BIOINFORMATICS APPLICATIONS NOTE Vol. 25 no. 14 2009, pages 1833–1835
doi:10.1093/bioinformatics/btp285

Genome analysis

baobabLUNA: the solution space of sorting by reversals

Marília D. V. Braga†

Université de Lyon, F-69000, Lyon; Université Lyon 1; CNRS UMR5558; Inria Grenoble Rhône-Alpes, Lyon, France

Received on February 19, 2009; revised on April 21, 2009; accepted on April 22, 2009

Advance Access publication April 28, 2009

Associate Editor: Alex Bateman

ABSTRACT

Summary: Computing the reversal distance and searching for an optimal sequence of reversals to transform a unichromosomal genome into another are useful algorithmic tools to analyse real evolutionary scenarios. Currently, these problems can be solved by at least two available softwares, the prominent of which are GRAPPA and GRIMM. However, the number of different optimal sequences is usually huge and taking only the distance and/or one example is often insufficient to do a proper analysis. Here, we offer an alternative and present baobabLUNA, a framework that contains an algorithm to give a compact representation of the whole space of solutions for the sorting by reversals problem.

Availability and Implementation: Compiled code implemented in Java is freely available for download at http://pbil.univ-lyon1.fr/software/luna/. Documentation with methodological background, technical aspects, download and setup instructions, interface description and tutorial are available at http://pbil.univ-lyon1.fr/software/luna/doc/luna-doc.pdf.

Contact: mdvbraga@gmail.com

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

Computing the reversal distance between two unichromosomal genomes without duplications, insertions and deletions and finding one optimal sequence of reversals (that is, a sequence with a minimum number of reversals) that transforms one genome into the other can be solved in polynomial time, thanks to Hannenhalli and Pevzner (1999). These two problems have been the topic of several works, such as Tannier *et al*. (2007), and their solutions are valuable tools to analyse evolutionary scenarios. Currently, there are at least two available softwares to solve these problems. One is the package GRAPPA and the other is the software GRIMM, described respectively, in Moret *et al*. (2001) and Tesler (2002).

Nevertheless, there are many different solutions, with each solution representing an optimal sequence of reversals that sort one genome into another, and finding only one is often insufficient. Exploring the whole set of solutions is thus an interesting strategy to do a more realistic analysis. The first step in this direction was the enumeration of all solutions, thanks to an algorithm proposed by Siepel (2003). However, since the number of solutions is usually huge, the whole set is very hard to handle and this could be as useless as finding one of them. Bergeron *et al*. (2002) then proposed a model to represent the solutions in a compact way, grouping them into classes of equivalence. This allows to reduce the set to be handled and an algorithm to directly enumerate the classes was given by Braga *et al*. (2008). The number of non-equivalent solutions can be still too large, therefore, a method was proposed for filtering solutions using constraints (Braga, 2009).

In this work, we describe baobabLUNA, a framework that contains the implementation of the algorithm developed by Braga *et al*. (2008) to directly enumerate all the classes of equivalent solutions and also the further use of biological constraints to filter the classes.

2 DESCRIPTION

2.1 Permutations, reversals and sorting sequences

Genomes are represented by the list of homologous markers between them. These markers correspond to the integers $1,2,\ldots,n$, with a plus or minus sign to indicate the strand they lie on. The order and orientation of the markers of one genome in relation to the other is represented by a *signed permutation* $\pi=(\pi_1,\pi_2,\ldots,\pi_{n-1},\pi_n)$ of size n over $\{-n,\ldots,-1,1,\ldots,n\}$, such that, for each value i from 1 to n, either i or $-i$ is mandatorily represented, but not both. The *identity permutation* $(1,2,3,\ldots,n)$ is denoted by I_n.

