Comparing Bacterial Genomes by Searching their Common Intervals

Sébastien Angibaud, Damien Eveillard, Guillaume Fertin, Irena Rusu

Comparing Bacterial Genomes by Searching their Common Intervals

Sébastien Angibaud, Damien Eveillard, Guillaume Fertin, and Irena Rusu

Computational Biology group (ComBi)
Laboratoire d'Informatique de Nantes-Atlantique (LINA), UMR CNRS 6241,
Université de Nantes, 2 rue de la Houssinière, 44322 Nantes Cedex 3, France
tel. +33 2.51.12.58.17 fax. +33 2.51.12.58.97
sebastien.angibaud@univ-nantes.fr, damien.eveillard@univ-nantes.fr
guillaume.fertin@univ-nantes.fr, irena.rusu@univ-nantes.fr

Abstract. Comparing bacterial genomes implies the use of a dedicated measure. It relies on comparing circular genomes based on a set of conserved genes. Following this assumption, the common interval appears to be a good candidate. For evidences, we propose herein an approach to compute the common intervals between two circular genomes that takes into account duplications. Its application on a concrete case, comparing E. coli and V. cholerae, is accurate. It indeed emphasizes sets of conserved genes that present high impacts on bacterial functions.

Keywords: comparative genomic, circular genome, common interval

1 Introduction

The bacterial world is ubiquitous and shows a great diversity [21]. Interestingly, it is responsible for the major key biological processes, but it remains far from being well-understood. For a while, understanding the bacterial diversity was considered as a pointless area because of the complexity and the versatility of this bacterial world, and because of its dynamic nature. Recent efforts in high-throughput methods for analyzing bacterial genomes, confirm its dynamic but emphasize, as well, a more comprehensive picture of the structure of genomes [12]. It gives various insights for investigating the bacterial complexity [6]. In particular, Doolittle [7] suggests sets of genes as convenient descriptors for comparing two bacterial species. Following this assumption, comparing two bacterial genomes implies (i) comparing their genes for finding orthologs and (ii) finding their invariants, which indicates the genes that are conserved in both species. Furthermore, it highlights common biological properties shared by the compared bacteria.

Previous points raises several computational questions. The first one deals with the inherent complexity of comparing one gene of a bacterial genome with genes from another genome. In [4], Berglund et al. tackled this problem by using a clustering technique as implemented in the `InParanoid` software. The second one concerns the choice of an appropriate measure to compare genome structures.

Based on a correct mapping of orthologs between species, various approaches propose a theoretical framework to compute a measure between genomes based on their structure similarities [5]. They particularly highlight the sets of genes that are conserved among the genomes.

When applied on prokaryotic world and its particular features, comparing genomes implies the use of a dedicated protocol. By nature, bacterial genomes are circular, which must be taken into account by a dedicated measure. In addition, experimental investigations show evidences of high rates of gene duplications in bacteria. Based on these assumptions, this paper depicts a *in silico* protocol for the bacterial genome comparison. We propose (i) to find bacterial homologies by using the `InParanoid` software. We then have to put forward (ii) a dedicated measure. It relies on comparing circular genomes based on a set of conserved genes (following the assumption detailed in [7] and mentioned above). In this purpose, we propose an adaptation of the *common interval* [20] with a special emphasis on the circularity and duplications.

The paper is organized as follows. We briefly give notations and definitions in Section 2. We show in Section 3 how to compute the number of common intervals between two circular genomes that takes into account duplications. Based on these theoretical features, we are able to define a complete approach to compare bacterial genomes. In Section 4, we propose its application on a concrete case: comparing *Escherichia coli* and *Vibrio cholerae.* These well-known γ-Proteobacteria appear as an appropriate benchmark for testing the measure (quantitively and qualitatively). Beyond the accuracy of the comparison, the measure emphasizes sets of genes that are conserved on genomes. These genes show particular functional properties and belong to operons, which reinforces the biological relevance of the common interval measure in comparative genomics.

2 Preliminaries

The following section gives some notations and definitions used in the paper.

