Fast identification of gene clusters in prokaryotic genomes

Karine St-Onge(2), Anne Bergeron(1,3), Cedric Chauve(1,2,3)
(1) Département d'Informatique, (2) LaCIM, (3) CGL
Université du Québec à Montréal

Algorithms and computational methods for biochemical and evolution networks, CompBioNets 2005, to appear in 2005.

Fast identification of gene clusters in prokaryotic genomes

Karine St-Onge[2], Anne Bergeron[1,3] and Cedric Chauve[1,2,3]

[1] Département d'Informatique, UQÀM (Université du Québec à Montréal, Canada).
[2] LaCIM (Laboratoire de Combinatoire et d'Informatique Mathématique), UQÀM.
[3] CGL (Comparative Genomics Laboratory), UQÀM.
Email: st-onge@lacim.uqam.ca,anne@math.uqam.ca,chauve@lacim.uqam.ca

Abstract. The detection of gene clusters that are conserved in several genomes, in terms of gene proximity and gene content, have proved to be an invaluable tool in the comparative analysis of prokaryotic genomes. It has applications, for example, in predicting functional association between groups of genes or putative genome rearrangements. We propose an efficient algorithm for computing gene clusters, based on the hypothesis of the prevalence of short reversals in the evolution of prokaryotic genomes.

1 Introduction

In this paper, we are interested in the detection of groups of chromosomal segments with identical or almost identical gene content, called *gene clusters*. In many cases, this conservation of gene content is probably due to a selection pressure that tends to preserve the very proximity of the genes [11]. As a consequence, the detection, across several genomes, of such clusters considerably helps the prediction of features of interest such as operons [7], physical interactions of proteins [2], functional annotation of genes [11] or identification of biological networks [21]. It can also be useful in detecting orthologous genes [1], and for phylogenetic reconstruction, through the identification of some of the rearrangements events that can occur during the evolution [6; 15; 20].

A first approach to the detection of gene clusters starts from a mathematically precise definition of gene cluster, and then proposes an algorithm detecting the groups of chromosomal segments that meet the criterions of this definition. In this category, one can cite the notion of gene teams, introduced by Nicolas *et al.* [10] and developed by several authors, including He and Goldwasser [5], and Pasek *et al.* [12]. However, there are some drawbacks with these methods. For some of them, the definition of gene clusters is too restrictive and does not capture essential characteristics of biological gene clusters: gene teams do not model data with duplicated genes for example [10]. A contrario, when the definition is flexible enough, the computation of gene clusters can become hard from the computational point of view, which can limit the comparison to only a few genomes: the domain teams of Pasek *et al.* [12] can require a computation time exponential in the size of the considered genomes, and they can not be used for data sets containing a lot of genomes, while the homology teams of He and Goldwasser [5] are defined for only two genomes.

A second approach attacks gene clusters detection on more pragmatic grounds, primarily based on the detection of gene segments that are conserved among several of the considered genomes (see [14] for a recent survey on this topic). This collection of conserved segments is then processed, in general using an heuristic, to obtain a set of gene clusters. The main problem with this approach is the lack of a formal definition of the computed clusters, that can lead to noisy results. Moreover, most of the algorithms developed under this model base their search of gene clusters on the detection of conserved pairs of adjacent genes, which is computationally efficient but makes it more difficult to detect clusters that have been rearranged. As far as we know, the only two algorithms using this second approach and that can detect rearranged clusters are due to Fujibuchi *et al.* [4] and Rogozin *et al.* [13], but both these methods have computational shortcomings: the method of Rogozin *et al.* [13] is based on the NP-hard problem of computing maximal paths in a graph, while Fujibuchi *et al.* [4] use a very time consuming P-quasi grouping clustering algorithm.

Note that the recent tool GECKO [19] uses both approaches: it defines primarily gene clusters as common intervals of sequences [18], but it introduces a fast post-processing phase of the set of common intervals that

compensates in some way for the weakness, in terms of flexibility, of the definition of clusters as common intervals.

In the present work, we follow the second approach, and we propose a time and space efficient algorithm based on the local conservation of gene segments. The main qualities of our algorithm are its ability to process data sets containing many genomes, as it has a time and space complexity that is quadratic in the size of the input, and its flexibility, as it can detect clusters that are present only in a few genomes and whose gene order have been rearranged. We present our algorithm in Section 2, and an application to the analysis of 12 genomes of γ-Proteobacteriae, with a focus on the tryptophan operon, in Section 3.

