

## The Complexity of Gene Placement

Leslie Ann Goldberg<sup>1</sup>, Paul W. Goldberg<sup>1</sup>, and Mike Paterson<sup>1</sup>

*Department of Computer Science, University of Warwick, Coventry,  
CV4 7AL, United Kingdom*

Pavel Pevzner

*Department of Computer Science, University of California,  
San Diego, San Diego, California 92093*

Süleyman Cenk Sahinalp<sup>2</sup>

*Department of Electrical Engineering and Computer Science, and Center for  
Computational Genomics, Case Western Reserve University, Cleveland, Ohio 44106*

and

Elizabeth Sweedyk

*Department of Computer Science, Harvey Mudd College,  
Claremont, California 91711*

Received July 23, 1999

We focus on algorithmic problems related to deriving gene locations on DNA sequences of closely related species by using comparative mapping data. Conventional genetic mapping generates intervals on the DNA sequence of given species for potential gene positions. The simultaneous analysis of gene intervals in related species, e.g., human and mouse, may eliminate some of the ambiguities and lead to better estimates of gene locations. We address the problem of eliminating the ambiguities in gene orders by means of minimizing the number of conserved regions among the species. This is equivalent to the problem of choosing gene coordi-

<sup>1</sup>Partially supported by the IST Programme of the EU under contract number IST-1999-14186 (ALCOM-FT) and RAND-II (Project 21726), by EPSRC Grant GR/L60982, and by NATO Grant CRG-972175.

<sup>2</sup>Supported in part by a grant from Charles B. Wang Foundation.



nates (gene placement) that satisfy the genetic mapping constraints and minimize the breakpoint distance between genomes. We first show that the gene ordering problem is hard: there is no polynomial-time approximation scheme unless  $P = NP$ , even under the restrictions that: (1) the order of genes in one of species is known, or (2) at most two intervals overlap at any location on the map of any of the species. Then we provide two polynomial-time algorithms under restriction (1) above; the first approximates the problem within a factor of 3, and the second exactly solves the problem under the additional restriction that (3) no more than  $O((\log n)/(\log \log n))$  intervals overlap at a location on the map of any of the species. We also prove the tractability of the general problem when there is a single conserved region (i.e., when there exists a gene placement resulting in identical gene orders). © 2001 Elsevier Science

## 1. INTRODUCTION

Let  $G_1, \dots, G_n$  be a set of known *genes* (or markers), which occur in the DNA of two distinct species, e.g., human and mouse. For each species, we are given partial information about the sequence in which the genes occur along its DNA. The goal is to construct, for each species, a sequence in which the genes occur (i.e., a permutation of  $G_1, \dots, G_n$ ) that is consistent with the partial information and minimizes the number of *conserved regions* between the permutations assigned to the two species. A “conserved region” between two permutations  $\Pi_1$  and  $\Pi_2$  of  $G_1, \dots, G_n$  is defined to be a maximal substring of  $\Pi_1$ , which either occurs in  $\Pi_2$  or occurs in reverse order in  $\Pi_2$ . For example, if  $\Pi_1 = G_1G_5G_2G_3G_7G_8G_4G_6$  and  $\Pi_2 = G_1G_2G_3G_5G_6G_4G_8G_7$  then the maximal conserved regions between  $\Pi_1$  and  $\Pi_2$  are  $[G_1]$ ,  $[G_5]$ ,  $[G_2G_3]$ , and  $[G_7G_8G_4G_6]$ . Thus, there are four conserved regions between  $\Pi_1$  and  $\Pi_2$ .

The number of conserved regions between two genomes is equal to the *breakpoint distance* between genomes plus one. If two genes are adjacent in  $\Pi_1$  but not in  $\Pi_2$ , they determine a breakpoint in  $\Pi_1$ . For example,  $\Pi_1 = G_1|G_5|G_2G_3|G_7G_8G_4G_6$  has three breakpoints breaking  $\Pi_1$  into four conserved regions. The breakpoint distance between  $\Pi_1$  and  $\Pi_2$  is defined as the number of breakpoints in  $\Pi_1$  (equal to the number of breakpoints in  $\Pi_2$ ). Breakpoint distance and breakpoint phylogenies have been studied in a number of papers [3, 4, 16].

For each species, the partial information that is available is essentially, for each gene  $G_i$ , an *interval* along the DNA of the species in which  $G_i$  may occur. The biological motivation for this assumption is given in Section 1.1. Each interval may be viewed abstractly as an interval on the real line, and we use that representation later in the paper (specifically, in the negative results). The idea is that each gene must be placed somewhere in its interval. (Although the intervals are usually statistical confidence intervals, here

we make no preference for locating a gene toward the center of the interval.) Once the genes are placed, the real line is forgotten, and only the sequence of genes remains. This version of the problem, however, contains geometrical information that is irrelevant to the combinatorial problem. We continue by giving a combinatorial description of the constraints which we use in our algorithms. For the sake of simplicity, in this paper we consider unichromosomal genomes but the results can be generalized for multichromosomal genomes.

If the endpoints of the intervals are arranged in order along the real line, then a sequence of consecutive left-hand endpoints may be treated as being located at a common point, and the combinatorial constraints are not changed. (A similar claim holds for consecutive right-hand endpoints.) As a result, the following combinatorial information can be easily derived. Formally, the partial information consists of a partition of the set of genes  $\{G_1, \dots, G_n\}$  into  $m$  “opening sets”  $O_1, \dots, O_m$  and another partition of  $\{G_1, \dots, G_n\}$  into  $m$  “closing sets”  $C_1, \dots, C_m$ . These partitions have the property that, for every  $r \in \{1, \dots, m\}$ ,  $C_r \subseteq O_1 \cup \dots \cup O_r$ . (The interval for every gene is “opened” before it is “closed”.) The partial information that the opening and closing sets provide is as follows: If a gene  $G_h$  is in opening set  $O_a$  and in closing set  $C_b$  (where, by definition,  $a \leq b$ ) then gene  $G_h$  must be placed in the sequence *after* all of the genes in  $C_1 \cup \dots \cup C_{a-1}$  and *before* all of the genes in  $O_{b+1} \cup \dots \cup O_m$ .

**DEFINITION 1.** A *gene placement* for a set of genes that are common to a pair of species is a pair of total orders for each of the given partial orders. The *gene placement* problem that we consider here is the problem of finding a gene placement so as to minimize the number of the conserved regions between the genomes.

We note in passing that it is straightforward to compute the number of conserved regions (or breakpoint distance) associated with two given total orders. We continue by classifying the gene placement problem under consideration here according to the “depth” of problem instances.

