Math. Modelling and Sci. Computing, Vol. 9, No.1 pp. 1-23, 1998 **ISSN 1067-0688/98 $7.00 + 0.00**
Principia Scientia, Printed in USA

AUTOMATED PAIR-WISE COMPARISONS OF MICROBIAL GENOMES

Arvind K. Bansal[1], Peer Bork[2], and Peter J. Stuckey[3]

[1]Department of Mathematics and Computer Science, Kent State University,
Kent, OH 44242, USA, Phone: +1.330.672-4004 ext. 214 E-mail: arvind@mcs.kent.edu
[2]Computer Informatics Division,, European Molecular Biology Laboratory, Meyerhofstr 1, Postfach 10 22 09, 69012 Heidelberg, Germany, Phone: +49.6221.387.534 E-mail: Bork@EMBL-Heidelberg.de
[3]Department of Computer Science, University of Melbourne, Parkville,
Victoria 3052, Australia, Phone +61.3.9287.9155 E-mail: pjs@cs.mu.oz.AU

ABSTRACT. In this paper, we describe a framework for an automated pair-wise comparison of complete microbial[2] genomes to derive putative orthologous genes — functionally equivalent counterparts of genes in different genomes, corresponding gene-groups — clusters of neighboring genes which have some natural pressure to occur in close proximity, fused genes — separate adjacent genes in one genome which join together to form a single gene in another genome, gene duplication, and duplicated corresponding gene-groups. The framework has three stages: BLAST (Basic Local Alignment Search Tool) — a popular sequence search technique — comparison to filter gene-pairs with high similarity, alignment of filtered gene-pairs using the Smith-Waterman alignment — a pair-wise sequence alignment based on dynamic programming technique, and bipartite graph matching and fuzzy logic techniques to identify orthologues and corresponding gene-groups. The identification of orthologues is based upon modeling gene matchings from two genomes as bipartite graph matching. Two different approaches and the corresponding algorithms to identify orthologues have been described. Both the approaches give very similar results. Five microbial genome pairs — partial *E. coli* (1304 genes)[3] versus *H. influenzae*, *H. influenzae* versus partial *B. subtilis* (1852 genes), partial *B. subtilis* (1852 genes) versus *M. genitalium*, partial *B. subtilis* (1852 genes) versus *M. jannaschii*, and *M. genitalium* versus *M. jannaschii* — have been compared. A comparative analysis of results related to putative orthologues, corresponding gene-groups, gene duplication, fused genes, gene shuffling — a gene correspondence such that one of the matching genes occurs outside a corresponding gene group — and gene conservation in various genomes is presented.
KEYWORDS. algorithms, archaea, bacteria, bipartite graph, computational biology, computer software, cross-species comparison, gene-groups, genome sequence analysis, graph matching, homologues, microbe, operons, orthologues, prokaryotes

[1]Supported in part by Research Council grant, Kent State University, Kent, Ohio, USA and Deutsche Forschungsgemeinschaft – The German Federal Agency – grant during his sabbatical in 1995 and 1996
[2]The automated comparison can handle both bacterial genomes and archaeal genomes, and for convenience we will refer to both as microbes.
[3]1304 genes of *E. coli* have been chosen for historical reason as the first experimental data available at EMBL during Fall 1995.

1. INTRODUCTION

Microbes (bacteria and archaea) serve as model organisms for understanding basic metabolic functions. Microbes are also important targets in biotechnology, disease treatment, and ecology, etc. The first complete genome of a cellular organism was sequenced in 1995 [7]. Since then, many others have been completed. Some of the completed ones are *E. coli*, *M. genitalium* [8], and *M. jannaschii* [4]. Seven bacterial and three archaeal genomes will have been reported by the end of 1997.

A natural step in understanding microbial genomes is to chart out the functionality and the variations in the functionality of genes and families of genes. Due to the size of genomes and increasing rate of availability of complete sequences, a cost-effective and time-efficient way is to perform a fully-automated pair-wise cross-species computational comparison of genome sequences.

A pair-wise cross-species computational comparison of complete genomes will help in the identification of gene functions, function and evolution of gene families, gene conservation, variations of functionality within various organisms [12], and mechanisms of evolution. The information about clustered gene-groups will help in identifying higher order functions, and help in relating gene-groups to metabolic pathways [17]. In addition, the identification of conserved genes will help in refining the phylogenetic tree — a hypothetical evolution tree — [13] further.

