ALGORITHMS AND METHODS FOR LARGE-SCALE GENOME REARRANGEMENTS IDENTIFICATION

Presented by
Jose Antonio Arjona Medina

Under the supervision of
Prof. Dr. Oswaldo Trelles
Algorithms and methods for large-scale genome rearrangements identification

Jose Antonio Arjona Medina
arjona@uma.es
Supervised by Dr. Oswaldo Trelles
Publications supporting the thesis

• “Computational Synten Block: A Framework to Identify Evolutionary Events”, *(IEEE Transaction in Nano Bioscience, 2015)*

• “Refining borders of genome-rearrangements including repetitions”, *(BMC Genomics, 2016)*

• “Computational workflow for the fine-grained analysis of metagenomic samples”, *(BMC Genomics, 2016)*

• “A multiple comparison framework for Synten Block detection” *(IWBBIO, 2017)*

• “Ancestral sequence reconstruction: A framework to detect Synten Blocks and revert rearrangements” *(in progress)*

arjona@uma.es
Overview

• Introduction
• Background
• Methods
• Results
• Conclusions and future work
Introduction

Synteny Blocks, Large-Scale Genome Rearrangements and Break Points

General Overview
Synteny Blocks

- The idea: Conserved blocks that share the same order and strand

Genome 0: *M. agalactiae* 5632

Genome 1: *M. bovis* PG45

High Score segments Pairs (HSPs) produced by GECKO

Synteny Blocks (SBs)
Large-Scale Genome Rearrangement

- A LSGR is an operation that changes the order or the strand of a SB

- Inversion
  - Change the strand

- Transposition
  - Change the order: moves the block to another position within the chromosome

- Duplication
  - Copy the block

- Translocation
  - Change the order: moves the block to another position in another chromosome
Break Point

• The point (or the region) in the sequence between two SBs that have suffered a LSGR

The SB in the middle has suffered a LSGR (inversion)

Dots represent BPs in the sequence
General Overview

HSPs
GECKO
(Torreño and Trelles, 2015)

Starting point

GECKO-CSB
Arjona and Trelles, 2015

SB and rearrangements pairwise detection

GECKO-Refinement
Arjona and Trelles, 2016

Refining SB borders and BPs

Rearrangements reconstruction (multi comparison)
(in progress)

GECKO-Evol
Arjona, Perez and Trelles, 2018?

GECKO-MGV
Diaz del Pino, Arjona, Torreño, Benavides and Trelles, 2016

Meta-GECKO
Perez, Arjona, Torreño, Ulzurrun and Trelles, 2016

arjona@uma.es
Objectives

• Formal definition of and detection of SBs
• Detection of LSGR and BP
• Refinement of SBs borders
• Reversion of LSGR
Background

“If I have seen further, it is by standing on the shoulders of giants”
State of the art

• **SB and BP detection**
  – No formal definition (difficult to compare methods)
  – The granularity problem
  – The BP contradiction
  – Dealing with repetitions

• **Methods to reverse LSGR**
  – Oriented to the “sorting permutation problem”
  – Reference depended
  – Not designed for dealing with repetitions
## The granularity problem

<table>
<thead>
<tr>
<th>Granularity</th>
<th>SB</th>
<th>BP</th>
<th>LSGR</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fine-grained</strong></td>
<td>Many (shorter and well conserved)</td>
<td>Many (shorter and better quality)</td>
<td>Small subset of total LSGR (short cycles)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td><strong>Coarse</strong></td>
<td>Few (larger and low percentage of identity)</td>
<td>Few (larger and noisy: Many short SB are included)</td>
<td>Small subset of total LSGR (Big picture)</td>
</tr>
</tbody>
</table>
An example
The break point contradiction

- Rearrangements do not occur randomly
- Fragile regions in the sequence, predispose to suffer a LSGR (hotspots)
  - BP should not be defined as a relation between two genomes
  - Although comparison is the only way (so far) to detect them
  - Most methods to refine SB take for granted that BPs are not conserved regions.

arjona@uma.es
Dealing with repetitions

• Driven the evolution in many ways
• Mostly associate with mobile elements
• Repetitions increase the model complexity
  – Most methods to detect SBs avoid repetitions
The sorting permutation problem

• Transform one sequence into another (the reference) through operations.