Notations. Let $\mathcal{F}$ be a set of *genes*, where each gene is represented by an integer. A circular genome G is represented by an *ordered* sequence of signed elements (*signed genes*) from $\mathcal{F}$, where we consider that the first gene and the last one are adjacent. Denote η_G the size of genome G. Let $G[p]$, $0 \leqslant p \leqslant \eta_G - 1$, be the signed gene that occurs at position p on a genome G. For any signed gene g, let $\overline{g}$ be the signed gene having the opposite sign. Let $occ_G(g)$, $g \in \mathcal{F}$, be the number of occurrences of the gene g in G. Given a genome G without duplicates (i.e. without signed genes having the same absolute value) and two signed genes a, b, let $G[a, b]$ be the set of unsigned genes located between genes a and b in G according to the circular order ($G[0]$ $G[1] \dots G[\eta_G - 1]$ $G[0] \dots$) on G. We also note $[a, b]_G$ the substring (i.e. the sequence of consecutive elements) of G starting by a and finishing by b.
For example, consider the set $\mathcal{F} = \{1, 2, 3, 4, 5, 6\}$ and the circular genome $G = +3 - 2 + 6 + 4 - 1 + 5$ without duplicates. Then, $G[4] = -1$ and $\overline{G[4]} = +1$, $G[-1, +3] = \{1, 3, 5\}$ and $[-1, +3]_G = -1 + 5 + 3$.

Common intervals. We define here the measure used in the paper. Let G_1, G_2 be two circular genomes without duplicates and with a similar gene content. A *common interval* [20] of (G_1, G_2) is a substring of G_1 such that G_2 contains a permutation of this substring (not taking signs into account). For example, consider $G_1 = +1+2+3+4+5$ and $G_2 = +2-4+3+5+1$. Then, substring $[+3, +5]_{G_1}$ is a common interval of (G_1, G_2). Note that the original definition of a common interval of [20] is extended here to consider circular sequences. Therefore, substring $[+5, +1]_{G_1}$ is also a common interval.

Given a set $\mathcal{I}$ of common intervals, a common interval $I \in \mathcal{I}$ is called *maximal*, if there is no common interval of $\mathcal{I}$ that strictly contains it. We call a *straight* interval, a maximal common interval for which the order and the signs of its genes are identical in G_1 and G_2. On the contrary, the maximal interval is called *reverse* if the order and the signs of genes are reversed in G_1 with respect to G_2. Finally, we call an *unstructured* interval, a maximal common interval that is neither straight nor reverse.

Genomes with duplicates. When genomes contain duplicates, we cannot directly compute the number of common intervals, because this measure is defined on permutations. A natural solution consists in i) finding a one-to-one correspondence (i.e. a matching) between signed genes of G_1 and G_2, ii) using this correspondence to rename genes of G_1 and G_2, and iii) deleting the unmatched signed genes in order to obtain two genomes G'_1 and G'_2 such that G'_2 is a *permutation* of G'_1. Computing the measure becomes thus possible. Our study proposes to focus on two matching models: the *exemplar* model [15] and the *maximum matching* model [18].

- *Exemplar model:* for each gene g, we keep in the matching only one occurrence of g in G_1 and in G_2.
- *Maximum matching model:* this model keeps the maximum number of signed genes in both genomes. In particular, we look for a one-to-one correspondence between signed genes of G_1 and G_2 that matches, for each gene g, exactly $min(occ_{G_1}(g), occ_{G_2}(g))$ occurrences.

For a given model, among all possible matchings, we look for one that optimizes the number of common intervals. However, given two genomes G_1 and G_2, the problem that consists in finding an exemplar (resp. a maximum matching) of (G_1, G_2) such that the number of common intervals is maximized, has been proved to be **APX**-Hard [1]. This complexity holds even when G_1 does not contain duplicates and each gene appears at most twice in G_2.

3 Maximizing the number of common intervals between two circular genomes

In [2], authors proposed three methods to compare two genomes that are modeled as linear sequences of genes. The first one is an exact algorithm based on transforming an optimization problem into a 0–1 linear program [16]. The second one is a heuristic based on the notion of Longest Common Substring (LCS) and the third method is an hybrid method that combines both previous approaches. An approach dedicated to the bacterial genome analysis implies to take into account the natural circularity of the input genomes. For that, we propose to show herein how to extend previous works.

3.1 Exact approach

The principle of this approach is based on (1) transforming an optimization problem into a 0–1 linear program [16] and (2) run this program on a powerful solver (e.g. `MiniSat+` [8]) in order to obtain optimal solutions. Given two circular genomes G_1 and G_2, we detail the above transformation for both *exemplar* and *maximum matching* models.