2 Computing gene clusters

2.1 Description of the algorithm

Our algorithm runs through three phases: identification of short conserved gene segments, segmentation of the genomes, and clustering of the genome segments, into gene clusters, based on their similarity in gene content. The algorithm is parameterized by three integers δ, ω and ρ.

The input of the algorithm is made of k *signed sequences*, denoted by $G_1, \ldots, G_k$, representing k genomes[1]. The value $G_i[j]$ is a signed integer that represents the j^{th} gene of the i^{th} genome G_i: the absolute value $|G_i[j]|$ of $G_i[j]$ describes the gene family this gene belongs to, and its sign is the orientation of this gene. We denote by $n_1, \ldots, n_k$ the number of genes respectively of $G_1, \ldots, G_k$, with $n = n_1 + \ldots + n_k$. We denote by f the total number of gene families that appear in the genomes.

Given three integers i, j_1, j_2, the *gene segment* $G_i[j_1..j_2]$ is the segment of G_i composed of the consecutive genes $G_i[j_1], G_i[j_1+1], \ldots, G_i[j_2]$. The length of a gene segment is the number of genes it contains, that is $j_2 - j_1 + 1$.

A *gene cluster* is a set of gene segments. The output of our algorithm is a list of gene clusters. We now describe the three phases of our algorithm.

Phase 1. Identification of short conserved gene segments. Given the parameter δ, a positive integer, a δ-segment is a gene segment of length at most $\delta + 1$. A δ-segment $G_{i_1}[j_1..j_2]$ is said to be *conserved* if there exists another δ-segment $G_{i_2}[k_1..k_2]$ that does not overlap it (either $i_1 \neq i_2$ or $k_1 > j_2$ or $j_1 > k_2$) and such that either $|G_{i_1}[j_1]| = |G_{i_2}[k_1]|$ and $|G_{i_1}[j_2]| = |G_{i_2}[k_2]|$, or $|G_{i_1}[j_1]| = |G_{i_2}[k_2]|$ and $|G_{i_1}[j_2]| = |G_{i_2}[k_1]|$.

The first phase of the algorithm computes the list $\mathcal{G}$ of all conserved δ-segments, which can easily be done in $O(n\delta \log(f))$ worst-case time and $O(f \log(f))$ space.

Phase 2. Segmentation of genomes based on supported points. A *point* in a genome is the region located between two consecutive genes. We denote by $P_i[k, k+1]$ the point located between the genes in positions k and $k+1$ of the genome G_i. Given the set $\mathcal{G}$ of all conserved δ-segments, the *support* of $P_i[k, k+1]$ is the number of these segments that contain both $G_i[k]$ and $G_i[k+1]$. Next, given an integer parameter ω, with $1 \leqslant \omega \leqslant 2\delta - 1$, a segment $G_i[j_1..j_2]$ is said to be ω-supported if all the points it contains have support at least ω while both surrounding points $P_i[j_1 - 1, j_1]$ and $P_i[j_2, j_2 + 1]$, if they exist, have support lower than ω.

The output of this second phase of the algorithm is the list of all the ω-supported segments, based on the conserved δ-segments computed in Phase 1. Note that these segments are two-by-two disjoint. Given $\mathcal{G}$, computing the list of all ω-supported segments can easily be done in time $O(\delta n)$ and space $O(n)$. Figure 1 illustrates the concepts of conserved δ-segments and ω-supported segments.

Phase 3. Clustering of ω-supported segments in gene clusters. In this third phase, the set of ω-supported segments computed during Phase 2 is clustered into disjoint subsets, each of these subsets representing a gene cluster. This clustering is defined in terms of a graph whose vertices are the ω-supported segments and edges are the pairs of segments $(G_{i_1}[j_1..j_2], G_{i_2}[k_1..k_2])$ such that at least ρ per cent of the genes of the first

[1] In the case of a genome with more than one chromosome, we consider that these chromosomes have been concatenated.

