# A branch-and-cut approach to physical mapping with end-probes

Thomas Christof [*] Michael Jünger [†] John Kececioglu [‡] Petra Mutzel [§] Gerhard Reinelt[*]

## Abstract

A fundamental problem in computational biology is the construction of physical maps of chromosomes from hybridization experiments between unique probes and clones of chromosome fragments in the presence of error. Alizadeh, Karp, Weisser and Zweig [AKWZ94] first considered a maximum-likelihood model of the problem that is equivalent to finding an ordering of the probes that minimizes a weighted sum of errors, and developed several effective heuristics. We show that by exploiting information about the end-probes of clones, this model can be formulated as a weighted Betweenness Problem. This affords the significant advantage of allowing the well-developed tools of integer linear-programming and branch-and-cut algorithms to be brought to bear on physical mapping, enabling us for the first time to solve small mapping instances to optimality even in the presence of high error. We also show that by combining the optimal solution of many small overlapping Betweenness Problems, one can effectively screen errors from larger instances, and solve the edited instance to optimality as a Hamming-Distance Traveling Salesman Problem. This suggests a new combined approach to physical map construction.

**Key words** Computational biology, physical mapping of chromosomes, betweenness problem, linear ordering problem, branch-and-cut.

## 1 Introduction and background

### 1.1 Motivation

Each human chromosome is a linear sequence of roughly $10^8$ bases. To aid manipulation of DNA of this scale in the laboratory, and to prepare it for sequencing, *physical maps* of chromosomes are constructed that give the location along the molecule of important features, such as the location of clones of DNA fragments.

In this paper, we consider the construction of physical maps by a protocol known as *STS-content mapping.* In this strategy, which is widely used in the Human Genome Project, each *clone* corresponds to an interval of the chromosome, and each *probe* corresponds to a unique point on the chromosome. While the position of clones and probes along the chromosome is unknown, it can be determined whether a clone contains a probe by a test called a *hybridization* experiment. (The experiment tests whether the probe DNA bonds, or *hybridizes*, with the clone DNA.) As hybridization experiments are inevitably imperfect, the resulting clone-probe incident data contains errors. In a *false positive* error, the experiment reports that a clone contains a probe when it does not, while in a *false negative* error, the experiment reports that a clone does not contain a probe when it does. The goal is to recover the probe or clone ordering from such hybridization data.

We consider the problem of STS-content mapping with false positive and false negative errors. In practice, an additional type of error, called *chimerism*, occurs. A chimeric clone is a clone that does not sample a single interval of the chromosome, but contains two or more unrelated fragments of DNA. While we concentrate on handling false positive and negative errors, our approach can also be extended to data with chimeric clones, as indicated in the final section.

In practice, the set of probes is obtained by extracting DNA from the ends of clones. Usually probes are obtained from a subset of the clones, and are not consistently extracted from both ends. For any given clone, however, the probes that were extracted from its ends are known, and can be identified.

In this paper, we assume that probes are extracted from *both* ends of *every* clone. (We do not assume any information about which end a probe came from.) We show that by consistently selecting probes from both ends of a clone, the problem of reconstructing the probe order in the presence of false positives and negatives can be successfully tackled by integer linear programming. Our computational results suggest that this change to the experimental protocol could significantly improve the quality of the physical maps that are constructed, while tolerating a much higher experimental error rate. Furthermore, our integer linear programming formulation, which is based on the Weighted Betweenness Problem, can handle partial-order information on probes, and is actually simplified by such information. Our approach

[*] *Institut für Angewandte Mathematik, Universität Heidelberg, Germany, e-mail:* thomas.christof@iwr.uni-heidelberg.de

[†] *Institut für Informatik, Universität zu Köln, Germany, e-mail:* mjuenger@informatik.uni-koeln.de

[‡] *Department of Computer Science, Univ. of Georgia, Athens, USA, e-mail:* kece@cs.uga.edu

[§] *Max-Planck-Institut für Informatik, Saarbrücken, Germany, e-mail:* mutzel@mpi-sb.mpg.de

can also be extended to general probe-clone hybridization data where clones do not have probes extracted from both ends, as indicated in the final section.

### 1.2 Related work

We use the following mathematical description of the problem, also used by Alizadeh et al. [AKWZ94] and Greenberg and Istrail [GI95]. The chromosome is mapped by $n$ probes, $P_1, \ldots, P_n$, and $m$ clones, $C_1, \ldots, C_m$. The outcome of the probe-clone hybridization experiments is given by an $m \times n$ 0-1 matrix $A$ whose rows correspond to clones and whose columns correspond to probes. An entry $a_{ij}$ is 1 if probe $P_j$ hybridizes to clone $C_i$, and 0 otherwise. Entry $a_{ij} = 1$ is a false positive if the entry should be 0, while entry $a_{ij} = 0$ is a false negative if it should be 1.

In the absence of error and knowledge of end-probes, the problem of reconstructing the probe order is equivalent to the Consecutive Ones Problem: find a permutation $\pi$ of the columns of matrix $A$ so that in the reordered matrix $A^\pi$, the ones in every row are consecutive. Using the PQ-tree of Booth and Lueker [BL76], a representation of all permutations $\pi$ that have the consecutive ones property can be computed in time linear in the number of ones in matrix $A$. In the presence of error, however, this approach breaks down.

Biologists and computer scientists have tackled the problem in several ways. Typically, maximum-likelihood functions are suggested and their solution is attempted by local search (see [AKWZ94, MHM$^+$93]). Another approach is to approximate the maximum-likelihood function by a well-studied combinatorial problem, such as the Hamming-Distance Traveling Salesman Problem ([AKWZ94, AKNW95, GI95]). Methods for filtering the data have also been offered as an attempt to remove typical errors such as false positives or chimeric clones [GDHC95, MGL94]. Further pointers to the literature are given in [VLM96]. It is worth emphasizing that all these approaches are particularly sensitive to false-positive errors, and are successful only for relatively low false-positive rates.

