Assignment of Orthologous Genes via Genome Rearrangement

Xin Chen, Jie Zheng, Zheng Fu, Peng Nan, Yang Zhong, Stefano Lonardi, and Tao Jiang

Abstract—The assignment of orthologous genes between a pair of genomes is a fundamental and challenging problem in comparative genomics. Existing methods that assign orthologs based on the similarity between DNA or protein sequences may make erroneous assignments when sequence similarity does not clearly delineate the evolutionary relationship among genes of the same families. In this paper, we present a new approach to ortholog assignment that takes into account both sequence similarity and evolutionary events at a genome level, where orthologous genes are assumed to correspond to each other in the most parsimonious evolving scenario under genome rearrangement. First, the problem is formulated as that of computing the *signed reversal distance with duplicates* between the two genomes of interest. Then, the problem is decomposed into two new optimization problems, called *minimum common partition* and *maximum cycle decomposition*, for which efficient heuristic algorithms are given. Following this approach, we have implemented a high-throughput system for assigning orthologs on a genome scale, called SOAR, and tested it on both simulated data and real genome sequence data. Compared to a recent ortholog assignment method based entirely on homology search (called INPARANOID), SOAR shows a marginally better performance in terms of sensitivity on the real data set because it is able to identify several correct orthologous pairs that are missed by INPARANOID. The simulation results demonstrate that SOAR, in general, performs better than the iterated exemplar algorithm in terms of computing the reversal distance and assigning correct orthologs.

Index Terms—Ortholog, paralog, gene duplication, genome rearrangement, reversal, comparative genomics.

1 Introduction

Orthologs and paralogs were originally defined by Fitch [10] in 1970. Orthologs are genes in different species that evolved from the same gene in the last common ancestor of the species, and paralogs are genes that were duplicated from a single gene on the same genome. In order to avoid ambiguity, Sonnhammer and Koonin [19] further divided paralogs into two subtypes: inparalogs and outparalogs. Outparalogs between two species are paralogs in a species that were duplicated before the speciation (i.e., split of the two species) and inparalogs are duplicated after the speciation. For a given set of paralogs on a genome, there commonly exists a gene that is the direct descendant of the ancestral gene of the set, namely, the one that best reflects the original position of the ancestral gene in the ancestral genome. Sankoff [18] called such a gene the *true exemplar* of the paralogous set. These concepts are illustrated in Fig. 1.

Orthologous genes are typically evolutionary and functional counterparts in different species. Many existing computational methods for solving various biological problems (e.g., the inference of functions of new genes or the analysis of phylogenetic relationship between different species) and many other tools in comparative genomics use orthologs in a critical way. As a consequence, the identification of orthologs, especially direct descendants of ancestral genes in current species, is a fundamental problem in computational biology. It follows from the definitions of ortholog and paralogs that the best way to recognize orthologs is to measure the divergence time between homologous genes in two different genomes. As the divergence time could be estimated by comparing the DNA/protein sequences of genes, most existing algorithms for ortholog assignment, such as the well-known COG system [24], [23] and INPARANOID program [17], employ a homology search algorithm such as BLAST [1]. However, the evolutionary rates of all genes in a homologous gene set may vary greatly and, thus, the estimation of divergence times from sequence similarity can be inaccurate. Therefore, information from homology search alone may not be sufficient for recognizing orthologs reliably, although it is usually sufficient for identifying paralogs. On the other hand, we observe that molecular evolution proceeds in two different forms: local mutations and global rearrangements. Local mutations include base substitution, insertion, and deletion and global rearrangements include genome inversion, translocation, transposition, etc. The homology-based ortholog assignment methods only use local mutations and neglect genome rearrangement events that might actually provide valuable information.

In this paper, we propose a new approach for assigning orthologs by taking into account both local mutations and genome rearrangement events. Our method starts by identifying sets of paralogs (i.e., gene families) on each genome and the family correspondences between two genomes by using homology search. The paralogs are then treated as copies of the same genes and ortholog assignment is formulated as a natural optimization problem of rearranging one genome consisting of a sequence of (possibly duplicated) genes into the other with the smallest

- *X. Chen, J. Zheng, Z. Fu, and S. Lonardi are with the Department of Computer Science and Engineering, University of California, Riverside, CA 92521. E-mail: {xinchen, zjie, zfu, stelo}@cs.ucr.edu.*
- *P. Nan and Y. Zhong are with Shanghai Center for Bioinformatics Technology, Shanghai, China. E-mail: {nanpeng, yangzhong}@fudan.edu.cn.*
- *T. Jiang is with the Department of Computer Science and Engineering, University of California, Riverside, CA 92521, and the Shanghai Center for Bioinformatics Technology, Shanghai, China. E-mail: jiang@cs.ucr.edu.*

Manuscript received 18 Nov. 2004; accepted 6 Apr. 2005; published online 1 Nov. 2005.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-0188-1104.

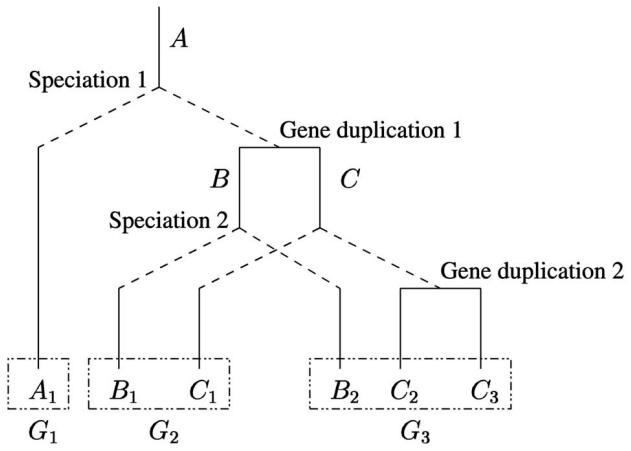


Fig. 1. An illustration of orthologous and paralogous relationships. After two speciation events and two gene duplications, three present genomes, $G_1 = (A_1)$, $G_2 = (B_1, C_1)$, and $G_3 = (B_2, C_2, C_3)$, are obtained. In this scenario, all genes in G_2 and G_3 are co-orthologous to gene A_1. Genes B_1 and C_1 are outparalogs with respect to G_3 (i.e., the second speciation) and are inparalogs with respect to G_1 (i.e., the first speciation). Gene C_2 is the direct descendant (i.e., true exemplar) of the ancestral gene C, while C_3 is not if C_3 is duplicated from C_2.

number of rearrangement events. This most parsimonious rearrangement process should suggest pairs of orthologous genes in a straightforward way. To simplify the discussion (and as a first attempt), we first consider only inversion events in genome rearrangement. The above optimization problem thus becomes a problem of computing the *signed reversal distance with duplicates* (SRDD) between two genomes. SRDD is a simple extension of the well-known problem of sorting by reversals [14]. Although the problem of sorting by reversals has been intensively studied in the past decade, SRDD has basically been untouched. We give an efficient and effective heuristic algorithm for solving SRDD, using the techniques of *minimum common partition* of two given genomes and *maximum cycle decomposition* on a *complete* graph. Based on this algorithm, we develop a high-throughput system for assigning orthologs on a genome scale, called System for Ortholog Assignment by Reversals (SOAR). The heuristic algorithm for SRDD and the system SOAR have been tested on both simulated and real genomic sequence data (from human, mouse, and rat X chromosomes) and, compared with two existing algorithms in the literature, the exemplar algorithm [18] (actually, an iterative version of it) and INPARANOID [17], which is an efficient ortholog assignment method based on homology search. The test results demonstrate that our heuristic algorithm for SRDD, in general, performs better than the iterated exemplar algorithm in terms of computing the reversal distance and assigning correct orthologs and SOAR is marginally better than INPARANOID in terms of the sensitivity of its predictions because it is able to find several true orthologous pairs that are missed by INPARANOID.[1]

The rest of the paper is organized as follows: The next section reviews some related work on ortholog assignment and genome rearrangement. Section 3 introduces the system SOAR that we have implemented for ortholog assignment. The heuristic algorithm for SRDD, which is used in SOAR, is described in Section 4. Preliminary experiments on simulated data and real data are presented in Section 5. Some concluding remarks are given in Section 6.

1. SOAR is slightly worse than INPARANOID in terms of specificity. However, the specificity numbers may not be very reliable here because we validate orthologs by matching their names in Genbank. It is well known that the gene names in Genbank are not completely consistent at the present stage and orthologous genes could be named differently. Besides, the validation method is biased in favor of homology-based ortholog assignment methods since many genes in Genbank were named via homology search.

2 Related Work

Since our approach for ortholog assignment combines homology search with genome rearrangement, we briefly review some related work in ortholog assignment and genome rearrangement.

2.1 Previous Results on Ortholog Assignment

One of the earliest systems developed to identify orthologs is the COG database [24], [23], which is widely used to infer the functions of new genes from annotated genomes. COG stands for *clusters of orthologous groups*, each of which consists of individual orthologous genes or orthologous sets of paralogs from at least three lineages. When a gene is queried against the COG database, it is classified into a COG if it has at least three best BLAST hits in this COG. However, as its name implies, the COG database actually does not report exactly which gene is the ortholog of the query gene if paralogs exist. That is, it does not distinguish the true ortholog of the query gene and its paralogs.

Yuan et al. proposed an approach based on the analysis of *reconciled trees* [28]. The authors first construct a gene tree for a set of homologous genes and then a phylogenetic (species) tree for the species from which the genes came. When there are discrepancies between these two trees, a reconciled tree is computed by taking into consideration the presence of gene duplications, which results in an assignment of orthologous genes. This approach fails when the involved species belong to the same genus because the species background cannot provide sufficient information for resolving the genes under consideration [28]. Furthermore, accurate reconstruction of gene and species trees is a nontrivial problem itself, which further limits the effectiveness of the approach as a method for assigning orthologous genes. A similar approach that uses a set of *bootstrap* trees instead of reconciled trees was proposed in [22] recently. These tree-based methods are not designed to assign orthologs between complete genomes.

To avoid potential errors that might be introduced by multiple alignments or phylogenetic trees, a homology-based tool, called INPARANOID, was introduced in [17]. It uses all-*versus*-all pairwise gene sequence comparison and *bidirectional best hits* (BBHs) to identify the so-called *main orthologs* (and their inparalogs) between a given pair of genomes. Although this approach is in general pretty reliable, it may miss many pairs of orthologs or even assign orthologs incorrectly when the similarity between gene sequences does not accurately reflect the evolutionary relationship among the genes (either because local mutations are not the only evolutionary events that occurred in the history of a gene family or because the sequence similarity measure used does not count local mutations accurately).