• Proven to be NP-hard
  – A reference is needed
  – No “natural” way to include repetitions in the model
  – No use of inside-block information
Methods

Pair-wise comparison method, refining blocks and multiple comparison framework: definitions and methods
Methods Overview

• 1) Pairwise SB and LSGR detection (GECKO-CSB)

• 2) SB refinement

• 3) Multi-genome SB and LSGR detection and reconstruction
1-Computational Synteny Blocks: A pair-wise framework to detect LSGR

- Set of properties to detect SBs
- Arrows represent strand
1-Computational Synteny Blocks: A pair-wise framework to detect LSGR

• These properties also describe rearrangements

```
Simple Linearity       Simple bi-Linearity
Complete Linearity     Complete bi-Linearity
Simple Colinearity    Simple bi-Colinearity
Complete Colinearity  Complete bi-Colinearity
```

```
Inversion
```
```
Transposition
```
```
Inverted Transposition
```
```
Duplication
```
2-Synteny Block refinement

• Using repetitions to refine (if any)
• Does not force the BP to be a point or region
Refining based on transitions including repeats

Illustrative representation of the Region of Interest (ROI). (a) A ROI region in an inversion event (CSB B). (b) Virtual CSBs and repetitions. (c) Same representation but including identity vectors and vector difference graphs
Finite State Machine to detect identity transitions

FSM detects the coordinates where the vector difference value was the last time at a certain threshold (U1) before reaching the second threshold (U2)
Result of the refinement

CSBs before and after the refinement. At the end of the refinement process, we detect BPs. We also extract PRASB and GAP sequences to analyse accuracy of the method. PRASB and BP have the same length.
3-Multiple comparison framework

• Motivation
  – Formal SB definition
  – Solve the BP contradiction
  – Solve the granularity problem
  – No reference-based
  – Combine sequence information and rearrangements
The Synteny Block concept

• **SB has two categories**
  
  – **Block:** The sequence
  
  – **Synteny:** The relation with other blocks
Block Element

• Subsequence in the sequence

1. $\alpha^h < \alpha^t$
2. $|\alpha| = \alpha^t - \alpha^h$
3. $|\alpha| \geq 0$ (As a consequence of 1 and 2)
Unitary Block Element

• A Block Element that does not overlap with others Unitary Block Elements
Unitary Conserved Element

• A Block Element originate from comparison
The Unitary Conserved Element problem

A) Two overlapped HSPs.
B) Result of the trimming process. Two fragments are still overlapped.
C) New overlapped Conserved Elements trigger a new trimming process.
D) Final result of the recursive trimming process.

The final pairs of Conserved Elements do not overlap.
The Unitary Conserved Element problem (II)

Representation of the trimming process in a multiple comparison.

In the comparison AB there is an inversion, that triggers a trimming process in the comparison BC.

As a result, another trimming process is triggered in comparison DC.
Unitary Synteny Element

- A set of Unitary Conserved Elements from different sequences
  - More than one block \( \pi = \{ \alpha, \alpha', \alpha'', ..., \beta, \beta', \beta'', ..., \gamma, \gamma', \gamma'', ..., \omega'' \} \)
  - Same length \( |\alpha| = |\alpha'| = |\alpha''| = ... = |\beta| = |\beta'| = |\beta''| = ... = |\omega''| \)
  - Every Unitary Conserved Block belong to one and only one Unitary Synteny Element

\[ \forall \pi_i, \pi_j \in \Pi, j \neq i : \pi_i \cap \pi_j = \emptyset \]

and

\[ \pi_1 \cup \pi_2 \cup \pi_3 \cup ... \cup \pi_{N\Pi} = \Lambda_{\Phi_A} \cup \Lambda_{\Phi_B} \cup \Lambda_{\Phi_\Gamma} \cup ... \cup \Lambda_{\Phi_\Omega} \]
Unitary Synteny Element

- Graphic representation

Fig. 3.6 Graphic representation of three Synteny Elements. Synteny Element $\pi_1$ links $\alpha_1, \beta_1$ and $\gamma_1$ Unitary Conserved Elements.
Break Point

• Defined as the region (or point) between two Unitary Conserved Elements
The transitivity property of Synteny Block: Inferred HSP

• This method does not increase the number of Unitary Conserved Blocks.
• It just reveals *synteny* relations that have not been detected by the chosen comparison method.
  
  — Hence, this supports the evidence why SBs must be defined in a N-dimensional space.

![Diagram showing transitivity property of Synteny Blocks: A-B, B-C, A-C]
Synteny Block concatenation

• If the succession is the same

\[ \Pi(\alpha_{a+i}) = \Pi(\beta_{b+i}) = \Pi(\gamma_{g+i}) = \ldots = \Pi(\omega_{o+i}) = \pi_i : i = \{-1, 0, +1\} \]

• All these Unitary Conserved Elements conform each a Unitary Synteny Element:

\[
\begin{align*}
\pi_{-1} &= \alpha_{a-1} \cup \beta_{b-1} \cup \gamma_{g-1} \cup \ldots \cup \omega_{o-1} \\
\pi &= \alpha_a \cup \beta_b \cup \gamma_g \cup \ldots \cup \omega_o \\
\pi_{+1} &= \alpha_{a+1} \cup \beta_{b+1} \cup \gamma_{g+1} \cup \ldots \cup \omega_{o+1}
\end{align*}
\]

