01.

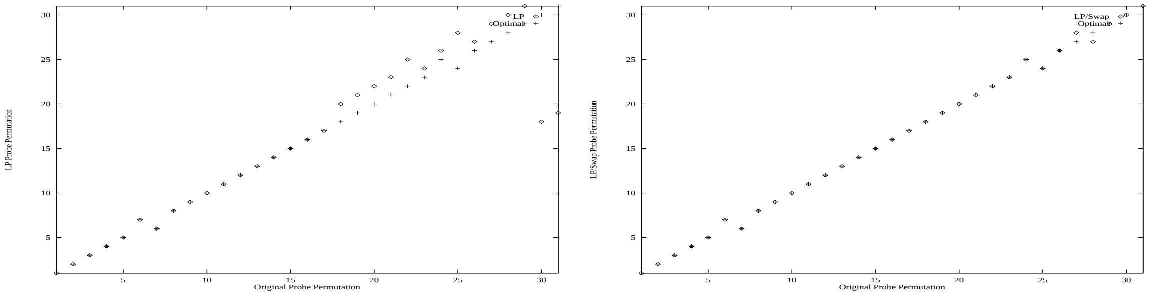

Figure 13: Simulated data. Clone coverage 5, Probe coverage 4, false negative rate 0.20, false positive rate 0.01, and chimera rate 0.01.

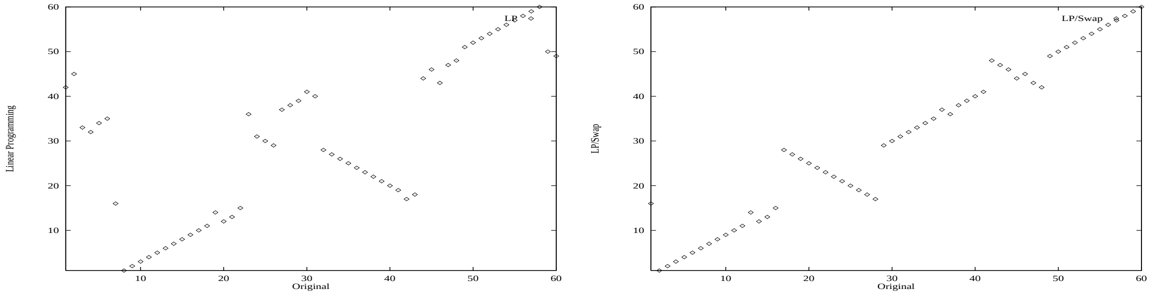

Figure 14: Simulated data. Clone coverage 5, Probe coverage 3, false negative rate 0.20, false positive rate 0.1, and chimera rate 0.1.

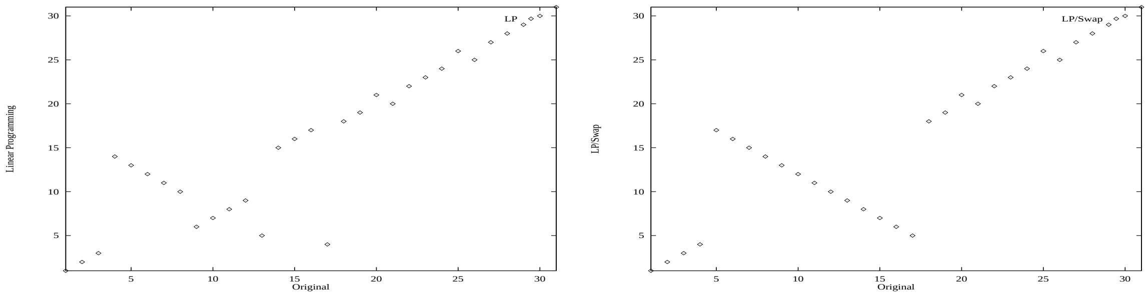

Figure 15: Simulated data. Clone coverage 5, Probe coverage 3, false negative rate 0.20, false positive rate 0.1, and chimera rate 0.1.

(ma) along with their weights in parentheses, called for the computed permutation, for each of the graphs displayed in Figures 6-13.

