W696–W699 *Nucleic Acids Research, 2006, Vol. 34, Web Server issue*
doi:10.1093/nar/gkl169

SPRING: a tool for the analysis of genome rearrangement using reversals and block-interchanges

Ying Chih Lin, Chin Lung Lu[1],*, Ying-Chuan Liu and Chuan Yi Tang

Department of Computer Science, National Tsing Hua University, Hsinchu 300, Taiwan, ROC and [1]Department of Biological Science and Technology, National Chiao Tung University, Hsinchu 300, Taiwan, ROC

Received February 14, 2006; Revised and Accepted March 20, 2006

ABSTRACT

SPRING (http://algorithm.cs.nthu.edu.tw/tools/SPRING/) is a tool for the analysis of genome rearrangement between two chromosomal genomes using reversals and/or block-interchanges. SPRING takes two or more chromosomes as its input and then computes a minimum series of reversals and/or block-interchanges between any two input chromosomes for transforming one chromosome into another. The input of SPRING can be either bacterial-size sequences or gene/landmark orders. If the input is a set of chromosomal sequences then the SPRING will automatically search for identical landmarks, which are homologous/conserved regions shared by all input sequences. In particular, SPRING also computes the breakpoint distance between any pair of two chromosomes, which can be used to compare with the rearrangement distance to confirm whether they are correlated or not. In addition, SPRING shows phylogenetic trees that are reconstructed based on the rearrangement and breakpoint distance matrixes.

INTRODUCTION

With an increase in the number of genomic data (DNA, RNA and protein sequences) available, the study of genome rearrangement has received a lot of attention in computational biology and bioinformatics, owing to its applications in the measurement of evolutionary difference between two species. In this study, chromosomes considered are usually denoted by permutations of ordered and signed integers with each integer representing an identical gene in chromosomes and its sign (e.g. + or −) indicating the transcriptional orientation. Here, we use permutation and chromosome interchangeably. Given two permutations representing two linear/circular chromosomes, the genome rearrangement study is to compute the *rearrangement distance* which is defined as the minimum number of rearrangement operations required to transform one chromosome into another. The commonly used rearrangement operations that affect a permutation include reversals (also called inversions) (1–3), transpositions (4,5), block-interchanges (i.e. generalized transpositions) (6,7) and even their combinations (8,9). *Reversals* act on the permutation by inverting a block of consecutive integers into the reverse order and also changing the sign of each integer, and *transpositions* act by swapping two contiguous (or adjacent) blocks of consecutive integers. Conceptually, *block-interchanges* are a generalization of transpositions allowing the swapped blocks to be not necessarily adjacent in the permutation.

Currently, many existing tools have focused on inferring an optimal series of reversals (10,11) or an optimal series of block-interchanges (12) for transforming one chromosome into another. In this paper, we have developed a web server, called SPRING (short for Sorting Permutation by Reversals and block-INterchanGes), to compute the rearrangement distance as well as an optimal scenario between two permutations of representing linear/circular chromosomes using reversals and/or block-interchanges.

If both reversals and block-interchanges are considered together, SPRING adopts a strategy of unequal weight by using weight 1 for reversals and weight 2 for block-interchanges. This is mainly due to the following reasons. First, reversals have been favored as more frequent rearrangement operations when compared with block-interchanges. Second, a reversal affecting the chromosome removes at the most two breakpoints, whereas a block-interchange removes at the most four, where a *breakpoint* denotes two adjacent genes (g_1, g_2) in a chromosome that does not appear consecutively as either (g_1, g_2) or ($-g_2$, $-g_1$) in another chromosome. Third, the rearrangement distance involving both reversals and block-interchanges can currently be computed

*To whom correspondence should be addressed. Tel: +886 3 5712121 ext. 56949; Fax: +886 3 5729288; Email: cllu@mail.nctu.edu.tw in polynomial time only when the weight of reversals is 1 and the weight of block-interchanges is 2 (please refer to Methods for further discussion).

In addition, SPRING computes the breakpoint distance between two permutations, which can be used to compare with the rearrangement distance to see whether they are correlated or not, where the *breakpoint distance* is the number of breakpoints between two permutations.

By integrating two existing programs, respectively, called Mauve (13) and PHYLIP (14), SPRING accept not only gene-order data but also sequence data as its input, and can output evolutionary trees that are inferred based on the calculated breakpoint and rearrangement distances. In particular, if the input is sequence data, SPRING can automatically search for identical landmarks, called LCBs (Locally Collinear Blocks), which are homologous/conserved regions shared by all input sequences. Basically, an LCB is a collinear set of multi-MUMs (which are exactly matching subsequences shared by all chromosomes considered that occur only once in each chromosome and that are bounded on either side by mismatched nucleotides). In practice, it may correspond to a homologous region of sequence shared by all genomes and does not contain any genome rearrangements.

