Rearrangement Scenarios Guided by Chromatin Structure

Sylvain Pulicani[1,4], Pijus Simonaitis[2], Eric Rivals[1,3], and Krister M. Swenson[1,3(✉)]

[1] LIRMM, CNRS – Université Montpellier, 161 Rue Ada, 34392 Montpellier, France
{pulicani,swenson}@lirmm.fr
[2] ENS Lyon, 46 Allée D'Italie, 69364 Lyon, France
[3] Institut de Biologie Computationnelle (IBC), Montpellier, France
[4] Institut de Génétique Humaine (IGH), UMR9002 CNRS-UM, Montpellier, France

Abstract. Genome architecture can be drastically modified through a succession of large-scale rearrangements. In the quest to infer these rearrangement scenarios, it is often the case that the parsimony principal alone does not impose enough constraints. In this paper we make an initial effort towards computing scenarios that respect chromosome conformation, by using Hi-C data to guide our computations. We confirm the validity of a model – along with optimization problems Minimum Local Scenario and Minimum Local Parsimonious Scenario – developed in previous work that is based on a partition into equivalence classes of the adjacencies between syntenic blocks. To accomplish this we show that the quality of a clustering of the adjacencies based on Hi-C data is directly correlated to the quality of a rearrangement scenario that we compute between *Drosophila melanogaster* and *Drosophila yakuba*. We evaluate a simple greedy strategy to choose the next rearrangement based on Hi-C, and motivate the study of the solution space of Minimum Local Parsimonious Scenario.

Keywords: Genome rearrangement · Double cut and join · DCJ · Hi-C · Chromatin conformation

1 Introduction

Genome architecture is modified on a large scale by *rearrangements*. Even somewhat distantly related species, such as human and mouse, can share almost all of the same genes yet have drastically different gene orders [6]. These differences are a result of a succession of rearrangements from an ancestral architecture, each rearrangement acting on *breakpoints* between conserved stretches of DNA. In the quest to infer accurate rearrangement *scenarios* that transform one modern-day architecture into another, it is often the case that the parsimony principal alone does not impose enough constraints [5].

One potentially useful biological factor that drives chromosome evolution is the 3-dimensional conformation of chromosomes in the nucleus; a current

© Springer International Publishing AG 2017
J. Meidanis and L. Nakhleh (Eds.): RECOMB CG 2017, LNBI 10562, pp. 141–155, 2017.
DOI: 10.1007/978-3-319-67979-2_8

hypothesis is that breakpoints which are close in 3D space are more likely to take part in a rearrangement than those which are distant. Evidence supporting this hypothesis has been reported for inter-species rearrangements [13], as well as for somatic rearrangements [3,17]. We call this the *locality hypothesis*.

Although the 3D conformation is not static, loci tend to group together into topologically associating domains (TADs). Existing techniques to get an accurate estimate of the distance between pairs of loci (like for example FISH), are not scalable to the whole genome, therefore the Hi-C method was developed as a surrogate [8,10]. Hi-C operates on several cells at once, binning genomic coordinates into windows and producing a matrix where position i, j contains the number of loci within window i that come in close physical contact with a loci from window j. Thus the *weight* of a rearrangement that acts on breakpoints in window i and j can be obtained by looking up that value in the matrix.

In previous work, we developed algorithms to use this 3D structure to guide the computation of rearrangement scenarios. We sought out *local* scenarios, relying on a partition of the adjacencies such that each class of the partition contains mutually proximal adjacencies. Rearrangements that act on breakpoints from the same class then have no cost, and are referred to as *local*, whereas those that act on breakpoints from different classes incur a cost of one, and are referred to as *non-local* [11,12]. We called these problems Minimum Local Scenario (MLS) and Minimum Local Parsimonious Scenario (MLPS). The former asks for the minimum number of non-local moves necessary when transforming one genome into the other, while the latter asks the same question for parsimonious scenarios. A partition of the adjacencies is part of the input to MLS and MLPS. It remains an open question how to create such a partition given genome adjacencies and Hi-C data.

In this paper, we explore practical aspects of finding rearrangement scenarios using *Drosophila melanogaster* and *Drosophila yakuba*. To this end, we compute a double cut and join (DCJ) scenario [2,15] that greedily chooses a next move based on the weight (Hi-C value) for its pair of breakpoints. In Sect. 2 we show that this scenario has exceptionally high weight with respect to scenarios where DCJs are sampled at random.

In Sect. 3 we address practical aspects of computing globally optimal local scenarios using MLS and MLPS. In particular, we show that MLS can be computed efficiently despite being NP-Hard (proved in [11]). We also show that a simple k-medoid clustering can be used as an informative partition of the adjacencies.

In Sect. 4.1 we evaluate how the greedy scenario from Sect. 2 performs with respect to the clusterings. Our results show that the scenario does not minimize the number of non-local moves, indicating that there is room for improvement. On the other hand, we show that a hybrid method which computes an MLPS before sampling random parsimonious DCJ moves finds higher weight scenarios than sampling alone. A hybrid method that attempts to greedily choose parsimonious local moves after computing non-local moves with MLPS does not, however, outperform the generic greedy algorithm. This indicates that further

work needs to be done to choose a set of non-local moves from the many possible MLPSs that maximize the weight with respect to the Hi-C data.