Variables. We define two types of variables: the *match variables* and the *interval variables.* One *match variable* is defined for each possible pair of signed genes that can be matched together. Any such variable will be set to 1 if the pair of corresponding signed genes is matched, and 0 otherwise. Thus, we define the set of match variables as follows:

$$X = \{x^i_j : 0 \leqslant i < \eta_{G_1} \wedge 0 \leqslant j < \eta_{G_2} \wedge |G_1[i]| = |G_2[j]|\}$$

$$\forall x^i_j \in X, x^i_j \in \{0,1\}$$

The *interval variables* correspond to the possible common intervals. Note that the whole genome G_1 is necessarily a common interval. Hence, we do not consider these intervals in order to reduce the linear program generated. These variables are defined as follows:

$$C = \{c^{i,n}_{j,m} : 0 \leqslant i < \eta_{G_1} \wedge 0 \leqslant j < \eta_{G_2} \wedge 0 \leqslant n < \eta_{G_1} - 1 \wedge 0 \leqslant m < \eta_{G_2} - 1\}$$

$$\forall c^{i,n}_{j,m} \in C, c^{i,n}_{j,m} \in \{0,1\}$$

A variable $c^{i,n}_{j,m} \in C$ corresponds to the common interval that begins at the position i on G_1 and that contains $n+1$ signed genes. Its corresponding interval on G_2 begins at the position j and contains $m+1$ signed genes.

Objective function. We want to maximize the number of common intervals between G_1 and G_2. For that, we define the objective function as the sum of the interval variables:

$$\text{maximize} \sum_{c^{i,n}_{j,m} \in C} c^{i,n}_{j,m}$$

Constraints. We define several constraints to insure that the assigned variables correspond to a valid matching, according to the considered model. For that, we first define two constraints to verify that each signed gene cannot be matched to more than one signed gene on the other genome:

$$(C1.a)\ \forall i, 0 \leqslant i < \eta_{G_1}, \sum_{\substack{0 \leqslant j < \eta_{G_2} \\ |G_1[i]|=|G_2[j]|}} x_j^i \leqslant 1$$

$$(C1.b)\ \forall i, 0 \leqslant j < \eta_{G_2}, \sum_{\substack{0 \leqslant i < \eta_{G_1} \\ |G_1[i]|=|G_2[j]|}} x_j^i \leqslant 1$$

Then, for each gene, we count the number of signed genes that are matched in order to respect the definition of the model. For the *maximum matching* model, we must have:

$$(C2)\ \forall g \in \mathcal{F}, \sum_{\substack{0 \leqslant i < \eta_{G_1} \\ |G_1[i]|=g}} \sum_{\substack{0 \leqslant j < \eta_{G_2} \\ |G_2[j]|=g}} x_j^i = \min\{occ_{G_1}(g), occ_{G_2}(g)\}$$

For the *exemplar* model, we must have:

$$(C2')\ \forall g \in \mathcal{F}, \sum_{\substack{0 \leqslant i < \eta_{G_1} \\ |G_1[i]|=g}} \sum_{\substack{0 \leqslant j < \eta_{G_2} \\ |G_2[j]|=g}} x_j^i = \min\{1, \min\{occ_{G_1}(g), occ_{G_2}(g)\}\}$$

In order to the interval variable validity, we introduce a new notation. For any $q \in \{1, 2\}$, we let $f_q(i, p) \equiv (i + p) \mod \eta_{G_q}$. This notation is necessary for taking into account the common intervals that contain both $G_q[0]$ and $G_q[\eta_{G_q} - 1]$.

First, we must make sure that each extremity of a common interval is matched with the two following constraints:

$$(C3.a)\ \forall c_{j,m}^{i,n} \in C, 2c_{j,m}^{i,n} - \sum_{\substack{0 \leqslant p \leqslant m \\ |G_1[i]|=|G_2[f_2(j,p)]|}} x_{f_2(j,p)}^{i} - \sum_{\substack{0 \leqslant p \leqslant m \\ |G_1[f_1(i,n)]|=|G_2[f_2(j,p)]|}} x_{f_2(j,p)}^{f_1(i,n)} \leqslant 0$$

$$(C3.b)\ \forall c_{j,m}^{i,n} \in C, 2c_{j,m}^{i,n} - \sum_{\substack{0 \leqslant p \leqslant n \\ |G_1[f_1(i,p)]|=|G_2[j]|}} x_{j}^{f_1(i,p)} - \sum_{\substack{0 \leqslant p \leqslant n \\ |G_1[f_1(i,p)]|=|G_2[f_2(j,m)]|}} x_{f_2(j,m)}^{f_1(i,p)} \leqslant 0$$