In order to compare two genomes, exact functional counterparts of genes in genomes have to be identified and compared. *Homologues* are genes derived from some common ancestral gene. *Paralogues* (para = in parallel) are homologous genes comprising a multigene family (as a result of gene duplication) with possible variations in functionality. Due to the presence of many multigene families in genomes which are homologues, one has to identify the exact functional counterpart of a gene in another species out of a multigene family. These functional counterparts are called *orthologues* (ortho = exact) that have arisen from speciation [6]. The *orthologues* can be the only basis of gene comparison since the history of orthologous genes represents the history of species. However, orthologues have to be carefully identified. Current similarity search techniques can identify homologues. Orthologues and paralogues are homologues, thus current similarity search techniques are unable to discriminate between the two groups. In this paper, we are mainly concerned with the identification of *putative orthologues* and the *corresponding gene-groups* — clusters of neighboring genes which have some natural pressure to occur in proximity — in genome-pairs comprising of putative orthologues. However, our technique also derives and separates duplicated genes and duplicated gene-groups from putative orthologues.

In this paper, we describe two techniques and the corresponding algorithms to derive putative orthologues and corresponding gene-groups. There are three phases in the framework: the BLAST [3] comparison phase, the Smith-Waterman alignment [19,20] phase, and a bipartite graph matching phase. The BLAST comparison phase provides a time-efficient mechanism to identify possible gene-pair matchings and prune out dissimilar gene-pair correspondences. The Smith-Waterman alignment phase aligns the filtered gene-pairs to identify the regions of similarity. The last phase models the matchings of two genomes as a weighted bipartite graph — a graph with two sets of nodes such that nodes in one set can have edges only with nodes in the other set — such that each genome is placed in a separate ordered set, and integrates heuristics with bipartite graph matching algorithms to identify putative orthologues and corresponding gene-groups. The bipartite graph matching uses two different variations: best first approach and a variation of the Hungarian method (Papadimitrou, 1982). In the best first approach, the putative corresponding gene-groups are first identified, and the gene-pairs in the matching gene-groups are positively biased during the identification of orthologues. The variation of the Hungarian method tries to achieve the local maximization of a cumulative sum of scores to identify corresponding gene-groups. The technique iteratively identifies putative

corresponding gene-groups, iteratively biases the gene-pair matchings within a putative gene-group, and treats orthologues as a special case of gene-groups with a group-size of one. Putative gene-groups with a distance less than a threshold are merged. The process is repeated until gene-groups can not be extended further. The developed prototype software uses the integration of Prolog [18] using Sicstus 3.2 [15] and C and Unix system routines, and is portable across different architectures supporting these languages and the BLAST software package. The current implementation uses the GCG software package [5] for local alignment variant of the Smith-Waterman algorithm. For a typical prokaryotic genome such as *H. influenzae* and *B. subtilis*, a typical Silicon Graphics workstation takes approximately 7200 seconds for the BLAST phase, approximately 1800 seconds for the Smith-Waterman alignment phase, and approximately twenty seconds for the bipartite matching phase. The use of Prolog and the dynamic invocation of shells in UNIX routines slows down the overall time-efficiency of the prototype software by approximately two to three times. However, the automated comparison of two genomes is handled in realistic time, and the technique is suitable for distributed computing.

The developed software has been utilized to compare five available genomes: partial *E. coli* (1304 curate genes) *H. influenzae*, partial *B. subtilis* (1852 genes), *M. genitalium*, and *M. jannaschii*. The comparison results suggest the presence of putative orthologues, putative corresponding gene-groups, different types of correspondence in gene-groups, duplicated gene-groups, putative fused genes, shuffling of genes, and conserved genes.

The contributions of this paper are:

(1) the development of algorithms to identify orthologues, corresponding gene-groups, types of corresponding gene-groups;
(2) the integration of the information derived from the Smith-Waterman algorithm to identify gene duplications, gene-group duplication, and fused genes;
(3) the development of an integrated software, and
(4) a comparative study of five genome-pairs.