Finally, Sect. 6.1 shows that the difference between the MLPS and the MLS is generally very low. As shown in Simonaitis and Swenson [11], this indicates that an algorithm which allows for an arbitrary difference of costs between local and non-local moves is possible.

1.1 Definitions and Experimental Setup

When comparing large-scale genome architecture, *syntenic blocks* of similar sequences of genes between a group of species are first inferred using sequence similarity [7]. The *adjacencies* between these blocks are the stretches of nucleotides with no homology between the genomes, and are the potential locations for *breakpoints* that rearrangements act on. These breakpoints could be close or far in three dimensional space, as indicated by Hi-C data. We refer to this spacial proximity as the *weight* of a pair of breakpoints, and by extension as the weight of a rearrangement. The *weight of a scenario* is the mean of the weights of its rearrangements.

Our general experimental setup is the following. Genomes of *Drosophila melanogaster* and *Drosophila yakuba* are first partitioned into 64 syntenic blocks (see Sect. 5.1). The adjacencies between the blocks are the potential breakpoints. Normalized Hi-C data for *melanogaster* are used as a locality measurement (see Sect. 5.2). We construct a partition of the adjacencies by clustering around medoids [9] using Hi-C data as the similarity function (see Sect. 3.2).

To weight a rearrangement we take the intervals $(i, j), (k, l)$ corresponding to the coordinates of the breakpoints of the DCJ. Say the interval (i, j) ((k, l), respectively) spans windows i' through j' (k' through l', respectively) of the Hi-C matrix. The weight of the move then is the average matrix value over all combinations of indices $\{i', i' + 1, \ldots, j'\} \times \{k', k' + 1, \ldots, l'\}$. Intervals are updated through DCJs as described in [12].

Figure 1 shows a toy example describing our model of genome rearrangement with respect to Hi-C data. Consider a DCJ that operates on $s_1 x s_2$ and $s_3 y s_4$ where s_i is a syntenic block or telomere, and x and y are adjacencies. Then there are two different ways to make s_1 adjacent to s_3: we can have either $s_1 x s_3$ (and $s_2 y s_4$) or $s_1 y s_3$ (and $s_2 x s_4$). The two possibilities are illustrated by the first DCJ in panels (a) and (b); they invert the same blocks, yet act on the adjacencies in the two different ways.

The distinction between *cost* and *weight* is an important one in our presentation. Weight always refers to a value or average value taken directly from a Hi-C matrix. Two examples are: (1) the weight of a scenario, which is a function of the values in the Hi-C matrix, and (2) the weight of a clustering, which is a function of the Hi-C values between pairs of adjacencies in the same cluster. Cost always refers to an assessment of a scenario of rearrangements with respect to a clustering of the adjacencies. For example, the cost of a scenario can be computed with respect to a clustering (of any weight) by assigning a cost of zero to adjacencies that are in the same cluster, and a cost of one to the others.

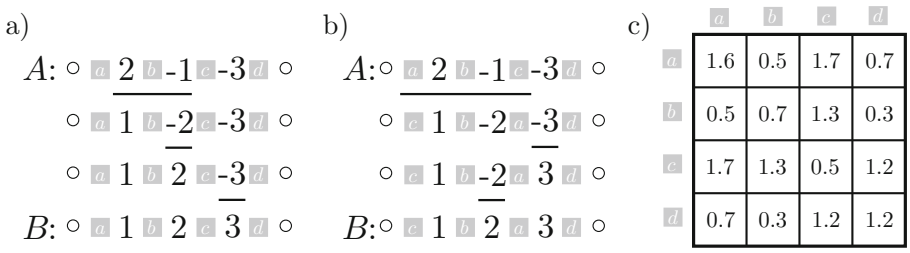


Fig. 1. Parsimonious DCJ scenarios transforming genome A into B. Syntenic blocks are represented by numbers, adjacencies by gray boxes, and telomeres (chromosome ends) by ∘. Each DCJ changes the blocks or telomeres surrounding two adjacencies. Panel (c) shows a matrix such that position i, j is the average value in the Hi-C matrix for the genomic intervals corresponding to adjacencies i and j. Panels (a) and (b) show two possible greedy parsimonious scenarios computed of the first having average weight 1.4 and the second with average weight 0.97 respectively.

2 Greedy Computation of Rearrangement Scenarios

2.1 Greedy Scenarios

In order to maximize the weight of a scenario, we designed Algorithm 1. This greedy algorithm computes a parsimonious rearrangement scenario, performing at each step the DCJ with the highest weight from all the available parsimonious DCJs.

Data:
- The genomes A and B,
- M_A, the corresponding Hi-C maps for A.

Result: L, a parsimonious scenario of DCJs transforming A into B.
$C := A$;
while C *is not equal to* B **do**
  $d :=$ a DCJ that transforms C into C' such that
   $dist(C', B) = dist(C, B) - 1$ and the weight of d in M_A is maximum;
  Apply d to C;
  Append d to L;
end

Algorithm 1. Greedy parsimonious scenario

2.2 Weight and Greedy Scenarios

We compared the weight of a greedy scenario with those of 1000 parsimonious scenarios where each DCJ is sampled uniformly at random. The results are shown in Fig. 2.