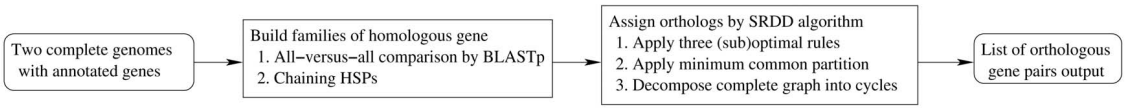


Fig. 2. An outline of SOAR.

Recently, Cannon and Young [4] developed a suite of programs, called OrthoParaMap, to distinguish orthologs from paralogs, which may represent the first effort to assign orthology by making use of comparative genomic positional information. This information is retrieved from conserved *synteny blocks* in the two species of interest and then mapped onto gene family phylogeny to infer a gene duplication mechanism like speciation, segmental duplications, or tandem gene duplications. However, it is widely believed that genome rearrangement scenarios can essentially provide more valuable positional information than synteny blocks. This makes the assignment of orthologous genes via genome rearrangement a very promising approach.

2.2 Genome Rearrangement

Hannenhalli and Pevzner [14] developed an elegant algorithm for computing *signed reversal distance* and *translocational distance* between two genomes (with distinct genes) in polynomial time. A permutation is represented as a *breakpoint graph*. The reversal distance between the permutation and the identity permutation is calculated by decomposing the breakpoint graph into a maximum number of edge-disjoint cycles and counting the numbers of breakpoints, cycles, and *hurdles*, as well as checking the existence of a *fortress*. The best running time for sorting a permutation by the minimum number of reversals is quadratic [15], although the signed reversal distance can be computed in linear time [2].

As mentioned before, the above model and algorithms only deal with genomes consisting of distinct genes. When studying divergent genomes that contain many highly homologous genes (i.e., paralogs) scattered across each genome, we need to consider genome rearrangement with *duplicated genes*. Sankoff [18] designed an *exemplar* algorithm to find the genes from two corresponding gene families that are direct descendants of the same gene in the most recent common ancestral genome. The basic idea is to delete all but one member of every gene family on each of the two genomes being compared so as to minimize some rearrangement distance over all choices of reduced genomes thus derived.

El-Mabrouk [9] recently proposed an algorithm to reconstruct an ancestral genome, giving rise to the minimal number of duplication transpositions and reversals, which works only for genomes containing sets of gene families of sizes at most two. Tang and Moret [25] presented a straightforward approach by enumerating all the possible assignments of orthologs between two genomes with duplicates. However, the number of such possible assignments grows exponentially as the number of paralogs increases, making their approach applicable only to genomes with a very small number of duplicated genes.

3 A System for Ortholog Assignment Based on Sorting by Reversals

Our proposed approach for ortholog assignment takes into account both local mutations at the gene level and global rearrangements at the genome level. Local mutations are measured by sequence similarity and genome rearrangements are measured by the minimum number of rearrangement events (e.g., inversions of contiguous segments of genes) necessary to transform one genome into the other. Following this approach, we have implemented a system, called SOAR, that consists of two major steps: 1) identify paralogs and construct gene families from an annotated genome via homology search and 2) assign orthologs via a heuristic algorithm for SRDD. The steps are illustrated in Fig. 2 and described separately below.

3.1 Construction of Gene Families

SOAR takes as input two annotated genomes (or chromosomes). Gene protein sequences, as well as their locations, are extracted to form a gene list for each genome, which is then formatted and indexed to facilitate an all-*versus*-all gene sequence comparison by BLASTp [1]. BLASTp generates several *high scoring segment pairs* (HSPs) for each pair of genes. Since the HSPs may overlap with each other, a simple summation of their scores may overestimate the similarity between the two genes. We apply a chaining algorithm [8] to find a set of compatible HSPs while maximizing their combined E-value. As in [17], two genes are considered homologous if 1) the combined E-value is less than $1e\text{-}20$ and 2) the compatible HSPs span 50 percent of each gene in length. Each set of homologous genes from the same genome constitutes a gene family. Since we do not deal with gene insertions and deletions in a genome rearrangement process (i.e., we assume that there were no gene loss or insertion events after the speciation), genes without homologous counterparts in the other genome will be removed from the gene lists.

In this preliminary version of SOAR, there is no attempt to identify and group inparalogs in a gene family because we do not know of any effective method for identifying inparalogs (except for the related work in [17]). This is a pitfall in the evolutionary model assumed in SOAR, although it should not be a serious problem for closely related genomes. Moreover, there is some speculation that small gene families (i.e., with sizes of at most 6) are mostly primordial, i.e., they have been maintained from the earliest time of evolution [20]. This speculation, if true, could somewhat justify our method. On the other hand, since we do not consider gene loss or insertion events after the speciation, we simply remove from a gene family those members that are the least homologous (i.e., with the largest combined E-value) to the members of the counterpart

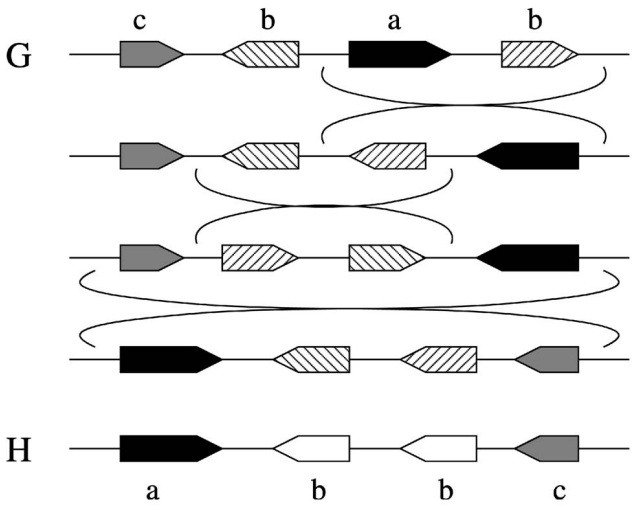


Fig. 3. The most parsimonious transformation using three reversals.

family so that corresponding gene families from both genomes always have equal sizes.

Finally, all the remaining genes from each genome are ordered according to their locations on the genome and the resulting two sequences of genes will be input into the following heuristic algorithm for SRDD.

3.2 Identification of Orthologs Using a Heuristic Algorithm for SRDD

Now, we have two genomes of equal content and all the genes are assumed to be the direct descendants of their ancestral genes in the most recent common ancestral genome. The sequence of genes on each genome thus evolved from the sequence of genes on the common ancestral genome by genome rearrangement events such as reversals and transpositions. By reconstructing the most parsimonious rearrangement scenario, we could identify pairs of genes, where each pair contains one gene from each genome, that came from the same ancestral gene and, thus, likely candidates for orthologs. Equivalently, we may consider the most parsimonious transformation from one genome into the other by reversals and transpositions. Since transpositions occur much less frequently than reversals, hereafter we will only consider reversals. The following is a simple example to illustrate the idea of ortholog assignment via sorting by reversals. Consider two genomes, $G = +c - b + a + b$ and $H = +a - b - b - c$, consisting of four genes and one multigene family each. Fig. 3 shows the parsimonious transformation from G into H using three reversals. In this transformation, the first (or second) copy of gene b in G is found to correspond to the first (or second, respectively) gene b in H, indicating that they might be a pair of orthologous genes.

In contrast, it may happen that orthologs are not conserved very well compared to nonorthogous pairs between two genomes. In the example above, we may have that the first copy of gene b in G is the bidirectional best hit of the second copy of gene b in H. If the sequence similarity does not accurately reflect its evolutionary relationship, orthologous pairs in this gene family will more likely be assigned incorrectly by any approach based solely on local mutations. Therefore, as one of the valuable evolutionary evidences, genome rearrangement events may allow us to reveal the true orthologous relationship, but local mutations cannot.

This approach for ortholog assignment raises a new computational problem, i.e., how to sort a sequence of genes (with duplicates) into another with the minimum number of reversals (SRDD). SOAR uses an efficient and effective heuristic algorithm for SRDD, which will be explained in the next section.

4 An Efficient Heuristic Algorithm for SRDD

In this section, a genome is represented as a string of signed symbols from a finite alphabet $\mathcal{A}$, where each sign (+ or -) represents a transcriptional orientation and a symbol denotes a gene. All occurrences of a symbol in a genome constitute a gene family. A gene is called a *singleton* if it is the only member of its family; otherwise, it is a *duplicated* gene. Two genomes G and H are *related* if they have the same gene content, i.e., an equal number of gene families and an equal size of each family. A reversal operation $\rho(i, j)$ transforms a genome $G = (g_1 \cdots g_{i-1} g_i g_{i+1} \cdots g_{j-1} g_j g_{j+1} \cdots g_n)$ into another genome $G \cdot \rho(i, j) = (g_1 \cdots g_{i-1} - g_j - g_{j-1} \cdots - g_{i+1} - g_i g_{j+1} \cdots g_n)$, where $-g_i$ means the gene g_i with an opposite orientation. Given two related genomes G and H, the *reversal distance problem* is to find the smallest number of reversals $\rho_1, \rho_2, \cdots, \rho_t$ such that $G \cdot \rho_1 \cdot \rho_2 \cdots \rho_t = H$. The *reversal distance* between G and H is thus $d(G, H) = t$. If all the genes in G and H are singletons, the reversal distance problem is usually referred to as the problem of *sorting by reversal* and the distance can be calculated by the Hannenhalli-Pevzner (H-P) formula [14]:

$$d(G, H) = b(G, H) - c(G, H) + h(G, H) + f(G, H),$$

where $b(G, H)$ is the number of black edges in the breakpoint graph for G and H, $c(G, H)$ the number of cycles in maximum cycle decomposition, $h(G, H)$ the number of hurdles, and $f(G, H)$ the number of fortresses, respectively.[2]

When G and H contain duplicated genes, however, the problem cannot be directly solved by the H-P algorithm anymore. Once ortholog assignment between the two genomes is accomplished, the signed reversal distance with duplicates (SRDD) can be simply computed using the H-P formula since every gene can then be regarded as unique. Let us denote by M the set of all the possible ortholog assignments. Given an assignment $m \in M$ and a genome G, let G^m denote the gene sequence of G after orthologs have been assigned by m. The following straightforward lemma relates SRDD to the assignment of orthologs.