Then, we define constraints to insure that each signed gene of G_1 inside a common interval I is correctly matched. For that, we consider two cases. If the corresponding interval of I on G_2 does not contain both $G_2[0]$ and $G_2[\eta_{G_2} - 1]$, we write:

$$(C4.a)\ \forall c_{j,m}^{i,n} \in C, j + m < \eta_{G_2}, \forall 1 \leqslant p < n, \forall 0 \leqslant r < j,$$

$$|G_1[f_1(i,p)]| = |G_2[r]|, c_{j,m}^{i,n} + x_r^{f_1(i,p)} \leqslant 1$$

$$(C4.b)\ \forall c_{j,m}^{i,n} \in C, j + m < \eta_{G_2}, \forall 1 \leqslant p < n, \forall j + m < r \leqslant \eta_{G_2},$$

$$|G_1[f_1(i,p)]| = |G_2[r]|, c_{j,m}^{i,n} + x_r^{f_1(i,p)} \leqslant 1$$

Else, if the corresponding interval of I on G_2 contains both $G_2[0]$ and $G_2[\eta_{G_2} - 1]$, we write:

$$(C4.c)\ \forall c_{j,m}^{i,n} \in C, j + m \geqslant \eta_{G_2}, \forall 1 \leqslant p < n, \forall f_2(j,m) < r < j,$$

$$|G_1[f_1(i,p)]| = |G_2[r]|, c_{j,m}^{i,n} + x_r^{f_1(i,p)} \leqslant 1$$

Finally, we also define the three symmetric constraints of $(C4.a)$, $(C4.b)$ and $(C4.c)$ to consider each signed gene in G_2. Appendix **??** gives on overview of the whole 0–1 linear program.

3.2 Non-exact approaches

IILCS heuristic. The $IILCS$ heuristic proposed in [2] is a greedy algorithm based on the notion of Longest Common Substring (LCS). The program matches genes of an LCS of the two genomes, up to a complete reversal, and iterates this process until no gene can be matched (see [2] for more details). We easily modify this algorithm in order to compare circular genomes by identifying the LCS that may overlap the end and the beginning of the genomes.

Hybrid method. This approach uses both previous methods. First, a partial matching is obtained by running $IILCS$ until the size of any LCS is smaller than a given parameter k, chosen by the user. Then, a 0–1 linear program is generated in order to match the remaining unmatched genes. Since both previous methods have been extended for taking into account genome circularity, the hybrid method is hence adapted to circular genomes.

4 Comparing bacterial genomes: a practical application

Based on the previous theoretical framework, we are now able to propose a practical comparison of proteobacterias. Here, we first define the protocol used and, secondly, put forward a precise analysis of biological results obtained.

4.1 Protocol

The different steps of our comprehensive approach might be described as follow (see also Figure 1 for illustration).

Step 1. Input data. One selects on the `NCBI` website two genomes G_1 and G_2 and transforms the corresponding data into two files in the FASTA format, one for each genome. These files contain the list of genes, each of which is described by a label followed by its protein sequence.