The greedy scenario has a significantly higher weight than the sampled scenarios. This is expected as the greedy scenario is a local maximum. We also

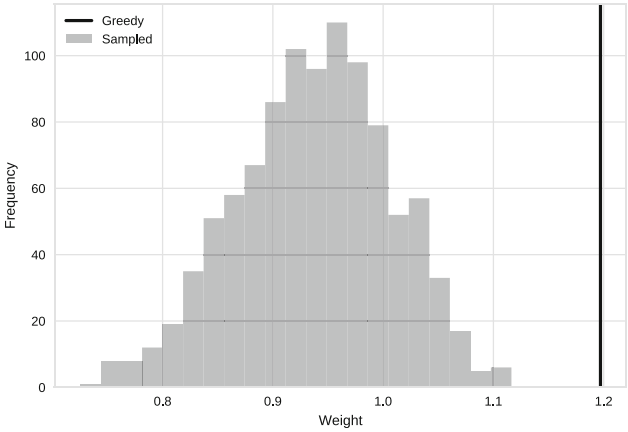


Fig. 2. Weight of the greedy scenario as a vertical bar along with the distribution of 1000 sampled parsimonious scenarios. The greedy scenario has a low probability of being sampled due to the exponential number of parsimonious scenarios (with respect to scenario length).

note that the weight along the greedy scenario decreases linearly ($r^2 = 0.92$), and that only 10 DCJs out of 51 are performed with inter-chromosomal adjacencies. Moreover, the last 8 DCJs (those with lowest weights) are done using inter-chromosomal adjacencies.

3 Global Computation of Locality

We saw in the last section that a scenario with exceptionally high weight can be obtained by greedily computing a parsimonious scenario according to Hi-C. In this section we discuss practical aspects behind the computation of scenarios that globally minimize non-local moves. First we define the optimization problems used for this task in previous work [11,12]. In Sect. 3.3 we show that the first of the two problems is tractable when computing scenarios between *melanogaster* and *yakuba*, despite being NP-hard.

These optimization problems rely on a partition of the adjacencies as input. In the final two subsections we show how we compute clusters of adjacencies to serve as these partitions, and how the quality of these clusters show a strong correlation to the number of non-local moves that must be done (the Minimum Local Scenario).

3.1 Optimization Problems

The idea behind the clustering of adjacencies is that pairs of adjacencies in the same cluster are more likely to take part as breakpoints in a rearrangement than pairs between different clusters. Thus we evaluate rearrangements by giving the

inter-cluster rearrangements a cost of 1, and the intra-cluster rearrangements a cost of 0. The *locality cost* (or just cost) of a scenario is the sum of the cost of the constituent moves. With such a model in hand we posed two optimization problems for which we developed algorithms for [11,12].

*Problem 1 (*MLS*).* Minimum Local Scenario.

INPUT: Adjacency sets A and B with a clustering of A.
OUTPUT: A scenario of rearrangements transforming A into B.
MEASURE: The locality cost of the scenario.

The problem Minimum Local Parsimonious Scenario introduces the constraint that the output is also a parsimonious scenario, *i.e.* a scenario of minimum length.

*Problem 2 (*MLPS*).* Minimum Local Parsimonious Scenario.

INPUT: Adjacency sets A and B with a clustering of A.
OUTPUT: A *parsimonious* scenario of rearrangements transforming A into B.
MEASURE: The locality cost of the scenario.

We will use the term MLS and MLPS to denote the locality cost of the scenario for an optimal solution of the problem.

3.2 Clustering

Both algorithms of the previous section require a clustering of the adjacencies for genome A. To that end, we use Hi-C data as a similarity measure for the clustering; the higher the weight for a pair of adjacencies, the more likely we are to have them in the same cluster. Clustering around medoids [9] was chosen for its simplicity and speed. A *medoid* of a cluster is an element that maximizes the sum of the similarities to the rest of a cluster. This sum is the cluster's *weight*, and when summed over all the clusters it provides us with a *weight function* for a clustering. An important property of this clustering method is that it provides many local optima that we can compare to the solutions obtained for our optimization problems (MLS and MLPS). We use three clustering algorithms: k-medoids, random, and mixed that generate k clusters for a fixed k, which in our case ranges from 1 to 70.

The k-medoids algorithm starts with randomly initialized centroids. The rest of the elements are then associated to the centroids that are most similar to them. The medoids of the obtained clusters are then computed and they become the new centroids around which the elements will be clustered. We continue this procedure until the weight of a clustering stops increasing.