METHODS

In SPRING, we have implemented algorithms developed by Kaplan *et al*. (2) and Lin *et al*. (7) to compute the rearrangement distances between two linear/circular chromosomes by reversals and by block-interchanges, respectively. In addition, when considering both reversals and block-interchanges with weights of 1 and 2, respectively, we have adopted a new algorithm in SPRING to calculate the rearrangement distance between two linear/circular chromosomes as well as its optimal scenario. In fact, this computation can be performed using the algorithm that was proposed by Yancopoulos *et al.* (15) based on the approach of breakpoint graph. The steps of their algorithm are as follows. First, represent the input of two chromosomes as a breakpoint graph. Second, search for all so-called oriented gray edges (i.e. gray edges joining the left/right ends of two black edges), each of which actually corresponds to a reversal, and apply a cut-and-proper-join operation to each oriented gray edge (i.e. cut and rejoin in the proper way the two black edges adjacent to each oriented gray edge). Notice that after this step all remaining gray edges are unoriented [i.e. gray edges joining the left (respectively, right) end of one black edge to the right (respectively, left) end of another black edge]. Finally, cut and properly rejoin the two black edges of each unoriented gray edge, followed by applying another cut-and-proper-join to the gray edge connecting a temporary circular intermediate (CI for short), which is a cycle consisting of one black edge and one gray edge. These two consecutive cut-and-proper-join then correspond to a block-interchange.

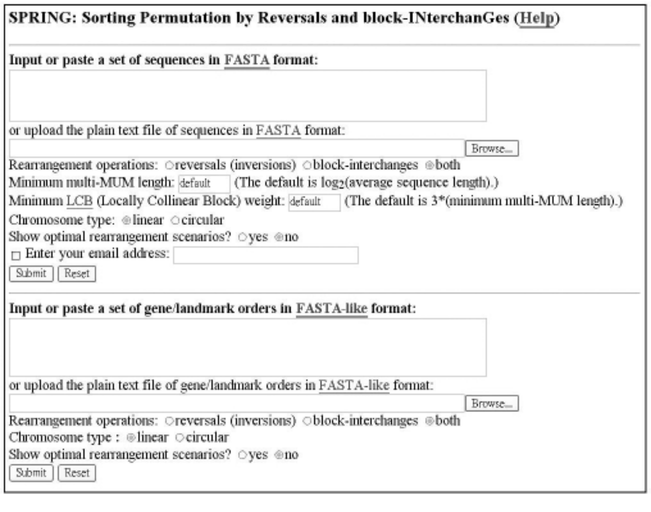
Instead of using the algorithm proposed by Yancopoulos *et al*. (15), we have adopted the following approach in SPRING to solve the same problem and with this approach we can ensure that the number of used block-interchanges in our optimal scenario is minimum over all possible optimal scenarios. First, we represent the input of two chromosomes as a breakpoint graph. Second, we identify all the so-called oriented components (i.e. those components with at least one vertex corresponding to an oriented edge) and use the algorithm proposed by Kaplan *et al*. (2) to find optimal reversals of each oriented component. Finally, we apply the algorithm proposed by Lin *et al*. (7) to each of the remaining components (that are unoriented) to find its optimal block-interchanges. In our approach, we can show that the number of block-interchanges in the optimal scenario is minimum, which seems to be reasonable from the biological viewpoint because block-interchanges have been less favored as fundamental evolutionary operations. We also show that using weight 1 for reversals and weight larger than or equal to 3 for block-interchanges will make SPRING return nothing but only reversals, meaning that in this case users can utilize SPRING to compute the rearrangement distance by choosing only reversals as rearrangement operations.

IMPLEMENTATION AND USAGE OF SPRING

The kernel algorithms of SPRING were written in C and the web interface was written in PHP. Currently, SPRING (see Figure 1 for its web interface) is installed on IBM PC with 2.8 GHz processor and 3 GB RAM under Linux system.

Input

Users can enter or paste two or more linear/circular genomic sequences or gene/landmark orders as the input of SPRING. If the input is a set of chromosomal sequences, SPRING will automatically identify all LCBs (i.e. homologous/conserved regions) as landmarks. Usually, each LCB identified is associated with a weight that can serve as a measure of confidence that it is a true homologous region rather than a random match, where the *weight* of an LCB is defined as the sum of lengths of multi-MUMs in this LCB. In SPRING, the minimum LCB weight is a user-definable parameter and its default is set to be three times the minimum multi-MUM length. Users can identify larger LCBs that are truly involved in the genome rearrangement by selecting a high minimum weight, whereas



SPRING: Sorting Permutation by Reversals and block-INterchanGes (Help)

Input or paste a set of sequences in FASTA format:

or upload the plain text file of sequences in FASTA format:

Rearrangement operations: ○reversals (inversions) ○block-interchanges ●both

Minimum multi-MUM length: default (The default is $\log_2$(average sequence length).)