Alizadeh et al. [AKWZ94] first introduced the maximum-likelihood model that we consider. The idea is to find a corrected matrix $B$ that maximizes $p(B|A)$, the probability that $B$ is the true hybridization matrix, given the observed matrix $A$. Using Bayes' Theorem, they show

$$\operatorname*{argmax}_B\{p(B|A)\} = \operatorname*{argmax}_B$$
$$\left\{ \left(\frac{p_\chi}{1-p_\chi}\right)^{n_\chi(A|B)} \left(\frac{p_\rho}{1-p_\eta}\right)^{n_\rho(A|B)} \left(\frac{p_\eta}{1-p_\rho}\right)^{n_\eta(A|B)} \right\},$$

where $n_\chi(A|B)$ is the number of rows of $A$ that are declared chimeric with respect to $B$, $n_\rho(A|B)$ is the number of false positives in $A$ with respect to $B$, $n_\eta(A|B)$ is the number of false negatives in $A$ with respect to $B$, $p_\chi$ is the probability that a clone is chimeric, $p_\rho$ is the probability that an entry is a false positive, and $p_\eta$ is the probability that an entry is a false negative. Choosing $\chi = -\ln \frac{p_\chi}{1-p_\chi}$, $\rho = -\ln \frac{p_\rho}{1-p_\eta}$, and $\eta = -\ln \frac{p_\eta}{1-p_\rho}$ gives

$$\operatorname*{argmax}_B\{p(B|A)\} =$$
$$\operatorname*{argmin}_B\{\chi\, n_\chi(A|B) + \rho\, n_\rho(A|B) + \eta\, n_\eta(A|B)\}.$$

The problem then is to find a probe ordering $\pi$ that minimizes the linear objective function

$$f(\pi) = \min_B\ \{\chi\, n_\chi(A^\pi|B) + \rho\, n_\rho(A^\pi|B) + \eta\, n_\eta(A^\pi|B)\}.$$

Evaluating $f$ for a given $\pi$ requires finding the best matrix $B$; Jain and Myers [JM95] show how this can be done efficiently using dynamic programming. Given the NP-hardness of minimizing $f(\pi)$ [Boo75], Alizadeh et al. attack the problem using local search.

### 1.3 Plan of the paper

We model mapping with end-probes, in the absence of chimeric clones, as the Weighted Betweenness Problem. Given a collection of betweenness and nonbetweenness constraints on a set of elements to be linearly ordered, the Weighted Betweenness Problem asks for an ordering $\pi$ of the elements that minimizes a weighted sum of the constraints violated by $\pi$ (see Section 2). We then give an integer linear programming formulation of Weighted Betweenness based on the Linear Ordering Problem.

When relaxing the integer linear program by dropping integrality constraints, one often needs additional inequalities to get good solutions with a linear programming method. Section 3 describes our approach for obtaining useful new inequalities. These inequalities are then used in a branch-and-cut algorithm (see Section 4).

Our experiments on generated data verify that under the maximum-likelihood objective, the correct order is an optimal solution, or very close to an optimal solution. Conversely, our exact solution of the betweenness problem gives, in most cases, the original probe ordering (see Section 5). Section 6 describes our plans for further research, and indicates how our approach can be extended to handle chimerism as well as the absence of end-probe information.

## 2 Reducing physical mapping with end-probes to integer linear programming

We formulate the problem of minimizing $f(\pi)$ in the absence of chimerism in terms of the Weighted Betweenness Problem. Consider the $r$-th row of $A$ corresponding to clone $C_r$. Let columns $i$ and $k$ correspond to the end-probes $P_i$ and $P_k$ of clone $C_r$. Consider a probe $P_j$, other than $P_i$ or $P_k$, that hybridized with clone $C_r$, i.e. where $a_{rj} = 1$. If entry $a_{rj} = 1$ is correct, in the correct order $\pi$ of the probes column $j$ should be between columns $i$ and $k$. We denote this by the triple $(i, j, k)$, which we call a *betweenness constraint.* Since we do not know the relative order of end-probes, both the ordering $i \cdots j \cdots k$ and $k \cdots j \cdots i$ are consistent with betweenness constraint $(i, j, k)$.

Now consider a probe $P_j$, again different from $P_i$ and $P_k$, that did *not* hybridize with clone $C_r$, i.e. where $a_{rj} = 0$. If entry $a_{rj} = 0$ is correct, in the correct order $\pi$ of the probes, column $j$ should *not* be between columns $i$ and $k$, which we denote by the triple $\overline{(i, j, k)}$, and call a *nonbetweenness constraint.* Both the orderings $k \cdots i \cdots j$, $i \cdots k \cdots j$, and their reverse, are all consistent with the constraint $\overline{(i, j, k)}$. For an instance of the problem, we denote the set of betweenness constraints by $\mathcal{B}$, and the set of nonbetweenness constraints by $\overline{\mathcal{B}}$. Notice that any triple $i, j, k$ occurs only once in $\mathcal{B}$ or $\overline{\mathcal{B}}$.

The key advantage of having end-probe information is that we can express the number of false positives and false negatives in a probe ordering $\pi$, which are *global* properties of $\pi$ usually requiring dynamic programming to compute [JM95, AKWZ94], in terms of *local* betweenness and nonbetweenness constraints. The false positives in row $r$ are exactly those columns $j$ for which $a_{rj} = 1$ but constraint $(i, j, k)$ is violated by $\pi$, where $P_i$ and $P_k$ are the end-probes of $C_r$; each such $j$ costs a penalty $\rho$ in the objective function $f(\pi)$. The false negatives in row $r$ are exactly those columns $j$ for which $a_{rj} = 0$ but constraint $\overline{(i,j,k)}$ is violated; each such $j$ costs a penalty $\eta$. Thus the problem is equivalent to finding a $\pi$ that minimizes the weighted sum of the constraints that it violates. We note that this formulation is slightly more general than the classical Betweenness Problem [Opa79], in that we have both betweenness and nonbetweenness constraints, and we are optimizing a weighted sum of violations.

Opatrny [Opa79] has shown that simply deciding whether a set of elements can be linearly ordered to satisfy a collection of betweenness constraints is NP-complete. Chor and Sudan [CS95] present an approximation algorithm for the classical Betweenness Problem that either finds a feasible solution, or finds a linear order that satisfies at least one-half of the constraints. They do not consider nonbetweenness constraints or weighted constraints.

### 2.1 Variables

Our formulation of the Weighted Betweenness Problem can be expressed as an integer linear program by introducing linear ordering variables. For every ordered pair $(i, j)$ of probes, we introduce a variable $x_{ij}$ with the interpretation

$$x_{ij} = \begin{cases} 1 & \text{if } i \text{ precedes } j \text{ in the ordering } \pi, \\ 0 & \text{otherwise.} \end{cases}$$

Thus the probe order $\pi$ is represented by the set of variables $x_{ij}$.

With every betweenness constraint $(i, j, k)$ from set $\mathcal{B}$, we associate the variable $b_{ijk}$ with

$$b_{ijk} = \begin{cases} 0 & \text{if constraint } (i,j,k) \text{ is met,} \\ 1 & \text{otherwise.} \end{cases}$$

Thus $b_{ijk}$ counts whether or not betweenness constraint $(i, j, k)$ is violated.

As it will turn out, it is more natural in the linear programming formulation to deal with betweenness constraints rather than nonbetweenness constraints; hence we will express nonbetweenness constraints in terms of betweenness constraints. With every nonbetweenness constraint $\overline{(i,j,k)}$ from set $\overline{\mathcal{B}}$, we associate a variable $b_{ijk}$ with

$$b_{ijk} = \begin{cases} 0 & \text{if constraint } (i,j,k) \text{ is met,} \\ 1 & \text{otherwise.} \end{cases}$$

Thus $1 - b_{ijk}$ counts whether or not $\overline{(i,j,k)}$ is violated.