A subset of numbers $\rho\subseteq\{1,2,\ldots,n-1,n\}$ is said to be an *interval* of a permutation π if there exist $i,j\in\{1,\ldots,n\}$, $1\leq i\leq j\leq n$, such that $\rho=\{|\pi_i|,|\pi_{i+1}|,\ldots,|\pi_{j-1}|,|\pi_j|\}$. Given a permutation π and an interval ρ of π, we can apply a *reversal* on the interval ρ of π, that is, the operation which reverses the order and flips the signs of the elements of ρ, that results in the permutation $(\pi_1,\ldots,\pi_{i-1},-\pi_j,\ldots,-\pi_i,\pi_{j+1},\ldots,\pi_n)$.

If $s=\rho_1\rho_2\ldots\rho_i$ is a *sequence of reversals* for a permutation π, we say that s *sorts* π into π_T if the result of the consecutive application of the reversals ρ_1, ρ_2, $\ldots\rho_i$ on π is π_T. The length of a shortest sequence sorting π into π_T is called the *reversal distance* of π and π_T, denoted by $d(\pi,\pi_T)$. Let $s=\rho_1\rho_2\ldots\rho_i$ be a sequence of reversals sorting π into π_T. If $d(\pi,\pi_T)=i$, then s is said to be an *optimal sorting sequence*. As an example, the sequence $\{1\}\{2\}\{4\}\{1,2,3\}$ sorts $(-3,2,1,-4)$ into I_4 and is optimal.

2.2 Main functionalities

2.2.1 Computing traces Given two permutations π and π_T, the enumeration of all solutions (sequences) that sort π into π_T can be done by iterating an algorithm given by Siepel (2003). However, the number of solutions is huge and the complexity of enumerating all of

†Present address: Universität Bielefeld, Technische Fakultät, AG Genominformatik, Postfach 10 01 31, 33501 Bielefeld, Germany

them is $O(n^{2n+3})$ (Braga, 2009). Bergeron *et al*. (2002) introduced a more compact representation of the space of solutions, grouping them into equivalence classes called *traces*. All equivalent solutions in a trace are composed by the same reversals but in different orders. Observe however that this is not the formal definition of a trace, which can be obtained in Braga (2009). Braga *et al*. (2008) later proposed an algorithm to directly give one representative solution and the number of solutions in each trace. The complexity of this algorithm is also exponential in a property of the traces called *width* (Braga, 2009), but, as the number of traces is usually much smaller than the number of solutions, enumerating traces runs considerably faster.

The framework `baobabLUNA` contains the implementation of the algorithm developed by Braga *et al*. (2008). As a simple example of the gain represented by this algorithm with respect to the enumeration of all solutions, the 28 solutions that sort $(-3,2,1,-4)$ into I_4, can be grouped in only two traces, one is represented by $\{1\}\{1,2,3\}\{2\}\{4\}$ and has 24 solutions, while the other is $\{1,2,4\}\{3\}\{1,3,4\}\{2,3,4\}$ and has 4 solutions. More details on how the algorithm generates directly the traces and also counts the number of solutions in each trace can be obtained in Braga (2009).

2.2.2 Filtering traces with constraints Biological constraints can be used to filter the traces of optimal sequences, as described in Braga (2009). Besides the two signed permutations π and π_T, this approach requires a list C of compatible constraints for selecting the sequences that sort π into π_T and respect the given constraints. Frequently, only a subset of the sorting sequences of a trace is in agreement with the constraints in C, and this subset is called C-induced subtrace. The result of applying this method is the complete set of non-empty C-induced subtraces of sequences sorting π into π_T. Generally, we have no guarantee that a sorting sequence that respects all constraints exists, thus this approach can lead to an empty result.

One of the considered constraints is the list of common intervals detected between the two initial permutations, that may correspond to the clusters of co-localized genes between the considered genomes—an optimal sequence of reversals that does not break the common intervals may be more realistic than one that does break. This approach was previously used in several studies [see for instance, Diekmann *et al*. (2007)]. We used the common intervals initially detected and also a variation of this approach, described in Braga (2009), that is the list of common intervals progressively detected when sorting one permutation into another by reversals.