Without loss of generality, we can assume that every opening set  $O_r$  and every closing set  $C_r$  is nonempty. (If  $C_r$  is empty, then it can be deleted and  $O_r$  and  $O_{r+1}$  can be merged without changing the set of sequences which the partial information allows. Similarly, if  $O_r$  is empty, then it can be deleted and  $C_{r-1}$  and  $C_r$  can be merged.) Therefore, we can assume  $m \leq n$ . We define  $X_r = \{O_1 \cup \dots \cup O_r\} \cap \{C_r \cup \dots \cup C_m\}$ . Thus,  $X_r$  is the set of all genes which could possibly be placed in the sequence *after* all of the genes in  $C_1 \cup \dots \cup C_{r-1}$  and *before* all of the genes in  $O_{r+1} \cup \dots \cup O_m$ . We define the *depth* of the species (really, the depth of the partial information provided for the species) to be  $\max_r |X_r|$  and we define the *depth* of the problem instance to be the *minimum* of the depths of the two species. (Returning to

the “intervals” description, the depth of a species is the maximum number of intervals which overlap at any single point.) Thus, a *depth-1* species is a species for which the partial information completely specifies the sequence of genes. (By definition,  $X_r$  contains both  $O_r$  and  $C_r$ , so since  $|X_r| = 1$ ,  $O_r$  and  $C_r$  contain the same single gene.)

The *depth-d gene placement problem* is the problem of minimizing the number of conserved regions, given a depth- $d$  problem instance. A *polynomial-time approximation scheme* (PTAS) for the depth- $d$  gene placement problem is an algorithm that takes the problem instance and a parameter  $\epsilon$  and outputs a gene placement such that the ratio between the number of conserved regions in the output and the optimal number of conserved regions is at most  $1 + \epsilon$ . The running time of the algorithm may depend arbitrarily on  $\epsilon$ , but must be bounded from above by a polynomial in the size of the problem instance. In this paper, we prove the following results about the depth-1 gene placement problem.

**THEOREM 1.** *There is no polynomial-time approximation scheme for the depth-1 gene placement problem unless P = NP.*

**THEOREM 2.** *We give a polynomial-time algorithm which approximates the depth-1 gene placement problem within a factor of 3.*

**THEOREM 3.** *We give a polynomial-time algorithm which exactly solves the special case of the depth-1 gene placement problem in which the depth of one species is 1 and the depth of the other species is  $O((\log n)/(\log \log n))$ .*

Next, we show that the depth-2 gene placement problem is much harder than the depth-1 gene placement problem. (Compare the following theorem with Theorem 3.)

**THEOREM 4.** *There is no polynomial-time approximation scheme for the gene placement problem with the restriction that both species have depth at most 2 unless P = NP.*

Finally, we show that some questions about the arbitrary-depth gene placement problem are tractable.

**THEOREM 5.** *We give a polynomial-time algorithm which determines whether the solution to the (arbitrary depth) gene placement problem is 1 (that is, whether it is possible to have only one conserved region).*

### 1.1. Biological Motivation

Waardenburg's syndrome is an inherited genetic disorder resulting in hearing loss and pigmentary dysplasia. Ten years ago genetic mapping and some luck narrowed the search for the Waardenburg's syndrome gene to human chromosome 2 but the exact localization remained unclear. There

was another clue that directed attention to chromosome 2. For a long time, breeders scrutinized mice for mutant characteristics and one of these, designated *splotch*, with patches of white spots has been considered a possible homolog to Waardenburg's syndrome. Through breeding (which is easier in mice than in humans) the *splotch* gene was mapped to mouse chromosome 2. As gene mapping proceeded it became clear that the position of the Waardenburg's syndrome gene on human chromosome 2 can be derived through the human-mouse comparative genetic map. Comparative genetic maps show the groups of genes that are linked to one another in both species. Therefore, mapping a gene in mice immediately gives a clue for a location of the homologous human gene.

However, the difficulty is that the conventional genetic mapping generates *intervals* for potential gene positions rather than gene *coordinates*. These intervals may overlap, differ in size, and even be in conflict with each other, thus leading to ambiguities in assigning *gene orders*. The simultaneous analysis of gene intervals in related species (e.g., humans and mice) may eliminate some of the ambiguities and lead to better estimates of gene locations.

Although rearrangements of gene orders have been extensively studied in the computer science literature (see, for example, [1, 2, 5-7, 9, 10, 12, 16, 18], and a review of the area in [15]) the problem of generating gene orders from experimental data remained largely unexplored. In particular, Hannenhalli and Pevzner [7] remarked that deriving gene orders is a nontrivial task since the map accuracy in humans is significantly lower than in mice and for many closely located genes in humans the relative ordering is still unknown. For many closely located genes in humans the relative ordering will remain unknown until the final sequences of human and mouse genomes become available [19, 20].

This problem forced Hannenhalli and Pevzner [7] to make a number of arbitrary decisions while deriving gene orders in humans and mice. Sankoff *et al.* [17] described a heuristic approach to identification of conserved regions that proved to be useful in detecting potential errors in genetic maps.

Gene mapping usually estimates the distance between two markers by a statistical procedure. The confidence interval for this distance may vary, depending on many factors like the amount of available genotyping data. Biologists attempt to merge the distance constraints into the overall map and to resolve the potential conflicts. Letovsky and Berlyn [11] developed CPROP, a program that integrates mapping information from numerous sources. Nadkarni [13] has developed Mapmerge, another program that synthesizes information about gene orders from multiple sources. Biologists ideally would like to assign a *genomic coordinate* to every marker. However, in view of genetic mapping uncertainties, this is frequently impos-

sible and coordinate-based representations have not been traditionally used by the human mapping community. In particular, the map information in the genome database uses a formalism that does not lend itself to direct translation to coordinates. The absence of coordinate-based genetic maps was the cause of difficulties Hannenhalli and Pevzner had while deriving tentative gene orders in men and mice in 1995 [7].

This paper addresses some algorithmic problems related to deriving gene orders from comparative mapping data. In the simplest case we assume that the genetic map is already assembled and every gene is assigned an interval of potential coordinates. We use such genetic maps from two species to eliminate the ambiguities in gene orders and to estimate the number of conserved groups in the species.

## 2. TECHNICAL DETAILS

The proofs of each of Theorems 1 to 5 are given in each of the following sections. We prove the positive results first, followed by the negative results (Theorems 1 and 4).