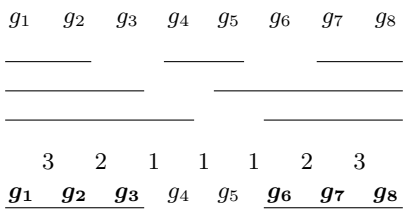

Fig. 1. This small genome has eight genes, and seven short conserved segments are underlined. Short conserved segments can overlap. The supports of the points range from 1 to 3, and there are two 2-supported segments: the segment from g_1 to g_3, and the segment from g_6 to g_8. These segments do not overlap.

segment belong to families that have occurrences in the second segment and vice-versa. The final list of gene clusters is then simply the list of the connected components of this graph. Computing this list only asks for all pairwise comparisons between ω-supported segments, which can be done in $O(n^2)$ and produces a graph with $O(n)$ vertices and $O(n^2)$ edges. Computing the connected components of this graph requires an $O(n^2)$ worst-case time.

Hence, all together, the three phases of our algorithm have a worst-case time complexity that is $O(n^2 + n\delta \log(f))$ and requires a space in $O(n^2 + f \log(f))$. Since in typical applications, δ has a low value (between 2 and 4 for example) and can be considered constant, and since f is in $O(n)$, the practical complexity is quadratic in n.

2.2 Discussion

Phase 1. Short conserved segments. The general idea of our algorithm is based on some recent observations by Sankoff *et al.* [16; 17] that small reversals seem to be prevalent in the evolution of prokaryotic genomes. This observation implies that close species should share many conserved δ-segments for short values of δ, even if, due to genome rearrangements, insertions or deletions of genes, some pairs of consecutive genes are not conserved. The goal of Phase 1 of our algorithm is to detect these segments. Note however that in most other works following the same approach of detecting conserved gene segments, the length of these segments is limited to two genes – this corresponds to $\delta = 0$ in our framework –, which makes them very sensitive to rearrangements, insertions and deletions. The only other models relaxing this constraint of adjacency are due to Fujibuchi *et al.* [4], and Rogozin *et al.* [13], where the notion of *conserved gene pair* can be seen as a restriction of our notion of conserved δ-segments with $\delta = 3$.

Phase 2. Support of points. The second phase, in which the parameter ω is central, aims at detecting conserved segments longer than δ-segments. Note that the parameter ω and the notion of support of a point is new, as far as we know, in the problem of computing gene clusters. Together with the parameter ρ, it gives a useful way to adjust to the quality of conservation one wants to detect. For detecting clusters of highly conserved segments, both in terms of contents and order, one will use high values for these two parameters, while, by lowering the value of ω, one will relax the gene order conservation.

However, as the example of the tryptophan operon described in Section 3 will show, our definition of support can hide some parts of interesting clusters, if the data set contains genomes that are phylogenetically close. Indeed this proximity implies the presence of long, strongly supported, and very similar, in terms of gene order, segments that will then belong to the same cluster. And in the case when just a small part of these segments is conserved in other genomes, this information can then be shadowed by these long strongly supported segments. It would then greatly improve the quality of the computed clusters if the support of points could take into account the phylogenetic distance between genomes: points that are only supported by the similarity of close genomes should be valued less that points supported by distant genomes which highlights the functional pressure on such points. The modification of our algorithm in this direction would also increase its time and space complexities, as it implies, for each conserved δ-segment, to remember

in which genomes it appears. However, using classical algorithmical technics and tools, like bit-vectors for example, should allow to maintain a good practical efficiency and we are currently updating our algorithm in this way.

Phase 3. Clustering The third phase was designed in order to be efficient from a computational point of view. Indeed, in the other works that aim at detecting gene clusters that may have been rearranged, this phase is time-consuming: Fujibuchi *et al.* [4] base their approach on a quartic time approximation algorithm for the quasi P-grouping clustering problem, that has been proved to be NP-hard [9], while Rogozin *et al.* [13] have to compute longest paths in a graph, another NP-hard problem. Our approach is very efficient in terms of computing time, but using a single-linkage clustering based on similarity of the gene content can produce heterogeneous clusters due to chain effects. Nevertheless, preliminary experiments showed that post-processing such clusters by removing weakly connected vertices – an approach that can be seen as an approximation of the P-quasi grouping used in [4] – is still very fast and reduces these clusters in much more homogeneous clusters.