Another constraint implemented in `baobabLUNA` is called *strata* and is specific to the evolution of sexual X and Y chromosomes in mammals and some other organisms. Although X and Y are usually very different, they still share an identical region (called 'pseudo-autosomal' region) at one of their extremities and are believed to have evolved from an identical pair of chromosomes. This process is at the origin of sexual differentiation: the female XX and the male XY pairs. Current theories suggest that the pseudo-autosomal region, which originally covered the whole chromosomes, was successively pruned by a few big reversals on the Y chromosome (Lahn and Page, 1999). The successive limits of the pseudo-autosomal region on the X chromosome represent the limits of what have been called the 'evolutionary strata' of X chromosome and a sequence of reversals that could have created the strata on human X chromosome is given by Ross *et al*. (2005). The use of the strata as a constraint to filter the space of solutions of the sorting by reversals problem is described in Braga *et al*. (2008) and is used by Lemaitre *et al*. (2009) to evaluate the scenario of reversals given by Ross *et al*. (2005).

Table 1. Computation results for each pair of permutations (the number of elements and reversal distance of each pair is given in the first column).

PERMUT.	Algorithm	N_S	N_T	Execution time
π_A, I_{12}	`enumSol`	8 278 540	–	$\simeq$ `13.5 min`
$n=12, d=10$	`traces`	8 278 540	2151	$\simeq$ `27 sec`
	`perfTrcs`	1 698 480	12	$\simeq$ `4 sec`
	`prgSubt`	453 600	3	$\simeq$ `2 sec`
π_B, I_{16}	`enumSol`	505 634 256	–	$\simeq$ `16 h`
$n=16, d=12$	`traces`	505 634 256	21902	$\simeq$ `7.3 min`
	`perfTrcs`	122 862 960	171	$\simeq$ `27 sec`
	`prgSubt`	5 963 760	6	$\simeq$ `14 sec`
Rfe, *R*2	`enumSol`	546 840	–	$\simeq$ `42 sec`
$n=12, d=9$	`traces`	546 840	13	$\simeq$ `3 sec`
	`prgSubt`	263 088	6	$\simeq$ `2 sec`
X, Y	`enumSol`	31 752	–	$\simeq$ `5 sec`
$n=12, d=8$	`traces`	31 752	6	$\simeq$ `1.3 sec`
	`strSubt`	420	1	$\simeq$ `0.5 sec`

The columns $\mathbf{N}_S$ and $\mathbf{N}_T$ give, respectively, the number of sorting sequences and traces computed by each algorithm. Experiments were made on a 64 bit personal computer with two 3 GHz CPUs and 2 GB of RAM.

2.3 Experiments

In order to evaluate the performance of the algorithm that computes directly the traces, named `traces`, we used the algorithm `enumSol` that enumerates all solutions. We also tested the filters `perfTrcs`, that selects traces whose solutions do not break common intervals initially detected, `prgSubt`, which selects subtraces whose solutions do not break common intervals progressively detected and `strSubt` that selects subtraces whose solutions produce a given strata in the origin permutation. The analysed permutations are $\pi_A = (-12, 11, -10, 6, 13, -5, 2, 7, 8, -9, 3, 4, 1)$ and $\pi_B = (-12, 11, -10, -1, 16, -4, -3, 15, -14, 9, -8, -7, -2, -13, 5, -6)$ (both fictitious), $Rfe = (1, 3, -2, -11, 5, -9, -10, 8, 6, -7, -4, 12)$ and $R2 = \mathrm{I}_{12}$ [the bacterium *Rickettsia felis* and its ancestor *R*2, reconstructed in Blanc *et al*. (2007)], $X = \mathrm{I}_{12}$ and $Y = (-12, 11, -2, -1, -10, -9, 8, -5, 7, 6, -4, 3)$, [human X and Y chromosomes, as the scenario proposed in Ross *et al*. (2005)]. The results are in Table 1 and show that computing traces directly indeed runs much faster than computing solutions. Moreover, the variants that take constraints in consideration usually run faster than computing all traces. Additional analyses and experimental results can be found in Braga (2009).