	Heuristic Score	Optimal Score	Ratio of Scores	Inversions & Translocations	Overcalled Errors
Figure 6	20.43	12.82	1.59	1 (i) + 1 (t)	3 fn (1.60) + 1 ma (2.94)
Figure 7	31.94	14.00	2.28	1 (i)	3 fp (13.14)+ 3 fn (4.80)
Figure 8	17.21	12.82	1.34	1 (i)	1 fp (4.38)
Figure 9	11.20	9.60	1.16	1 (i)	1 fn (1.60)
Figure 10	38.76	25.17	1.54	1 (i)	1 fp (4.38) + 1 ma (9.21)
Figure 11	15.58	15.58	1.00	0	
Figure 12	9.18	4.80	1.90	1 (t)	1 fp (4.38)
Figure 13	11.19	9.60	1.16	1 (i)	1 fn (1.60)

Table 2: A perspective on the quality of the permutation as estimated by the ratio of the heuristic score to the optimal score, versus the number of inversions and translocations required to edit the heuristic permutation into the optimal one. The last column represents the extra false negatives (fn), false positives (fp), and mislocated anchors (ma), called for the heuristic permutation.

- Most differences between the computed permutation and the original permutation involve swaps between probes or blocks of probes at adjacent positions. In general, these swaps are the result of an abundance of false negatives in the observed data. Thus the computed permutation actually conforms to the observed data. Due to these swaps, the percentage of correct adjacencies is a somewhat pessimistic measure of the quality of permutations produced. Counting translocations and inversions is a better measure but unfortunately difficult to compute.

- It is also clear that for all data, the swap-refined LP order is very nearly optimal/correct.

3 Data Decomposition and Anchor ordering

A real mapping project may have anywhere between five thousand to eight thousand clones and twelve hundred to three thousand probes. But our LP-based algorithm for physical mapping is only viable for problems up to 100 probes. This is also the size range of the largest linear ordering problems (recall the similarity) solved by cutting plane methods and the size of problems reported in the work of Alizadeh et al. [AKWZ95].

We therefore need to decompose a large problem into smaller, relatively independent subsets or clusters that can be ordered separately without significant loss of accuracy. We further desire that the clusters be small enough that we can solve them optimally using our LP-algorithm to provide an initial solution to a branch-and-bound algorithm. Given ordered clusters we then seek to connect those that can be linked with high confidence and position the clusters using whatever anchored probes are available. It is our opinion that producing such a partial map is superior to trying to optimize a complete solution as the links chosen across stretches where data is sparse or inaccurate are usually wrong. We discuss approaches for the three phases — clustering, ordering, and linking — in a sequence of subsections concluding with empirical results. Before launching into the details we give the following synopsis of our overall strategy for determining a physical map:

1. **Clustering.** Cluster together probes and clones that are highly likely to be in a contiguous region of the chromosome.

2. **Cluster Ordering.** For each cluster, compute an initial probe permutation for the probes within it, using the linear programming heuristic followed by the optimizing phase. If the number of probes in the cluster is less than 60, use branch and bound to compute the set of optimal probe permutations for the cluster. Otherwise use branch and bound to compute optimal probe permutations for small sections of the cluster.

3. **Linking Clusters.** Identify the highly likely connections between clusters. To do so, first delete the connections highly likely to be spurious as follows.

   - Delete connections between clusters containing anchors that are highly likely to be non-adjacent on the genome.
   - Lock in one and two edge paths between clusters containing anchors that are highly likely to be adjacent on the chromosome. Delete all other edges that interfere with the path, from the clusters being connected.

3.1 Clustering

Chimeras and false positives incorrectly suggest that regions of a genome may be close together when they are actually far apart. This is why these two types of errors are much more troublesome in comparison to false negatives, which at best lead to small inversions in the probe ordering and at worst make the map discontinuous. We therefore focused on a clustering method which filters false positives and chimeras. Recall that $N(p)$ is the set of clones incident to probe p. Given a threshold u, we say that p and q are u-connected, written $p \sim_u q$, iff $|N(p) \cap N(q)| \geq u$. Informally, two probes are highly likely to belong to the same region of the map if they are both incident to at least u of the same clones. The equivalence classes of probes induced by the reflexive and transitive closure, $\sim_u^\star$, of the relation $\sim_u$ are called *u-contigs.*

This partitioning of the probes constitutes our initial clustering of the data set. If δ is the rate of false positives and chimeras in the data, then the probability that a probe does not belong to its u-contig is δ^u. In practice, given $\delta \leq 0.1$, u set to two or three achieves an adequate level of accuracy in the cluster. Given a u-contig P, the corresponding *clone-cluster* $C_P = \cup_{p,q \in P} N(p) \cap N(q)$. Informally, a clone belongs to a cluster C_P if at least two probes in P are incident to it. Note that clone clusters are not necessarily disjoint.