Lemma 4.1. *Given genomes G and H, we have $d(G, H) = min_{m \in M} d(G^m, H^m)$.*

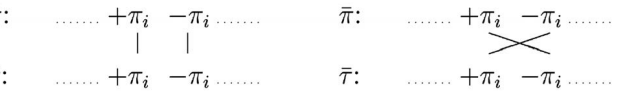
Unfortunately, the following theorem shows that SRDD is NP-hard. Observe that the NP-hardness of sorting unsigned strings over a fixed alphabet [7] does not imply Theorem 4.2.

2. Refer to [14] for the definitions of black edges, hurdles, and fortresses.

Theorem 4.2. *SRDD is NP-hard, even when the maximum size of a gene family is limited to two.*

Proof. The proof is by a simple reduction from the NP-hardness of sorting an unsigned permutations by reversals [5], [6]. Consider a permutation $\pi = (\pi_1 \pi_2 \cdots \pi_n)$ as an instance of sorting an unsigned permutation by reversals, where each element π_i is a positive integer and distinct from any other element. We encode such a permutation by simply replacing each π_i with a pair of two signed integers $(+\pi_i - \pi_i)$ and obtain a new sequence $\bar{\pi} = (+\pi_1 - \pi_1 + \pi_2 - \pi_2 \cdots + \pi_n - \pi_n)$ of size $2n$. The identity permutation $\tau = (1\ 2 \cdots n)$ can be encoded in the same way as $\bar{\tau} = (+1 - 1 + 2 - 2 \cdots + n - n)$. In the following, we prove that there is an optimal transformation from $\bar{\pi}$ to $\bar{\tau}$ that does not include any reversal that breaks an adjacent pair $(+\pi_i - \pi_i)$ in $\bar{\pi}$.

Consider an optimal transformation from $\bar{\pi}$ to $\bar{\tau}$. The transformation defines a correspondence between each element in $\bar{\pi}$ and an element in $\bar{\tau}$. Since there are exactly two duplicates for each π_i in $\bar{\pi}$ (and $\bar{\tau}$), only one of two possible correspondences between the duplicates of π_i could occur, as shown below:



In both cases, each pair of elements $(+\pi_i - \pi_i)$ are adjacent in both $\bar{\pi}$ and $\bar{\tau}$. In other words, there are no breakpoints [14] between $(+\pi_i - \pi_i)$. Let $(\bar{\pi}^m, \bar{\tau}^m)$ be the genome sequences after assigning orthologs according to the above optimal transformation from $\bar{\pi}$ to $\bar{\tau}$. The H-P theory [14] shows that there exists an optimal solution to transform $\bar{\pi}^m$ into $\bar{\tau}^m$ in which no reversal operation would break a pair of adjacent elements. By Lemma 4.1, this transformation is also an optimal solution to transform $\bar{\pi}$ into $\bar{\tau}$. Such an optimal solution to transform $\bar{\pi}$ into $\bar{\tau}$ will naturally give an optimal solution to transform π into τ. Since sorting unsigned permutations by reversals is NP-hard [5], [6], the theorem follows. □

In the following, we devise an efficient and effective heuristic algorithm for SRDD. To simplify the discussion, we assume without loss of generality that the first genes and the last genes of the two related genomes are identical and are positive singletons.

4.1 A Useful Lower Bound

The following definitions are adapted from [7], [9]. We convert a (signed) genome $G = (g_1 g_2 \cdots g_n)$ to an unsigned one by replacing each gene g_i with a string $g_i^h g_i^t$ if g_i is positive or $g_i^t g_i^h$ if g_i is negative, as is done in the breakpoint graph of H-P. A *partial graph* [9] associated with a genome $G = (g_1 g_2 \cdots g_n)$ is the graph $\mathcal{G}(V, E)$, where $V = \{g_i^s | 1 \le i \le n, s \in \{h, t\}\}$, and each (undirected) edge in E links two nodes in V that correspond to adjacent symbols in the genome G except pairs of g_i^h and g_i^t from the same gene g_i. Let $\tilde{V}$ be the set of distinct symbols in V, where g_i^h and g_j^h are viewed as the same symbol if g_i and g_j are from the same family. Clearly, the partial graphs of a pair of related genomes have an identical vertex set V and set $\tilde{V}$. For each pair of elements $\{\tilde{v}_1, \tilde{v}_2\} \in \tilde{V}$, let $f_G(\tilde{v}_1, \tilde{v}_2)$ denote the number of edges in E that link two nodes in the partial graph $\mathcal{G}(V, E)$ of G with symbols $\tilde{v}_1$ and $\tilde{v}_2$, respectively. The *number of reversal breakpoints* between two related genomes G and H is defined as:

$$b_r(G, H) = \sum_{\{\tilde{v}_1, \tilde{v}_2\} \in \tilde{V}} \delta(f_H(\tilde{v}_1, \tilde{v}_2) - f_G(\tilde{v}_1, \tilde{v}_2)),$$

where $\delta(x) = x$ if $x > 0$ and 0 otherwise [7]. It is easy to see that $b_r(G, H) = b_r(H, G)$, although the above definition is not explicitly symmetric with respect to the two genomes. The definition is a natural extension to the concept of the number of breakpoints employed in the problem of sorting by reversals when duplicated genes are present. The following theorem follows from the observation that a reversal operation could reduce the number of breakpoints by at most two.

Theorem 4.3. *Let G and H be a pair of related genomes. Their reversal distance is lower bounded by $d(G, H) \ge \lceil b_r(G, H)/2 \rceil$.*

For example, the partial graph for genomes $G = +c - a - b + a + d$ and $H = +c + a + b + a + d$ is shown in Fig. 6a where the dashed cross-genome edges are masked out. From the above definition, we have

$f_G(c^t,a^t)=1$ $f_G(a^h,b^t)=1$ $f_G(b^h,a^h)=1$ $f_G(a^t,d^h)=1$ $f_G(c^t,a^h)=0$ $f_G(a^t,b^h)=0$
$f_H(c^t,a^t)=0$ $f_H(a^h,b^t)=1$ $f_H(b^h,a^h)=0$ $f_H(a^t,d^h)=1$ $f_H(c^t,a^h)=1$ $f_H(a^t,b^h)=1.$

These imply that the number of reversal breakpoints is two, i.e., $b_r(G, H) = 2$. By the above theorem, we need at least one reversal to transform G to H.

4.2 (Sub)Optimal Assignments

Our heuristic algorithm begins by finding individual ortholog assignments that are (nearly) optimal with respect to SRDD. Note that the (correct) identification of each ortholog reduces the number of duplicates in the related genomes and, thus, makes the SRDD problem easier. The following lemma gives a nearly optimal rule by making use of the H-P formula [14]. Let $G = (g_1 g_2 \cdots g_n)$ and $H = (h_1 h_2 \cdots h_n)$ be two related genomes.

Lemma 4.4. *Assume that $g_{i-1} g_i g_{i+1}$ is identical to $h_{j-1} h_j h_{j+1}$ or its reversal, where g_{i-1} and g_{i+1} are singletons, but g_i is not. Define two new genomes G' and H' from G and H by assigning orthology between g_i and h_j. Then, $d(G, H) \le d(G', H') \le d(G, H) + 1$.*

Proof. By Lemma 4.1, we have $d(G, H) \le d(G', H')$. Let us prove the second inequality. Let (G^m, H^m) be the genome sequences after orthologies have been assigned by an optimal solution to transform G into H, i.e., $d(G^m, H^m) = d(G, H)$. If there is an orthology assigned between g_i and h_j, then the lemma clearly holds. In the following, we assume that the two orthologous pairs assigned between G^m and H^m are (g_i, h_l) and (g_k, h_j), where $i \ne k$ and $j \ne l$.

Denote by (G'^m, H'^m) the genome sequences with the same orthology assigned as in (G^m, H^m), except that g_i is assigned to h_j and g_k is assigned to h_l. We now prove that $d(G'^m, H'^m) \le d(G^m, H^m) + 1$, which then implies $d(G', H') \le d(G, H) + 1$. Since the genome sequences G^m, H^m, G'^m, and H'^m contain only singletons, the H-P formula can be applied here to calculate the reversal distances $d(G^m, H^m)$ and $d(G'^m, H'^m)$. Therefore, we will

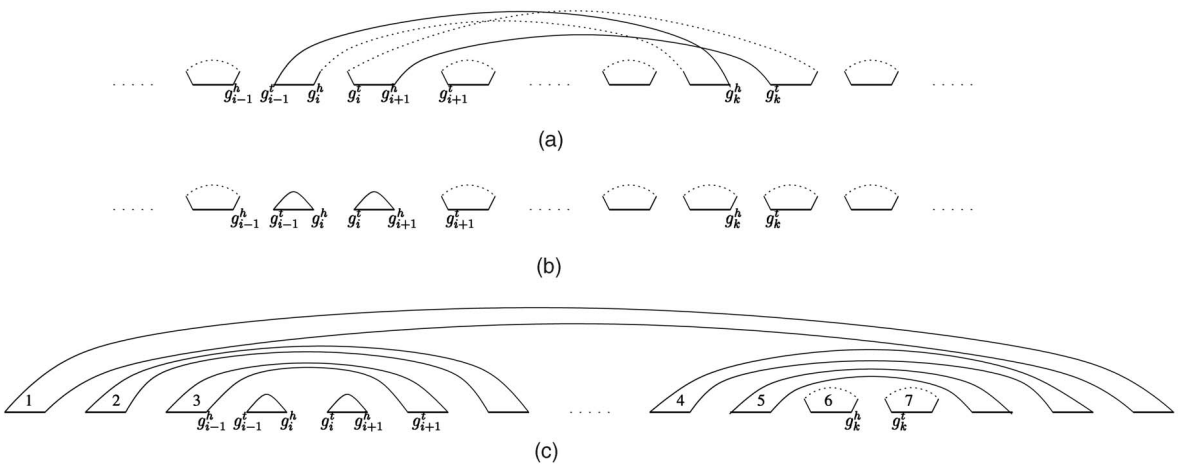


Fig. 4. (a) The breakpoint graph $B(G^m, H^m)$, (b) the breakpoint graph $B(G'^m, H'^m)$, and (c) seven possible hurdles in $B(G'^m, H'^m)$.

count the numbers of black edges, cycles, hurdles, and fortresses in the breakpoint graphs $B(G^m, H^m)$ and $B(G'^m, H'^m)$, respectively.