### 2.1. Proving Theorem 2

We give a polynomial-time algorithm which approximates the depth-1 gene placement problem within a factor of 3. Let  $I$  be an instance of the depth-1 gene placement problem. Let  $\pi$  denote the placement of the depth-1 species. We will use  $O_1, \dots, O_m$  and  $C_1, \dots, C_m$  to denote the (nonempty) opening and closing sets of the other species (species  $S$ ). We start with the following observation, which allows us to do useful preprocessing on the problem instance  $I$ .

**OBSERVATION 6.** *If  $G$  is in  $O_a \cap C_b$  and  $G'$  is a gene in  $O_{a'} \cap C_{b'}$  which is adjacent to  $G$  in  $\pi$  then, without loss of generality, either  $a < a'$  and  $b < b'$  or  $a > a'$  and  $b > b'$ .*

*Proof.* Suppose that  $a \leq a'$  and  $b' \leq b$  and that  $\pi'$  is a gene placement for  $S$ . Note that there is a gene placement  $\pi''$  for  $S$  which is as good as  $\pi'$  and has  $G$  and  $G'$  adjacent. (If  $G$  and  $G'$  are not adjacent in  $\pi'$  then  $G$  can be moved next to  $G'$  without creating a new conserved region.) Thus, the problem instance can be replaced by one in which gene  $G$  is deleted from both species. Once a placement for  $S - \{G\}$  has been found it can be extended to a placement for  $S$  by inserting  $G$  next to  $G'$ . ■

*Notation.* For any gene  $H \in C_1$ , let  $forward(H)$  be the suffix of  $\pi$  which starts at gene  $H$  and let  $prefix^+(H)$  denote the longest prefix of  $forward(H)$  which is a prefix of a feasible gene placement for  $S$ . Let  $backward(H)$  be

the substring formed by starting at gene  $H$  in  $\pi$  and proceeding back to the beginning of  $\pi$ . (That is,  $\text{backward}(H)$  is the reversal of the prefix of  $\pi$  which ends at  $H$ .) Let  $\text{prefix}^-(H)$  denote the longest prefix of  $\text{backward}(H)$  which is a prefix of a feasible gene placement for  $S$ .

In what follows, when we apply set-theoretic operators to these strings, we are treating them as the sets of their elements.

**OBSERVATION 7.** *Given a problem instance including a gene  $H$ , it is straightforward to identify (in polynomial time) the strings  $\text{forward}(H)$  and  $\text{backward}(H)$ .*

The 3-approximation algorithm is as follows. We assume that before each (recursive) call to the algorithm we preprocess the problem instance to ensure that the opening and closing sets are nonempty (see the Introduction) and that the instance is consistent with Observation 6.

1. If the input is the trivial problem instance with no genes then output the empty gene placement. If  $S$  consists of a *single* opening set and a single closing set then output  $\pi$ .
2. Otherwise, if  $|C_1| \geq 2$  then let  $P$  be a sequence consisting of the genes in  $C_1$  (in any order). Let  $I'$  be the subinstance formed by removing the genes in  $P$  from both species. The output consists of  $P$  followed by a (recursively generated) 3-approximation for  $I'$ .
3. Otherwise, let  $C_1 = \{H\}$  and let  $O_1^*$  be the set  $O_1 - \{\text{prefix}^+(H) \cup \text{prefix}^-(H)\}$ . If  $O_1^* = \emptyset$  then let  $P$  consist of  $\text{prefix}^+(H)$  followed by  $\text{prefix}^-(H) - \{H\}$ . Let  $I'$  be the subinstance formed by removing the genes in  $P$  from both species. The output consists of  $P$  followed by a (recursively generated) 3-approximation for  $I'$ .
4. Otherwise,  $C_1 = \{H\}$  and let  $O_1^*$  is nonempty. Let  $j$  be an integer which is as large as possible, given that for all  $k < j$ ,  $O_1^* \cap C_k = \emptyset$ . Let  $G$  be any element of  $O_1^* \cap C_j$ . Let  $P$  consist of  $G$  followed by  $\text{prefix}^+(H)$  followed by  $\text{prefix}^-(H) - \{H\}$ . Let  $I'$  be the subinstance formed by removing the genes in  $P$  from both species. The output consists of  $P$  followed by a (recursively generated) 3-approximation for  $I'$ .

It is easy to see that the algorithm terminates after at most  $n$  iterations, since  $I'$  has fewer genes than  $I$ . The theorem follows from the following lemma, which implies that if the algorithm correctly generates a 3-approximation for  $I'$  then it correctly generates a 3-approximation for  $I$ . For any problem instance  $I''$ , the notation  $\text{OPT}(I'')$  denotes the optimum (minimum) number of conserved regions that can be achieved for the problem instance.

**LEMMA 8.** *In Cases 2–4, there is a feasible gene placement for  $I$  which has  $P$  as a prefix. The number of conserved regions in  $P$  (with respect to the depth-1 gene) is at most  $3(\text{OPT}(I) - \text{OPT}(I'))$ .*

*Proof.* Case 2: Clearly,  $C_1 \subseteq O_1$ . Thus, there is a feasible gene placement for  $I$  which has  $P$  as a prefix. By Observation 6, no two genes in  $O_1$  are adjacent in  $\pi$ . Thus, in any gene placement, at least  $|C_1| - 1$  of the genes in  $C_1$  are in singleton conserved regions. Thus, the number of conserved regions in  $P$  is  $|C_1|$  and  $\text{OPT}(I) - \text{OPT}(I')$  is at least  $|C_1| - 1$ .

Case 3: Since there is a feasible gene placement with prefix  $\text{prefix}^+(H)$  and one with prefix  $\text{prefix}^-(H)$ , there is one with prefix  $P$ .  $P$  contains at most two conserved regions. However, any feasible solution has a conserved region contained in  $P$  (by Observation 6, any gene placed before  $H$  is a singleton), so  $\text{OPT}(I) - \text{OPT}(I')$  is at least 1.

Case 4: As in Case 3, there is a feasible gene placement with prefix  $P$ . Also,  $P$  contains at most three conserved regions. We claim that there exists an optimal gene placement for  $S$  which has at least one of its conserved regions contained in  $P$ , implying that  $\text{OPT}(I) - \text{OPT}(I') \geq 1$ . The claim is clearly true if there exists an optimal gene placement for  $S$  in which the conserved region containing  $H$  is a substring of  $\text{prefix}^+(H)$  or a substring of  $\text{prefix}^-(H)$ .