Minimum LCB (Locally Collinear Block) weight: default (The default is 3*(minimum multi-MUM length).)

Chromosome type: ●linear ○circular

Show optimal rearrangement scenarios? ○yes ●no

□ Enter your email address:

Input or paste a set of gene/landmark orders in FASTA-like format:

or upload the plain text file of gene/landmark orders in FASTA-like format:

Rearrangement operations: ○reversals (inversions) ○block-interchanges ●both

Chromosome type : ●linear ○circular

Show optimal rearrangement scenarios? ○yes ●no

Figure 1. The web interface of SPRING.

by selecting a low minimum weight they can trade some specificity for sensitivity to identify smaller LCBs that are possibly involved in the genome rearrangement.

Before running SPRING, users also need to choose the used rearrangement operations that can be reversals, block-interchanges or both, the input chromosome type that can be either linear or circular, and to determine whether or not to show the optimal rearrangement scenarios. In particular, showing optimal scenarios of rearrangement is a little time-consuming for cases in which the number of input genes (or identified landmarks) is large. In these cases, users are recommended to run SPRING in a batch way, which is also suitable to cases of large-scale sequences, instead of in an immediate way (the default). In the batch way, users will be notified of the output via email when their submitted jobs are finished.

Output

If the input is a set of chromosomal sequences, then SPRING will first output the order of identified common LCBs shared by all input sequences, and then output the rearrangement distance matrix (in which each entry denotes the rearrangement distance between a pair of two input chromosomes), as well as the breakpoint distance matrix. Breakpoint distances can be used to compare with rearrangement distances to see whether they are correlated or not. In addition, SPRING shows two phylogenetic trees that are reconstructed based on the rearrangement and breakpoint distance matrixes, respectively, using a program of neighbor-joining method from the PHYLIP package.

In each of the identified LCB orders, users can see their detailed information just by clicking the associated link, such as the position (denoted by left and right end coordinates), length and weight of each LCB, and the overall coverage of all LCBs. It should be noted that if both the left and right coordinates of an identified LCB are negative values, then this LCB is the inverted region on the opposite strand of the given sequence and the sign of its corresponding integer is '−'.

If users chose to show optimal scenarios before running SPRING, then they can view the optimal scenario between any pair of two input sequences just by clicking the link associated with each entry in the computed distance matrix. In the display of an optimal scenario, operations of reversals are marked with green color and those of block-interchanges with red and blue colors.

On the other hand, if the input is a set of gene/landmark orders, SPRING just outputs breakpoint and rearrangement distance matrixes along with their evolutionary trees and optimal scenarios between pairs of any two gene/landmark orders.

EXPERIMENTAL RESULTS

To validate SPRING, we have tested it with two sets of chromosomal sequences and a set of gene orders for detecting evolutionary relationships of the input species. All the tests were run using SPRING with default parameters and their detailed input data and experimental results can be accessed and referred in the help page of SPRING.

Chromosomal sequences of 11 γ-proteobacteria

Genome rearrangements by reversals have recently been studied in γ-proteobacterial complete genomes by comparing the order of a reduced set of genes on the chromosome (16). For our purpose, we selected 11 γ-proteobacterial complete sequences and tried to use SPRING to infer their phylogenetic tree by considering reversals and block-interchanges together. As a result, there are 58 identified LCBs in total and topologies of the constructed phylogenetic trees based on the breakpoint and rearrangement distance matrixes, respectively, are very similar. In fact, we calculated that the correlation coefficient between the breakpoint and rearrangement distance matrixes is 0.996, indicating high correlation between these two distances.

Chromosomal sequences of three human *Vibrio* pathogens

Vibrio vulnificus is an etiological agent for severe human infection acquired through wounds or contaminated seafood, and shares morphological and biochemical characteristics with other human *Vibrio pathogens*, including *V.cholerae* and *V.parahaemolyticus*. Currently, genomes of these three *Vibrio* species, each consisting of two circular chromosomes, have been sequenced, and it has been reported that *V.vulnificus* is closer to *V.parahaemolyticus* than to *V.cholerae* from the evolutionary point of view (7,12,17). In this experiment, we re-inferred their evolutionary relationships by applying SPRING to their complete sequences in a chromosome by chromosome manner. The adopted rearrangement operations include both reversals and block-interchanges. Consequently, *V.vulnificus* is closer to *V.parahaemolyticus* than to *V.cholerae* in the phylogenetic tree reconstructed according to the breakpoint/rearrangement distance matrix, which agrees with previous results.