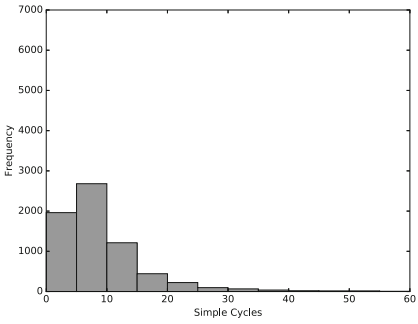
The random algorithm partitions the elements at random into k non-empty clusters. On our data we observe that the weights of the random clusters are always smaller than those provided by the clustering around medoids. In order to bridge the gap between the obtained weights we mix k-medoids and random algorithms to obtain a mixed algorithm that initializes the centroids randomly and then chooses at random the number of resulting elements to be assigned to the clusters based on the similarity function, and how many of them will be assigned at random (without performing further iterations).

3.3 Feasibility of Computing MLS

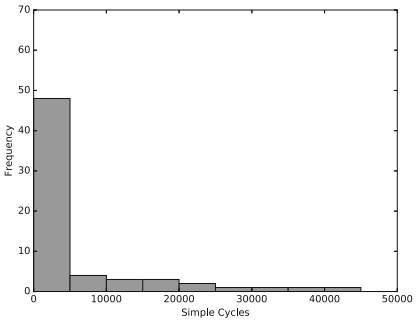
In general, finding the MLS is computationally costly, as we have proven it to be NP-hard [11]. In other words, we cannot expect to be able to compute a scenario that minimizes the number of non-local moves on any pair of genomes. We established an exact algorithm, however, that runs efficiently if a certain parameter called the number of *simple cycles* is "small enough". We expect "small enough" to roughly be in the hundreds of thousands.

Although the number of simple cycles between two genomes is hard to predict since it depends on the entire problem input (*i.e.* the syntenic blocks and the clustering), we find that between *melanogaster* and *yakuba* the number of simple cycles is always very small using the clusters computed by the K-MEDOIDS algorithm. Particularly encouraging is the fact the even completely random clustering yield a practical number of simple cycles.

In particular, we ran 100 instances of K-MEDOIDS using Hi-C data from *melanogaster* for every k ranging from 2 to 70, and computed the number of simple cycles for those clusterings. The values for all 6900 runs are presented in Fig. 3a. The average number of simple cycles is 16.9 for $k = 42$, the maximum that we observed. The average number of simple cycles over all k is 8.5. Figure 3b presents a similar histogram for the runs of RANDOM for *melanogaster*. As expected, RANDOM clusterings produce larger numbers of simple cycles.



(a) Using K-MEDOIDS clusterings. Average is 8.5, standard deviation is 8.4. The highest average is 16.9 for $k = 42$.



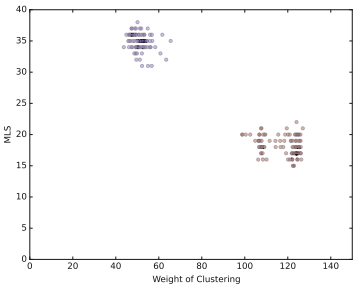
(b) Using RANDOM clusterings. Average is 9,164, standard deviation is 18,227. Four values higher than 50,000 were detected with a maximum of 86,319.

Fig. 3. The frequency of the number of simple cycles computed for all possible values of k (ranging from 2 to 70).

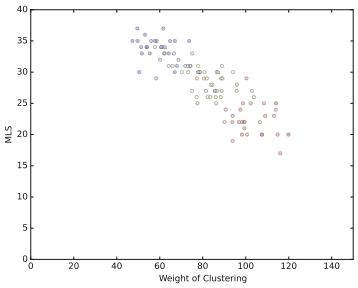
3.4 MLS and the Weight of the Clustering

Figure 4a presents 200 independent clusterings of the adjacencies of *Drosophila melanogaster* into $k = 15$ clusters. Half of them are generated using the RANDOM algorithm, the others using the K-MEDOIDS algorithm. There is a clear separation

on both axes between these two sets of clusterings. The MLS and clustering weight are always significantly better on a K-MEDOIDS clustering than a RANDOM clustering. Similar results are found for values of k ranging from 5 to 50.



(a) 100 RANDOM clusterings (blue dots) and 100 K-MEDOIDS clusterings (red dots).



(b) 100 MIXED clusterings with varying amounts of randomly assigned adjacencies. Point colors go from blue (random) to red. Pearson's correlation:
$r = -0.87$ (p-value $= 1 \times 10^{-31}$)

Fig. 4. The number of non-local moves computed for MLS compared to the weight of the clustering for $k = 15$ clusters on *D. melanogaster* Hi-C data. (Color figure online)

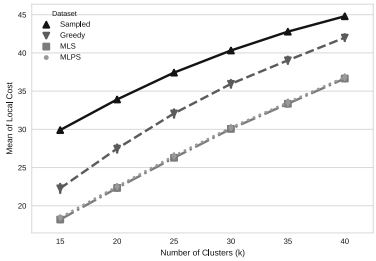
The MIXED clustering was introduced in order to bridge the gap between the weights of the clusterings provided by RANDOM and K-MEDOIDS. We note a significant inverse correlation between the weights of the clusterings and MLS cost. Figure 4b depicts 100 independent clusterings for the adjacencies of *Drosophila melanogaster* obtained using the MIXED algorithm. In this plot the color of a point indicates how many of the adjacencies in that particular clustering got assigned to the clusters at random during a run of MIXED. Blue shows that a clustering is mostly random and red, on the other hand, means that most of the adjacencies got assigned to the centroids based on the similarity function. In this example Pearson's correlation r is found to be -0.87 with a 2-tailed p-value of 1×10^{-31}. Similar results were found for values of k ranging from 5 to 50, where the correlation slightly increases up to $k = 25$ before slowly decreasing.