Suppose alternatively that every optimal gene placement for  $S$  has  $H$  contained in a conserved region which is a proper superstring of  $\text{prefix}^+(H)$  or  $\text{prefix}^-(H)$ . Let  $\pi'$  be such a gene placement and let  $\mathcal{H}$  be the conserved region of  $\pi'$  containing  $H$ . Without loss of generality, suppose that  $\mathcal{H}$  is a superstring of  $\text{prefix}^+(H)$ . Let  $j'$  be the minimum integer such that  $C_{j'}$  is not contained in  $\text{prefix}^+(H)$  and every element in  $\text{prefix}^+(H)$  is in  $O_1 \cup \dots \cup O_{j'}$  but some element in  $\mathcal{H}$  is in  $O_{j'+1} \cup \dots \cup O_m$ . Since  $\mathcal{H}$  is contained in a feasible solution,  $C_{j'} \subseteq O_1$ . Furthermore, every member of  $C_{j'}$  must precede  $\mathcal{H}$  in  $\pi'$ . By Observation 6, each of these is in a singleton conserved region in  $\pi'$ . We now have two cases. If  $C_{j'}$  contains an element of  $\text{prefix}^-(H)$  then this element (and therefore its conserved region) is in  $P$ . Otherwise,  $C_{j'}$  contains an element of  $O_1^*$ . The minimality of  $j'$  implies that  $j \geq j'$  (otherwise  $\text{prefix}^+(H)$  would be shorter). Thus,  $j = j'$ , so  $G$  is a conserved region of  $\pi'$  and therefore  $P$  contains a conserved region of  $\pi'$ . ■

## 2.2. Proving Theorem 3

We give a polynomial-time exact algorithm for the special case of the depth-1 gene placement problem in which the depth of one species is 1 and the depth of the other species (which we call species  $S$ ) is  $b = O((\log n)/(\log \log n))$ . Let  $O_1, \dots, O_m$  and  $C_1, \dots, C_m$  be the opening and closing sets of  $S$ . As before, let  $X_r = \{O_1 \cup \dots \cup O_r\} \cap \{C_r \cup \dots \cup C_m\}$ , where  $|X_r| \leq b$ . (Let  $X_{m+1} = \emptyset$ .) For  $r \in \{0, \dots, m\}$ , for any  $Y_{r+1} \subseteq X_{r+1}$ ,

and  $G \in C_r \cup Y_{r+1}$ , let  $P_r(Y_{r+1}, G)$  be an optimal gene placement for species  $S$  (one with a minimum number of conserved regions between it and the gene placement of the other species) given that

- we ignore all genes other than those in  $C_1 \cup \dots \cup C_r \cup Y_{r+1}$  (i.e., we remove all other genes from both species) and
- we only consider gene placements which end in gene  $G$ .

We will show how to compute the placements  $P_r(Y_{r+1}, G)$  in polynomial time by dynamic programming.

First, observe that there are at most  $2^b$  choices of  $Y_{r+1}$  and at most  $2b$  choices of  $G$ . Second, observe that if  $Y_{r+1} \subseteq X_r$  then  $C_r \cup Y_{r+1} \subseteq X_r$ , so  $P_r(Y_{r+1}, G) = P_{r-1}(C_r \cup Y_{r+1}, G)$ . Third, suppose that  $Y_{r+1} \not\subseteq X_r$ . Let  $H$  be a fixed gene in  $Y_{r+1} \cap O_{r+1}$ . Now for every

- $Y_r \subseteq Y_{r+1} \cap X_r - \{G\}$ ,
- $G' \in C_{r-1} \cup Y_r$ , and
- permutation  $\pi$  of  $\{C_r \cup Y_{r+1}\} - Y_r - \{H\} - \{G\}$ ,

let  $P(Y_r, G', \pi)$  be the gene placement formed by taking  $P_{r-1}(Y_r, G')$  followed by  $H$  followed by  $\pi$  followed by  $G$ . Clearly, we can choose  $P_r(Y_{r+1}, G)$  by taking an optimal placement  $P(t_r, G', \pi)$  (over all choices of  $Y_r$ ,  $G'$ , and  $\pi$ ). Furthermore, there are at most  $2^b$  choices for  $Y_r$ , at most  $2b$  choices for  $G'$ , and at most  $(2b)!$  choices for  $\pi$ . Since  $(2b)! = n^{O(1)}$  the algorithm runs in polynomial time.

### 2.3. Proving Theorem 5

We give a simple polynomial-time algorithm which determines whether the solution to the (arbitrary depth) gene placement problem is 1 (that is, whether it is possible to have just one conserved region). It suffices to give polynomial-time algorithm (such as the following algorithm) which determines whether there exists a *single* gene placement  $\pi$  which is consistent with the partial information  $(O_1, \dots, O_m)$  and  $(C_1, \dots, C_m)$  provided for the first species and the partial information  $(O'_1, \dots, O'_{m'})$  and  $(C'_1, \dots, C'_{m'})$  provided for the second species. We claim that the following method finds a single conserved region that is not reversed between the species (if one exists).

1. If  $O_1 \cap O'_1 = \emptyset$  then there is no such  $\pi$ .

2. Otherwise, pick any  $G \in O_1 \cap O'_1$  and let  $G$  be the first gene in  $\pi$ . Recursively find the rest of  $\pi$ .

Before recursing in the second step of the algorithm, we delete any empty opening and closing sets and merge adjacent opening sets as described in Section 1.

To justify the above method, note that if the partial information is consistent with a single conserved region, then that conserved region must start with an element of  $O_1 \cap O'_1$  (assuming that the conserved region is not reversed between the two species). Then if an arbitrary element  $O_1 \cap O'_1$  is used, that does not then make the information on the remaining genes inconsistent with a single conserved region.

We may then search for a conserved region that is reversed between the species, by reversing the sequence of opening and closing sets of one of the species (and deeming the opening sets to be closing sets and vice versa), then repeating the above method.

#### 2.4. Proving Theorem 1

We show that there is no polynomial-time approximation scheme for the depth-1 gene placement problem unless  $P = NP$ . We start by defining the *3-bounded max-2SAT* problem. An instance (or formula)  $\Phi$  of 3-bounded max-2SAT consists of a set of clauses  $\{C_1, \dots, C_k\}$  where each  $C_i$  is the disjunction of two literals over a set  $\text{var}(\Phi)$  of boolean variables. Each *literal* occurs in at most three clauses. The goal is to find an assignment of values to the elements of  $\text{var}(\Phi)$  which maximizes the number of clauses that are satisfied. We will use the following fact from [14].

*Fact 9.* There is no polynomial-time approximation scheme for 3-bounded max-2SAT unless  $P = NP$ .

We will prove Theorem 1 by showing that a polynomial-time approximation scheme for the depth-1 gene placement problem could be turned into a polynomial-time approximation scheme for 3-bounded max-2SAT.