The paper is organized as follows. Section 2 describes some background and definitions needed for modeling. This section has been made generic to improve the readability to the researchers in both Computational Science and Biological Science communities. Section 3 describes an overview of various stages of software in finding the orthologues. Sections 4 and 5 describe the first approach: Section 4 describes an algorithm to identify corresponding gene-groups, and Section 5 describes an algorithm to find the best matchings in a bipartite graph. Section 6 describes a variant of the Hungarian method and the corresponding algorithm to identify orthologues and corresponding gene-groups. Section 7 presents a technique to derive fused genes. Section 8 briefly presents a comparison of the five genome-pairs. Section 9 concludes the paper.

2. BACKGROUND AND DEFINITIONS

In this section we briefly describe basic definitions related to microbial genomes, basic alignment techniques, some mathematical concepts needed in this paper, and definitions related to graphs and bipartite matching. Subsection 2.1 describes the basics of genomes. Subsection 2.2 describes the basic concepts of sequence searching and alignment. Subsection 2.3 describes the basic notions related to gene-groups and operons — an ordered set of genes involved in a complex functionality in a metabolic pathway. Subsection 2.4 describes some mathematical concepts needed in this paper. Subsection 2.5 lists the mathematical notations used in this paper. Subsection 2.6 describes the notations needed to model a genome comparison using bipartite graph matching. Section 2.7

describes the basics of the Hungarian method — a well known bipartite graph matching technique — used in one of the algorithms.

2.1 Genome Related Background. The genome of an organism is encoded within molecules of DNA. A molecule of DNA is a sequence of four nucleotides: adenine 'A', cytosine 'C', guanine 'G', and thymine 'T'. A DNA molecule is represented by a sequence of characters from the alphabet {'A', 'C', 'G', 'T'}. A protein is a sequence of different types of molecules collectively known as amino acids. Commonly, the sequence of proteins comprises alanine 'A', arginine 'R', asparagine 'N', aspartic acid 'D', cysteine 'C', glutamic acid 'E', glutamine 'Q', glycine 'G', histidine 'H', isoleucine 'I', leucine 'L', lysine 'K', methionine 'M', phenylalanine 'F', proline 'P', serine 'S', threonine 'T', tryptophan 'W', tyrosine 'Y', and valine 'V'. The alphabet {'A', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'V', 'W', 'Y'} is denoted by A.

A *microbial genome,* denoted by subscripted capital Greek letter Γ_I, is an ordered set of pairs of the form $<(\chi_1, \gamma_1), \ldots, (\chi_n, \gamma_n)>$. Each γ_I is a sequence of the form $<s_1, ..., s_N>$ $(1 \le I \le N)$ where $s_I \in \mathsf{N}$ for DNA and $s_I \in \mathsf{A}$ for protein. Each χ_I is a control region preceding γ_i. In this paper, we are interested in the comparison of protein sequences of the form $<\gamma_{11}, \ldots, \gamma_{1N}>$ and $<\gamma_{21}, \ldots, \gamma_{2M}>$.

Homologues are similar genes derived from some common ancestral gene. *Paralogues* are homologous genes in a multigene family as a result of gene duplication [6]. Paralogous genes may have possible variations in functionality. An *orthologue* is the exact functional counterpart of a gene in another genome that has arisen from speciation [6]. The history of an orthologous gene reflects the history of the species.

2.2. Sequence Searching and Sequence Alignment Techniques. Two genetic sequences are similar, if there is a significant match between them after limited shifting of characters. *Sequence alignment* is the process of aligning similar sequences together in a way that asserts a correspondence between characters that are thought to derive from a common ancestral sequence. Aligning a set of sequences requires the introduction of spacing characters referred to as *indels*. If the aligned sequences did indeed derive from a common ancestral sequence, then indels represent possible evolutionary events in which characters were either inserted or deleted.

The BLAST software [3] uses local alignment to identify similar sequences. It is based upon selecting a small subsequence, using string matching to identify locations of matches, and expanding the size of matched segments at the identified locations. The similarity searches of BLAST are asymmetric: comparing a sequence γ_I with γ_J may not give the same similarity score as comparing a sequence γ_J with γ_I, and the comparison of the same sequences γ_I to γ_I and γ_J to γ_J gives different scores for different sequences. However, BLAST is time efficient.

The Smith-Waterman algorithm [19] is a matrix-based dynamic programming technique to find out the optimum pair-wise sequence alignment. The Smith-Waterman algorithm is more precise than BLAST, and the order of gene-pair comparison does not affect the similarity score derived using the Smith-Waterman algorithm. The time-complexity of the Smith-Waterman algorithm is $O(N^2)$ where N is the size of two aligned sequences.