• and the sign relation between them is the same along adjacent Elementary Conserved Blocks

\[
\begin{align*}
\text{sign}(\alpha_{a-1}, \beta_{b-1}) &= \text{sign}(\alpha_a, \beta_b) = \text{sign}(\alpha_{a+1}, \beta_{b+1}) \\
\text{sign}(\alpha_{a-1}, \gamma_{g-1}) &= \text{sign}(\alpha_a, \gamma_g) = \text{sign}(\alpha_{a+1}, \gamma_{g+1}) \\
\text{sign}(\beta_{b-1}, \gamma_{g-1}) &= \text{sign}(\beta_b, \gamma_g) = \text{sign}(\beta_{b+1}, \gamma_{g+1}) \\
\ldots \\
\text{sign}(\psi_{p-1}, \omega_{o-1}) &= \text{sign}(\psi_p, \omega_o) = \text{sign}(\psi_{p+1}, \omega_{o+1})
\end{align*}
\]
SB concatenation: Example (I)
Synteny Block concatenation

• Then, Unitary Synteny Elements $\pi - 1$, $\pi$ and $\pi + 1$ can be merged into a single one by concatenating their Unitary Conserved Elements as follows:

$$\pi_{\text{new}} = \{\alpha_{\text{new}}, \beta_{\text{new}}, \ldots, \omega_{\text{new}}\}$$

where

$$\alpha_{\text{new}} = (\alpha_{-1}^h, \alpha_{+1}^t)$$
$$\beta_{\text{new}} = (\beta_{-1}^h, \beta_{+1}^t)$$
$$\ldots$$
$$\omega_{\text{new}} = (\omega_{-1}^h, \omega_{+1}^t)$$
SB concatenation: Example (II)
Inversions

• If

\[ \Pi(\alpha_{a-1}) = \Pi(\beta_{b-1}) = \Pi(\gamma_{g-1}) = \ldots = \Pi(\omega_{o-1}) = \pi_{-1} \]
\[ \Pi(\alpha_a) = \Pi(\beta_b) = \Pi(\gamma_g) = \ldots = \Pi(\omega_o) = \pi \]
\[ \Pi(\alpha_{a+1}) = \Pi(\beta_{b+1}) = \Pi(\gamma_{g+1}) = \ldots = \Pi(\omega_{o+1}) = \pi_{+1} \]

• And

\[ \text{sign}(\alpha_{a-1}, \beta_{b-1}) = \text{sign}(\alpha_{a+1}, \beta_{b+1}) = -\text{sign}(\alpha_a, \beta_b) \]
\[ \text{sign}(\alpha_{a-1}, \gamma_{g-1}) = \text{sign}(\alpha_{a+1}, \gamma_{g+1}) = -\text{sign}(\alpha_a, \gamma_g) \]
\[ \ldots \]
\[ \text{sign}(\beta_{b-1}, \gamma_{g-1}) = \text{sign}(\beta_{b+1}, \gamma_{g+1}) = \text{sign}(\beta_b, \gamma_g) \]
\[ \ldots \]
\[ \text{sign}(\psi_{p-1}, \omega_{o-1}) = \text{sign}(\psi_{p+1}, \omega_{o+1}) = \text{sign}(\psi_p, \omega_o) \]

• Then, either \( \alpha_a \) or \( \beta_b, \gamma_g, \ldots, \omega_o \) are inversions
Detection of an Inversion: Example
Transpositions

• If

\[
\begin{align*}
\pi(\alpha_{a-1}) &= \pi(\beta_{b-1}) = \pi(\gamma_{g-1}) = \ldots = \pi(\omega_{o-1}) = \pi_{-1} \\
\pi(\alpha_a) &= \pi(\beta_{b+1}) = \pi(\gamma_{g+1}) = \ldots = \pi(\omega_{o+1}) = \pi_{+1}
\end{align*}
\]

• And

\[
\begin{align*}
\pi(\alpha_{i-1}) &= \pi(\beta_{j-1}) = \pi(\gamma_{k-1}) = \ldots = \pi(\omega_{l-1}) = \pi_{m-1} \\
\pi(\alpha_i) &= \pi(\beta_b) = \pi(\gamma_g) = \ldots = \pi(\omega_o) = \pi \\
\pi(\alpha_{i+1}) &= \pi(\beta_{j+1}) = \pi(\gamma_{k+1}) = \ldots = \pi(\omega_{l+1}) = \pi_{m+1}
\end{align*}
\]

• Then, either \(\alpha_a\) or \(\beta_b, \gamma_g, \ldots, \omega_o\) are transpositions
Detection of a Transposition: Example