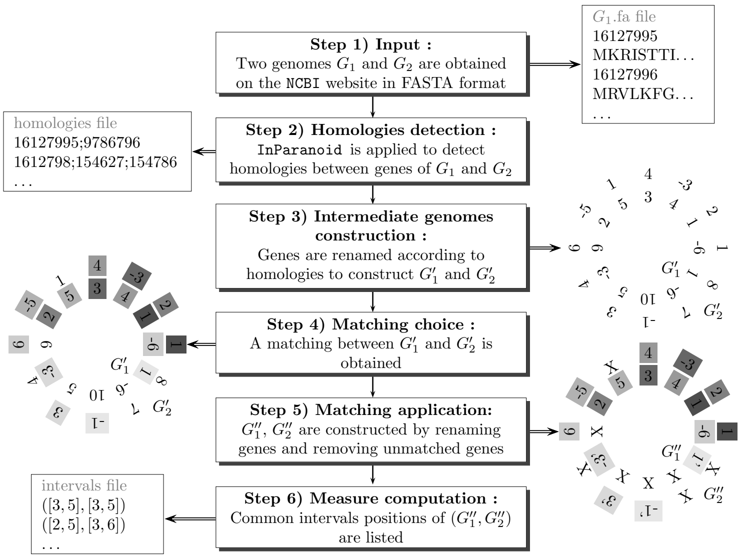

Step 1) Input :
Two genomes G_1 and G_2 are obtained on the NCBI website in FASTA format
G_1.fa file
16127995
MKRISTTI...
16127996
MRVLKFG...
...
Step 2) Homologies detection :
InParanoid is applied to detect homologies between genes of G_1 and G_2
homologies file
16127995;9786796
1612798;154627;154786
...
Step 3) Intermediate genomes construction :
Genes are renamed according to homologies to construct G'_1 and G'_2
G'_1
G'_2
Step 4) Matching choice :
A matching between G'_1 and G'_2 is obtained
Step 5) Matching application:
G''_1, G''_2 are constructed by renaming genes and removing unmatched genes
G''_1
G''_2
Step 6) Measure computation :
Common intervals positions of (G''_1, G''_2) are listed
intervals file
([3, 5], [3, 5])
([2, 5], [3, 6])
...

Fig. 1. Step by step description of the bacterial genome analysis.

Steps 2 and 3. Homologies detection. We call the InParanoid software [13], which clusters orthologs and inparalogs genes based on a Blastp (step 2). The output file contains the clusters of the homologous genes. According to these sets, we tag these homologous genes with a similar label. It hence builds two intermediate genomes G'_1 and G'_2 (step 3).

Steps 4, 5 and 6. Measure computation. We use one of the approaches described in Section 3 to obtain a matching between the two genomes (step 4). In this purpose, two parameters must be specified:

- *The model:* exemplar or maximum matching model (see Section 2).
- *The method:* the exact method based on a pseudo-boolean programming (PSB), the $IILCS$ heuristic or the hybrid method with a parameter k bounding the size of the LCS (see Section 3).

We generate the list of gene pairs that are matched. Thus, we rename the genes according to this matching and remove the unmatched genes to construct two genomes G''_1 and G''_2 without duplications and with the same gene content (step 5). We then compute the common intervals of (G''_1,G''_2) (step 6).

Filtering relevant common intervals. Among the set $\mathcal{I}$ of common intervals we have obtained, we select the relevant common intervals as follows:

1. We remove common intervals that contain all matched signed genes (the whole genome being a trivial common interval).

2. Since genomes are circular, for each common interval $I \in \mathcal{I}$, there exists a unique common interval $I' \in \mathcal{I}$, such that I' contains all matched signed genes that do not belong to I. The pair (I, I') will be called a *complementary* pair. Given any complementary pair (I, I'), we assume that the smallest interval $I_s \in \{I, I'\}$ (i.e., the one that contains less signed genes) is the most biologically informative, and thus we choose to keep only I_s in our set.
3. However, an interval that contains only one gene carries no biological information. We thus remove such intervals from our set.

4.2 Practical application

We apply our method on a benchmark composed of proteobacteria's genomes (*Escherichia coli* and *Vibrio cholerae*). Table 1 lists the considered set of genomes and Table 2 shows quantitative details concerning common intervals under the maximum matching model. Note that the computational time of `InParanoid` is much longer than the one needed to obtain the matching. However, for the longest sequences (*NC*_000913 vs *NC*_002505 or *NC*_000913 vs *NC*_009457), only the *IILCS* heuristic gives results in an acceptable amount of time. Note also the distribution of straight, reverse and unstructured intervals (see Table 2). On average, 41% of maximal common intervals are straight, 49% are reverse and 10% are unstructured.

NCBI label	Name
NC_000913	*Escherichia coli* K12
NC_002505	*Vibrio cholerae* 01 biovar eltor str. N16961 chromosome I
NC_002506	*Vibrio cholerae* 01 biovar eltor str. N16961 chromosome II
NC_009456	*Vibrio cholerae* 0395 chromosome I
NC_009457	*Vibrio cholerae* 0395 chromosome II

Table 1. Set of genomes analyzed.