2.3. Gene-groups and Operons. A *gene-group* is a cluster of neighboring genes $<\gamma_I\ \gamma_J\ \gamma_K \ldots>$ $(I < J \le I + r, J < K \le J + r$ where r is a positive bounded integer with a small upper bound such as ten) which have a natural pressure to occur in proximity. A gene-group may have insertions, permutations, or deletions of genes with reference to a corresponding gene-group in another genome. A *corresponding gene-group* $<\gamma_{1I}\ \gamma_{1J}\ \gamma_{1K} \ldots >$ $(I < J < K)$ in the genome Γ_1 matches with a corresponding gene-group $<\gamma_{2M}\ \gamma_{2N}\ \gamma_{2P} \ldots>$ $(M < N < P)$ in Γ_2 such that, in general, γ_{1I} and γ_{1J} and γ_{1K}

etc. are similar to one of the genes in the sequence $<\gamma_{2M}\gamma_{2N}\gamma_{2P}\ldots>$. A *conserved gene-string* $<\gamma_I\gamma_{I+1}\gamma_{I+2}\ldots\gamma_J>$ $(J > I)$ [17] is a special case of corresponding gene-groups such that there is no gap between two consecutive genes. If we allow deletion or insertion of single genes within a conserved gene string, the resulting cluster is still considered a gene-group. A *shuffling* is a correspondence between two putative orthologous genes such that one of the genes lies in a corresponding gene-group. However, the number of shuffled genes is much smaller than the number of matching genes in the corresponding gene-groups.

An *operon* [1] is a set of neighboring genes which are transcribed as a single unit and has common regulatory elements. Often, genes within an operon belong to the same pathway or encode different units of a single protein.

2.4. Some Mathematical Concepts. A sequence is in partial order if two consecutive elements in the sequence are related with a relationship π which is transitive: $s_I \pi s_J$ (s_I precedes s_J) and $s_J \pi s_K$ implies $s_I \pi s_K$, and antisymmetric: $s_I \pi s_J$ implies $s_J \phi s_I$ where 'ϕ' is inverse relation of 'π'. However, $S_I \pi S_J$ and $S_I \pi S_K$ does not imply any order between S_J and S_K. For example, the sets {1}, {1, 2}, {1, 2, 3}, {1, 2, 4} are in partial order with relationship '$\subset$' (proper subset). *Greatest lower bound*, denoted by $\sqcap$, of two consecutive elements in partial order is given by the lower value in the relationship. For example, if $S_I \pi S_J$ then $S_I \sqcap S_J = S_I$.

A set S of data elements of the form $(\gamma_{1I}, \gamma_{2K})$ when transposed will give a transposed set with data elements of the form $(\gamma_{2K}, \gamma_{1I})$. We will denote the transposed set by the set S^T.

2.5. Mathematical Notations. We denote genomes by subscripted capital Greek letter Γ_I, mapping of two genomes Γ_1 and Γ_2 by $\Gamma_1 \alpha \Gamma_2$ (or $\Gamma_2 \alpha \Gamma_1$), a set within curly brackets {…}, a tuple within brackets (…), a sequence within angular brackets <…>, a range within square brackets […], number of elements in a set S by |S|, membership of a set by $\in$, union of two sets by $\cup$, union of multiple sets by $\bigcup$, intersection of two sets by $\cap$, empty set by $\varnothing$, set difference by –, deletion of an element in a set by –, insertion in a set by +, subset by $\subseteq$, existence of an element by $\exists$, forall by $\forall$.

We denote the Smith-Waterman alignment of two genes γ_{1I} and γ_{2J} by $\gamma_{1I} \leftrightarrow \gamma_{2J}$, the aligned version of γ_{1I} by $\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J})$, the aligned version of γ_{2J} by $\pi_2(\gamma_{1I} \leftrightarrow \gamma_{2J})$, the start position of aligned version of γ_{1I} by $start(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J}))$, the end position of aligned version of γ_{1I} by $end(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J}))$, the start position of aligned version of γ_{2J} by $start(\pi_2(\gamma_{1I} \leftrightarrow \gamma_{2J}))$, the end position of aligned version of γ_{2J} by $end(\pi_2(\gamma_{1I} \leftrightarrow \gamma_{2J}))$.