### 2.2 Integer programming formulation

To ensure that the variables $x_{ij}$ encode a linear ordering $\pi$ of the probes, the following conditions must be met:

$$\begin{array}{rll} 0 \le x_{ij} \le 1 & \text{for} & 1 \le i \ne j \le n \qquad (1) \\ x_{ij} + x_{ji} = 1 & \text{for} & 1 \le i \ne j \le n \qquad (2) \\ x_{ij} + x_{jk} + x_{ki} \le 2 & \text{for} & 1 \le i \ne j \ne k \le n \qquad (3) \\ x_{ij} \text{ integral} & \text{for} & 1 \le i \ne j \le n. \qquad (4) \end{array}$$

Conditions (2) and (3) enforce the antisymmetry and transitivity properties of a linear ordering.

With these conditions, every feasible assignment of the $x$-variables corresponds to a linear ordering $\pi$ of the probes. To ensure that the $b_{ijk}$ count violations of the betweenness and nonbetweenness constraints, we add the following inequalities. To force a $b_{ijk} \in \mathcal{B}$ to be 1 if $(i, j, k)$ is violated, we add

$$\begin{aligned} b_{ijk} &\ge x_{ij} - x_{jk} && (5) \\ b_{ijk} &\ge x_{jk} - x_{ij}. && (6) \end{aligned}$$

Thus $b_{ijk} \in \mathcal{B}$ is 1 if $|x_{ij} - x_{jk}| = 1$.

To force a $b_{ijk} \in \overline{\mathcal{B}}$ to be 0 if $\overline{(i,j,k)}$ is violated, we add

$$\begin{aligned} b_{ijk} &\le x_{ij} + x_{jk} && (7) \\ b_{ijk} &\le 2 - (x_{ij} + x_{jk}). && (8) \end{aligned}$$

Thus $b_{ijk} \in \overline{\mathcal{B}}$ is 0 if $|x_{ij} - x_{jk}| = 0$.

Due to the form of the objective function, we do not have to explicitly require integrality of the $b_{ijk}$. The coefficient on a $b_{ijk} \in \mathcal{B}$ in the objective function is positive, while the coefficient on a $b_{ijk} \in \overline{\mathcal{B}}$ is negative, so it suffices to require

$$0 \le b_{ijk} \le 1. \qquad (9)$$

Finally, we seek an assignment of the variables that minimizes

$$\rho \sum_{b_{ijk} \in \mathcal{B}} b_{ijk} + \eta \sum_{b_{ijk} \in \overline{\mathcal{B}}} (1 - b_{ijk}) =$$

$$\rho \sum_{b_{ijk} \in \mathcal{B}} b_{ijk} - \eta \sum_{b_{ijk} \in \overline{\mathcal{B}}} b_{ijk} + \eta|\overline{\mathcal{B}}|.$$

Since $\eta|\overline{\mathcal{B}}|$ is a constant for any particular problem instance, this is equivalent to minimizing the objective function

$$\rho \sum_{b_{ijk} \in \mathcal{B}} b_{ijk} - \eta \sum_{b_{ijk} \in \overline{\mathcal{B}}} b_{ijk}.$$

Note that while the number of variables is $\Theta(n^2 + mn)$ and the number of inequalities is $\Theta(n^3 + mn)$, during the execution of a branch-and-cut algorithm only a subset of the inequalities are included in the linear programs.

Constraints (1) to (9) guarantee that the solutions are precisely all possible linear orderings. Since the Betweenness Problem is NP-hard, we cannot hope for a general solution of the above integer linear program (ILP). Instead, we approach the ILP by solving the relaxed linear program (LP) in which the integrality constraints are removed. Successively tighter relaxations are achieved by adding valid, but violated linear inequalities to the LP. In fact, the set of solutions to the ILP is given by some set of linear inequalities, though the NP-hardness of the problem makes it unlikely that such a complete linear description can be found in general and exploited algorithmically. In the next section, we describe further valid inequalities that have enabled us to solve mapping instances exactly. To simplify the exposition, we refer to only one set $\mathcal{B}$ of betweenness variables, where

$$b_{ijk} = \begin{cases} 0 & \text{if } j \text{ is between } i \text{ and } k \text{ in the ordering,} \\ 1 & \text{otherwise.} \end{cases}$$

## 3 Valid inequalities

Suppose we are given an instance of the Weighted Betweenness Problem by a number $n$ of probes and a set $\mathcal{B}$ of $t$ triples. With every permutation $\pi = (\pi_1\ \pi_2\ \cdots\ \pi_n)$ of the probes we associate incidence vectors $x^\pi \in \{0,1\}^{n(n-1)}$ and $b^\pi \in \{0,1\}^t$ with

$$x^\pi_{ij} = \begin{cases} 1 & \text{if} \quad \pi_i < \pi_j, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$b^\pi_{ijk} = \begin{cases} 0 & \text{if} \quad \pi_i < \pi_j < \pi_k \quad \text{or} \quad \pi_k < \pi_j < \pi_i \\ 1 & \text{otherwise.} \end{cases}$$

The polytope $\mathcal{P}(n, \mathcal{B})$ associated with an instance of the betweenness problem is defined as

$$\text{conv}\left(\left\{\begin{pmatrix} x^\pi \\ b^\pi \end{pmatrix} \mid \pi \text{ is a permutation of the probes}\right\}\right).$$

The vertices of $\mathcal{P}(n, \mathcal{B})$ are 0-1 vectors and correspond to feasible solutions of the Weighted Betweenness Problem.

### 3.1 Trivial lifting

Let $a^T x \le b$ be a valid inequality for $\mathcal{P}(n', \mathcal{B}')$. The inequality $a^T x \le b$ is also valid for any $\mathcal{P}(n, \mathcal{B})$ with $n \ge n'$ and $\mathcal{B} \supseteq \mathcal{B}'$. Thus linear descriptions of small problem instances give valid inequalities for larger problem instances.

### 3.2 Small problem instances

We can easily compute the complete linear description of small problem instances by enumerating all feasible incidence vectors and applying the double-description method for computing the linear description of the convex hull of the set of the vectors. Christof and Reinelt [CR96] discuss algorithmical details and software for that transformation. They successfully use inequalities from small polytopes in branch-and-cut algorithms for the linear ordering problem and the traveling salesman problem. We present here the linear inequalities of those polytopes that proved to be useful in the separation procedure of our branch-and-cut algorithm. The polytopes are associated with problem instances on 3 and 4 nodes, and certain combinations of betweenness conditions. We do not list trivial inequalities $x_{ij} \le 1$ and $x_{ij} \ge 0$, or dicycle inequalities $x_{ij} + x_{jk} + x_{ki} \le 2$ on the linear ordering variables. These inequalities define facets of the polytopes only in certain cases.