It is important to observe that our clone clusters based on the u-contigs of probes, are not the same as the clusters of clones more traditionally built by linking clones that share u probes in common. The latter can lead to incorrect joinings due to the presence of chimeras in the data. For example, suppose $N(p) = N(q) = \{a, b\}$, $N(s) = N(t) = \{b, c\}$ and clone b is chimeric in that probes p and q are far away from s and t. Then our technique produces 2-contigs $\{p, q\}$ and $\{s, t\}$ and corresponding clone clusters $\{a, b\}$ and $\{b, c\}$, where as the clone-based linking would place all four probes and three clones in the same vicinity.

Our 2-contigs are the *double-linked islands* proposed by Arratia et al.[ALTW91]. These authors studied the coverage statistics of these versus 1-contigs. If there are an average of $w = 3$ probes per clone, than

95% of the genome will be covered by 1-contigs if the clone coverage c is 3.5, where as c must be 6 to guarantee the same level of 2-contig formation. To achieve 99% coverage, $c = 10$ is required for 2-contigs. Most experimentalists are using $c = 5$ in practice, so we cannot simply discard the clones and probes not in a 2-cluster. We attempt to utilize these in the final phase of our global strategy.

For the next phase, cluster ordering, the clusters must be on the order of 60 probes or less in order for our algorithms to solve them optimally. If clone coverage c is high, one may have a number of 2-contigs that are significantly larger than this. One approach is to subdivide the 2-contigs into 3-contigs and then apply a block-swapping heuristic to the 3-contigs. However, one invariably has end-probe information that can effectively be used to partially order the probes in a large contig.

Using End Clones for Ordering. Probes from the ends of clones are natural *splitters* of a u-contig, say P, into smaller sets. Consider a clone c in C_P for which $e, f \in P$ are end-probes extracted from the ends of c. Let L, M, and R be three probe sets that are initially $\{e\}$, all probes in $P - \{e, f\}$ incident to c, and $\{f\}$, respectively. For each probe p in $P - (L \cup M \cup R)$, add p to the set containing the probe q for which $N(p) \cap N(q)$ is the largest. Continue until all probes of p have been placed in one of the three sets. While clearly a heuristic procedure, it is highly likely that the partitioning of P into the three parts L, M, and R is correct, where, of course, M is between L and R.

By selecting a sequence of end-sequenced clones and computing the refinement of the 3-partitions computed heuristically from each, one can arrive at a progressively finer partitioning of a u-contig into ordered sub-blocks of the probes in a contig. Indeed, if one desired and there was sufficient end-clone information, one could use this procedure to compute an ordering of the probes. We chose to use it, however, primarily as a mechanism for accurately decomposing large 2-contigs into blocks sufficiently small for our intra-cluster ordering algorithms. In general, end-sequenced contigs with the greatest number of probes incident to them are preferable for this purpose as they best seed the set M in the heuristic. For example, Figure 16 is the result of ordering a probe set of size 84. The ends from four clones were used to partition the set into smaller sets of size 1, 29, 4, 4, 4, 3, 28, 4, 11, and 1, each of which were individually ordered, and the results then combined.

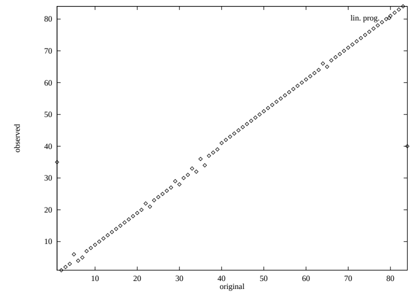

Figure 16: Simulated data. Clone coverage 8, probe coverage 4, false negative rate 0.20, false positive rate 0.01, and chimera rate 0.01.