4 Evaluation

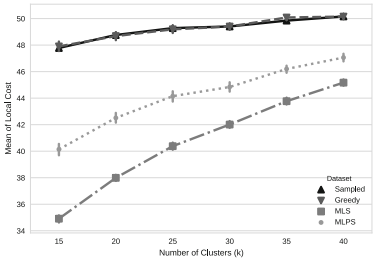
As we have seen, the MLS and MLPS are the results of a binary cost function. On the other side, the greedy scenario is a mean of Hi-C values. Therefore we cannot compare them directly. In this section, we compare the scenarios in two complementary ways. First we compute the locality cost of the greedy scenario using the clustering. Second we compute the weight of hybrid algorithms that first compute non-local moves with MLPS, and then complete the scenario using sampling or greedy methods.

4.1 Evaluating Greedy on Clusters

Using the same clusters as below, we computed the cost for the sampled scenarios and for the greedy one. The results are shown in Fig. 5, along with the costs of MLS and MLPS.



(a) Locality costs using K-MEDOIDS clusterings. The MLS and MLPS curves almost perfectly fit each other.



(b) Locality costs using RANDOM clusterings. The greedy and sampled curves almost perfectly fit each other.

Fig. 5. Locality costs for the greedy, sampled, MLS and MLPS scenarios. Since the DCJ distance is 51, the maximum possible cost is 51.

As a sanity check, note that for the same values of k, the cost using RANDOM clusterings is significantly higher than when using K-MEDOIDS clusterings. For K-MEDOIDS clustering, the cost of the greedy scenario is always lower than that of a sampled scenario. By greedily preferring the adjacencies that maximize the Hi-C weight, we give preference to the intra-cluster adjacencies. Moreover, the difference in cost between the greedy scenario and the sampled ones increases for small numbers of clusters. The costs of the MLS and MLPS are always lower than for greedy. This shows that room for improvement remains over the purely greedy algorithm.

4.2 The Weight of MLPS

A solution to MLPS does not directly give a Hi-C weight. In order to calculate it, we arbitrarily choose one of the many optimal solutions to MLPS, each giving a set of non-local moves. We then compute the local moves to build the rest of the scenario by either sampling, or by choosing moves greedily. Call these algorithms MLPS-*sampled* and MLPS-*greedy*, respectively. We compare the distributions computed using the sampling methods to the values obtained using the greedy methods in Fig. 6.

As expected, the MLPS-sampled distribution is significantly higher (*i.e.* more weight) than the sampled distribution. Moreover, there is no MLPS-sampled scenario with a particularly low weight. This indicates that the clustering captures the Hi-C information, allowing us to build biologically meaningful scenarios with MLPS. This further buttresses the link between clustering and MLPS observed in Sect. 6.1.

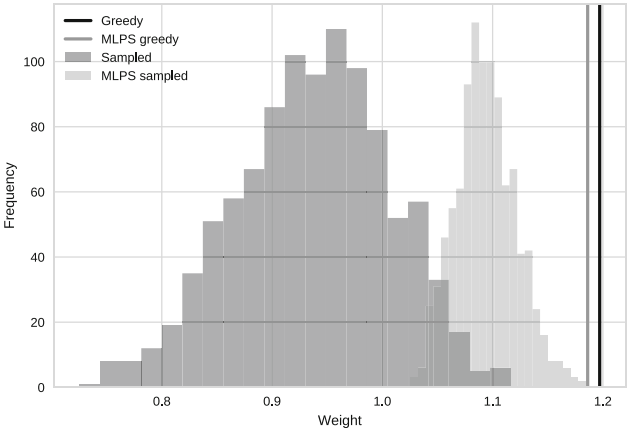


Fig. 6. Weights of the greedy and MLPS-greedy scenarios as vertical bars along with the distribution of 1000 parsimonious sampled and MLPS-sampled scenarios. The number of cluster is $k = 20$.

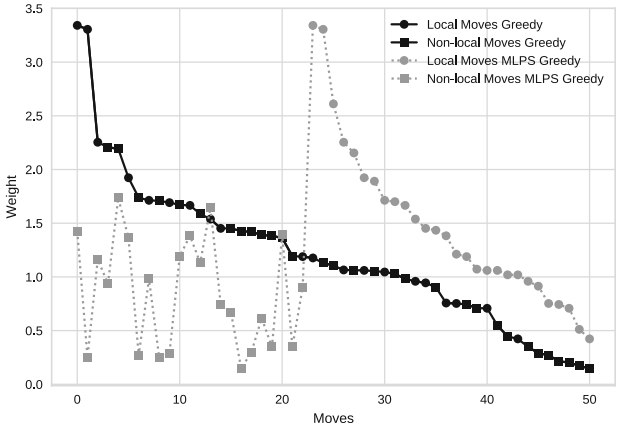


Fig. 7. Weight per move of the greedy and the MLPS-greedy scenarios.