$\mathcal{P}(3, \{\{1-2-3\}\})$

$$x_{12} + x_{32} \quad -b_{123} \quad \le 1 \tag{10}$$
$$x_{21} + x_{23} \quad -b_{123} \quad \le 1 \tag{11}$$
$$x_{12} + x_{23} \quad +b_{123} \quad \le 2 \tag{12}$$
$$x_{21} + x_{32} \quad +b_{123} \quad \le 2 \tag{13}$$
$$x_{12} + x_{23} + 2x_{31} \quad -b_{123} \quad \le 2 \tag{14}$$
$$2x_{13} + x_{21} + x_{32} \quad -b_{123} \quad \le 2 \tag{15}$$

In the following, we do not list facets that are already given by $\mathcal{P}(3, \{\{1-2-3\}\})$.

$\mathcal{P}(4, \{\{1-2-3\}, \{1-4-3\}\})$

$$x_{12} + 2x_{24} + x_{32} + x_{41} + x_{43} \quad -b_{123} + b_{143} \quad \le 4 \tag{16}$$
$$x_{12} + 2x_{24} + x_{32} + x_{41} + x_{43} \quad +b_{123} - b_{143} \quad \le 4 \tag{17}$$
$$x_{14} + x_{21} + x_{23} + x_{34} + 2x_{42} \quad -b_{123} + b_{143} \quad \le 4 \tag{18}$$
$$x_{14} + x_{21} + x_{23} + x_{34} + 2x_{42} \quad +b_{123} - b_{143} \quad \le 4 \tag{19}$$

$\mathcal{P}(4, \{\{1-2-3\}, \{2-1-4\}\})$

$$x_{14} + x_{32} + 2x_{43} \quad -b_{123} - b_{214} \quad \le 2 \tag{20}$$
$$x_{23} + 2x_{34} + x_{41} \quad -b_{123} - b_{214} \quad \le 2 \tag{21}$$
$$x_{14} + x_{23} + 2x_{34} + 2x_{42} \quad -b_{123} - b_{214} \quad \le 3 \tag{22}$$
$$x_{14} + x_{23} + 2x_{31} + 2x_{43} \quad -b_{123} - b_{214} \quad \le 3 \tag{23}$$
$$2x_{13} + x_{32} + 2x_{34} + x_{41} \quad -b_{123} - b_{214} \quad \le 3 \tag{24}$$
$$2x_{24} + x_{32} + x_{41} + 2x_{43} \quad -b_{123} - b_{214} \quad \le 3 \tag{25}$$

$\mathcal{P}(4, \{\{1-2-3\}, \{2-1-4\}, \{1-4-3\}\})$

$$x_{32} + x_{43} - b_{123} - b_{214} - b_{143} \quad \le 0 \tag{26}$$
$$x_{23} + x_{34} - b_{123} - b_{214} - b_{143} \quad \le 0 \tag{27}$$
$$x_{23} + 2x_{31} + x_{43} - b_{123} - b_{214} - b_{143} \quad \le 1 \tag{28}$$
$$2x_{13} + x_{32} + x_{34} - b_{123} - b_{214} - b_{143} \quad \le 1 \tag{29}$$
$$x_{12} + 2x_{24} + 2x_{32} + x_{43} - b_{214} - b_{143} \quad \le 3 \tag{30}$$
$$x_{14} + x_{23} + 2x_{34} + 2x_{42} - b_{123} - b_{214} \quad \le 3 \tag{31}$$
$$x_{21} + 2x_{23} + x_{34} + 2x_{42} - b_{214} - b_{143} \quad \le 3 \tag{32}$$
$$2x_{24} + x_{32} + x_{41} + 2x_{43} - b_{123} - b_{214} \quad \le 3 \tag{33}$$
$$x_{12} + 2x_{24} + x_{32} + x_{41} + x_{43} - b_{123} + b_{143} \quad \le 4 \tag{34}$$
$$x_{12} + 2x_{24} + x_{32} + x_{41} + x_{43} + b_{123} - b_{143} \quad \le 4 \tag{35}$$
$$x_{14} + x_{21} + x_{23} + x_{34} + 2x_{42} - b_{123} + b_{143} \quad \le 4 \tag{36}$$
$$x_{14} + x_{21} + x_{23} + x_{34} + 2x_{42} + b_{123} - b_{143} \quad \le 4 \tag{37}$$

Note that inequalities (30) – (33) can be transformed to inequalities (22) and (25) and inequalities (34) – (37) are inequalities (16) – (19).

$\mathcal{P}(4, \{\{1-2-3\}, \{2-1-4\}, \{1-4-3\}, \{2-3-4\}\})$

$$+b_{123} + b_{214} + b_{143} + b_{234} \quad \ge 2 \tag{38}$$
$$+b_{123} + b_{214} - b_{143} + b_{234} \quad \le 2 \tag{39}$$
$$+b_{123} - b_{214} + b_{143} + b_{234} \quad \le 2 \tag{40}$$
$$-b_{123} + b_{214} + b_{143} + b_{234} \quad \le 2 \tag{41}$$
$$+b_{123} + b_{214} + b_{143} - b_{234} \quad \le 2 \tag{42}$$

In addition inequalities (28) – (29) and the inequalities (16) – (19) define facets of the polytope.

### 3.3 Projection

By making use of the equation $x_{ij} + x_{ji} = 1$ we can define a standard form of inequalities with all coefficients of the linear ordering variables having nonnegative value. Within a branch-and-cut algorithm, it is more efficient to use the equation $x_{ij} + x_{ji} = 1$ to eliminate variables $x_{ij}$ with $j > i$, since it reduces the number of variables substantially . This elimination converts the linear ordering dicycle inequalities $x_{ij} + x_{jk} + x_{ki} \le 2$ to inequalities $x_{ij} + x_{jk} - x_{ik} \le 1$ and $x_{ij} + x_{jk} - x_{ik} \ge 0$ with $i < j < k$.

## 4 The branch-and-cut algorithm

Our physical-mapping algorithm is based on the branch-and-cut algorithm of [GJR84, GJR85] for the classical Linear Ordering Problem. Reimplementation was done using the software system ABACUS [Thi95, JRT95], a general, object-oriented framework for implementing branch-and-cut algorithms. ABACUS' object-oriented design in C++ proved to be extremely useful. Since classical linear ordering and betweenness are strongly related, functions and data structures common to both problems could be implemented in common base classes.

The algorithm starts with a linear program containing only constraints (1), and iteratively adds violated constraints after an LP has been solved. We test violation of the constraints in the ILP and the ones described in Section 3. The above step is repeated until we get an integral solution vector satisfying all the constraints of type (3), (5)–(9) or until we cannot find any violated inequalities. In the first case, we have found a provably optimal solution, while in the second case the same procedure is applied recursively to two subproblems, one in which a fractional variable $x_{ij}$ is set to 1, and the other in which the variable is set to 0.