Vizualisation and editing. Finally, due to the fact that our approach does not rely on a formal definition of a gene clusters, the computed clusters often need to be manually adjusted. For example, it can happen that the extremities of some segments computed with a low value of ω should not be part of a cluster. It can also happen that a cluster should be split or that two clusters should be joined. It is also frequent that, given a cluster, a core of gene families that define this cluster appear clearly. It then would be useful to "project" these families on the genomes, as it is done in [13]. This would lead to the detection of segments that should be part of a given cluster but are not detected by our current algorithm, due to a high value of ω or of ρ, or due to the fact that a segment has been extensively rearranged.

3 Application to γ-proteobacterial genomes

We now describe the application of our algorithm to a data set composed of 12 genomes of γ-Proteobacteriae, and we focus on the results for the well known tryptophan operon. All the files and results of our analysis are available on the website `http://adn.bioinfo.uqam.ca/~genoc/CLUSTERS`.

3.1 Data set, gene families and gene clusters

We downloaded from the NCBI Microbial Complete Genomes database, the genomes of the following organisms: *Escherichia coli K12* (*E. coli*), *Salmonella typhimurium* (*S. typhimurium*), *Yersinia pestis CO92* (*Y. pestis CO92*), *Yersinia pestis KIM* (*Y. pestis KIM*), *Buchnera aphidicola* (*B. aphidicola*), *Wigglesworthia glossinidia* (*W. glossinidia*), *Haemophilus influenzae* (*H. influenzae*), *Pasteurella multocida* (*P. multocida*), *Pseudomonas aeruginosa* (*P. aeruginosa*), *Xylella fastidiosa* (*X. fastidiosa*), *Xanthomonas axonopodis* (*X. axonopodis*), *Xanthomonas campestris* (*X. campestris*). This data set is interesting, as it contains very distant and different genomes and at the same time, three pairs of close genomes: (*E. coli* and *S. typhimurium*, the two *Yersina pestis*, and the two *Xanthomonas*).

We computed the gene families by considering all coding genes, tRNA genes and rRNA genes. Families for tRNA and rRNA are based on the genes annotations. Coding genes families are based on amino acid sequences similarity, using pairwise sequences comparisons with BLAST and single-linkage clustering with the following parameters: two amino-acids sequences are linked if the best BLAST alignment overlaps at least 70% of the length of both sequences and has a similarity of at least 30%, on the aligned segments, for both sequences. The total number of genes in the 12 genomes is 39050. The gene families computation process resulted in 11771 gene families. Of these, 6221 families contains only one gene, and thus are present in only one genome, and one contains 434 genes, composed almost exclusively of genes coding for ATP-binding proteins. A total of 6538 families are present in only one genome, while 227 are present in at least 10 genomes, and 284 families are present in all 12 genomes.

We then computed gene clusters for two data sets: the full data set of 12 genomes, and a subset of 9 genomes obtained by removing the genomes of *S. typhimurium*, *Y. pestis KIM* and *X. campestris*, to avoid

pairs of very close genomes. For both data sets, we computed gene clusters with all combinations of the three parameters δ, ω and ρ, with $\delta = 2, 3$, $\omega = 1, \ldots, 2\delta - 1$ and $\rho = 50, 60, 70, 80, 90, 100$, which yields 48 different combinations.

For the 12 genomes, the computation was completed[2] in approximatively 10 minutes , while for the 9 genomes, it took 4 minutes. This difference is a consequence of the absence of pairs of very similar genomes that reduces the number of ω-supported segments and hence the time required for the clustering phase. In terms of space complexity, the two possible bottlenecks are the computation of conserved δ-segments, that can require a space in $O(f \log(f))$, and the encoding of the edges of the graph whose vertices are ω-supported segments and components define gene clusters, since there can be $O(n^2)$ edges. But in the light of our experiments, one can assume that the space complexity, in practice, is linear in the number of genes in the data set.

The number of clusters for the data set of 12 genomes goes from 199 clusters ($\delta = 3$, $\omega = 2$ and $\rho = 100$) to 930 clusters ($\delta = 2$, $\omega = 2$ and $\rho = 60$). For the 9 genomes, the smallest number of clusters is 249 ($\delta = 3$, $\omega = 5$ and $\rho = 100$) and the largest is 913 ($\delta = 2$, $\omega = 1$ and $\rho = 60$). The complete list of clusters is available on the companion website.