The greedy scenario remains better than the MLPS-greedy scenario. We conjecture that this is due to the fact that we choose an arbitrary set of non-local moves that correspond to an optimal MLPS. Moreover, we plotted the weight of each move of the greedy and the MLPS-greedy scenarios in Fig. 7. The effect of choosing an arbitrary MLPS is apparent: the largest non-local moves chosen by the greedy scenarios are not chosen by the MLPS-greedy method.

We conjecture that choosing – out of all possible MLPS solutions – the solution that optimizes the weight of the non-local moves would yield a heavier MLPS-greedy solution than the purely greedy solution. Indeed, by replacing in the MLPS-greedy solution only the two lowest weighted non-local moves, with the two highest weighted non-local moves from the pure greedy scenario, the score of the new scenario would already be as good as the purely greedy algorithm.

5 Data Treatment

5.1 Creation of Syntenic Blocks

We computed syntenic blocks in two steps. First, we took the orthologs for *D. melanogaster* and *D. yakuba*. This was done using the OMA groups database [1]. We removed each gene that overlaps or intersects another, along with its ortholog in the other species. Then, the blocks were constructed using the Orthocluster tool [16]. The basic idea of Orthocluster is to aggregate orthologs to make the biggest possible blocks without breaking certain constraints that define the synteny; these constraints are the maximal and minimal number of genes per blocks, the absolute gene order between the genomes, the genes strandedness, the quantity of non-ortholog genes and the possibility to make nested blocks. We forbid the creation of nested blocks as we wanted a 1-1 block mapping. All other parameters were default.

Orthocluster outputs clusters of genes. We interpret the clusters as syntenic blocks by taking the smallest gene position in a cluster as the start position of the block from this cluster, and the largest position as the end for that block.

5.2 Hi-C as a Measure of the Locality

The Hi-C experiment provides a rough estimate of how many times a pair of genomic loci are in close proximity [8]. Roughly speaking, formaldehyde is introduced in a population of cells so that parts of the genomes that are in spacial proximity are linked together. Each side of the link contains segments of DNA that are then sequenced. The sequences are mapped to a reference genome so that the pair of spatially proximal loci are determined. Finally, the raw data are corrected for experimental biases (see [14] for the method applied to the matrices that we use, published in [10]).

The chromosomes in the nucleus have a dynamic structure. Thus, cells from the same cell type will yield different sets of pairs of loci. Therefore, since the experiment is done on a cell population, we observe the average of all these sets of loci among all the cells of the population.

Due to the nature of contact probability, which decreases dramatically with respect to chromosomal distance (it roughly follows a power law), we applied the normalization done by Lieberman-Aiden *et al.* (see the appendix of [8]) to the matrices published in [10]. This gives the long rearrangements (in the genetic coordinate sense) to have the same importance as the short ones. Specifically, a normalized intrachromosomal heatmap entry $INTRA_{ij}$ gets the value

$$INTRA_{ij} = H_{ij}/averageAtDist(|i-j|),$$

where $averageAtDist(d)$ is the expected value of an entry d off of the diagonal of any intrachromosomal matrix. A normalized inter-chromosomal heatmap entry $INTER_{ij}$ gets the value

$$INTER_{ij} = H_{ij}/\Big(\frac{interaction_i * interaction_j}{interaction_{all}}\Big),$$

where $interaction_x$ is the sum of all interactions for position x, and $interaction_{all}$ is the total sum of all entries in all matrices (intra *and* inter-chromosomal).

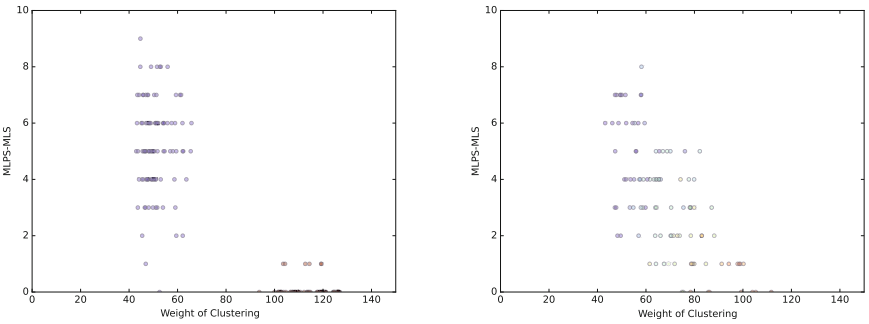
6 Future Work and Conclusions

We wrap things up by introducing future directions. The first subsection discusses the feasibility of using more general weight functions when computing globally optimal scenarios. We showed in [11] that this question is directly related to the difference MLPS − MLS. The second subsection briefly discusses the possibility of using Hi-C data from two different species.