Obviously for every solution for the betweenness problem corresponding to a permutation $\pi = (\pi_1 \pi_2 \cdots \pi_n)$ the reverse permutation $(\pi_n \cdots \pi_2 \pi_1)$ is a feasible solution with the same objective function. This allows us to fix one arbitrary variable $x_{ij}$ as $x_{ij} = 1$ before solving the first relaxation.

## 5 Computational results

### 5.1 Generator

For our computational experiments we used a generator similar to the one suggested by Greenberg and Istrail [GI95]. The distinct advantage of using simulated data is that we know the correct order of the probes, and hence we can reliably measure the success of our method. For testing our algorithm, we generated data where each clone gave rise to two end-probes, all clones were of the same length, and were randomly distributed across the chromosome. We used a coverage varying from 3 to 5, a false negative rate of 10%, and a false positive rate varying from 0% to 5%. To generate a false positive or false negative, a coin was flipped at each entry with the given probability. Across experiments with varying coverage, the clone length is held constant.

### 5.2 Exact solutions

For comparison, we also considered the common approach to physical mapping of solving a Hamming-Distance Traveling Salesman Problem (HDTSP) (see, e.g., [AKWZ94] and [GI95]). The nodes of a HDTSP are the columns of our hybridization matrix $A$, and the distances $d_{ij} = d_{ji}$ are given by the Hamming distance between the corresponding columns, i.e., the number of rows of $A$ with $a_{ri} \neq 0, a_{rj} = 0$. To obtain a Hamilton cycle problem, an artifical column is introduced in $A$ with all entries equal to 0. The resulting symmetric Traveling Salesman Problems were solved to optimality using the ABACUS version [Thi95] of the branch-and-cut code of Jünger, Reinelt, and Thienel [JRT94].

Figures 1 to 12 show results from 150 problem instances. Within each plot, the horizontal axis indicates the false positive rate. The three plots in each figure are for coverage 3, 4, and 5.

We first compared the solution values of the Weighted Betweenness Problem and the HDTSP with the value of the known correct solution, to determine whether exact solution of the maximum likelihood model is worthwhile for recovering the true map. As shown in Figures 1–3, the values of the optimal solution of the betweenness problem and the correct solution differ in only a fraction of the instances, and then by quite small amounts. This suggests that the maximum-likelihood function may be useful in identifying the correct probe order, even on instances with high false positive rates.

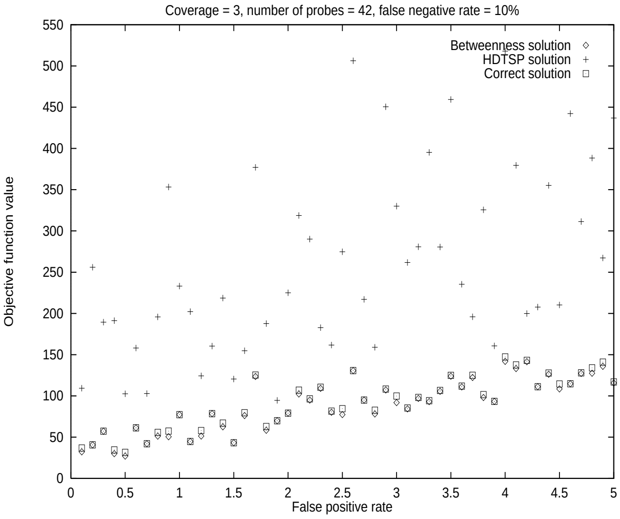

Figure 1: Optimal Solution Values (coverage=3)

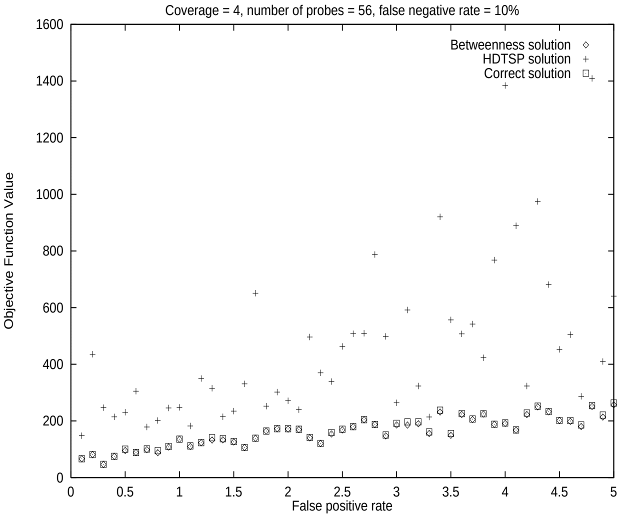

Figure 2: Optimal Solution Values (coverage=4)

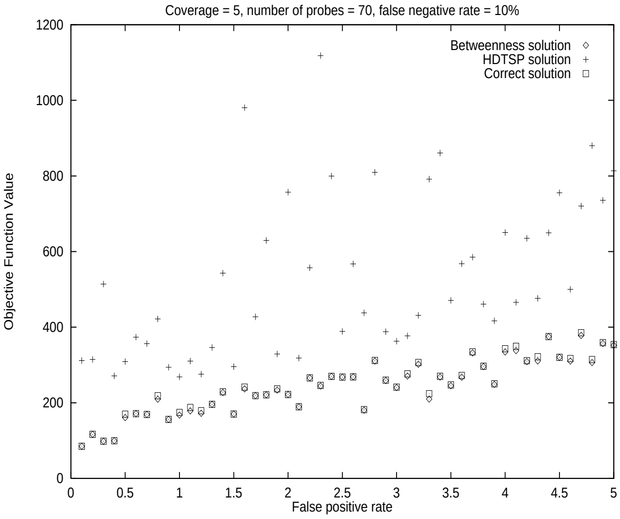

Figure 3: Optimal Solution Values (coverage=5)

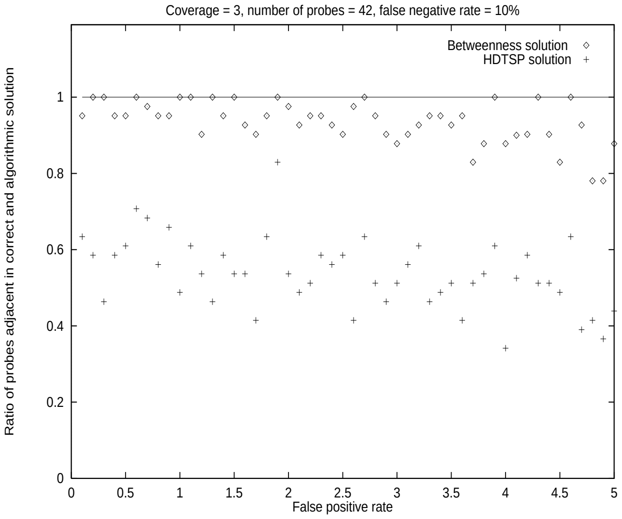

Figure 4: Adjacency-Quality (coverage=3)