3.2 Intra-Cluster Probe Ordering

Given a cluster of 60 probes or less the next problem is to optimally order the probes within the cluster. We use a branch-and-bound procedure that explores the space of all permutations of probes. As with all B&B algorithms the quality of the initial upper bound on the score is critical in determining how much of the space can be eliminated by the bounding step, and thus how large a problem one can solve. We use the swap-refined LP order as the initial permutation and its cost as the initial upper bound. At each step in the algorithm, we need to compute both the cost of extending a partial permutation π_i at the right end with a probe p, and the upper bound on the cost of completing this permutation. Both these costs are computed incrementally, using extensions to the recurrences of [JM95] for handling end-probe and anchor information, as described in a bit later in this section.

The algorithm explores the space of permutations by adding probes to the right-end of an initial prefix, π_i, of length i of a permutation. Given π_i as the current prefix, probes are considered for the $i+1^{\text{st}}$ spot in decreasing order of the number of clones they share in common with the rightmost probe of π_i, i.e., for probe p: $|N(p) \cap N(\pi_i(i))|$. Ties are broken arbitrarily. Given the selection of a next probe p, the algorithm sets its current prefix π_{i+1} to $\pi_i \cdot p$ and determines if any extension of π_{i+1} to a full permutation will score less than the current upper bound. If not then the algorithm backtracks, otherwise it proceeds with the extension of π_{i+1}. If a complete permutation π_m is achieved and it scores better than the upper bound, then the upper bound is reset and π_m is recorded as the instance achieving it. If the score equals the current upper bound, then π_m is added to the list of instances achieving the bound.

It remains to describe the lower bound on the score of the best possible extension of a prefix π_i used to cull the search. In effect what we know is the first i rows of D_π and we seek to bound the cost of the best completion of the matrix. To efficiently compute a lower bound in $O(n + lgm)$ time, we consider the best extension in which the values in each column may be permuted *independently* of each other. These can easily be scored using the recurrences presented in [JM95] extended to accommodate the longest increasing sequence computation required for scoring anchors.

Note that this B&B subprocedure is not a heuristic. It solves mapping problems with up to 60 clones optimally in less than an hour of compute time. This meets the size range handled by previous work which cannot guarantee optimality. For example, this procedure gave us the optimal orderings of the Drosophila contigs used for the x-axis in Figures 6-8. We conclude this section with a detailed treatment of how we compute the score of a potential permutation prefix π_i and its best possible extension, in the presence of end-clone and anchored probe information.

Modeling End-Clone Information: Given a permutation prefix π_i, it induces a column prefex V_i for the column of D_{π_i} for clone V. If the data were perfect, we would expect V's complete column V_m to be matche by the regular expression $R^1 = 0^\star e 1^\star e 0^\star$ where the two e's represent the probes known to be the two ends of V. But in general, if it is a k-chimeric clone, it would have the form $R^k = 0^\star e(1^\star 0^\star)^{k-1} 1^\star e 0^\star$. The best possible corrective sequence for V_m is the minimum cost series of false positive and negative corrections that take one from V_m to the given regular expression. In other words, we seek the minimum weighted Hamming distance, $\Delta(V_m, R^c)$ for each $c \in [1,k]$ where replacing a 1 with a 0 costs P and replacing a 0 with a 1 costs N. The overall minimum cost correction to the column is then $\min_c \Delta(V_m, R^c) + (c-1)C$. As in our paper [JM95], we developed a set of intertwining dynamic programming recurrences which compute $\Delta(V_i, Q)$ for all i and Q for each of the patterns $0^\star$, $0^\star e(1^\star 0^\star)^{c-1} 1^\star$, $0^\star e(1^\star 0^\star)^c$, and $0^\star e(1^\star 0^\star)^{c-1} 1^\star e 0^\star$ where $c \in [1,k]$. This permits the evaluation of the best possible correction and extension of V_i for π_i in $O(k)$ time given π_{i-1} and $\Delta(V_{i-1}, Q)$ for the $4k$ choices of Q.