2.4 Download, setup and tutorial

Download and setup instructions, interface description and tutorial for computing traces (including the versions that take constraints in consideration) are available in http://pbil.univ-lyon1.fr/software/luna.

3 FINAL REMARKS

The framework `baobabLUNA` contains the implementation of a method proposed by Braga *et al*. (2008), that gives a compact

representation of the solution space of the sorting by reversals problem, grouping solutions into traces. This is an interesting alternative to most of the previous methods that give either only one or all solutions, and are provided by tools such as GRIMM (Tesler, 2002) and GRAPPA (Moret *et al*., 2001). However, although the number of traces is much smaller than the number of solutions, it may be still too big to be interpreted, and in some cases, too big to be computed. Indeed, currently we are unable to compute traces for permutations with a reversal distance of about 20 or higher.

Different biological constraints can be used to filter the traces and reduce the universe to be handled. Nevertheless, there is no guarantee that a solution that respects the given constraints exists, thus this approach may lead to empty results.

ACKNOWLEDGEMENTS

The author is grateful to Marie-France Sagot and Christian Gautier for their constructive comments and to the Pôle Bioinformatique Lyonnais (PBIL) for hosting baobabLUNA web site.

Funding: Programme Alβan (E05D053131BR); French projects ANR (REGLIS NT05-3_45205 and MIRI BLAN08-1_335497); INRIA ArcoIris (associated with the University of São Paulo, Brazil); Rhône-Alpes Bioinformatics Center (PRABI).

Conflict of Interest: none declared.

REFERENCES

Bergeron,A. *et al*. (2002) On the properties of sequences of reversals that sort a signed permutation. In *Journées Ouvertes en Biologie, Informatique et Mathématiques 2002*, Saint Malo, France, pp. 99–108.

Blanc,G. *et al*. (2007) Reductive genome evolution from the mother of Rickettsia. *PLoS Genet.*, **3**, 103–114.

Braga,M.D.V. (2009) *Exploring the Solution Space of Sorting by Reversals When Analyzing Genome Rearrangements*, PhD Thesis, Université Lyon 1, France.

Braga,M.D.V. *et al*. (2008) Exploring the solution space of sorting by reversals with experiments and an application to evolution. *Trans. Comput. Biol. Bioinform.*, **5**, 348–356.

Diekmann,Y. *et al*. (2007) Evolution under reversals: parsimony and conservation of common intervals. *Trans. Comput. Biol. Bioinform.*, **4**, 301–309.

Hannenhalli,S. and Pevzner,P. (1999) Transforming cabbage into turnip (polyn. algorithm for sorting signed permutations by reversals). *J. ACM*, **46**, 1–27.

Lahn,B.T. and Page,D.C. (1999) Four evolutionary strata on the human X chromosome. *Science*, **286**, 964–967.

Lemaitre,C. *et al*. (2009) Footprints of inversions at present and past pseudoautosomal boundaries in human sex chromosomes. *Genome Biol. Evol.* [Epub ahead of print, doi:10.1093/gbe/evp006, April 30, 2009].

Moret,B.M.E *et al*. (2001) A new implementation and detailed study of breakpoint analysis. *Proceedings of the 6th Pacific Symposium on Biocomputing*, Big Island, Hawaii, pp. 583–594.

Ross,M.T. *et al.* (2005) The DNA sequence of the human X chromosome. *Nature*, **434**, 325–337.

Siepel,A. (2003) An algorithm to enumerate sorting reversals for signed permutations. *J. Comput. Biol.*, **10**, 575–597.

Tannier,E. *et al*. (2007) Advances on sorting by reversals. *Discrete Appl. Math.*, **155**, 881–888.

Tesler,G. (2002) GRIMM: genome rearrangements web server. *Bioinformatics*, **18**, 492–493.