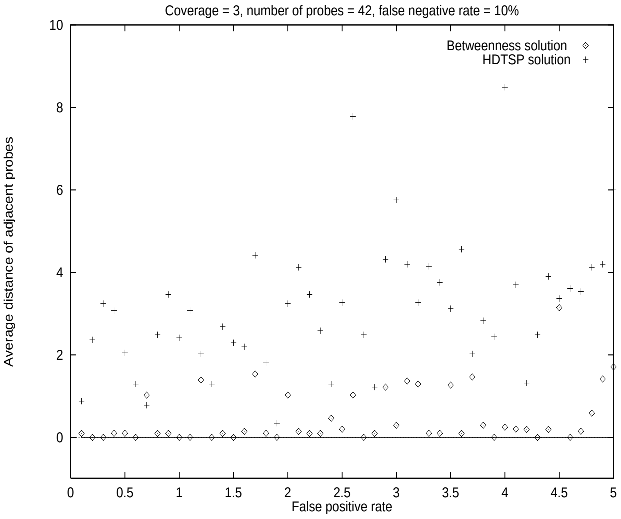

Figure 7: Distance-Quality (coverage=3)

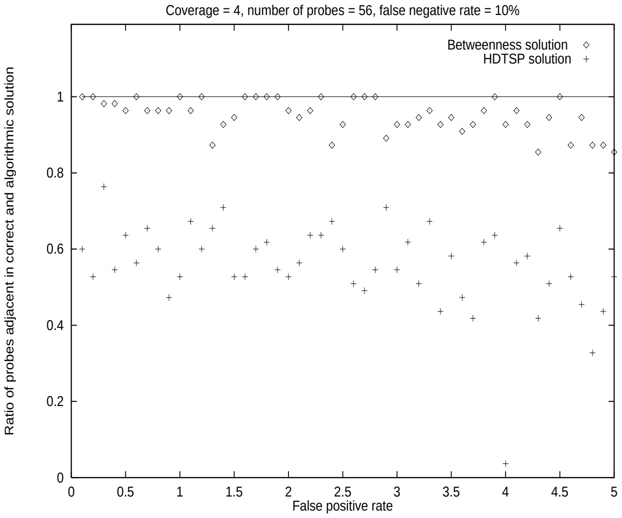

Figure 5: Adjacency-Quality (coverage=4)

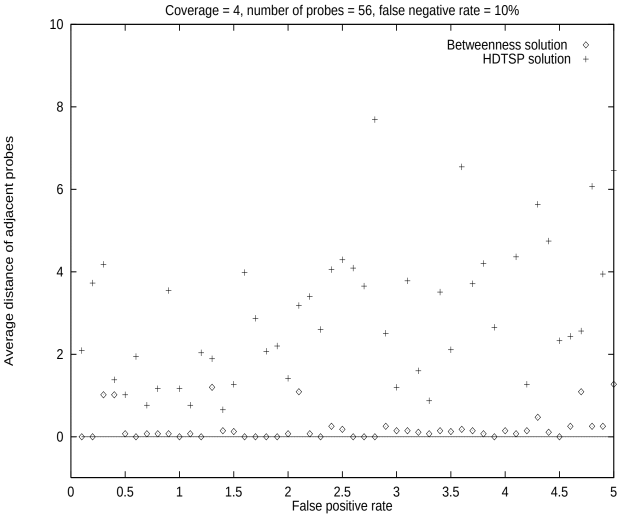

Figure 8: Distance-Quality (coverage=4)

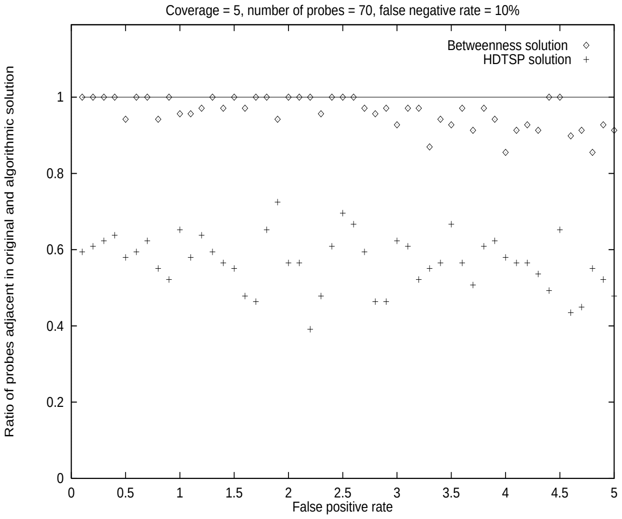

Figure 6: Adjacency-Quality (coverage=5)

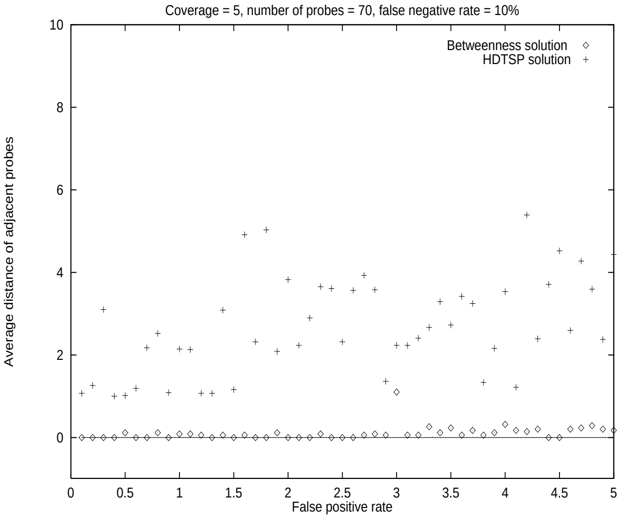

Figure 9: Distance-Quality (coverage=5)

In contrast, the optimal solution of the HDTSP is rarely an optimal solution for the original problem. Moreover, the discrepancy increases with higher false positive rates, while the betweenness solution remains stable.

As in Greenberg and Istrail [GI95], we define the adjacency-quality of a solution $\pi$ as the ratio of the number of adjacencies of probes that are common to both $\pi$ and the correct ordering $\pi^*$, divided by the number of adjacencies in $\pi^*$. Figures 4–6 show that the quality of the solution of the betweenness problem is 1 in many cases, indicating that the optimal solution of the betweenness problem is exactly the correct ordering. An alternative measurement of quality of a solution is the average number of probes that lie between two probes which are adjacent in the correct ordering. Figures 7–9 show the corresponding values. With respect to the HDTSP, both qualities of the solutions of the betweenness problems are appreciably better in most every instance than the corresponding qualities of the optimal solution of the HDTSP.