These quantitative results are computationnaly relevant. We therefore propose to go further by investigating their qualitative properties. In particular, we focus on the comparison of the chromosome pair composed by *Escherichia coli* (NC_000913) and *Vibrio cholerae* (NC_009457), for which the quantitative results are given in Table 2. We consider their comparison as an appropriate benchmark for testing the biological properties highlighted by the common interval measure. As shown in Figure 2, it emphasizes three kinds of common intervals:

Straight intervals. They show a perfectly conserved gene arrangements like in Figure 2 a). Based on experimental knowledges refered in EcoCyc[11], *CcmA*, *CcmB*, *CcmC*, *CcmD*, *CcmE*, *CcmF*, *CcmG* and *CcmH* are the genes that encode for proteins that are sub-units of the cytochrome C complex. These genes

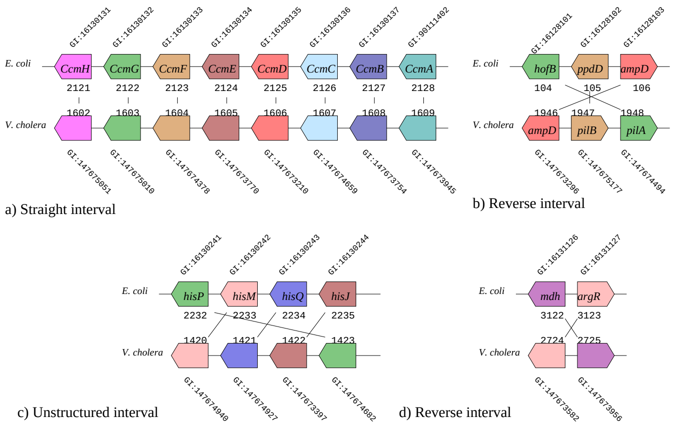
	genome size			computational time		relevant common intervals cardinality				
genome G_2	*E. coli*	G_2	method	Inparanoid (s)	matching (s)	number	maximal	straight	reverse	unstructured
NC_002505	4243	2742	$IILCS$	1144	62	3710	275	117(43%)	134(48%)	24(9%)
NC_002506	4243	1093	PSB	638	23	123	50	18(36%)	23(46%)	9(18%)
NC_009456	4243	1133	PSB	651	22	132	55	17(31%)	27(49%)	11(20%)
NC_009457	4243	2742	$IILCS$	1199	59	3602	278	114(41%)	141(51%)	23(8%)

Table 2. Relevant common intervals obtained under the maximum matching model by comparing *Escherichia coli* (NCBI label NC_000913) with four *Vibrio cholerae* chromosomes (NC_002505, NC_002506, NC_009456 and NC_009457).

belong to the same operon promoted by ccmAp[17]. In particular, experimental studies show that mutants of one of these genes are deficient in the ability to produce c-type cytochromes [19]. Their presence into a unique straight common interval, emphasizes the fact that these genes are conserved by their DNA sequence (i.e. homology shown using `InParanoid`), but also by their arrangement on the bacterial genomes (i.e. determination of the common interval). This result confirms the interest to find highly functionnal genes into a single common interval.

Reverse intervals. They show sets of genes that presents a conservation between two genomes in a reverse order, like the one in Figure 2 b). This particular interval depicts the pilus gene clusters that encode for extracellular pilus structures. They are common among bacteria and have been involved in several colonization functions, like those involved in the intestinal mucosa colonization. For clinical motivations, the homology of these genes is already shown between V. cholerae and Vibrio fisheri [14]. In [9], Fullner and Mekalanos depict the organization of the pilus assembly operon with 7 genes. The reverse common interval shown in Figure 2 b) suggests a conservation of 3 genes only, which is not fully accurate with the experimental assumption. Nevertheless, note that EcoCyc [10] not shows neither high-quality evidences of a common transcriptionnal unit for the seven genes, which not invalidates our method.