6.1 MLPS − MLS and the Weight of the Clustering

We study the value of the difference MLPS − MLS due to its potential role in computing a more general cost function. When this difference is low the number of non-parsimonious rearrangements in the MLS is few. In this case a non-binary cost function that has an arbitrary, but fixed, difference between local and non-local move cost is easier to compute and approximate [11]. In this section we show that the difference is usually very low for *Drosophila*, and that higher quality clusters reduce this difference.

A significant correlation between the difference MLPS − MLS and the weights of the clusterings is found, as depicted in Fig. 8a and b for $k = 10$. Similar results hold for values of k from 5 to 50.

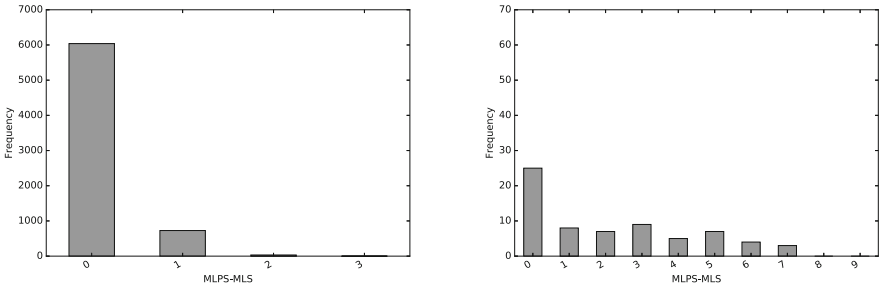


(a) 100 RANDOM (blue dots) and 100 K-MEDOIDS clusterings (red dots)

(b) 100 MIXED clusterings with varying amounts of randomly assigned adjacencies. Dots color goes from blue (random) to red. Pearson's correlation: $r = -0.69$ (p-value $= 2 \times 10^{-15}$)

Fig. 8. The difference in the number of non-local moves computed for MLPS and MLS compared to the weight of the clustering for $k = 10$ clusters on *D. melanogaster* Hi-C data. (Color figure online)

Further, for the clusterings provided by K-MEDOIDS this value of MLPS − MLS is low in general, as displayed in Fig. 9a. As for Fig. 3a we did 100 runs of K-MEDOIDS for every k, and found the average for MLPS − MLS to be highest at $k = 24$ with a value 0.29. The average over all k was 0.12. A similar histogram for the runs of RANDOM for *melanogaster* can be found in Fig. 9b.



(a) Using K-MEDOIDS clusterings. The average is 0.12, the standard deviation is 0.34. The highest average is 0.29 for $k = 24$.

(b) Using RANDOM clusterings. The average is 2.19, the standard deviation is 2.2. The highest average is 7.

Fig. 9. The frequency of the difference MLPS − MLS over all possible values of k (ranging from 2 to 70).

6.2 Using Hi-C from *D. yakuba*

No data for *D. yakuba* are publicly available at this time. However, we gained access to unpublished preliminary data. All results for greedy scenarios and clusterings on *melanogaster* also apply to *yakuba*, and tend to be more pronounced.

When computing the parsimonious scenario from species A to B we use the Hi-C data of A. In doing so, we ignore the differences in chromatin conformation between A and B that are specific to species B. Between human and mouse, for example, the conformation is in general very similar with a small number of important differences [4]. Thus, using the Hi-C data from both species is a current challenge.

As a preliminary investigation, we *reverse* the greedy scenario from A to B and use the Hi-C data of species B. In that case, the weight is closer to that of the sampled scenarios. This could be a reflection of the 3D spacial features specific to each species; by selecting adjacencies of very high weight we could select some special features in the chromatin conformation of one species over the other. If these characteristics have no correspondence in the other genome, they lead to low weight in the reversed scenario. Assuming that the differences in 3D are linked to post-speciation evolutionary events, there is the potential to use these adjacencies to place rearrangements on phylogenetic branches. We plan to further investigate this direction.

6.3 Conclusions

We have demonstrated the existence of scenarios corresponding exceptionally well to Hi-C data. They can be computed using a simple greedy strategy. In an effort to find a global optimum we had developed the MLS and MLPS methods in previous work, but it remained unclear how to use them since part of the input is a partition of adjacencies into equivalence classes. Given such a clustering of adjacencies, they give lower bounds on the number of non-local moves required in a scenario. We showed that meaningful clusters can be found even with a rudimentary clustering technique, and further, that a better clustering implies a scenario requiring fewer non-local moves. These results were based on computing MLS exactly, which was feasible since the number of simple cycles is demonstrated to be small enough between *D. melanogaster* and *D. yakuba*.

A hybrid method, that first chooses a minimum number of non-local parsimonious moves before greedily choosing high-weight moves (using Hi-C directly), did not find a better scenario (with respect to Hi-C weight) than the purely greedy strategy. There is evidence, however, of room for improvement. This motivates the study of the solution space of MLPS so that one could pick, among all optimal solutions, the one that also has the best Hi-C values. In general, the best way to balance weight and cost remains an open question.

From a practical perspective several improvements to this work are in order. First, development of the same methods on the inversion model would be appropriate for certain branches of the tree of life. In *Drosophila*, extra care would have to be taken since inversions and transpositions are the two main drivers behind architecture transformations. Second, a world of more sophisticated clustering methods exist. Application of the right method may partition the adjacencies in a more relevant way when choosing local rearrangement scenarios.