Modeling Anchored Probes: Suppose the set of anchored probes A are known to be in the order $a_1 < a_2 < a_3 < \cdots < a_{|A|}$. Given a proposed probe order π, we need to determine the minimum number of anchored probes that can be considered to be out of order with respect to π. This amounts to solving the longest increasing subsequence problem [Man89] over the sequence $\pi_{a_1} \pi_{a_2} \cdots \pi_{a_{|A|}}$: the number of anchored probes not in the increasing subsequence are counted as out of ordered and penalized $M = \ln \frac{\alpha}{1-\alpha}$ apiece as described in the introduction. Such a subsequence can be found in $O(|A| \lg |A|)$. Moreover, given the state of this well-known algorithm for the partial order π_i and an ordered list of those probes not yet positioned by π_i, an easy exercise gives an upperbound on the best possible completion in $O(\lg A)$ time.

In real data sets, the anchored probes are not necessarily totally ordered but only positioned within intervals of the genome which may overlap with each other. This is true, for example, of the the Drosophila mapping project at Lawrence Berkeley Laboratories: probes are extracted both from genes and clones whose location along the previously constructed physical maps is known. Recall that both genes and clones are subintervals of the chromosome. Hence for each probe extracted from a pre-mapped gene, marker, or clone, the anchor information is in the form of a subinterval of the chromosome — the probe may occur at any point along this subinterval. For large subintervals such as genes, several probes may be extracted from the subinterval. For probes extracted from clones, the differences from total order occur when probes are extracted from overlapping subintervals.

To translate such subintervals back to our model of anchors, we order all subintervals according to their beginning locations along the chromosome. The probes are then assigned anchor locations according to the order of the subinterval in which they occur. Hence there may be several probes assigned to the same anchor location. In terms of scoring a permutation of such probes, the method for computing longest increasing subsequences is easily adapted to account for non-unique integers in the integer sequence. Unfortunately, our method of assigning anchor locations to probes may run into trouble with probes extracted from overlapping subintervals. As an example, let probe A be extracted from interval $[5,50]$, and probe B be extracted from interval $[30,70]$. Then by our method probe A is assigned to be anchor 1, and probe B is assigned to be anchor 2. It should be clear that this assigned order may be wrong. Probe A may have occurred at location 45, and probe B may have occurred at location 35. However, such irregularities in the assignment of an ordering to anchors are local, and limited to reversals in the ordering of adjacent anchors. So we accommodate such swaps by adapting the computation of the longest increasing subsequence accordingly. In effect, an incoming anchor a_i is taken to extend the longest increasing subsequences that end with a_x, where a_x is equal in value to a_{i-1}, a_i, or a_{i+1}.

3.3 Linking Clusters

Long range continuity of the physical map may be achieved by determining the connections of highest likelihood between clone clusters. To begin, consider a graph G_C whose vertices model each clone cluster of the probe partitioning step and each singleton clone that does not belong to any clone cluster. For a clone set C, let $P(C)$ be the set of probes incident to a clone in C. Note that for a clone cluster C_P for probe contig P, $P(C_P) \supseteq P$. Initially there is an undirected edge from C to D in the graph G_C iff $P(C) \cap P(D) \neq \emptyset$.

Our goal is to remove and/or direct as many of the spurious edges of G_C as possible so that in the culled graph unambiguous links between clusters become evident. Note that initially G_C contains so many spurious edges when the rates of false positives and chimeras are high, that it is highly likely that a path exists between any given pair of clusters. Note that unlike optimization procedures which are trying to select *a* path through G_C, our approach is to eliminate the clearly spurious parts and then report what is clearly unambiguously ordered in what remains.

The first and most obvious method of identifying spurious edges to and from a cluster is to use the set of optimal or near-optimal probe permutations determined in the previous stage. Clearly, any edges due to probes in the "middle" of such a permutation are highly likely to be spurious. Proceeding formally, suppose there is an edge between C and D because $P(C) \cap P(D) = \{p\}$, that $C = C_P$ is the clone cluster for probe contig P, and p is the x^{th} probe on a computed order for P. The probability that the edge is due to a true overlap is $\epsilon^{\min(x,|P|-x)}$ where ϵ is the rate of false negatives. We thus chose to delete all edges to and from a clone cluster that are due to probes that never occur at the first three locations or last three locations in any optimal probe ordering. For simulated data with probe coverage 3, clone coverages 5, 7, and 9, false negatives up to 20%, false positives up to 10%, and chimeras up to 25%, at least 98% of the edges thus deleted were observed to indeed be spurious.