Consider a general case illustrated in Fig. 4. Since $g_{i-1}g_ig_{i+1}$ is identical to $h_{j-1}h_jh_{j+1}$ or its reversal and both g_{i-1} and g_{i+1} are singletons, we have $b(G'^m, H'^m) - c(G'^m, H'^m) = b(G^m, H^m) - c(G^m, H^m) - 2$. Further, observe that the number of fortresses $f(\cdot)$ takes only 0 or 1 values. Therefore, we need only to prove that the number of hurdles is increased by no more than three in (G'^m, H'^m), i.e., $h(G'^m, H'^m) - h(G^m, H^m) \leq 3$, and that the breakpoint graph of (G'^m, H'^m) is not a fortress when $h(G'^m, H'^m) - h(G^m, H^m) = 3$. This would imply $h(G'^m, H'^m) + f(G'^m, H'^m) \leq h(G^m, H^m) + f(G^m, H^m) + 3$.

Consider the *components* (i.e., the union of overlapping cycles) in the two breakpoint graphs $B(G'^m, H'^m)$ and $B(G^m, H^m)$. Observe that the components containing end nodes of g_i or g_k in $B(G^m, H^m)$ may be decomposed into multiple components in $B(G'^m, H'^m)$ since some edges have been removed (while some other edges added), while other components in $B(G^m, H^m)$ remain the same in $B(G'^m, H'^m)$. As a result, seven possible (unoriented) components could become new hurdles in the breakpoint graph of (G'^m, H'^m), as illustrated in Fig. 4c. In the figure, component 1 does not overlap with any old component containing nodes from g_i or g_k in $B(G^m, H^m)$ (see also Fig. 4a) and has a large span; components 2-5 overlap with one of the old components in $B(G^m, H^m)$ and they were created due to the removal of some edges; and components 6-7 are the two each of which contains one of the two end nodes of g_k (i.e., g_k^h or g_k^t).

By the H-P theory [14], we observe that if component 1 is a hurdle in $B(G'^m, H'^m)$ but not a hurdle in $B(G^m, H^m)$, then none of the components 2-7 could become a new hurdle in $B(G'^m, H'^m)$. Further, observe that at most three of the components 2-7 could be hurdles simultaneously in the breakpoint graph $B(G'^m, H'^m)$. Therefore, the number of hurdles in $B(G'^m, H'^m)$ is increased by no more than three, and there is only one case where three new hurdles are added. In this case, both components 6 and 7 must be hurdles. By the H-P theory [14], two *super hurdles* must be separated by some unoriented component in a breakpoint graph. Since each of the components 6 and 7 contains an end node of g_k, these components cannot be super hurdles. As a result, the breakpoint graph $B(G'^m, H'^m)$ cannot be a fortress if both components 6 and 7 are hurdles because a fortress is defined as a permutation with an odd number of hurdles, all of which are super hurdles. □

The above bound is tight. For example, $d(G'^m, H'^m) = d(G^m, H^m) + 1$ holds for genomes $G = +c + b + a + d + g + f + e + h + j + l + d + m - k + n + i$ and $H = +a + b + c + d + e + f + g + h + i + j + k + l + d + m + n$, where only one duplicated gene d exists. It implies that the second gene of d in G will correspond to the first one in H in the most parsimonious scenario transforming from G to H by reversals. This is a surprising finding as it contradicts our common intuition regarding gene order conservedness under reversals.

The next lemma gives an optimal assignment rule, and can be proven similarly to the above. Note that its optimality is due to the fact that either component 6 or component 7 must represent an adjacent pair in the breakpoint graph $B(G'^m, H'^m)$ (such that the number of hurdles cannot increase by more than two).

Lemma 4.5. *Assume that $g_{i-1}g_i$ and $g_{j-1}g_j$ are identical to $h_{k-1}h_k$ and $h_{l-1}h_l$ or their reversals, respectively. Suppose that each of the above four pairs is composed of a singleton and a duplicated gene and the four duplicated genes are from the same family and without any other gene included. Define two new genomes G' and H' from G and H by assigning orthology between the duplicated genes in $g_{i-1}g_i$ and $h_{k-1}h_k$ and between the duplicated genes in $g_{j-1}g_j$ and $h_{l-1}h_l$. Then, $d(G, H) = d(G', H')$.*

It is well known that, in the problem of sorting permutations by reversals, the distance value is highly dominated by the first two terms of the H-P formula, i.e., $\tilde{d}(G, H) = b(G, H) - c(G, H)$. Caprara [6] proved that the probability that $d(G, H) > \tilde{d}(G, H)$ is $O(1/n^5)$ for a random unsigned permutation of n elements and $O(1/n^2)$ for a random signed permutation. As an extension to genomes with duplicate genes, we can naturally define $\tilde{d}(G, H) = min_{m \in M}\tilde{d}(G^m, H^m)$, where M is the set of all the possible ortholog

assignment between G and H as mentioned earlier. By working with $\tilde{d}(G,H)$, we could potentially approximate $d(G,H)$ in SRDD. Based on this observation, one can devise the following assignment rule, which, unfortunately, does not guarantee optimality with respect to SRDD. The proof can again be done in a similar way as for Lemma 4.4.

Lemma 4.6. *Assume that $g_{i-1}g_i$ is identical to $h_{j-1}h_j$ or its reversal, where each of the pairs consists of a singleton and a duplicated gene. Define two new genomes G' and H' from G and H by assigning orthology between the duplicated genes in $g_{i-1}g_i$ and $h_{j-1}h_j$. Then, $\tilde{d}(G,H) = \tilde{d}(G',H')$.*

4.3 Minimum Common Partition

In order to formulate an efficient algorithm for the problem of SRDD, we introduce here, for the first time, a new optimization problem, called *minimum common partition* (MCP).[3] Let us first define a *segment* $\bar{g}_i$ as a substring (or its reversal) of a genome sequence G. A *partition* is a list $\{\bar{g}_1, \bar{g}_2, \cdots, \bar{g}_n\}$ of segments of a genome G such that the concatenation of the segments (or their reversals) in some order results in the genome G. The list can be thought as a *contracted representation* of G if we consider each segment $\bar{g}_i$ as a symbol. A list of segments is called a *common partition* of two related genomes G and H if it is a partition of G and a partition of H as well. Note that the contracted representations of two related genomes induced by a common partition are still related to each other. Furthermore, a *minimum common partition* is a partition with the minimum cardinality (denoted as $L(G,H)$) over all choices of common partitions of G and H. For example, for genomes $G = +c + a + b - a + d$ and $H = +c + a - b - a + d$, a minimum common partition is $\{+c + a, +b, -a + d\}$ with $L(G,H) = 3$. The *minimum common partition* (MCP) problem is naturally defined as the problem of finding the minimum common partition between two given genomes.[4] The following theorem establishes the relationship between SRDD and MCP. As mentioned earlier, the first genes of G and H, as well as the two last genes, are assumed to be identical and positive singletons.

Theorem 4.7. *Given two related genomes G and H, we have $\lceil (L(G,H)-1)/2 \rceil \le d(G,H) \le L(G,H) - 1$.*

Proof. Consider the distance between the contracted representations $\bar{G}$ and $\bar{H}$ of G and H obtained from a minimum common partition between them, respectively. Because the size of $\bar{G}$ (and $\bar{H}$) is equal to $L(G,H)$, by the H-P theory we have $d(\bar{G},\bar{H}) \le L(G,H) - 1$, where $d(\bar{G},\bar{H})$ is the reversal distance between $\bar{G}$ and $\bar{H}$. Notice that any solution that transforms $\bar{G}$ into $\bar{H}$ by reversals is also a feasible solution to transform G into H by reversals, which implies that $d(G,H) \le d(\bar{G},\bar{H})$. It follows that $d(G,H) \le L(G,H) - 1$.

Let G^m (and H^m) be the genome of G (and H, respectively) after orthologs are assigned based on an optimal solution to transform G to H by reversals, i.e., $d(G^m,H^m) = d(G,H)$. Note that any common partition between G^m and H^m is also a common partition between G and H such that $L(G,H) \le L(G^m,H^m)$ holds. On the other hand, any gene in G^m or H^m can be considered as a singleton, which makes $b_r(G^m,H^m) = L(G^m,H^m) - 1$ correct. By Theorem 4.3, we have $\lceil b_r(G^m,H^m)/2 \rceil \le d(G^m,H^m)$, from which $\lceil (L(G,H)-1)/2 \rceil \le d(G,H)$ follows. □

Theorem 4.7 suggests a way to approximate SRDD by MCP. Unfortunately, MCP is also NP-hard.

Theorem 4.8 (Goldstein et al. [12]). *Let k-MCP denote the version of MCP where each gene family is of size at most k. The problem k-MCP is NP-hard, for any $k \ge 2$.*

Now, we present an approximation algorithm for MCP. Given two related genomes G and H, a *single-match* is a pair of identical genes g_i and h_j from G and H that may have different signs. A *pair-match* is a pair of adjacent gene pairs g_ig_{i+1} and h_jh_{j+1} that are identical or the reversal of each other. Clearly, a pair-match consists of two single-matches. A common partition $\{\bar{g}_1, \bar{g}_2, \cdots, \bar{g}_n\}$ between G and H can be considered as a one-to-one mapping M from G to H. When $M(g_i) = h_j$, g_i and h_j form a single-match. If M induces a pair-match between g_ig_{i+1} and h_jh_{j+1}, we say that g_i is a *nonbreakpoint*; otherwise, it is a *breakpoint*. Denote by b the number of breakpoints in G (or H). Clearly, $b+1$ is the cardinality of the common partition induced by M. Therefore, MCP is equivalent to maximizing the number of nonbreakpoints, which is in turn equivalent to maximizing the number of pair-matches in a common partition.

Observe that two pair-matches may not coexist in a common partition. Such pair-matches are said to be *incompatible*. For two related genomes G and H, we can construct a *pair-match* graph $\mathcal{P}(V,E)$, where V consists of all possible pair-matches between G and H and E includes edges connecting incompatible pair-matches. The following lemma is straightforward.

Lemma 4.9. *The maximum independent set problem on $\mathcal{P}(V,E)$ is equivalent to the minimum common partition problem on (G,H).*

Since the complement of an independent set of $\mathcal{P}(V,E)$ is a vertex cover of $\mathcal{P}(V,E)$, we can approximate MCP by using an efficient approximation algorithm for the vertex cover (VC) (e.g., the standard greedy algorithm with approximation ratio 2). An outline of our approximation algorithm is shown in Fig. 5.