Given an instance  $\Phi$  of 3-bounded max-2SAT, we construct an instance  $\Gamma$  of the depth-1 placement problem. For each variable  $x$ ,  $\Gamma$  contains a set of genes  $\Gamma(x)$  which has two distinct optimal placements. For two boolean variables  $x, y \in \text{var}(\Phi)$ ,  $\Gamma(x)$  and  $\Gamma(y)$  are designed to interact if and only if  $x$  and  $y$  appear in a clause  $C$  of  $\Phi$ . Suppose this happens in a feasible solution of  $\Gamma$  that has one of the optimal arrangements for each of  $\Gamma(x), \Gamma(y)$ . Then we will find that a conserved region will be saved if and only if the gene arrangements in  $\Gamma(x)$  and  $\Gamma(y)$  encode a satisfying assignment of  $C$ . One conserved region is saved for each clause satisfied. Given a feasible solution  $S(\Gamma)$  for  $\Gamma$ , let  $S(\Gamma(x))$  denote  $S(\Gamma)$  restricted to  $\Gamma(x)$ . We cannot generally assume that because  $S(\Gamma)$  is approximately optimal, each  $S(\Gamma(x))$  is one of the two optimal solutions to  $\Gamma(x)$ . However, we show how to construct an alternative solution  $S'(\Gamma)$  from  $S(\Gamma)$  in

polynomial time, such that

- $S'(\Gamma)$  has no more conserved regions than  $S(\Gamma)$ ,

•  $S'(\Gamma(x))$  (which denotes  $S'(\Gamma)$  restricted to  $\Gamma(x)$ ) does encode one of the truth values (hence  $S'(\Gamma(x))$  is optimal).

Given such a construction, it just remains to derive an approximation ratio  $1 - \epsilon$  for 3-bounded max-2SAT which would be associated with a hypothetical approximation ratio  $1 + \delta$  for gene arrangement, such that  $\epsilon \rightarrow 0$  as  $\delta \rightarrow 0$ .

In this section we will work with the alternative formulation of the gene placement problem which is mentioned in the Introduction. Each gene will be described by an interval along the real line. Each gene must be placed somewhere in its interval. Once the genes are placed, the real line is forgotten, and only the sequence of genes remains. In our diagrams, we will denote gene intervals by vertical lines. We will displace the vertical lines sideways for readability (to allow intervals to be distinguished). A sequence of consecutive (in the depth-1 ordering) but nonoverlapping gene intervals is depicted by line segments that lie on a common line. Genes are denoted by the symbols  $\bullet$  or  $\circ$  (the two symbols are used to show two alternative gene placements in one diagram), and feasible solutions are depicted by placing a gene symbol on each vertical line. A gene symbol without a line depicts a gene whose position is fixed, that is, a gene whose interval starts and stops at the same point. Without loss of generality, we assume that the order of placement of the depth-1 gene is  $G_1, \dots, G_n$ .

#### 2.4.1. Representing a Boolean Variable

For  $x \in \text{var}(\Phi)$ ,  $\Gamma(x)$  uses 27 genes, plus an additional 13 separator genes as described later. First there are two sequences of length 15 and 3. Let  $X_1, \dots, X_{15}, H_1, H_2, H_3$ , denote these sequences. The 13 separator genes will prevent conserved regions from containing more than two genes. Observe that when gaps between two different pairs of consecutive gene intervals coincide, then only one of those pairs may form a conserved region. The alignment of the gaps is as shown in Fig. 1. We will obtain at most eight pairings of adjacent genes from these sequences, consisting of either  $(H_1, H_2)$  or  $(H_2, H_3)$ , and seven alternating pairings in the  $X_i$ 's. Assuming that one of these optimal placements has been made, associate with “true” the one that joins  $H_1$  with  $H_2$ , and with “false” the one that joins  $H_2$  and  $H_3$ .

We next introduce some genes to  $\Gamma(x)$  which “reinforce” the optimality of joining alternate pairs in the  $X$ -sequence. Introduce three sequences of three consecutive genes,  $R_1, R_2, R_3$  and  $R'_1, R'_2, R'_3$  and  $R''_1, R''_2, R''_3$ , with gaps between consecutive genes aligned as in Fig. 1. Observe that provided

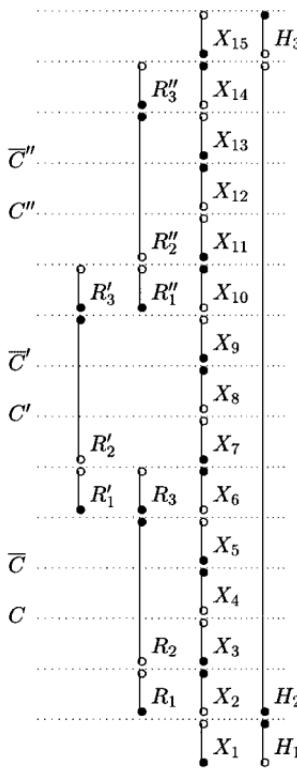


FIG. 1. Representing a variable: Vertical lines indicate intervals associated with individual genes. The sets of  $\bullet$  and  $\circ$  symbols indicate the two optimal placements of genes.  $C, C', C''$  indicate gaps between consecutive genes associated with a clause containing the unnegated variable,  $\bar{C}, \bar{C}', \bar{C}''$  gaps for clauses containing the negated variable.

alternate pairing are made with genes in the range  $X_2, \dots, X_6$ , then we can pair either  $\{R_1, R_2\}$  or  $\{R_2, R_3\}$  and similarly for the reinforcers  $R'$  and  $R''$ .

The construction uses a number of sequences of consecutive genes ( $X$ 's,  $R$ 's,  $R''$ 's,  $H$ 's), constrained to lie in intervals that are separated by gaps. In all intervals in these sequences, other than the interval at each end, we will additionally place a gene whose position is fixed, and which is cut off from its neighbors. The effect of these extra genes is to prevent any feasible solution from having conserved regions of length more than 2. These extra genes are omitted from the descriptions, for simplicity. Note that these separators for the  $X$ 's also have the effect of separating the  $H$ 's and the reinforcers in the same way.

The total number of genes in the above description of  $\Gamma(x)$  (for  $x \in \text{var}(\Phi)$ ) is 40. The extra 13 separators occur in the intervals for  $x_2, \dots, X_{14}$ . A set  $\Gamma(x)$  is constructed for each  $x \in \text{var}(\Phi)$ , and they are

placed consecutively on the real line. For  $x, y \in \text{var}(\Phi)$ , if  $x \neq y$  then the genes in  $\Gamma(x)$  are placed so as not to overlap the genes in  $\Gamma(y)$ .