3.2 The tryptophan operon

In order to get a first idea on the quality of the computed clusters, we now look our algorithm is able to capture the structure of the tryptophan (*trp*) operon. In our data set, this operon is composed of the genes *trpA*, *trpB*, *trpC*, *trpD* and *trpE* [22]. Note that none of the genes of this operon is present in *W. glossinidia*. The computation of gene families gives the following result[3]:

- the *trpA* genes form the family 1, and the *trpB* genes the family 2,
- the *trpC* genes are split in two families, labeled 3 (*E. coli*, *S. typhimurium*, the two *Y. pestis*, *B. aphidicola*, *H. influenzae* and *P. multocida*) and 7 (*P. aeruginosa*, *X. fastidiosa*, *X. axonopodis* and *X. campestris*),
- the *trpD* genes are split in two families, labeled 4 (all organisms but *E. coli*, *S. typhimurium* and *Y. pestis KIM*), and 5 (*E. coli* and *S. typhimurium*); no gene named *trpD* appears in *Y. pestis KIM*,
- the *trpE* genes form the family labeled 6.

We then computed gene clusters on the two data sets, with parameters $\delta = 2$, $\rho = 50$ and $\omega = 1, 2, 3$. We considered all possible values of ω since one of our goal in this work is to study the usefulness of the notion of support in the detection of interesting gene clusters. We filtered each list of gene clusters so obtained to keep only the clusters containing at least one gene belonging to the families 1, 2, 3, 4, 5, 6 or 7. For the data set composed of 12 genomes, we obtained 11 clusters for $\omega = 1$, and 8 clusters for both $\omega = 2$ and $\omega = 3$. The most remarkable clusters are obtained with $\omega = 2$ and $\omega = 1$ and are shown in Figures 3 and 4.

One can notice that the cluster of Figure 3 does not capture the gene *trpE* in position 277 of *Y. pestis KIM*, due to the fact that the segment *trpC trpE* in this genome is not conserved with $\delta = 2$. Indeed, in all other genomes, these two genes, if they appear, are part of segments containing 3 points. Three segments that contains only the gene *trpE* are missing in this cluster, which is normal since every conserved segment contains at least 2 genes. The segment 6 (98) (3477) 4 3 of *H. influenzae* is missing due to the insertion of the gene (3477) that causes the point 6 (98) to have a support of only 1. However, with $\omega = 1$, this segment appears in the cluster displayed in Figure 4. This example illustrates the usefulness to consider several level of support and the need for a method that could join clusters computed with different values of ω. Among the segments of this operon that do not appear in the clusters of Figures 3 and 4, the two segments of *E. coli* and *S. typhimurium* and the four segments of *X. axonopodis* and *X. campestris* appear in other clusters but as small subsegments of large segments that are common either to *E. coli* and *S. typhimurium*, or *X. axonopodis* and *X. campestris*. This illustrates the phenomenon discussed in Subsection 2.2.

[2] On a Intel Dual Xeon 2.8 GHz/512K bi-processor SUN server.

[3] For clarity reasons, we change here the numbers associated to the gene families corresponding to genes belonging to the *trp* operon: 1 corresponds to 260, 2 replaces 261, 3 replaces 262, 4 replaces 263, 5 replaces 1428, 6 replaces 1429 and 7 replaces 3944. Others gene families numbers were not modified.