Gene orders of 29 γ-proteobacteria

In this experiment, we selected 29 γ-proteobacteria from the online supplementary material provided by Belda *et al.* (16), and ran SPRING using both reversals and block-interchanges to infer their evolutionary trees according to their gene orders. As a result, the tree topology inferred by breakpoint distances is very similar to that inferred by rearrangement distances, but with two following differences. Both the *Shigella flexneri* and *Blochmannia floridanus* strains move closer to *Escherichia coli* in the rearrangement-based topology. The correlation coefficient between the breakpoint and rearrangement distance matrixes is 0.997. It is worth mentioning that in the rearrangement-based topology inferred by Belda *et al.* (16) using only reversals, the *Shigella oneidensis* strains are away from the three *Pseudomonas* species, which is contrary to our rearrangement-based topology by considering both reversals and block-interchanges.

ACKNOWLEDGEMENTS

This work was supported in part by National Science Council of Republic of China under grant NSC94-2213-E-009-141. Funding to pay the Open Access publication charges for this article was provided by National Science Council of Republic of China under grant NSC94-2627-B-007-002.

Conflict of interest statement. None declared.

REFERENCES

1. Hannenhalli,S. and Pevzner,P.A. (1999) Transforming cabbage into turnip: polynomial algorithm for sorting signed permutations by reversals. *J. ACM*, **46**, 1–27.
2. Kaplan,H., Shamir,R. and Tarjan,R.E. (1999) A faster and simpler algorithm for sorting signed permutations by reversals. *SIAM J. Comput*., **29**, 880–892.
3. Bader,D.A., Moret,B.M.W. and Yan,M. (2001) A linear-time algorithm for computing inversion distance between signed permutations with an experimental study. *J. Comput. Biol*., **8**, 483–491.
4. Bafna,V. and Pevzner,P.A. (1998) Sorting by transpositions. *SIAM J. Dis. Math*., **11**, 221–240.
5. Elias,I. and Hartman,T. (2005) A 1.375-approximation algorithm for sorting by transpositions. In *Proceedings of the 5th Workshop on Algorithms in Bioinformatics (WABI'05), LNCS 3692*, pp. 204–215.
6. Christie,D.A. (1996) Sorting by block-interchanges. *Inf. Process. Lett*., **60**, 165–169.
7. Lin,Y.C., Lu,C.L., Chang,H.Y. and Tang,C.Y. (2005) An efficient algorithm for sorting by block-interchanges and its application to the evolution of *Vibrio* species. *J. Comput. Biol*., **12**, 102–112.
8. Lin,G.H. and Xue,G. (2001) Signed genome rearrangement by reversals and transpositions: models and approximations. *Theoret. Comput. Sci*., **259**, 513–531.
9. Eriksen,E. (2002) (1+ε)-approximation of sorting by reversals and transpositions. *Theoret. Comput. Sci*., **289**, 517–529.
10. Tesler,G. (2002) GRIMM: genome rearrangements web server. *Bioinformatics*,, **18**, 492–493.
11. Darling,A.E., Mau,B., Blattner,F.R. and Perna,N.T. (2004) GRIL: genome rearrangement and inversion locator. *Bioinformatics*, **20**, 122–124.
12. Lu,C.L., Wang,T.C., Lin,Y.C. and Tang,C.Y. (2005) ROBIN: a tool for genome rearrangement of block-interchanges. *Bioinformatics*, **21**, 2780–2782.
13. Darling,A.E., Mau,B., Blattner,F.R. and Perna,N.T. (2004) Mauve: multiple alignment of conserved genomic sequence with rearrangements. *Genome Res*., **14**, 1394–1403.
14. Felsenstein,J. (2005) PHYLIP (Phylogeny Inference Package) version 3.6. Distributed by the author, Department of Genome Sciences, University of Washington, Seattle, WA.
15. Yancopoulos,S., Attie,O. and Friedberg,R. (2005) Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, **21**, 3340–3346.
16. Belda,E., Moya,A. and Silva,F.J. (2005) Genome rearrangement distances and gene order phylogeny in γ-proteobacteria. *Mol. Biol. Evol*., **22**, 1456–1467.
17. Chen,C.Y., Wu,K.M., Chang,Y.C. and Chang,C.H. (2003) Comparative genome analysis of *Vibrio vulnificus*, a marine pathogen. *Genome Res*., **13**, 2577–2587.