If one assumes that the approximation algorithm for vertex cover has ratio r, we can obtain an upper bound on the performance of APPROX-MCP as follows:

Lemma 4.10. *If the size of the common partition found by the APPROX-MCP algorithm is l, then $l \le (r-1)(|V|-n) + r \cdot L(G,H)$, where $|V|$ is the size of the vertex set of the pair-match graph and n is the size of genome G. In particular, for 2-MCP, the above algorithm achieves an approximation ratio of 1.5.*

Proof. Let C^* denote an optimal solution of minimum vertex cover on $\mathcal{P}(V,E)$ and C the solution output from an approximation algorithm, where $|C|/|C^*| \le r$. Let l denote the number of segments in the common partition obtained from the solution C for minimum vertex cover. By Lemma 4.9, we have $|C^*| = |V| - n + L(G,H)$ and $|C| = |V| - n + l$, from which $l \le (r-1)(|V|-n) + r \cdot L(G,H)$ follows.

3. The problem of MCP was also independently introduced recently in [21], under the name *sequence cover*.

4. The MCP problem can be naturally generalized. One variant is to find a minimum common partition between two strings where no sign is involved and another is to find a minimum common partition among multiple genomes.

Algorithm APPROX-MCP(G, H)
/* G and H are a pair of related genomes. */
1. Construct the pair-match graph $\mathcal{P}(V, E)$ for G and H
2. Find an approximation of the vertex cover C of $\mathcal{P}$
3. Identify the segments based on the pair-matches in $V - C$
4. Output all the segments as a common partition of G and H

Fig. 5. An efficient approximation algorithm for MCP.

In order to prove the approximation ratio for 2-MCP, we first prove $|V| \leq n$ by considering two cases of gene duplicates as shown below:

$$G : \ldots ab \ldots ab \ldots \quad G : \ldots ab \ldots a \ldots b \ldots$$
$$H : \ldots ab \ldots ab \ldots \quad H : \ldots ab \ldots ab \ldots$$

In the case shown on the left, gene b can be absorbed into gene a, i.e., these two genes can be regarded as only one gene during the process of assigning orthology while guaranteeing the optimality. Therefore, we can assume that there are no such pair-matches in G or H. In the case on the right, there are exactly two pair-matches beginning with gene a in G, which implies $|V| \leq n$ and, thus, $l \leq r \cdot L(G, H)$ for 2-MCP.

Moreover, we observe that the pair-match graph of $\mathcal{P}(V, E)$ is 6-claw-free [13] since the degree of a vertex is at most four. Using a 1.5-approximation algorithm for vertex cover on a 6-claw-free graph [13], a 1.5-approximation algorithm for 2-MCP can be obtained. □

4.4 Maximum Cycle Decomposition

Although the contracted representations of the genomes G and H may still contain duplicates, we do not expect the number of duplicates to be large. Again, we define another new problem, called *maximum cycle decomposition* (MCD), to complete the solution for SRDD.

The *complete graph* associated with a pair of related genomes $G = (g_1 g_2 \cdots g_n)$ and $H = (h_1 h_2 \cdots h_n)$, denoted $\mathcal{G}(V, E)$, includes the partial graphs of both G and H as subgraphs, plus cross-genome edges joining g_i^s and h_j^s, where $1 \leq i \leq n, 1 \leq j \leq n, s \in \{t, h\}$ if g_i and h_j are identical genes [9]. For example, the complete graph for genomes $G = +c - a - b + a + d$ and $H = +c + a + b + a + d$ is shown in Fig. 6.

The *maximum cycle decomposition* (MCD) problem is the problem of decomposing a given complete graph into a maximal set of cycles such that 1) every vertex belongs to exactly one cycle, except for the first and last vertices of each genome; 2) the two vertices representing each gene must be connected, respectively, to the two vertices of some (identical) gene in the other genome by edges of the cycles, i.e., the connections satisfy a pairing condition; and 3) edges within a genome and across genomes alternate in a cycle. The following theorem is a simple extension of the H-P formula and gives tighter bounds on reversal distance in terms of MCD compared to the bounds in terms of MCP in the previous section.

Theorem 4.11. *Given two related genomes G and H, we have $n - 1 - C \leq d(G, H) \leq n - 1 - C_4$, where n is the size of G, C is the number of cycles in the maximum cycle decomposition, and C_4 is the maximal number of cycles of size four in any feasible solution to MCD.*

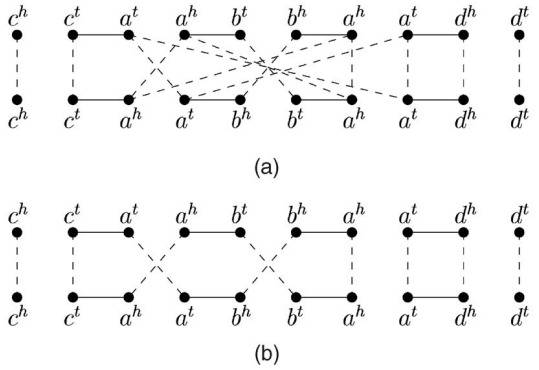


Fig. 6. (a) The complete graph of two related genomes $G = +c - a - b + a + d$ and $H = +c + a + b + a + d$, in which each gene is represented by two vertices. Dashed lines are cross-genome edges, while solid lines connect two adjacent vertices belonging to different genes of the same genome. (b) A maximum cycle decomposition consisting of two cycles.

Proof. Observe that any optimal solution to transform G into H by reversals could result in a feasible cycle decomposition (of size C^*) of the complete graph between G and H. The latter, however, may not be maximal (i.e., $C^* \leq C$). By the H-P theory, we have $n - 1 - C \leq n - 1 - C^* \leq d(G, H)$. On the other hand, any feasible cycle decomposition (e.g., the one has the maximal number of cycles of size four) could provide a solution to transform G into H by reversals (of size $d^*(G, H)$) because the complete graph after cycle decomposition is a standard breakpoint graph as defined in [14], which, however, may not be optimal (i.e., $d(G, H) \leq d^*(G, H)$). Further observe that any cycle of size four in a cycle decomposition would not be broken by reversals in its corresponding solution (i.e., the one whose breakpoint graph includes this cycle of four) since it actually represents an adjacent consecutive pair between G and H. Again, by the H-P theory we have $d(G, H) \leq d^*(G, H) \leq n - 1 - C_4$. □

The complexity of MCD has been proven to be NP-hard [11] by a reduction from the problem of MAX-ACD [6], i.e., finding the maximum-cardinality of edge-disjoint alternating cycle decomposition in a breakpoint graph associated with an unsigned permutation without duplicates. Therefore, we use a greedy algorithm in SOAR to solve MCD, as outlined in Fig. 7. The basic idea of the algorithm is to find small cycles that satisfy the above three conditions. Intuitively, small cycles result in large cycle decompositions, although it is not always the case.

Our heuristic algorithm for SRDD combines the three (sub)optimal ortholog assignment rules, APPROX-MCP and GREEDY-MCD, as outlined in Fig. 8. We note that this heuristic algorithm also works for genomes with unequal gene contents, although the above discussion and claimed properties assume two related genomes with equal gene content.

5 Experimental Results

In order to test the performance of SOAR as a tool to assign orthologs, we have applied it to both simulated data and real genome sequence data and compared its results with two algorithms in the literature, namely, an iterated version of the exemplar algorithm [18] and INPARANOID [17].

Algorithm GREEDY-MCD(G, H)
/* G and H are a pair of related genomes */
1. Construct the complete graph $\mathcal{G}(V, E)$ for G and H
2. **while** V is not empty **do**
   a. Select a vertex from V
   b. Find a shortest cycle that passes through the selected vertex while not violating the pairing constraint
   c. Remove all vertices and edges in the shortest cycle
3. Output all the cycles found as a cycle decomposition

Fig. 7. The greedy algorithm for maximum cycle decomposition.

Algorithm ALG-SRDD(G, H)
/* G and H are a pair of related genomes */
1. Apply the three (sub)optimal rules given in Lemmas 4.4 – 4.6
2. Run the APPROX-MCP algorithm
3. Run the GREEDY-MCD algorithm
4. Sort G into H according to the above cycle decomposition

Fig. 8. An outline of the heuristic for SRDD.

5.1 Simulated Data

We use simulated data to assess the performance of our heuristic algorithm for SRDD. In order to make a comparison test, we implemented the exemplar algorithm of Sankoff [18] and extended it into a tool for assigning orthologs. Although the exemplar algorithm was not originally proposed for the purpose of ortholog assignment, its objective is closely related to that of solving the ortholog assignment problem. More precisely, the algorithm looks for a pair of homologous (or identical) genes from two genomes that are direct descendants of the same gene in their most recent common ancestral genome. Therefore, for any two given genomes, the pairs of genes found by the exemplar algorithm should be orthologous to each other. The basic idea of the exemplar algorithm is to delete all but one member of each gene family (i.e., its exemplar) in each genome being compared and return a pair of two reduced genomes with the minimum rearrangement distance. We further iterate the exemplar algorithm by marking the output orthologs as singletons and repeating the algorithm on the new genomes again and again until no duplicated genes are left in the genomes.

The simulated data is generated as follows: Start from a genome G with n distinct symbols whose signs are generated randomly. Each symbol defines a single gene family. Then, randomly combine two gene families into a new family until r singletons are left in the genome G. In order to obtain a related genome H, perform k reversals on the genome G. The boundaries of these reversals are uniformly distributed within the size of the genome. Therefore, the triple (n, r, k) specifies the parameters for generating a pair of related genomes.