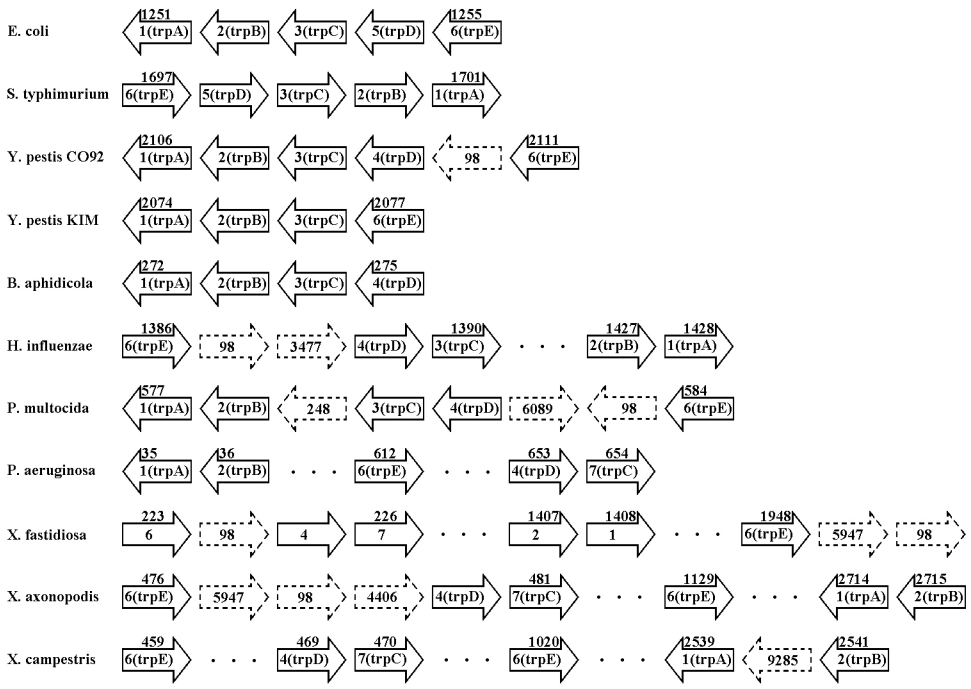

Fig. 2. The *trp* operon in the data set of 12 proteobacterial genomes. The numbers above the genes indicate the position of the first and last gene of each segment. Dashed genes indicate genes not belonging to the tryptophan pathway.

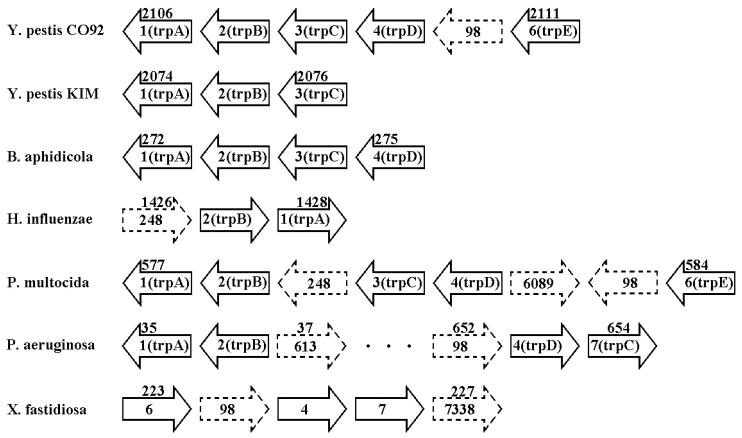

Fig. 3. A cluster obtained with the 12 genomes, $\delta = 2$, $\omega = 2$ and $\rho = 50$.

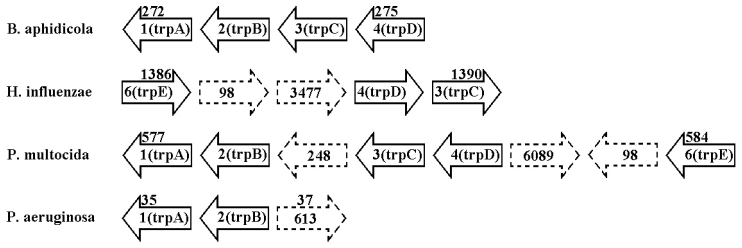

Fig. 4. A cluster obtained with the 12 genomes, $\delta = 2$, $\omega = 1$ and $\rho = 50$.

For the data set composed of 9 genomes, we obtained 6 clusters for $\omega = 1$, 3 clusters for $\omega = 2$ and 2 clusters for $\omega = 3$. We show in Figures 5 and 6 two interesting clusters that illustrates the influence of the inclusion of similar genomes in the data set, since they contain short segments that contains all genes of the *trp* operon.