![Diagram showing the detection of a transposition with G0, G1, and G2 groups, and C1 and C5 categories.](image-url)
Insertions and deletions

- When concatenating, not detected inserted blocks can be inferred if the length of the new Synteny Element is not the same.
  - A multiple alignment is needed

- An insertion can be detected as follows:

\[
\begin{align*}
\Pi(\alpha_{a-1}) &= \Pi(\beta_{b-1}) = \Pi(\gamma_{g-1}) = \ldots = \Pi(\omega_{o-1}) = \pi_{-1} \\
\Pi(\alpha_{a}) &= \Pi(\beta_{b}) = \Pi(\gamma_{g}) = \ldots = \Pi(\omega_{o}) = \pi \\
\Pi(\beta_{b+1}) &= \Pi(\gamma_{g+1}) = \pi_{in} \\
\Pi(\alpha_{a+1}) &= \Pi(\beta_{b+2}) = \Pi(\gamma_{g+2}) = \ldots = \Pi(\omega_{o+1}) = \pi_{+1}
\end{align*}
\]
Detection of an Insertion/deletion: Example
Duplications

- **If**

\[ \pi = \{ \alpha_1, \beta_2, \gamma_3, \ldots, \alpha_4 \} \]

- **And**

\[
\begin{align*}
\Pi(\alpha_{a-1}) &= \Pi(\beta_{b-1}) = \Pi(\gamma_{g-1}) = \ldots = \Pi(\omega_{o-1}) = \pi_{-1} \neq \Pi(\alpha'_{d-1}) \\
\Pi(\alpha_a) &= \Pi(\beta_b) = \Pi(\gamma_g) = \ldots = \Pi(\omega_o) = \pi = \Pi(\alpha'_d) \\
\Pi(\alpha_{a+1}) &= \Pi(\beta_{b+1}) = \Pi(\gamma_{g+1}) = \ldots = \Pi(\omega_{o+1}) = \pi_{+1} \neq \Pi(\alpha'_{d+1})
\end{align*}
\]

- **Then,** \( \alpha'_d \) is a duplication

arjona@uma.es
How to select the genome to perform the reversion?

Building a phylogenetic tree, using the block information (subsequences)
How to select the genome to perform the reversion?

Building a phylogenetic tree, using the block information (subsequences)
• 1) Pairwise SB and LSGR detection (GECKO-CSB)

• 2) SB refinement

• 3) Multi-genome SB and LSGR detection and reconstruction
Results and discussion
Experiments

• Our methods were compared with state-of-art methods, implemented by progressiveMauve, GRIMMsynteny and CASSIS.

• Data set of 68 Mycoplasmas, 2,278 pairwise genome comparisons.
Pairwise framework

- Better % coverage at all levels of similarity, especially in the less related genomes
Pairwise framework

• More coverage over both types of regions
  – For coding regions, around 90% against 75%
  – For non-coding regions 76% against 60%
Pairwise framework

- Differences of SB detection for a certain region in the genomes using Gecko-CSB and progressiveMauve methods

(a) Gecko-CSB detects one SB. (b) progressiveMauve detects three SBs (B, C and D).

(a) Gecko-CSB detects three SBs (A, B and C). (b) progressiveMauve detects one large SB.
Refining Synteny Blocks

• In a massive comparison, around 70% of the BPs detected by our method are sized below 100 bps and 95% below 300 bps.

– In a particular example of two genomes (~800Kbps) highly related, our method reports BPs sized below 100bps whereas CASSIS reports BPs sized up to 86,000 bps.
CSBs before and after the refinement. At the end of the refinement process, we detect BPs. We also extract PRASB and GAP sequences to analyse accuracy of the method. PRASB and BP have the same length.
Reconstruction of LSGR solves the granularity problem
Reconstruction of LSGR solves the granularity problem
Conclusions, contributions and future work
Advances in the State of the art

• **SB and BP detection**
  - Formal definition of SB
  - The granularity problem solved
  - The BP contradiction solved
  - Repetitions included in the model

• **Methods to reverse LSGR**
  - Combined with the SB detection
  - No Reference depended
  - Designed for dealing with repetitions
Conclusions and contributions

- More coverage
- Formal definition of SB and rearrangements
- LSGR reversion and SB concatenation as solution for the granularity problem
- Method to refine SB and BPs
Open Research Lines

• Frequencies of LSGR to improve inter-genome distances and phylogenetic organizations

• The rearrangement history reconstruction could also be helpful for ancestral genome reconstruction.

• Refined BPs can be used as input to find hidden patterns or extract features in order to set up a formal definition of BP.

• BPs may help the understanding of LSGR and the prediction of future LSGRs
Acknowledgments
Questions?