Acknowledgements. The authors would like to thank the reviewers for their helpful comments. Sylvain PULICANI is funded by NUMEV grant AAP 2014-2-028 and EPIGENMED grant ANR-10-LABX-12-01. This work is partially supported by the IBC (Institut de Biologie Computationnelle) (ANR-11-BINF-0002) and by the Labex NUMEV flaship project GEM.

References

1. Altenhoff, A.M., Škunca, N., Glover, N., Train, C.-M., Sueki, A., Piližota, I., Gori, K., Tomiczek, B., Müller, S., Redestig, H., et al.: The OMA orthology database in 2015: function predictions, better plant support, synteny view and other improvements. Nucleic acids research, p. gku1158 (2014)
2. Bergeron, A., Mixtacki, J., Stoye, J.: A unifying view of genome rearrangements. In: Bücher, P., Moret, B.M.E. (eds.) WABI 2006. LNCS, vol. 4175, pp. 163–173. Springer, Heidelberg (2006). doi:10.1007/11851561_16
3. Campbell, P.J., Stephens, P.J., Pleasance, E.D., O'Meara, S., Li, H., Santarius, T., Stebbings, L.A., Leroy, C., Edkins, S., Hardy, C., et al.: Identification of somatically acquired rearrangements in cancer using genome-wide massively parallel paired-end sequencing. Nat. Genet. **40**(6), 722–729 (2008)

4. Chambers, E.V., Bickmore, W.A., Semple, C.A.: Divergence of mammalian higher order chromatin structure is associated with developmental Loci. PLoS Comput. Biol. **9**(4), e1003017 (2013)
5. Chauve, C., Gavranovic, H., Ouangraoua, A., Tannier, E.: Yeast ancestral genome reconstructions: the possibilities of computational methods II. J. Comput. Biol. **17**(9), 1097–1112 (2010)
6. Chinwalla, A.T., Cook, L.L., Delehaunty, K.D., Fewell, G.A., Fulton, L.A., Fulton, R.S., Graves, T.A., Hillier, L.D.W., Mardis, E.R., McPherson, J.D., et al.: Initial sequencing and comparative analysis of the mouse genome. Nature **420**(6915), 520–562 (2002)
7. Ghiurcuta, C.G., Moret, B.M.E.: Evaluating synteny for improved comparative studies. Bioinformatics **30**(12), i9–i18 (2014)
8. Lieberman-Aiden, E., van Berkum, N.L., Williams, L., Imakaev, M., Ragoczy, T., Telling, A., Amit, I., Lajoie, B.R., Sabo, P.J., Dorschner, M.O., Sandstrom, R., Bernstein, B., Bender, M.A., Groudine, M., Gnirke, A., Stamatoyannopoulos, J., Mirny, L.A., Lander, E.S., Dekker, J.: Comprehensive mapping of long-range interactions reveals folding principles of the human genome. Science **326**(5950), 289–293 (2009)
9. Park, H.-S., Jun, C.-H.: A simple and fast algorithm for k-medoids clustering. Expert Syst. Appl. **36**(2, Part 2), 3336–3341 (2009)
10. Sexton, T., Yaffe, E., Kenigsberg, E., Bantignies, F., Leblanc, B., Hoichman, M., Parrinello, H., Tanay, A., Cavalli, G.: Three-dimensional folding and functional organization principles of the drosophila genome. Cell **148**(3), 458–472 (2012)
11. Simonaitis, P., Swenson, K.M.: Finding Local Genome Rearrangements. Springer, Heidelberg (2017)
12. Swenson, K.M., Simonaitis, P., Blanchette, M.: Models and algorithms for genome rearrangement with positional constraints. Algorithms Mol. Biol. **11**(1), 13 (2016)
13. Véron, A.S., Lemaitre, C., Gautier, C., Lacroix, V., Sagot, M.-F.: Close 3D proximity of evolutionary breakpoints argues for the notion of spatial synteny. BMC Genom. **12**(1), 303 (2011)
14. Yaffe, E., Tanay, A.: Probabilistic modeling of Hi-C contact maps eliminates systematic biases to characterize global chromosomal architecture. Nat. Genet. **43**(11), 1059–1065 (2011)
15. Yancopoulos, S., Attie, O., Friedberg, R.: Efficient sorting of genomic permutations by translocation, inversion and block interchange. Bioinformatics **21**(16), 3340–3346 (2005)
16. Zeng, X., Nesbitt, M.J., Pei, J., Wang, K., Vergara, I.A., Chen, N.: Orthocluster: a new tool for mining synteny blocks and applications in comparative genomics. In: Proceedings of the 11th International Conference on Extending database technology: Advances in Database Technology, pp. 656–667. ACM (2008)
17. Zhang, Y., McCord, R.P., Ho, Y.-J., Lajoie, B.R., Hildebrand, D.G., Simon, A.C., Becker, M.S., Alt, F.W., Dekker, J.: Spatial organization of the mouse genome and its role in recurrent chromosomal translocations. Cell **148**(5), 908–921 (2012)