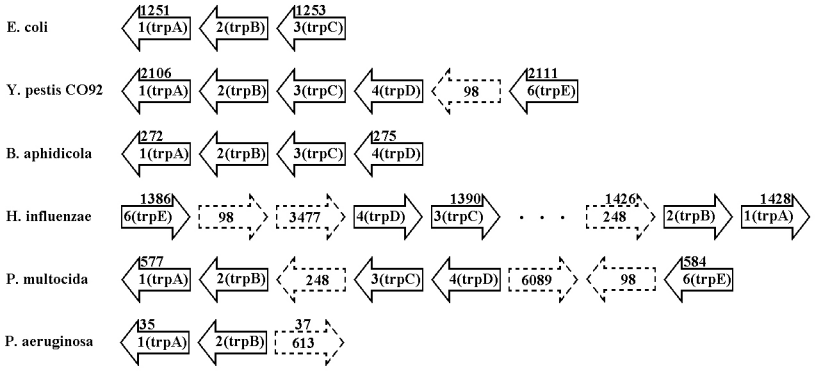

Fig. 5. A cluster obtained with the 9 genomes, $\delta = 2$, $\omega = 1$ and $\rho = 50$.

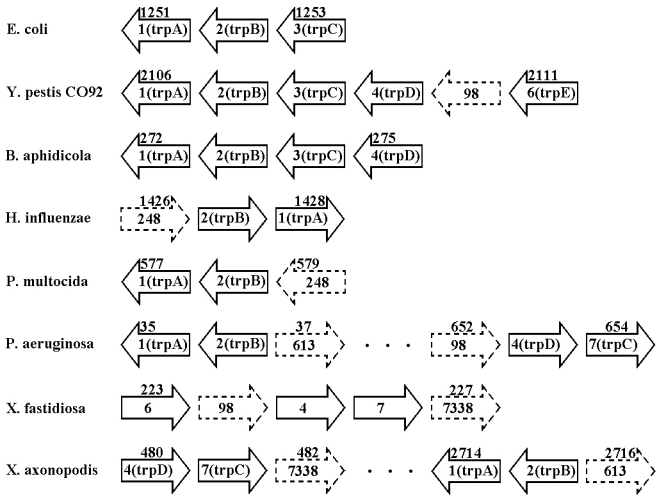

Fig. 6. A cluster obtained with the 9 genomes, $\delta = 2$, $\omega = 2$ and $\rho = 50$.

4 Conclusion

We presented in this work a novel algorithm for the computation of gene clusters based on the conservation of short gene segments, the notion of support of a point between adjacent genes and simple strategies for clustering gene segments. The main qualities of this algorithm are its flexibility and its computational efficiency that allows to use it with several combination of parameters. Preliminary experiments on a data set of 12 bacterial genomes showed promising results and illustrated the usefulness to be able to use several values of the parameters δ, ω and ρ. An other interesting consequence of the efficiency of this algorithm is that it leaves the door open for improving the computed clusters by post-processing in order to compensate the chaining effect associated to the single linkage clustering, or merging clusters obtained with different values of the parameters.

We think that among the important ways to extend our algorithm is the need to develop more precise notions of support, in order to relate the support of a point to the phylogenetic distance of the δ-conserved segments that contribute to this support. A probabilistic approach of this question has recently be proposed in [23].

The other challenge that has to be tackled with programs computing gene clusters relies in the mining of the computed clusters to find interesting ones, as generally these programs. Several possibilities exist, like computing a statistical significance of clusters [3], but more sophisticated methods need to be developed, especially methods taking into account the phylogenetic pattern of the considered genomes.

Acknowledgments. K. St-Onge is supported by a FQRNT scholarship. A. Bergeron is supported by an NSERC grant. C. Chauve is supported by NSERC and FQRNT grants. We thank the participants to the GénomeQuébec project "Comparative and integrative bioinformatics" for interesting feedback, and Jens Stoye and Thomas Schmidt for useful discussions and providing their paper [19].