#### 2.4.2. Representing a Clause

We have observed that  $\Gamma(x)$ , as described above, has two optimal gene placements. Here we assume that for each  $x \in \text{var}(\Phi)$  one or another of these gene placements has been used, and we add some more genes in such a way that a conserved region is saved for each clause satisfied by the truth assignment represented by the  $S(\Gamma(x))$  placements. In the next section we justify our assumption of local optimality for the  $S(\Gamma(x))$  placements (required for representing a truth assignment).

The two optimal placements do not use all the gaps between consecutive genes in the  $X$ -sequence, and consequently, if other gaps between consecutive genes happen to coincide with them, we may make further connections across these gaps without cost. In particular, consider the gaps  $(X_3, X_4)$ ,  $(X_4, X_5)$ ,  $(X_7, X_8)$ ,  $(X_8, X_9)$ ,  $(X_{11}, X_{12})$ ,  $(X_{12}, X_{13})$ . (All the others have connections across them, if not by the  $X$ 's then by the  $H$ 's,  $R$ 's,  $R'$ 's, or  $R''$ 's.) Only an alternating subsequence of three of these six have connections across them in an optimal  $S(\Gamma(x))$ .

Recall that the placement connecting  $H_1$  and  $H_2$  is associated with *true*, and this placement does not use the gaps  $(X_3, X_4)$ ,  $(X_7, X_8)$ , and  $(X_{11}, X_{12})$ . Likewise the *false* assignment does not use  $(X_4, X_5)$ ,  $(X_8, X_9)$ , and  $(X_{12}, X_{13})$ .

For  $x, y \in \text{var}(\Phi)$ , suppose  $\Gamma(x)$  precedes  $\Gamma(y)$  on the line. A clause containing  $x$  and  $y$  is represented by three consecutive genes  $K_1, K_2, K_3$ , where the gap  $(K_1, K_2)$  coincides with a gap in  $\Gamma(x)$  which is unused by the assignment to  $x$  satisfying the literal containing  $x$  in the clause, and the gap  $(K_2, K_3)$  coincides with a similarly chosen gap in  $\Gamma(y)$ . Each gap used by a clause is not used by any other clause, but recall that there are at most three literals of each kind in  $\Phi$ , so we have enough clause gaps in each  $\Gamma(x)$ .

Each clause uses three genes. It is not necessary to include an explicit separator gene as there will always be a fixed gene from some  $X$ -interval within  $K_2$ 's interval.

#### 2.4.3. Conversion from $S(\Gamma)$ to $S'(\Gamma)$

Given an approximately optimal gene placement  $S(\Gamma)$ , we cannot of course assume that for all  $x \in \text{var}(\Phi)$ ,  $S(\Gamma(x))$  is locally optimal, hence representing a truth value. We need to prove the following claim.

*Claim.* Given a feasible solution  $S(\Gamma)$ , we can convert it in polynomial time to an alternative feasible solution  $S'(\Gamma)$ , where

- the number of conserved regions is not increased, and
- the genes for each variable encode one of the truth values.

The conversion from  $S(\Gamma)$  to  $S'(\Gamma)$  works by performing local optimization on  $\Gamma(x)$  for each  $x \in \text{var}(\Phi)$ . We do not rearrange any of the clause-encoding genes. In the process, we may make connections in a  $\Gamma(x)$  which breaks a connection between a pair of clause-encoding genes; however, it is argued that when this happens, the local optimization gains at least as many “makes” as it loses in “breaks.”

For each  $x \in \text{var}(\Phi)$ , let  $C, \bar{C}, C', \bar{C}', C'', \bar{C}''$  denote the six clause gaps in  $\Gamma(x)$ , in the order in which they appear as we traverse  $\Gamma$  from one end to the other. Consider the pairs  $(C, \bar{C})$ ,  $(C', \bar{C}')$ , and  $(C'', \bar{C}'')$ .

We perform the local optimization in stages. First, if there are three successive open gaps in the  $X$ -sequence, we make a connection across the middle gap and, if necessary, break an existing connection using the gap. Secondly, if the gaps corresponding to  $D$  and  $\bar{D}$  are both open, where  $D \in \{C, C', C''\}$ , then the neighboring gaps must both be being used by the  $X$ 's, and therefore neither is used by the corresponding reinforcer,  $R, R'$ , or  $R''$ . We exchange either one of these two connections with its neighboring gap in  $C$  or  $\bar{C}$  and make the new connection in the corresponding reinforcer. We have introduced one extra make in  $\Gamma(x)$  at the possible cost of one break in clause.

The final stage to produce  $S'(\Gamma)$  is as follows. Exactly one of each adjacent pair of gaps corresponding to clauses is now used by the  $X$ -sequence. The effect of any remaining gaps is just to change the “parity” from one pair to the next, allowing perhaps a clause in one pair to be “satisfied” by the positive literal and a clause in another pair to be satisfied by the negative literal. In such a case we select the parity of the majority of the clause gap pairs and use the corresponding alternation of open and closed gaps. Since the alternating sequences are the only ways to achieve eight makes within the  $X$ - and  $H$ -sequences, such a change gains at least one make at the expense of at most one break in a clause.

The result of performing these local optimizations on each  $\Gamma(x)$  is  $S'(\Gamma)$ .

#### 2.4.4. Approximation Ratios

Suppose  $\Phi$  has  $k$  clauses and  $n$  variables. We know that  $n \leq 2k$ ,  $n \geq k/6$ . let  $m$  be the maximum number of clauses satisfiable. We know that  $k/2 \leq m \leq k$ . (It is easy to satisfy  $k/2$  clauses by a simple greedy algorithm.) In particular,  $n \leq 4m$  and  $k \leq 2m$ .

$\Gamma$  has  $40n + 3k$  genes, 40 for each  $\Gamma(x)$  and 3 for each clause. The number of conserved regions equals the number of genes minus the number of pairings that can be made. From the previous section, the optimal (largest) number of pairings is the number of pairings per (locally optimal)  $S'(\Gamma(x))$  times the number  $n$  of variables, plus the number of satisfied clauses, which is  $11n + m$ . The optimal (minimum) number of conserved regions is

$$40n + 3k - (11n + m) = 29n + 3k - m.$$

We know that a feasible solution with  $29n + 3k - m'$  conserved regions (for  $m' \leq m$ ) can be used to derive an assignment satisfying  $m'$  clauses. Suppose we can approximate the number of conserved regions within  $1 + \delta$ . So we can find a solution within  $(1 + \delta)(29n + 3k - m)$  conserved regions. This is equal to

$$\begin{aligned} & 29n + 3k - m + \delta(29n + 3k - m) \\ & \leq 29n + 3k - m + \delta(116m + 6m - m) \\ & = 29n + 3k - (1 - 121\delta)m. \end{aligned}$$

So we could then approximate 3-bounded max-2SAT within  $1 - 121\delta$ . Hence a PTAS for the minimum number of conserved regions would give a PTAS for 3-bounded max-2SAT.