Next we begin to exploit anchors in order to remove more edges and orient others. First we attempt to determine which anchors are *reliable*. If anchors a_i and a_j are both in probe contig P and $|i-j|$ is large, then it is more likely that one of the anchors is misplaced if $\delta^{\min(N(a_i),N(a_j))} < \alpha$ where α is the rate of mislocated anchors. If P contains several anchors then the anchor that is not near the majority is labeled unreliable, otherwise both a_i and a_j are labeled unreliable. All remaining anchors are considered reliable. Treating reliable anchors as correct, it then follows that all edges between clusters that contain reliable anchors that are far apart should be classed spurious. Also, all anchors in $P(C_P)$ far apart from the anchors in P must be false positives, and all edges adjacent to C_P dependent on these false positives should be deemed spurious. On simulated data at least 96% of the edges thus deleted are indeed spurious.

Finally, short paths between C and D when they contain reliable adjacent anchors are usually correct. We find that single edge paths are correct 95-97% of the time under high error-rate conditions. Two edge paths are found to be correct about 90% of the time. We thus chose to deem paths between such vertices *reliable* if their length was 2 or less. Once one "locks in" such connections it is then possible to eliminate more edges as spurious with respect to these reliable paths. Consider an edge (C, D) of a reliable path. All edges adjacent to C involving probes at the end of C overlapping D have a high probability of being spurious and are therefore deleted. Similarly, all edges adjacent to D involving probes at the end of D overlapping C are deleted. Also, the direction of the edge is established by the order of the reliable anchors in the probe sets of the vertices connected by the path containing the edge. Finally, if the edge (C, D) exists because $P(C) \cap P(D) = \{p\}$, we have effectively assigned the location of p and all other edges dependent on p may be deleted. In our empirical trials, at least 95% of the edges so deleted were indeed spurious.

At this point we place all clones at their reliably anchored locations and link them according to the reliable paths. We then seek any unambiguous connections that remain between clone clusters. We find in practice that the graph is disconnected, has the occasional unique path between clone clusters, but that even after the extensive culling above, there are still regions of the graph where alternatives exist. While one could select a path on the basis of some formal optimization criterion, it seems clear to us that doing so is no better than picking a path at random. There is simply insufficient reliable data to make the decision.

3.4 Computational Results

For the Drosophila data set, we compared our results with those produced at Lawrence Berkeley Labs. We found a 99% correlation between our results and their current maps. Nearly all differences involved an inversion of adjacent probes. To evaluate the quality of the permutations produced on simulated data, we use the following criteria.

- The fraction of the probes that were placed.

- The percentage of pairs of probes detected as being adjacent, and are actually so in the original permutation.

Table 3 summarizes the results.

Probe Coverage	Clone Coverage	Av. Cluster Size	Av. Path Size	Fraction of Probes Placed	% Adjacencies Found
2	5	4.35	4.91	0.85	79.55
2	7	5.71	5.90	0.90	87.53
3	5	5.66	7.89	0.88	84.93
3	7	8.92	9.51	0.97	90.66
4	5	9.26	12.00	0.90	84.04
4	7	15.20	16.94	0.96	71.90
5	5	13.02	20.63	0.93	83.69
5	7	17.65	30.16	0.96	72.97

Table 3: The fraction of the genome covered and the fraction of adjacencies detected for different probe rates, given clone coverage 5. Also, the average size of a clone cluster with $w = 2$, and the average size of a path after connections between clusters have been found.

- The table indicates that there is a significant difference between the size of the average cluster and the size of the average path after links between clusters have been frozen. This indicates that joining clusters is very necessary and leads to larger sets of ordered clones on average.

- The column on adjacencies shows encouraging results. The percentage of adjacencies discovered increases as the probe rate increases from 2 to 5, and as the clone coverage increases from 5 to 7. However, keeping clone coverage constant at 5, we discover that increasing the probe rate does not lead to an increase in the percentage of adjacencies discovered.