We ran the exemplar algorithm and ALG-SRDD on 10 random instances for each combination of parameters. Each run of ALG-SRDD typically takes less than 1 second on a PC with 2.4GHz CPU and 1GB RAM, while the exemplar algorithm needs about 4 seconds. Fig. 9 shows the average performance of both algorithms over 10 instances in terms of the number of incorrectly assigned orthologs and the reversal distance. On average, the number of genes with incorrectly assigned orthologs generally increases as the number of reversals k increases. However, our heuristic always produces fewer incorrect ortholog assignments than

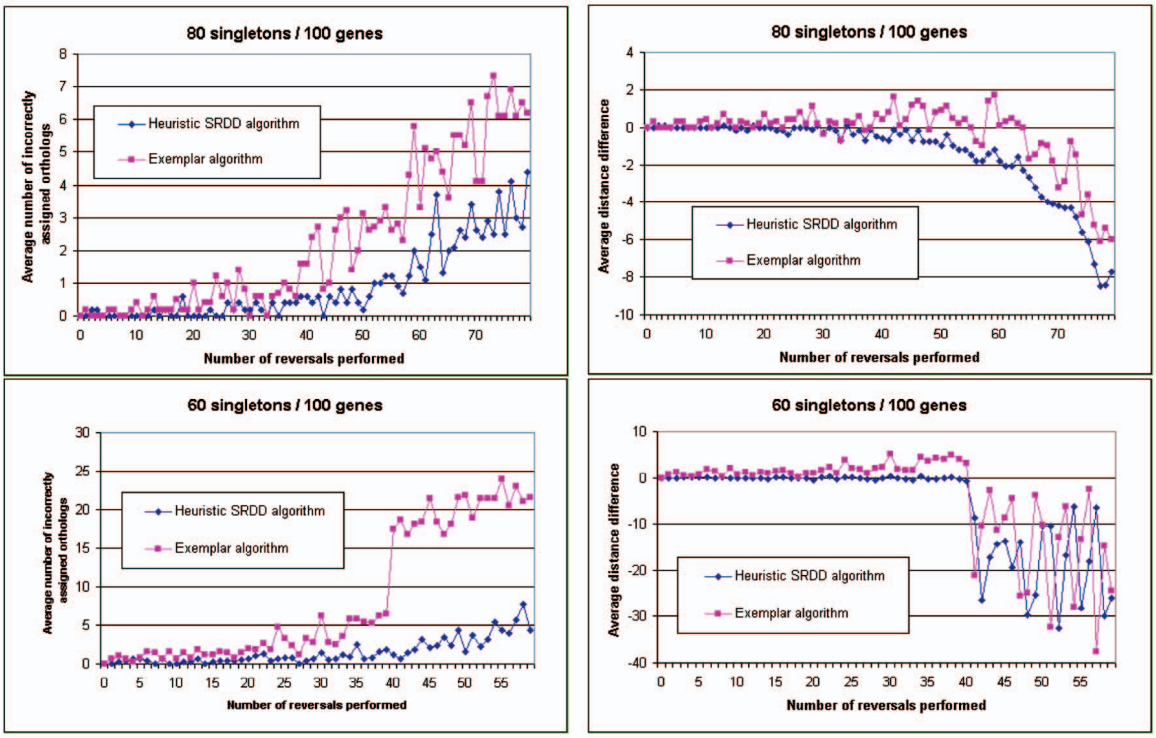


Fig. 9. Comparison of ALG-SRDD and the exemplar algorithm on simulated data. The two pictures on the left depict the average numbers of incorrectly assigned orthologs on instances with different numbers of reversals and the pictures on the right depict the average difference between the calculated reversal distance and the actual reversal distance.

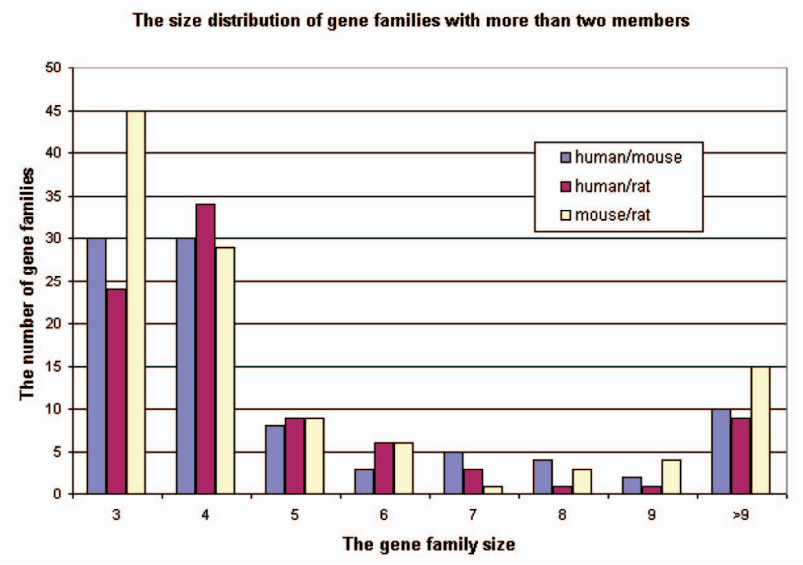


Fig. 10. The size distribution of gene families constructed for each pair of genomes. The numbers of gene families of size two are not shown in the figure for clarity (they are 355, 321, and 348, respectively).

the exemplar algorithm. In fact, the gap grows with the number of reversals and the number of duplicates. For $n = 100$ and $r = 80$, our algorithm produces an average of one or fewer errors for each $k \leq 50$ and an average of 5 or fewer errors for each $k \leq 80$ reversals. For $r = 60$, our algorithm then produces an average of one or fewer errors for each $k \leq 20$ and 5 or fewer errors for each $k \leq 50$ reversals. These statistics indicate that our method is quite reliable in assigning orthologs. We also observe that the distance values calculated by both algorithms are almost equal to the actual reversal distance k when k is small (or moderately large). However, as k increases, both algorithms constantly underestimate the actual reversal distance (even when no duplicated gene exists in every genome of interest [27]).

5.2 Real Genome Sequence Data

Here, we analyzed the X chromosomes of human (*Homo-sapiens*, NCBI build 34, July 2003; UCSC hg16), mouse (*Mus musculus*, NCBI build 32, October 2003; UCSC mm4), and rat (*Rattus norvegicus*, Baylor HGSC v. 3.1, June 2003; UCSC rn3). We excluded genes whose (protein) sequences are not available in Genbank (as of January 2004), we downloaded 922 genes from the human X chromosome, 1,030 genes from mouse, and 899 genes from rat, respectively. In the first step of SOAR, an all-*versus*-all sequence comparison was performed between each pair of chromosomes using BLASTp and then families of homologous genes were constructed. The size distribution of gene families (consisting of genes from a pair of genomes) is illustrated in Fig. 10. Because orthology between two singletons (one from each genome) from gene families of size two can be trivially assigned, we are left with 92 families between human and mouse (87 between human and rat and 102 between mouse and rat, respectively) whose orthology needs to be determined. By using ALG-SRDD, SOAR assigned 583 pairs of orthologous genes between human and mouse, 518 pairs between human and rat, and 599 pairs between mouse and rat (as shown in Table 1).

The genome rearrangement between human and mouse X chromosomes found by ALG-SRDD consists of 123 reversal operations and its breakpoint graph includes 143 breakpoints, which is depicted in Fig. 11. In contrast, by studying synteny blocks, Pevzner and Tesler [16] reported that there are at least seven macrorearrangements (with a DNA sequence span > 1 Mb) and 177 microrearrangements, some of which may be inaccurate due to assembly errors. Note that the synteny blocks are required to be unique when generated so that the H-P formula can be applied to calculate the reversal distance. The breakpoint graphs showing genome rearrangement for human/rat (117 reversals and 135 breakpoints) and for mouse/rat (155 reversals and 188 breakpoints) obtained by ALG-SRDD are also shown in Fig. 11. Given these

TABLE 1
Comparison of Ortholog Assignments by INPARANOID and SOAR

	assignable	INPARANOID			SOAR			common
	orthologs[†]	assigned[‡]	specificity[η]	sensitivity[ζ]	assigned	specificity	sensitivity	orthologs[ξ]
human/mouse	300	527	65.5%	92.6%	583	62.4%	94.6%	509
human/rat	116	468	78.8%	93.1%	518	75.8%	92.2%	448
mouse/rat	115	542	81.8%	98.2%	599	81.4%	99.1%	524

[†] *The total number of assignable ortholog pairs between two chromosomes.* [‡] *The total number of ortholog pairs assigned.* [η] *The number of true positives divided by the total number of true positives and false positives (excluding pairs with unknown names).* [ζ] *The percentage of true positives among all assignable ortholog pairs.* [ξ] *The number of orthologs assigned by both INPARANOID and SOAR.*

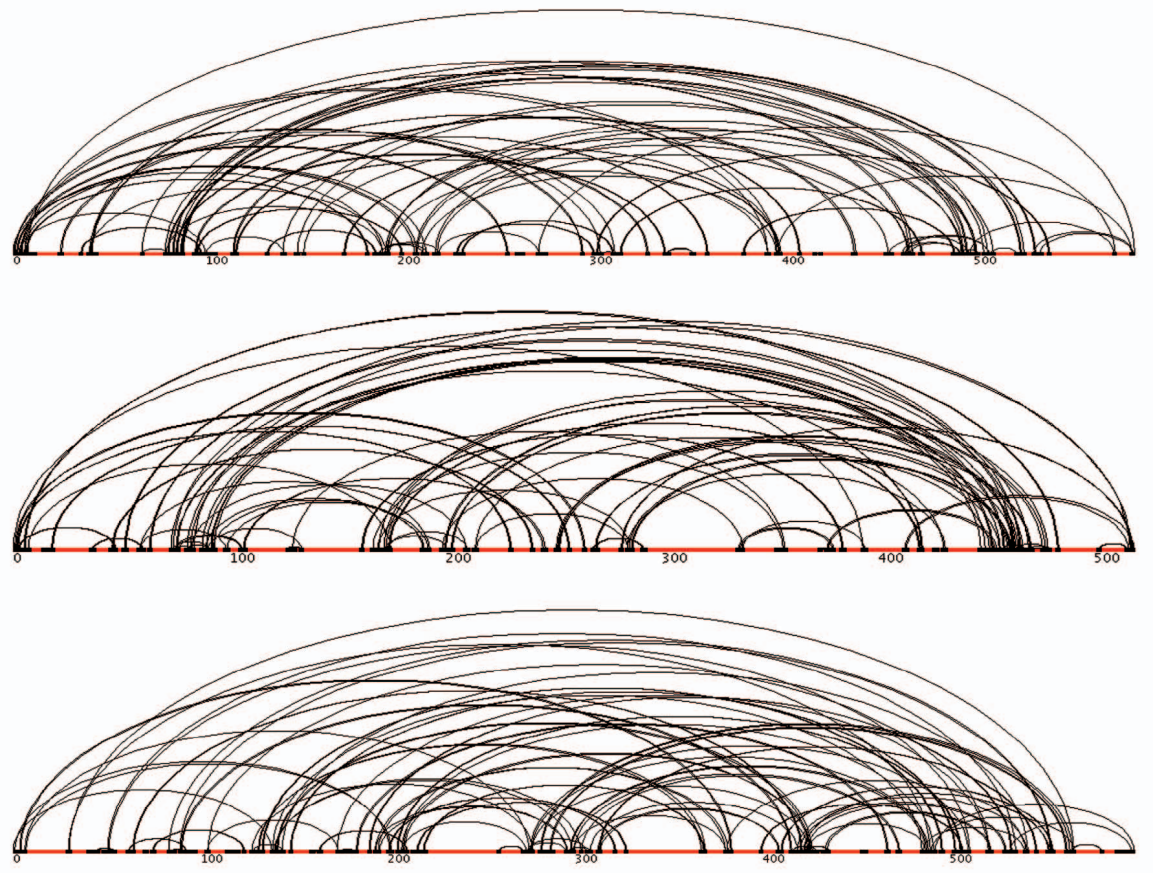


Fig. 11. The breakpoint graph of human and mouse X chromosomes (top), of human and rat X chromosomes (middle), and of mouse and rat X chromosomes (bottom). Note that the pictures only show the orderings of the genes in the genomes, not their physical locations.