## 2.5. Proving Theorem 4

We show that there is no polynomial-time approximation scheme for the gene placement problem with the restriction that both species have depth at most 2, unless P = NP.

As in Section 2.4, we proceed by reduction from 3-bounded max-2SAT. We use some notation from Section 2.4.

### 2.5.1. Representing a Boolean Variable

We begin by defining a cyclic structure as shown in Fig. 2.

Define a *7-unit*,  $U$ , to be an ordered set of seven genes  $(a, a', b', b, k, k', k'')$  which in one species (which we refer to as  $U$ 's species) has  $a, a', b', b$  at fixed consecutive locations, with  $k$  constrained to be adjacent to  $a$ ,  $k'$  adjacent to  $b$ , and  $k''$  adjacent to either or both of  $a', b'$ . (See Fig. 2.) In the other species,  $k, k'', k'$  are fixed and consecutive.

Define a *q-cycle* (where  $q$  is a positive even integer) to be a sequence  $S$  of  $q$  7-units,  $S = (U_1, \dots, U_q)$ , where for  $U = U_i$ ,  $U' = U_{(i+1) \bmod q}$  we have

- the species of  $U$  is not the same species as the species of  $U'$ ,
- the third and fourth genes in  $U$  are the second and first in  $U'$ , and
- the separate 7-units do not overlap each other.

Figure 2 depicts a 6-cycle. The nonfixed genes have been placed so that, for each 7-unit, two of them form a conserved region. (So  $(k, k'')$  are paired in Fig. 2.) We may also allow alternate pairs from the sequence  $(a, a'), (b, b'), (c, c'), \dots$  to form conserved regions, for a total of  $3q/2$  conserved regions of size 2.

Observe that, regardless of the size of  $q$ , there exist two optimal placements of the union of the last three genes of each  $U_i$ , for  $1 \leq i \leq q$ . We cannot make more than  $3q/2$  pairings, and  $3q/2$  pairings can only be made by alternating the orientations of the 7-units.

For each  $x \in \text{var}(\Phi)$ , let  $c(x)$  denote the number of clauses in  $\Phi$  in which  $x$  or its negation appears.  $\Gamma(x)$  will be a  $6c(x)$ -cycle. The total number of genes in  $\cup_{x \in \text{var}(\Phi)} \Gamma(x)$  is  $6 \cdot 7 \cdot k/2 = 21k$  genes. We associate the truth settings of  $x$  with the two optimal arrangements of the genes in  $\Gamma(x)$ .

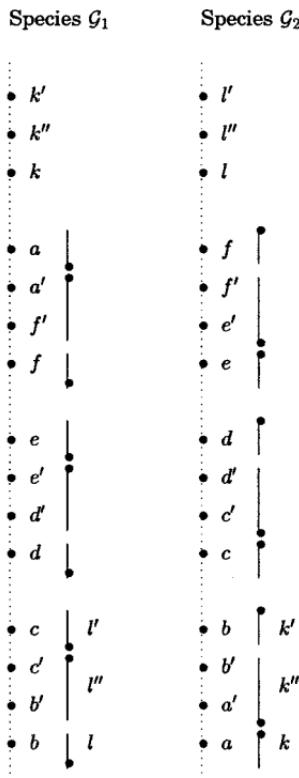


FIG. 2. Depth 2 gene arrangement: a cyclic structure with two optimal solutions. The alternative optimal solution is obtained by moving *all* the nonfixed genes to the opposite ends of their intervals. (Note that only two of the six 7-units are depicted in full; the unlabeled genes whose positions are not fixed correspond to sequences of three consecutive fixed genes (not shown) on the other species.)

### 2.5.2. Representing a Clause

We next allow pairs of cycles to interact so that additional pairings of genes may be affected whenever either cycle encodes a chosen truth value, corresponding to satisfaction of a clause in  $\Phi$ . For a clause  $C_i$  let  $x, y$  be the variables it contains. Choose two consecutive 7-units in  $\Gamma(x)$  and  $\Gamma(y)$  and make them adjacent on each species, but with a gene  $X$  between them on each species, and on one species an additional gene  $x$  which is constrained to be adjacent to  $X$  (and on the other species  $x$  is isolated). This will allow  $X$  to pair with one of the first four genes in the 7-units, provided that it encodes the correct truth value. See Fig. 3. Let  $(a, a', b, b', k, k'', k') \in \Gamma(x), (C, C', D, D', K, K'', K') \in \Gamma(x), (B, B', C, C', L, L'', L') \in \Gamma(y)$  be the 7-units. In Fig. 3,  $X$  may be paired with either  $C$  or  $b$ , provided at least one of  $\Gamma(x)$  and  $\Gamma(y)$  has an appropriate truth value.

It is important to note the following about this method for representing a clause containing  $x$ , say. The choice of which pair of consecutive 7-units from  $\Gamma(x)$  is not constrained by whether the literal contains  $x$  or  $\neg x$ . If we decide to use  $U_i, U_{i+1}$  from  $\Gamma(x)$ , then we can negate  $x$  in the clause being represented by reversing their directions.

Next we want to reinforce the optimality of an alternating sequence of orientations of 7-units, so that we can perform local optimization on every  $\Gamma(x)$  and obtain a  $S'(\Gamma)$  which has no more conserved regions than  $S(\Gamma)$ . Divide the clauses within which  $x$  appears into at most three sets  $S_1, S_2, S_3$ , each of which is either a singleton set or contains a clause with  $x$  and a clause with  $\neg x$ . So we use up to 4 7-units for each  $S_i$ , and we make these 7-units consecutive in the cycle representing  $\Gamma(x)$ . Let  $U_i, U_{i+1}, U_{i+2}, U_{i+3}$  be these 7-units. For each  $S_i$  we then add a tautologous clause that represents  $x \vee \neg x$ , and uses  $U_{i-2}, U_{i-1}$  for  $x$  and  $U_{i+4}, U_{i+5}$  for  $\neg x$ . Hence  $\Gamma(x)$  needs at most 24 7-units (eight for each  $S_i$ ).

### 2.5.3. Conversion from $S(\Gamma)$ to $S'(\Gamma)$

As in section 2.4.3 we want to do the following in polynomial time. Given a feasible solution  $\Gamma$ , convert it to a feasible solution  $\Gamma'$  in which

- there are least as many pairings, and
- alternate 7-units have alternate orientations.