References

1. X. Chen and J. Zheng and Z. Fu and P. Nan and Y. Zhong and S. Lonardi and T. Jiang. Assignment of orthologous genes via genome rearrangement. To appear in *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2005.
2. T. Dandekar, B. Snel, M. Huynen and P. Bork. Conservation of gene order: a fingerprint for proteins that physically interact. *Trends Biochem. Sci.* 23:324–328, 1998.
3. D. Durand and D. Sankoff. Tests for gene clustering. *J. Comput. Biol.* 10:453–482, 2003.
4. W. Fujibuchi, H. Ogata, H. Matsuda and M. Kanehisa. Automatic detection of conserved gene clusters in multiple genomes by graph comparison and P-quasi grouping. *Nucleic Acids Res.* 28:4029–4036, 2000.
5. X. He and M. Goldwasser. Identifying conserved gene clusters in the presence of homology families. To appear in *J. Comput. Biol.*, 2005.
6. J.O. Korbel, B. Snel, M.A. Huynen and P. Bork. SHOT: a web server for the construction of genome phylogenies. *Trends Genetics.* 18:158–162, 2002.
7. W.C. Lathe III, B. Snel and P. Bork. Gene context conservation of a higher order than operons. *Trends Biochem. Sci.* 25:474–479, 2000.
8. E. Lerat, V. Daubin, and N.A. Moran. From gene tree to organismal phylogeny in prokaryotes: the case of γ-Proteobacteria. *PLoS Biology*, 1:101–109, 2003.
9. H. Matsuda, T. Ishihara and A. Hashimoto. Classifying molecular sequences using a linkage graph with their pairwise similarities. *Theoretical Comput. Sci.* 210:305–325, 1999.
10. L. Nicolas, J.-L. Risler, A. Bergeron and M. Raffinot. Gene teams: a new formalization of gene clusters for comparative genomics. *Comput. Biol. Chem.* 27:59–67, 2003.
11. R. Overbeek, M. Fonstein, M. D'Souza, G.D. Pusch and N. Maltsev. The use of gene clusters to infer functional coupling. *Proc. Natl. Acad. Sci.* 96:2896–2901, 1999.
12. S. Pasek, A. Bergeron, J.L. Risler, A. Louis, E. Ollivier and M. Raffinot. Identificatrion of genomic features using microsyntenies of domains: Domain teams. *Genome Res.* 15:867–874, 2005.
13. I.B. Rogozin, K.S. Makarova, J. Murvai, E. Czabarka, Y.I. Wolf, R.L. Tatusov, L.A. Szekely and E.V. Koonin. Connected gene neighborhoods in prokaryotic genomes. *Nucleic Acids Res.* 30:2212–2223, 2002.
14. I.B. Rogozin, K.S. Makarova, Y.I. Wolf and E.V. Koonin. Computational approaches for the analysis of gene neighborhoods in prokaryotic genomes. *Brief Bioinform.* 5:131–149, 2004.
15. D. Sankoff. Rearrangements and chromosomal evolution. *Curr. Opin. Gen. Dev.* 13:583–587, 2003.
16. D. Sankoff. Short inversions and conserved gene clusters. *Bioinformatics* 18:1305–1308, 2002.
17. D. Sankoff, J.-F. Lefebvre, E. Tillier, A. Maler and N. El-Mabrouk. The distribution of inversion lengths in prokaryotes. In *RCG 2004*, vol. 3388 of *Lecture Notes in Bioinformatics*, pp. 97–108, Springer (Berlin), 2005.
18. T. Schmidt and J. Stoye. Quadratic time algorithms for finding common intervals in two and more sequences. In *CPM 2004*, vol. 3109 of *Lecture Notes in Comput. Sci.*, pp. 97–108, Springer (Berlin), 2004.
19. T. Schmidt, C. Rückert, J. Kalinowski and J. Stoye. GECKO: a tool for efficient gene cluster detection in prokaryotic genomes. Submitted, 2005.
20. M. Suyama and P. Bork. Evolution of prokaryotic gene order: genome rearrangements in closely related species. *Trends Genet.* 17:10–13, 2001.

21. C. von Mering, L.J. Jensen, B. Snel, S.D. Hooper, M. Krupp, M. Foglierini, N. Jouffre, M.A. Huynen and P. Bork. Known and predicted protein-protein associations, integrated and transferred across organisms. *Nucleic Acids Res.* 33:D433–437, 2005.
22. G. Xie, N.O. Keyhani, C.A. Bonner and R.A. Jensen. Ancient origin of the tryptophan operon and the dynamics of evolutionary change. *Microbiol. Mol. Biol. Rev.* 67:303–342, 2003.
23. Y. Zheng, B.P. Anton, R.J. Roberts and S. Kasif. Phylogenetic detection of conserved gene clusters in microbial genomes. *BMC Bioinformatics* 6:243, 2005.