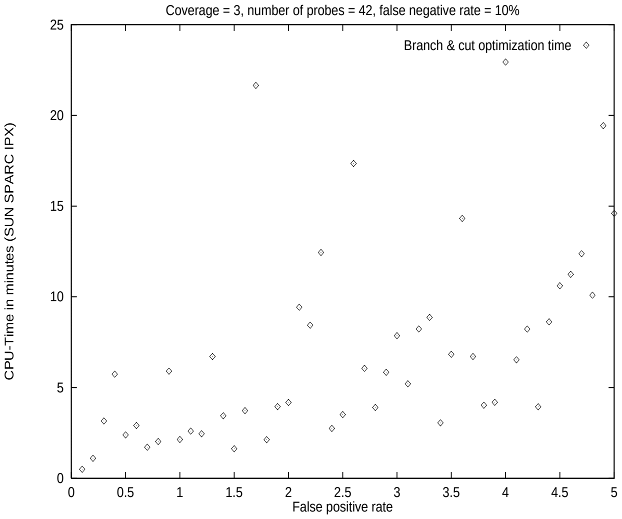

Figure 10: Solution Times (coverage=3)

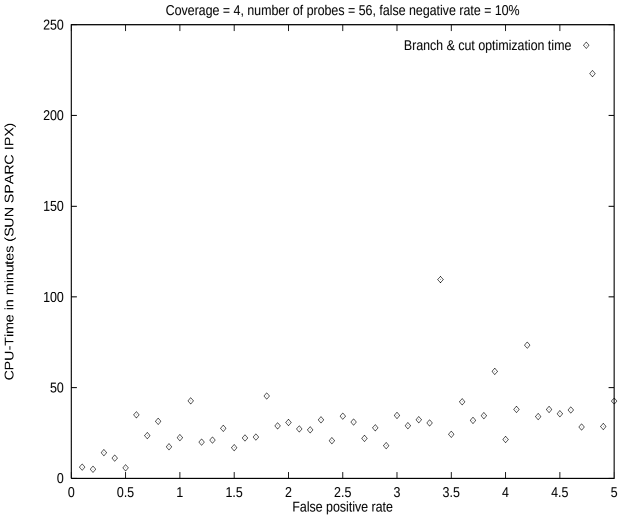

Figure 11: Solution Times (coverage=4)

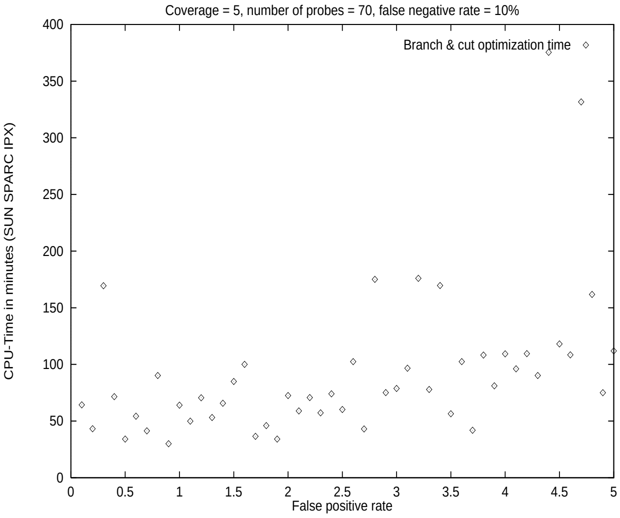

Figure 12: Solution Times (coverage=5)

Figures 10–12 display the running times of the algorithm on a SUN SPARC IPX using CPLEX2.2 as the linear programming solver. It can be observed that the problems tend to become harder as the false positive ratio increases.

### 5.3 Combined strategy

Unfortunately, we are currently able to solve only relatively small instances of the betweenness problem to optimality (up to 100 probes depending on the ratio of false positives to false negatives). For larger instances, the following approach appears promising. We use the exact solution of local betweenness problems to screen out errors (largely false positives) in the full problem, and then solve a HDTSP (which has been demonstrated to work well in the absence of false positives and negatives) on the edited full problem.

The intuition is that on a subset of the probes that are adjacent in the correct solution, it is likely that a false positive in an optimal solution of a local betweenness problem on the correspondig subset of columns of $A$ is also a false positive in the original problem. We implement this procedure as follows. Let $s$ be a fixed parameter, here taken to be $s = 30$, which bounds the maximum number of probes in a local betweenness problem that we will solve. For each probe in the data, we generate a local betweenness suproblem over the $s/2$ nearest neighbours of the probe with respect to the Hamming distance, and further include all end-probes that are opposite end-probes of the clones for which the first $s/2$ probes are end-probes. We call a betweenness condition of the original problem a false positive if there is no subproblem in which the condition is met. Since after removing these presumed false positives, the Hamming distances and the resulting subproblems change, the complete procedure may be executed several times.

Figure 13 compares the optimal HDTSP-solutions with the corresponding optimal HDTSP-solution after running the screening procedure for an instance with coverage 4, 400 probes, false positive rate 5%, and false negative rate 10%. (This false positive rate is more than an order of magnitude larger, and the coverage much lower, than what has been attempted previously [AKWZ94].) The original probe order is given as the identity permutation, and the computed order is plotted against the identity. A perfect solution would

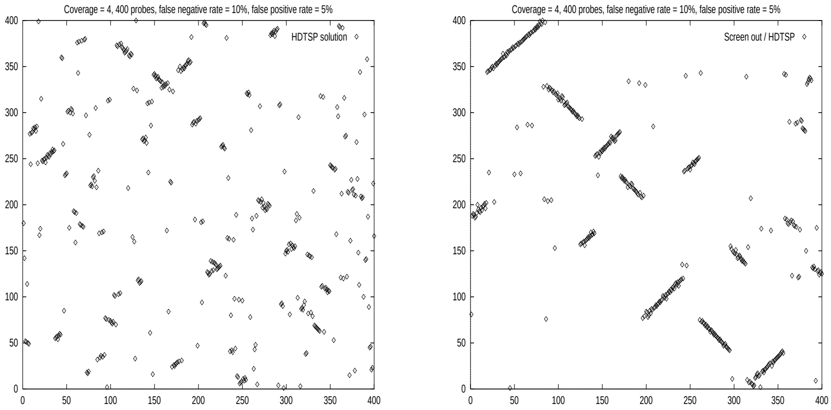

Figure 13: Exact solution of HDTSP without and with screening

appear as a line of slope 1 or $-1$. Compared to the raw HDTSP-solution, the second solution contains longer line segments of ±45 degrees, which suggest that the combined strategy may offer a superior approach. With screening we could reduce the number of false positives from 3902 to 109, while increasing the number of false negatives from 179 to 418.

## 6 Conclusion and future directions

We have shown that physical mapping from clone-probe hybridization data in the presence of false positive and false negative errors using unique probes with end-probe information can be formulated as a weighted betweenness problem. With this formulation we can solve small instances to optimality using methods from polyhedral combinatorics. Computational experiments with the exact branch-and-cut algorithm support the validity of the maximum-likelihood model. Comparison with the results obtained by solving Hamming-Distance Traveling Salesman Problems to optimality suggests that exact optimization of the maximum-likelihood function may be a better approach for solving the physical mapping problem than the widely-used transformation to a Hamming-Distance Traveling Salesman Problem.