We do not rearrange the clause-representing genes. The set of 7-units used for a set  $S_i$  as defined above is optimized by making the orientations of the 7-units satisfy one clause but not the other. For each  $S_i$  in a  $\Gamma(x)$ , observe whether the clauses give it a bias toward representing true or false. Then take the majority vote of all three, by analogy with Section 2.4.3. When one is overruled, we lose a conserved region through failure to satisfy

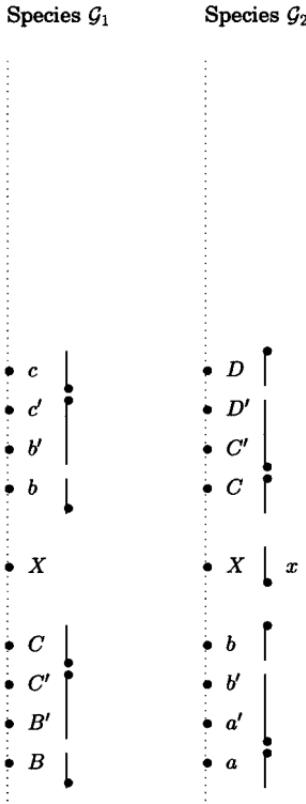


FIG. 3. Depth 2 gene arrangement: a satisfied clause allows an additional pairing to be made, namely  $(X, C)$ . If the  $\Gamma(x)$  containing  $a, a', b', b$  had its other optimal arrangement, then we could choose between  $(X, C)$  and  $(X, b)$ . (Gene  $x$  prevents both pairings simultaneously.)

its clause, but we gain one through recovery of the alternating pattern of orientations of the 7-units.

#### 2.5.4. Approximation Ratios

Suppose  $\Phi$  has  $k$  clauses and  $n$  variables, of which  $m$  are satisfiable. Each appearance of a literal uses a tautological clause, so that  $3k$  clause gadgets in total are used in  $\Gamma$ . Therefore  $6k$  7-units are used,  $3k$  on each species, for a total of  $27k$  genes in  $\Gamma$ . The number of pairings that can be made is 1.5 for each 7-unit, plus  $2k$  for the tautological clauses, plus  $m$ . Hence the optimal number of conserved regions is  $27k - (9k + 2k + m) = 16k - m$ .

Suppose that we can find a feasible solution with  $(1 + \delta)(16k - m)$  conserved regions. This number is

$$(16k - m) + \delta(16k - m) \leq (16k - m) + \delta(32m - m) = 16k - (1 - 31\delta)m.$$

So we could then approximate 3-bounded max-2SAT within  $1 - 31\delta$ . Hence a PTAS for this gene placement problem would give a PTAS for 3-bounded max-2SAT.

## REFERENCES

1. V. Bafna and P. Pevzner, Genome rearrangements and sorting by reversals, *SIAM J. Comput.* **25** (1996), 272–289.
2. V. Bafna and P. Pevzner, Sorting by reversals: Genome rearrangements in plant organelles and evolutionary history of X chromosome, *Mol. Biol. Evol.* **12** (1995), 239–246.
3. M. Blanchette, G. Bourque, and D. Sankoff, Breakpoint phylogenies, in “Genome Informatics’97,” S. Miyano and T. Tayagi (Eds.), pp. 25–43, Tokyo, Japan, 1997.
4. M. Blanchette and D. Sankoff, The median problem for breakpoint phylogenies, in “COCOON’97,” Lecture Notes in Computer Science, No. 1276, pp. 251–263, Springer-Verlag, Berlin/New York 1997.
5. B. DasGupta, T. Jiang, S. Kannan, M. Li, and Z. Sweedyk, On the complexity and approximation of syntenic distance, in “RECOMB’97,” pp. 99–108, Assoc. Comput. Mach., New York.
6. S. Hannenhalli and P. A. Pevzner, Transforming cabbage into turnip (polynomial algorithm for sorting signed permutations by reversals), *J. Assoc. Comput. Mach.* **46** (1999), 1–27.
7. S. Hannenhalli and P. A. Pevzner, Transforming men into mice (polynomial algorithm for genomic distance problem), in “36th Annual IEEE Symposium on Foundations of Computer Science,” pp. 581–592, 1995.
8. J. Kececioglu and R. Ravi, Of mice and men: Evolutionary distances between genomes under translocation, in “Proc. 6th Annual ACM-SIAM Symposium on Discrete Algorithms,” pp. 604–613, 1995.
9. J. Kececioglu and D. Sankoff, Exact and approximation algorithms for the inversion distance between two permutations, *Algorithmica* **13** (1995) 180–210.
10. J. Kececioglu and D. Sankoff, Efficient bounds for oriented chromosome inversion distance in “Combinatorial Pattern Matching, Proc. 5th Annual Symposium (CPM’94), Lecture Notes in Computer Science 807,” pp. 307–325, Springer-Verlag, Berlin, 1994.
11. S. Letovsky and M. B. Berlyn, CPROP: A Rule-Based Program for Constructing Genetic Maps, *Genomics* **12** (1992), 435–446.
12. J. H. Nadeau and B. A. Taylor, Lengths of chromosomal segments conserved since divergence of man and mouse, *PNAS* **81** (1984), 814–818.
13. P. Nadkarni, Mapmerge: Merge genomic maps, *Bioinformatics* **14**, No. 4 (1998), 310–316.
14. C. H. Papadimitriou and M. Yannakakis, Optimization, approximation and complexity classes, *J. Comput. System Sci.* **43** (1991), 425–440.
15. P. A. Pevzner, “Computational Molecular Biology: An Algorithmic Approach,” MIT Press, Cambridge, MA, 2000.
16. D. Sankoff and M. Blanchette, Multiple genome rearrangement, in “RECOMB’98,” pp. 243–247, Assoc. Comput. Mach., New York, 1998.
17. D. Sankoff, V. Ferretti, and J. H. Nadeau, Conserved segment identification, *J. Comput. Biol.* **4**, No. 4 (1997), 559–565.
18. D. Sankoff, G. Leduc, N. Antoine, B. Paquin, B. F. Lang, and R. Cedergren, Gene order comparisons for phylogenetic inference: Evolution of the mitochondrial genome, *Proc. Nat. Acad. Sci. USA* **89** (1992), 6575–6579.
19. J. C. Venter *et al.* The sequence of the human genome, *Science* **291** (2001), 1304–1352.
20. The Genome International Sequencing Consortium, Initial sequencing and analysis of the human genome, *Nature* **409** (2001), 860–921.