breakpoint graphs, orthologs between each pair of chromosomes can be assigned in a straightforward way. The resulting dot-plot graphs showing orthology mappings are illustrated in Fig. 12. The dot-plot graph between human and mouse is very similar to the one obtained by Pevzner and Tesler [16], which uses (unique) synteny blocks. However, if we disregard the difference in coordinate units, there is an interesting and noticeable difference between them. The difference occurs in the synteny block (ID 279) spanning from coordinate 78,691,856bp to 83,712,889bp in human X chromosome with the block spanning from coordinate 100,126,658bp to 105,951,929bp in mouse X chromosome. Pevzner and Tesler's results were based on the old versions of human assembly (NCBI build 30, June 2002; UCSC hg12) and mouse assembly (MGSC v3, Februry 2002; UCSC mm2) and reported that these two synteny blocks correspond to each other with opposite orientations [16]. We looked at the genes that are located in these blocks. Because different versions of genome assemblies were investigated, the coordinates of synteny blocks have greatly changed. Using a tool that converts genome coordinates between assemblies (from http://genome.ucsc.edu/cgi-bin/hgLiftOver), we found that the corresponding block in human NCBI build 34 spans from 82,083,913bp to 87,212,145bp, where 13 annotated genes were located. Unfortunately, we could not find a way to convert coordinates between different mouse assemblies. Thus, the exact coordinates of block 279 in mouse NCBI build 32 are unknown to us. Out of the 13 human genes, however, 10 are assigned orthologous genes that are closely located in a block chrX:101,348,568-106,260,520 in the new version of mouse assembly and two genes have no homologous hits at all.[5] In addition, by comparing gene names between these two mouse assemblies (via http://genome-mm2.cse.ucsc.edu/cgi-bin/hgGateway), three orthologous genes can also be found in the synteny block 279 of mouse MGSC v3 assembly, further indicating that the block chrX:101,348,568-106,260,520 in the new assembly probably corresponds to synteny block 279 of the previous version of mouse assembly (MGSC v3, February 2002; UCSC mm2). Of all these 10 orthologous pairs, each has the same transcription direction, which suggests the same orientation of the two human and mouse blocks. This discrepancy can be viewed as a sampling problem where gene-based maps are examined instead of sequence-based maps, and would be worth an in-depth investigation.

5. One may wonder why the remaining gene, named *RPS6KA6*, was assigned a mouse orthologous gene that is far from the block chrX:101,348,568-106,260,520 in mouse genome. We examined all the annotated genes in mouse NCBI build 32 assembly and found no gene with name *RPS6KA6* (see our Web site for the detailed experimental data, which was downloaded from the NCBI GenBank database as of January 2004). It implies that the human gene *RPS6KA6* was destined to be given a wrong ortholog by our approach because its true mouse ortholog was missing. We further looked at the latest version of mouse assembly (NCBI build 33 May 2004; UCSC mm5), and found a gene with the name *Rps6ka6*, which is located at chrX:102,716,041-102,790,574 and really within the block chrX:101,348,568-106,260,520 (if we assume no or small coordinate shift between mouse NCBI build 32 assembly and NCBI build 33 assembly). Detailed information can be found on the Web site http://www.cs.ucr.edu/~xinchen/soar.htm.

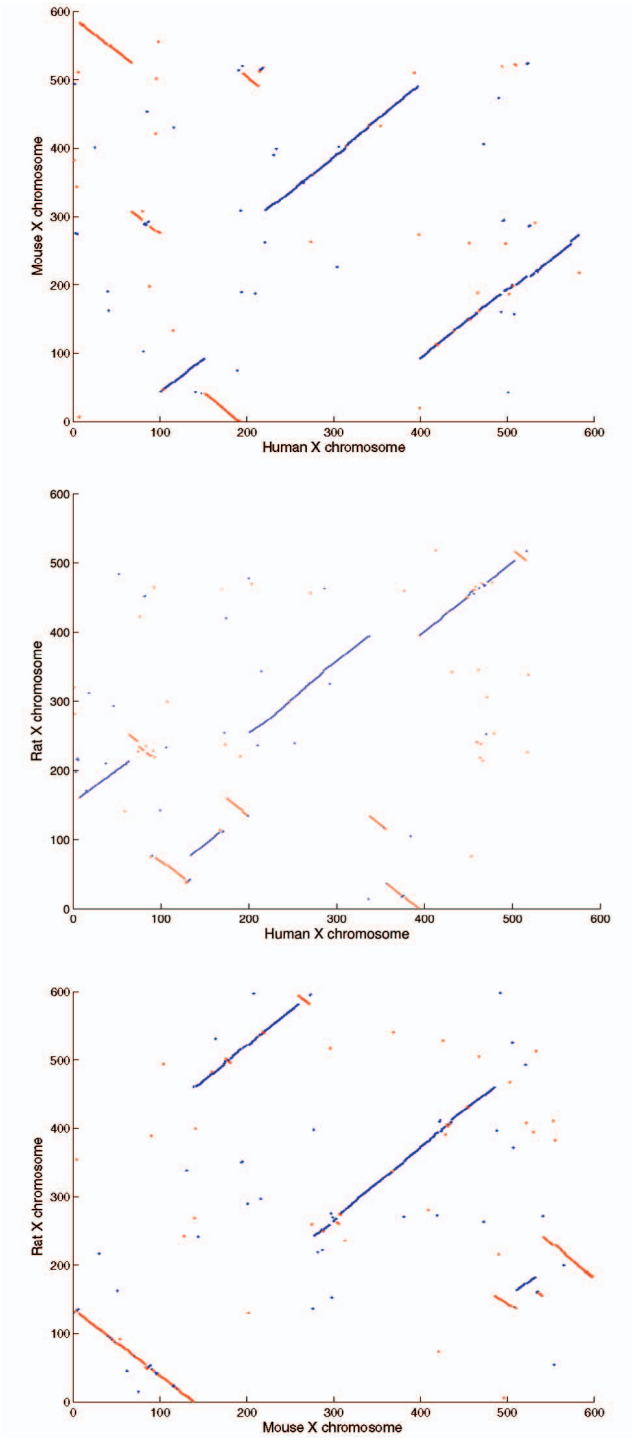


Fig. 12. Dot-plot graphs showing ortholog mapping between pairs of human, mouse, and rat X chromosomes. The X-axis/Y-axis represent the human/mouse/rat X chromosomes in terms of genes instead of DNA sequences as used in [16]. A blue point in the graph represents an orthologous pair of genes having the same transcriptional direction, while a red point represents an orthologous pair of genes that have different transcriptional directions.

The number of rearrangement events that occurred between mammalian X chromosomes is believed to be well below 40 percent of the number of common genes involved [3]. According to the preceding simulation results, among the 583 pairs of assigned orthologs between human and mouse X chromosomes, less than eight pairs are expected to be incorrectly assigned if we could assume that no other rearrangement events than reversals have taken place since the split of human and mouse.

Unfortunately, not much is known (or has been experimentally verified) about the true orthology among these genes because the functions of them are mostly unknown. Therefore, we took an indirect approach to validate our assignments by using the gene annotation information in Genbank, namely, gene names (or symbols). The name of a gene is usually given to convey the character or function of the gene [26]. Genes with identical names are likely to be an orthologous pair, although genes with different names could still be orthologs due to naming inconsistency in Genbank. Some of the names begin with "LOC," implying that these genes have not yet been assigned official names/symbols. Gene pairs involving such names are ignored in all subsequent statistics. If a pair of genes output by SOAR have completely identical names, we count them as a true positive pair; otherwise, it is counted as a false positive pair. We also calculate the total number of *assignable* pairs of orthologs between all three pairs of X chromosomes in our dataset, i.e., the total number of pairs of genes with identical names. For example, there are 300 assignable ortholog pairs between human and mouse. Among the 583 ortholog pairs predicted by SOAR, 284 are true positives, 128 involve genes without official names, and 171 are false positives, resulting in a sensitivity of 94.6 percent and a specificity of 62.4 percent (see Table 1 for definitions). Observe that the specificity number may not be accurate because genes with different names could still be orthologs.

In order to compare SOAR with existing homology-based methods for ortholog assignment, we implemented the INPARANOID algorithm described in [17]. The latter algorithm relies on BBHs (i.e., bidirectional best hits) between genes from two genomes. In practice, BBHs have been widely used to assign orthologs between two species, e.g., the HomoloGene database of Genbank (http://www.ncbi.nlm.nih.gov/HomoloGene). In our experiment, INPARANOID reported 527 ortholog pairs between human and mouse with a sensitivity of 92.6 percent, which is slightly worse than that of SOAR, and a specificity of 65.5 percent, which is slightly better than that of SOAR (but recall that the specificity numbers are not very reliable). One of the ortholog pairs that were correctly assigned by SOAR but not by INPARANOID is gene MAGEA4 of human and gene Magea4 of mouse. Since BLASTp did not give any BBH between MAGEA4 and Magea4, INPARANOID failed to assign orthology between them. There are a total of 509 pairs of orthologs assigned by both INPARANOID and SOAR, accounting for 96.5 percent of all the ortholog pairs assigned by INPARANOID, indicating the agreement between these two different approaches. More experimental results can be found on the SOAR Webpage (http://www.cs.ucr.edu/~xinchen/soar.htm).

The comparative results on all three pairs of genomes are summarized in Table 1. The results demonstrate that SOAR and INPARANOID make similar assignments, although they use different methods to assign orthologs, i.e., one is entirely based on homology search and the other relies mainly on genome rearrangement. SOAR performs slightly better than INPARANOID in terms of sensitivity, even though the validation method is a bit biased in favor of

homology-based ortholog assignment methods since many genes in Genbank were named via homology search.

6 Conclusion and Future Research

In this paper, we presented a novel approach to ortholog assignment that takes into account both sequence similarity and evolutionary events (i.e., reversals) at the genome level. We formulated the problem as that of computing the signed reversal distance with duplicates (SRDD) between the two genomes of interest. The problem was decomposed into two new optimization problems (MCP and MCD), for which we designed and analyzed efficient algorithms. We implemented the algorithm in a system for assigning orthologs on a genome scale, called SOAR.