We hypothesize that the reason for this is insufficient clone coverage of some regions of the chromosome. For the regions of the chromosome covered by only one clone, for example, it does not matter how much we increase the probe rate, as the unreliability of inferences in such regions causes the algorithms to not make decisions here, and thus miss the adjacencies in that region. In effect, the percent adjacencies found reflects the percentage of the chromosome with good clone coverage. This reasoning is borne out by the fact that the percentage of adjacencies increases dramatically when the clone coverage is increased from 5 to 7, for probe rates 2 and 3.

We also believe that these figures indicate that the percentage of clones ordered correctly may be a better indicator of algorithm quality than the percentage of adjacencies computed.

4 Conclusions and Future Work

In this paper we have presented a versatile framework for the representation of STS physical mapping data. The use of constraints makes it easy to incorporate most current data types — random probe-clone incidences, end-probes, genetic and in-situ information, and results of repeated and pooled experiments. This implies easy incorporation of future data types too. A relaxation of the framework proves to be practical for large data sets, and provides near-optimal probe orderings. This approach proves to be more robust to errors in general, and false positive errors in particular, as compared to previous approaches. We show good probe orderings for data sets containing high error rates.

Some interesting future work is to design approximation algorithms using this framework. The use of branch-and-cut methods to derive probe orderings with this formulation deserve exploration. Finally, this formulation assumes the uniqueness of STS probes. An approach dual to the one used to filter false positives may work to filter repeats.

References

[AKWZ95] Farid Alizadeh, Richard M. Karp, Deborah K. Weisser, and Geoffrey Zweig. Physical mapping of chromosomes using unique probes. *Proceedings of the 5th ACM-SIAM Symposium on Discrete Algorithms*, 1995.

[ALTW91] Richard Arratia, Eric S. Lander, Simon Tavare, and Michael S. Waterson. Genomic mapping by anchoring random clones: A mathematical analysis. *Genomics*, 11:806–827, 1991.

[BL76] Kellogg S. Booth and George S. Leuker. Testing for the consecutive ones property, interval graphs, and graph planarity using pq-tree algorithms. *J. Comp. and Syst. Sci.*, 13:335–379, 1976.

[CJK+97] Thomas Christof, Michael Jünger, John Kececioglu, Petra Mutizel, and Gerhard Reinelt. A branch-and-cut approach to physical mapping with end-probes. *Proceedings of the First Annual International Conference on Computational Molecular Biology*, pages 84–92, 1997.

[CR96] Vijay Chandru and M. R. Rao. Combinatorial optimization: An integer programming perspective. *ACM Computing Surveys*, 28:55–58, March 1996.

[GKS94] Martin Charles Golumbic, Haim Kaplan, and Ron Shamir. On the complexity of dna physical mapping. *Advances in Applied Mathematics*, 15:251–261, 1994.

[JM95] Mudita Jain and Eugene W. Myers. A note on scoring clones given a probe ordering. *Journal of Computational Biology*, 2:33–37, 1995.

[JRT95] Michael Jünger, Gerhard Reinelt, and Stefan Theinel. Practical problem solving with cutting plane algorithms in combinatorial optimization. *DIMACS Series in Discrete Mathematics*, 20:111–152, 1995.

[Kou77] Lawrence T. Kou. Polynomial complete consecutive information retrieval problems. *SIAM Journal on Computing*, 6(1):67–75, March 1977.

[Man89] Udi Manber. *Introduction to Algorithms — A Creative Approach.* Addison-Wesley, Reading, Massachussetts, 1989.

[Opa79] J. Opatrny. Total ordering problem. *SIAM Journal of Computing*, 8:111–114, 1979.

[PG85] M. W. Padberg and M. Grötschel. Polyhedral computations. In E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, editors, *The Travelling Salesman Problem*, pages 307–360. John Wiley & Sons Ltd., 1985.

[Rei85] Gerhard Reinelt. *The Linear Ordering Problem: Algorithms and Applications.* Helderman Verlag, Berlin, 1985.

[RT87] P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[Shm95] David B. Shmoys. Computing near-optimal solutions to combinatorial optimization problems. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 20:355–397, 1995.