Unstructured intervals. They represent common intervals which are neither straight nor reverse. Figure 2 c) illustrates this case. Like in a), this particular interval also describes the arrangement of four genes that belong to the same operon: *hisJp* operon. It is activated by Hns and repressed by ArgR. It products a subunit of an ABC Transporter. Note again that *ArgR* that encodes ArgR belongs, with *mdh*, to a single reverse common interval (see Figure 2 d)). *mdh* produces Mdh that is a subunit of a malate dehydrogenase. It interacts in several



a) Straight interval

b) Reverse interval

c) Unstructured interval

d) Reverse interval

Fig. 2. Illustration of the three kinds of common intervals observed between E. coli (NC_000913) and V. cholerae (NC_009457). Each representation indicates the gene indexes (e.g., GI16130131), the genes position (e.g., 2121), the available labels of genes (e.g., $CcmH$), the gene orientation. Homologies are summarized using colors.

biological processes (carbohydrate metabolism, gluconeogenesis, glycolysis, tricarboxylic acid cycle, tricarboxylic acid cycle intermediate metabolism, malate metabolism, fermentation, anaerobic respiration, glyoxylate catabolism, carbohydrate catabolism). Again, these results confirm that common intervals might be associated with operons, but moreover, like in this case, with a set of genes that controls operons. Assuming that a common interval emphasizes a biological function conservation over compared genomes, regulatory function appears as important as other metabolic functions. Like in this case, common interval shows a specific regulatory unit that controls distinct biological processes via distinct operons. It is particularly interesting for studying genomes and their comparisons from a functionnal viewpoint.

A close look at these qualitative results highlights that common intervals (either straight, reverse or unstructured) may indicate the presence of functional components within bacterial genomes. Such an information is particularly valuable for investigating the bacterial functional diversity, in particular in an environmental context for which the genomic data remain the corner stone for a better understanding.

5 Extension of the method to conserved intervals

Apart of the common intervals, Bergeron and Stoye introduced in [3] the notion of *conserved intervals*. Given two circular genomes G_1 and G_2 with same gene content and without duplication, consider two signed genes a and b of G_1, with possibly $a = b$. Substring $[a, b]_{G_1}$ is called a *conserved interval* of (G_1, G_2) if $[a, b]_{G_1}$ is a common interval and if it satisfies one of the two following properties: either a and b appear in G_2 and $G_1[a, b] = G_2[a, b]$; or $\overline{a}$ and $\overline{b}$ appear in G_2 and $G_1[a, b] = G_2[\overline{b}, \overline{a}]$. For example, if $G_1 = +1 + 2 + 3 + 4 + 5$ and $G_2 = -5 - 4 - 2 + 3 - 1$, substrings $[+4, +5]_{G_1}$ and $[+4, +1]_{G_1}$ are two conserved intervals of (G_1, G_2).

Our previous approach (see Section 3) can be adapted to compute the conserved intervals instead of the common intervals. Concerning the exact approach, since a conserved interval is also a common interval with restrictions on its extremities, an additional filter on interval variables suffices to consider the number of conserved intervals. In that case, we change the set C of variables such that

$$\begin{gathered} C = \{c^{i,n}_{j,m} : 0 \leqslant i < \eta_{G_1} \wedge 0 \leqslant j < \eta_{G_2} \wedge 0 \leqslant n < \eta_{G_1} - 1 \wedge 0 \leqslant m < \eta_{G_2} - 1\} \wedge \\ (\ (\ G_1[i] = G_2[j] \wedge G_1[f_1(i, n)] = G_2[f_2(j, m)]\ ) \vee \\ (\ G_1[i] = \overline{G_2[f_2(j, m)]} \wedge G_1[f_1(i, n)] = \overline{G_2[j]}\ )\ ) \end{gathered}$$

For $IILCS$ and hence for the hybrid method, the process remains the same in the greedy phase. However, the intervals computation is modified to count only conserved intervals, which does not affect the complexity of the algorithm.

6 Conclusion

In this paper, we have presented a comprehensive method to find sets of genes that are conserved between two circular genomes. We present for that an approach to compute or approximate the number of common intervals between two circular genomes. Each resulting interval can be considered as a set of genes that is conserved between the two genomes during the evolution process. Our method was tested on *Escherichia coli* and four chromosomes of *Vibrio cholerae*. From these experimentations, the results strongly suggest that common intervals is a measure that provide useful information on bacterial genomes and help the user to focus on specific sets of genes that possess functionnal and regulatory properties. It confirms the interest of the common interval measure for giving more functionnal insights in comparative genomics studies. A thorough biological analysis of each maximum common interval, along with tests on larger benchmarks, are planned in the very next future.

Acknowledgment: the authors thank Dr. Richard A. Long for the fruitful discussions that initiated this work.