Our preliminary experiments on simulated and real data have demonstrated that ortholog assignment via genome rearrangement is a very promising method. The current version of SOAR does not consider genome rearrangement events such as transposition, gene loss, and gene insertion. It also ignores the issue of inparalogs and works only with single-chromosomal genomes. We plan to look into these extensions in the future.

Acknowledgments

The authors thank the anonymous referees for many valuable comments and several useful references. This project is supported in part by US National Science Foundation grants ITR-0085910, CCR-0309902, and DBI-0321756, a US Department of Energy Genomics:GTL subcontract, and National Key Project for Basic Research (973) grant 2002CB512801.

References

[1] S. Altschul, T. Madden, A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. Lipman, "Gapped BLAST and PSI-BLAST: A New Generation of Protein Database Search Programs," *Nucleic Acids Research,* vol. 25, no. 17, pp. 3389-3402, 1997.

[2] D. Bader, B. Moret, and M. Yan, "A Linear-Time Algorithm for Computing Inversion Distance between Signed Permutations with an Experimental Study," *J. Computational Biology,* vol. 8, no. 5, pp. 483-491, 2001.

[3] G. Bourque and P. Pevzner, "Genome-Scale Evolution: Reconstructing Gene Orders in the Ancestral Species," *Genome Research,* vol. 12, pp. 26-36, 2002.

[4] S.B. Cannon and N.D. Young, "OrtholParaMap: Distinguishing Orthologs from Paralogs by Integrating Comparative Genome Data and Gene Phylogenies," *BMC Bioinformatics,* vol. 4, no. 1, p. 35, 2003.

[5] A. Caprara, "Sorting by Reversals Is Difficult," *Proc. First Ann. Int'l Conf. Computational Molecular Biology,* pp. 75-83, 1997.

[6] A. Caprara, "Sorting Permutations by Reversals and Eulerian Cycle Decompositions," *SIAM J. Discrete Math.,* vol. 12, no. 1, pp. 91-110, 1999.

[7] D. Christie and R. Irving, "Sorting Strings by Reversals and by Transpositions," *SIAM J. Discrete Math.,* vol. 14, no. 2, pp. 193-206, 2001.

[8] K. Chao and W. Miller, "Linear Space Algorithms that Build Local Alignments from Fragments," *Algorithmica,* vol. 13, pp. 106-134, 1995.

[9] N. El-Mabrouk, "Reconstructing an Ancestral Genome Using Minimum Segments Duplications and Reversals," *J. Computer and System Sciences,* vol. 65, pp. 442-464, 2002.

[10] W.M. Fitch, "Distinguishing Homologous from Analogous Proteins," *Systematic Zoology,* vol. 19, pp. 99-113, 1970.

[11] Z. Fu, "Assignment of Orthologous Genes for Multichromosomal Genomes Using Genome Rearrangement," UCR CS technical report, 2004.

[12] A. Goldstein, P. Kolman, and J. Zheng, "Minimum Common String Partition Problem: Hardness and Approximations," *Proc. 15th Int'l Symp. Algorithms and Computation (ISAAC),* pp. 473-484, 2004.

[13] M.M. Halldorsson, "Approximating Discrete Collections via Local Improvements," *Proc. Sixth ACM-SIAM Symp. Discrete Algorithms,* pp. 160-169, 1995.

[14] S. Hannenhalli and P. Pevzner, "Transforming Cabbage into Turnip (Polynomial Algorithm for Sorting Signed Permutations by Reversals)," *Proc. 27th Ann. ACM Symp. Theory of Computing,* pp. 178-187, 1995.

[15] H. Kaplan, R. Shamir, and R. Tarjan, "Faster and Simpler Algorithm for Sorting Signed Permutations by Reversals," *Proc. Eighth Ann. ACM-SIAM Symp. Discrete Algorithms,* pp. 344-351, 1997.

[16] P. Pevzner and G. Tesler, "Genome Rearrangements in Mammalian Evolution: Lessons from Human and Mouse Genomes," *Genome Research,* vol. 13, pp. 37-45, 2003.

[17] M. Remm, C. Storm, and E. Sonnhammer, "Automatic Clustering of Orthologs and In-Paralogs from Pairwise Species Comparisons," *J. Molecular Biology,* vol. 314, pp. 1041-1052, 2001.

[18] D. Sankoff, "Genome Rearrangement with Gene Families," *Bioinformatics,* vol. 15, no. 11, pp. 909-917, 1999.

[19] E. Sonnhammer and E. Koonin, "Orthology, Paralogy and Proposed Classification for Paralog Subtypes," *Trends in Genetics,* vol. 16, pp. 227-231, 2000.

[20] P. Slonimski, M. Mosse, P. Golik, A. Henault, Y. Diaz, J. Risler, J. Comet, J. Aude, A. Wozniak, E. Glemet, and J. Codani, "The First Laws of Genomics," *Microbial & Comparative Genomics,* vol. 3, p. 46, 1998.

[21] K. Swenson, M. Marron, J. Earnest-DeYoung, and B. Moret, "Approximating the True Evolutionary Distance between Two Genomes," Technical Report TR-CS-2004-15, Univ. of New Mexico, 2004.

[22] C. Storm and E. Sonnhammer, "Automated Ortholog Inference from Phylogenetic Trees and Calculation of Orthology Reliability," *Bioinformatics,* vol. 18, no. 1, 2002.

[23] R.L. Tatusov, M.Y. Galperin, D.A. Natale, and E.V. Koonin, "The COG Database: A Tool for Genome-Scale Analysis of Protein Functions and Evolution," *Nucleic Acids Research,* vol. 28, pp. 33-36, 2000.

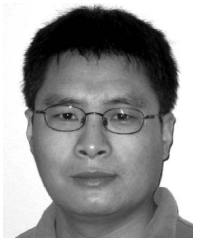
[24] R.L. Tatusov, E.V. Koonin, and D.J. Lipman, "A Genomic Perspective on Protein Families," *Science,* vol. 278, pp. 631-637, 1997.

[25] J. Tang and B. Moret, "Phylogenetic Reconstruction from Gene Rearrangement Data with Unequal Gene Contents," *Proc. Eighth Workshop Algorithms and Data Structures, (WADS '03)* pp. 37-46, 2003.

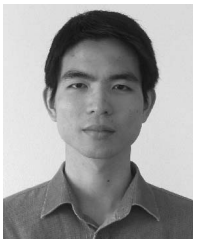
[26] H.M. Wain, E.A. Bruford, R.C. Lovering, M.J. Lush, M.W. Wright, and S. Povey, "Guidelines for Human Gene Nomenclature," *Genomics,* vol. 79, no. 4, pp. 464-470, 2002.

[27] L. Wang and T. Warnow, "Estimating True Evolutionary Distances between Genomes," *Proc. 33rd Ann. ACM Symp. Theory of Computing,* pp. 637-646, 2001.

[28] Y.P. Yuan, O. Eulenstein, M. Vingron, and P. Bork, "Towards Detection of Orthologues in Sequence Databases," *Bioinformatics,* vol. 14, no. 3, pp. 285-289, 1998.



Xin Chen received the PhD degree in mathematics from Peking University, Beijing, People's Republic of China, in 2001. He is now a postdoctorate in the Department of Computer Science and Engineering at the University of California, Riverside. His recent research interests fall in the general areas of computational biology and bioinformatics. He won the Best Paper Award twice at the International Conference on Genomic Informatics (GIW), in 1999 and 2004, respectively.



Jie Zheng received the BE degree in computer science and engineering from Zhejiang University, Hangzhou, People's Republic of China, in June 2000. From July 2000 to August 2001, he served as a research assistant in the State Key Lab of CAD & CG, Zhejiang University. He is a PhD candidate, coadvised by Dr. Tao Jiang and Dr. Stefano Lonardi, in the Department of Computer Science and Engineering at the University of California, Riverside. His research interests include bioinformatics, combinatorial optimization, computational complexity, and stringology.

Zheng Fu reveived the BS and MS degrees in computer science and technology from Renmin University of China in 1999 and 2002. She is a PhD candidate in computer science at the University of California, Riverside. Her research interests fall in the general areas of computational biology and bioinformatics.

Peng Nan received the BS degree from the Department of Chemistry at Wuhan University of China, Wuhan, People's Republic of China, in July 1987 and the PhD degree in biology from the South China Institute of Botany, Academia Sinicca, Guangzhou, China, in October 2002. She was a research associate at the Wuhan Institute of Botany, Academia Sinica, China, from July 1987-June 2000 and is now an associate professor in the School of Life Sciences, Fudan University, China. She is also a member of the Shanghai Center for Bioinformatics Technology. Her recent research interests include botanical products for bioactivity testing, drug discovery from medicinal plants, and data mining from specialty databases.

Yang Zhong received the BS degree in information science from the University of Science and Technology of China, Hefei, China, in 1984. He was a researcher at the Wuhan Institute of Botany, Academia Sinica from August 1984-December 1999 and is now a professor of botany and bioinformatics and the executive dean of the School of Life Sciences at Fudan University, Shanghai, China. He is also a deputy director of the Shanghai Center for Bioinformation Techology and a visiting professor at Beijing University Center for Theoretical Biology and Tibet University. His recent research interests include plant molecular evolution and biodiversity informatics.

Stefano Lonardi received the "Laurea cum laude" from the University of Pisa in 1994 and the PhD degree in the Summer of 2001 from the Department of Computer Sciences, Purdue University, West Lafayette, Indiana. He also holds a doctorate degree from the University of Padua (1999). He is an assistant professor at the University of California, Riverside. Stefano's recent research interests include algorithms, computational molecular biology, and data mining. He has published in several major theoretical computer science and computational biology journals and conferences. In the year 2005, he received the CAREER award from US National Science Foundation. He is a member of the IEEE Computer Society.

Tao Jiang received the BS degree in computer science and technology from the University of Science and Technology of China, Hefei, People's Republic of China, in July 1984 and the PhD degree in computer science, from the University of Minnesota in November 1988. He was a faculty member at McMaster University, Hamilton, Ontario, Canada, from January 1989-July 2001 and is now a professor of computer science and engineering at the University of California, Riverside. He is also a member of the UCR Institute for Integrative Genome Biology, Center for Plant Cell Biology, Shanghai Center for Bioinformatics Technology, Beijing University Center for Theoretical Biology, and the Tsinghua University Theoretical Computer Science graduate program. His recent research interests include algorithms, computational molecular biology, computational complexity, and computational aspects of information gathering and retrieval. He is a senior member of the IEEE and the IEEE Computer Society.