### 6.1 Data without end-probes

If end-probes are not given for a clone, we can add artificial probes for the clone that simulate its missing end-probes. If one end-probe of the clone is known, this adds one artificial end-probe, and if no end-probes are known, this adds two artificial end-probes. We can then generate betweenness and nonbetweeness constraints for the clone as before, taking care that the artificial end-probes only induce constraints for the given clone. Finding a linear order of all the probes (including the artificial ones) that minimizes the objective function will again minimize the weighted sum of false positive and false negative errors. Remarkably, this simple modification allows us to handle arbitrary probe-clone hybridization data, and shows that the betweenness formulation is far more general than may appear.

### 6.2 Chimerism

Our model can also be extended to handle chimerism. For each clone with end-probes $i$ and $k$, we add two artificial probes $i'$ and $k'$ that simulate the internal boundaries of the ends of its two chimeric fragments. (This addresses 2-chimerism, but can also be extended to 3-chimerism, etc.) We ensure that these two artificial probes lie between the clone's end-probes by introducing betweenness conditions $(i, i', k)$ and $(i, k', k)$ and fixing the corresponding variables by setting

$$b_{ii'k} = 0 \qquad (43)$$

$$b_{ik'k} = 0. \qquad (44)$$

If a probe that does not hybridize to the clone falls between the two artificial probes, it should not be counted as a false negative, while if a probe that does hybridize to the clone falls between the artificial probes, it should be counted as a false positive. The first requirement can be modelled by introducing nonbetweenness conditions $\overline{(i', j, k')}$ with weight $-\eta$ for all nonbetweenness conditions $b_{ijk} \in \overline{\mathcal{B}}$, the second requirement can be achieved with nonbetweenness conditions $\overline{(i', j, k')}$ with weight $\rho$ for all betweenness conditions $b_{ijk} \in \mathcal{B}$. We also introduce an additional variable $c_{ik}$ for the clone with end-probes $i$ and $k$ that is 1 if and only if a probe falls between these two artificial internal probes. To force $c_{ik}$ to be equal 1 if a probe lies between the artificial probes we add the inequalities

$$c_{ik} \geq 1 - b_{i'jk'} \quad \text{for all} \quad b_{ijk} \in \mathcal{B} \cup \overline{\mathcal{B}}. \qquad (45)$$

The objective function to be minimized changes to

$$\rho \sum_{b_{ijk} \in \mathcal{B}} b_{ijk} + \eta \sum_{b_{ijk} \in \overline{\mathcal{B}}} (1 - b_{ijk}) +$$

$$\chi \sum_{c_{ik} \in \mathcal{C}} c_{ik} + \rho \sum_{b_{i'jk'} \in \mathcal{B}'} (1 - b_{i'jk'}) - \eta \sum_{b_{i'jk'} \in \overline{\mathcal{B}'}} (1 - b_{i'jk'}),$$

where $\mathcal{C}$ denotes the set of all clones and $\mathcal{B}' = \{b_{i'jk'} \mid \exists\, b_{ijk} \in \mathcal{B}\}$ and $\overline{\mathcal{B}'} = \{b_{i'jk'} \mid \exists\, b_{ijk} \in \overline{\mathcal{B}}\}$.

### 6.3 Final remarks

We wish to emphasize that this approach to physical mapping has still to stand the test of real-world problem instances.

## References

[AKNW95] F. Alizadeh, R.M. Karp, L.A. Newberg, and D.K. Weisser. Physical mapping of chromosomes: A combinatorial problem in molecular biology. *Algorithmica*, 13(1/2):52–76, 1995.

[AKWZ94] F. Alizadeh, R.M. Karp, D.K. Weisser, and G. Zweig. Physical mapping of chromosomes using unique probes. In *Proceedings of the 5th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 489–500. ACM Press, 1994.

[BL76] K. Booth and G. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *J. of Computer and System Sciences*, 13:335–379, 1976.

[Boo75] K.S. Booth. *PQ-Tree Algorithms.* PhD thesis, Univ. of California, Berkeley, 1975.

[CR96] T. Christof and G. Reinelt. Combinatorial optimization and small polytopes. *Top*, 4(1):1–64, 1996. Spanish Statistical and Operations Research Society.

[CS95] B. Chor and M. Sudan. A geometric approach to betweenness. In P. Spirakis, editor, *Algorithms – ESA '95*, volume 979 of *Lecture Notes in Computer Science*, pages 227–237. Springer, 1995.

[GDHC95] W. Gillett, J. Daues, L. Hanks, and R. Capra. Fragment collapsing and splitting while assembling high-resolution restriction maps. *J. Computational Biology*, 2(2):185–205, 1995.

[GI95] D.S. Greenberg and S. Istrail. Physical mapping by STS hybridization: Algorithmic strategies and the challenge of software evaluation. *J. Computational Biology*, 2(2):219–273, 1995.

[GJR84] M. Grötschel, M. Jünger, and G. Reinelt. A cutting plane algorithm for the linear ordering problem. *Operations Research*, 32:1195–1220, 1984.

[GJR85] M. Grötschel, M. Jünger, and G. Reinelt. Facets of the linear ordering polytope. *Mathematical Programming*, 33:43–60, 1985.

[JM95] M. Jain and G. Myers. A note on scoring clones given a probe ordering. *Journal of Computational Biology*, 2(1):33–38, 1995.

[JRT94] M. Jünger, G. Reinelt, and S. Thienel. Optimal and provably good solutions for the symmetric traveling salesman problem. *Zeitschrift für Operations Research*, 40:183–217, 1994.

[JRT95] M. Jünger, G. Reinelt, and S. Thienel. Practical problem solving with cutting plane algorithms in combinatorial optimization. In L. Lovász W. Cook and P. Seymour, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Volume 20: Combinatorial Optimization.* Amer. Math. Soc., 1995.

[MGL94] R. Mott, A. Grigoriev, and H. Lehrach. An algorithm to detect chimeric clones and random noise in genomic mapping. *Genomics*, 22:482–486, 1994.

[MHM$^+$93] E. Maier, J. Hoheisel, R. Mott, A. Grigoriev, and H. Lehrach. Algorithms and software tools for ordering clone libraries: Application to the mapping of the genome of schizosaccharomyces pombe. *Nucleic Acids Res.*, 21:1965–1974, 1993.

[Opa79] J. Opatrny. Total ordering problem. *SIAM J. Comput.*, 8(1):111–114, 1979.

[Thi95] S. Thienel. *ABACUS A Branch-And-CUt System.* PhD thesis, Universität zu Köln, 1995.

[VLM96] M. Vingron, H.P. Lenhof, and P. Mutzel. Computational molecular biology. In M. Dell'Amico, F. Maffioli, and S. Martello, editors, *to appear in Annotat. Bibliographies in Comb. Opt.*, chapter 23. 1996. Tech. Rep. MPI-I-96-1-012, Max-Planck-Institut f. Informatik (1996).