References

1. S. Angibaud, G. Fertin, and I. Rusu. On the approximability of comparing genomes with duplicates. In *Proc. 2nd International Workshop on Algorithms and Computation (WALCOM 2008)*, volume 4921 of *LNCS*, pages 34–45. Springer, 2008.
2. S. Angibaud, G. Fertin, I. Rusu, and S. Vialette. A pseudo-boolean general framework for computing rearrangement distances between genomes with duplicates. *Journal of Computational Biology*, 14(4):379–393, 2007.

3. A. Bergeron and J. Stoye. On the similarity of sets of permutations and its applications to genome comparison. In *Proc. 9th International Computing and Combinatorics Conference (COCOON 2003)*, volume 2697 of *LNCS*, pages 68–79, 2003.
4. A. Berglund, E. Sjölund, G. Ostlund, and E. L. L. Sonnhammer. Inparanoid 6: eukaryotic ortholog clusters with inparalogs. *Nucleic Acids Res*, 36(Database issue):D263–6, 2008.
5. G. Blin, C. Chauve, G. Fertin, R. Rizzi, and S. Vialette. Comparing genomes with duplications: A computational complexity point of view. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 4(4):523–534, 2007.
6. T. P. Curtis and W. T. Sloan. Prokaryotic diversity and its limits: microbial community structure in nature and implications for microbial ecology. *Curr Opin Microbiol*, 7(3):221–6, 2004.
7. W. F. Doolittle. Phylogenetic classification and the universal tree. *Science*, 284(5423):2124–9, 1999.
8. N. Eén and N. Sörensson. Translating pseudo-boolean constraints into SAT. *Journal on Satisfiability, Boolean Modeling and Computation*, 2:1–26, 2006.
9. K. J. Fullner and J. J. Mekalanos. Genetic characterization of a new type IV-a pilus gene cluster found in both classical and El Tor biotypes of *Vibrio cholerae*. *Infect Immun*, 67(3):1393–404, 1999.
10. P. D. Karp, I. M. Keseler, A. Shearer, Ma. Latendresse, M. Krummenacker *et al.* Multidimensional annotation of the *Escherichia coli* K-12 genome. *Nucleic Acids Res*, 35(22):7577–90, 2007.
11. I. M. Keseler, J. Collado-Vides, S. Gama-Castro, J. Ingraham, S. Paley *et al.* Ecocyc: a comprehensive database resource for *Escherichia coli.* *Nucleic Acids Res*, 33(Database issue):D334–7, 2005.
12. H. Ochman and L. M. Davalos. The nature and dynamics of bacterial genomes. *Science*, 311(5768):1730–3, 2006.
13. M. Remm, C.E. Strom, and E.L. Sonnhammer. Automatic clustering of orthologs and in-paralogs from pairwise species comparisons. *J. Molecular Biology*, 314:1041–1052, 2001.
14. E. G. Ruby, M. Urbanowski, J. Campbell, A. Dunn, M. Faini *et al.* Complete genome sequence of *Vibrio fischeri*: a symbiotic bacterium with pathogenic congeners. *Proc Natl Acad Sci USA*, 102(8):3004–9, 2005.
15. D. Sankoff. Genome rearrangement with gene families. *Bioinformatics*, 15(11):909–917, 1999.
16. A. Schrijver. *Theory of Linear and Integer Programming.* John Wiley and Sons, 1998.
17. S. Tanapongpipat, E. Reid, J. A. Cole, and H. Crooke. Transcriptional control and essential roles of the *Escherichia coli* ccm gene products in formate-dependent nitrite reduction and cytochrome C synthesis. *Biochem J*, 334 (Pt 2):355–65, 1998.
18. J. Tang and B. M. E. Moret. Phylogenetic reconstruction from gene-rearrangement data with unequal gene content. In *Proc. Workshop on Software Architectures for Dependable Systems*, volume 2748 of *LNCS*, pages 37–46. Springer, 2003.
19. L. Thöny-Meyer. Haem-polypeptide interactions during cytochrome C maturation. *Biochim Biophys Acta*, 1459(2-3):316–24, 2000.
20. T. Uno and M. Yagiura. Fast algorithms to enumerate all common intervals of two permutations. *Algorithmica*, 26(2):290–309, 2000.
21. W. B. Whitman, D. C. Coleman, and W. J. Wiebe. Prokaryotes: the unseen majority. *Proc Natl Acad Sci USA*, 95(12):6578–83, 1998.