For bipartite graph matching, we denote an edge by $(\gamma_{1I}, \gamma_{2J})$ (or $(\gamma_{2J}, \gamma_{1I})$) where $\gamma_{1I} \in \Gamma_1$ and $\gamma_{2J} \in \Gamma_2$, the mapping of a gene $\gamma_{1I} \in \Gamma_1$ to a gene $\gamma_{2J} \in \Gamma_2$ by $\gamma_{1I} \alpha \gamma_{2J}$, the source-node γ_{1I} of an edge $(\gamma_{1I}, \gamma_{2J})$ by $\pi_1(\gamma_{1I}, \gamma_{2J})$, the sink-node γ_{2J} of an edge $(\gamma_{1I}, \gamma_{2J})$ by $\pi_2(\gamma_{1I}, \gamma_{2J})$, the set of source-nodes of a set of edges S by $\Pi_1(S)$, and the set of sink-nodes of a set of edges S by $\Pi_2(S)$.

We use natural language with C-like syntax to express algorithms, and "%" to denote comments in the algorithms.

2.6 Graphs and Genome Comparison Modeling. A graph has a set of nodes of the form $\{V_1, \ldots, V_N\}$ and a set of edges connecting two nodes such that each edge is of the form (V_I, V_J) where $1 \le I, J$

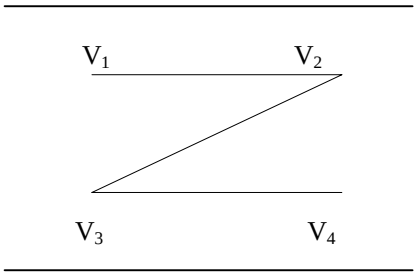

Figure 1. A strongly connected graph

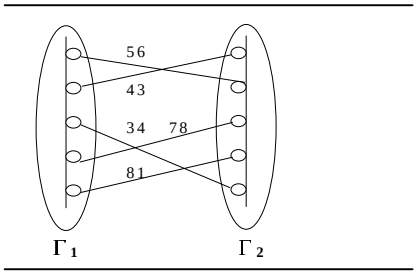

Figure 2. Genome comparison as bipartite graph

≤ N. A degree of a node is the number of edges incident upon the node. For example, the node V_2 in Figure 1 has a degree 2. A path in a graph is a sequence of edges which connect two nodes in a graph. A strongly connected graph (see Figure 1) has a path between any pair of two nodes. A bipartite graph has two sets of nodes such that there are edges from one set of nodes to another set of nodes. However, there are no edges within the same set of nodes. A weighted bipartite graph has different weights for different edges as shown in Figure 2.

A genome is a sequence of genes. Each gene is modeled as a node. A pair of genomes is modeled as a weighted bipartite graph such that two similar genes in different genomes have edges between them, and the similarity score between two genes is the weight of the corresponding edge.

2.7 The Hungarian Method for Matching Bipartite Graph. The Hungarian method [14] is a well-known technique to find maximal matchings in bipartite graphs. Essentially it works as follows.

Initially the matching is empty: there are no edges in the matching. The Hungarian method continually searches for an *augmenting path* in the graph, that is, a sequence of edges $<(\gamma_{1I}, \gamma_{2J})\ (\gamma_{2J}, \gamma_{1K})\ (\gamma_{1K}, \gamma_{2L})\ (\gamma_{2L}, \gamma_{1M})\ \ldots>$ such that each odd-numbered edge is not a part of the current matching, and each even-numbered edge is a part of the current matching, and the sum of the weights of the odd-numbered edges is greater than the sum of the weight of the even-numbered edges. When an augmenting path is found the matching is updated by adding the odd-numbered edges to the matching and removing the even-numbered edges. Clearly the new matching has a greater total weight than the previous matching. When no augmenting paths exist the matching is maximal. The time-complexity of the Hungarian method to find a maximal matching is $O(N^3)$, where N is the number of genes.

Example 1

The bipartite graph illustrated in Figure 3 has four nodes in the first set and four nodes in the second set. The set of edges is $\{(\gamma_{11}, \gamma_{21}), (\gamma_{11}, \gamma_{22}), (\gamma_{12}, \gamma_{21}), (\gamma_{12}, \gamma_{24}), (\gamma_{13}, \gamma_{22}), (\gamma_{13}, \gamma_{24}), (\gamma_{14}, \gamma_{23})\}$.

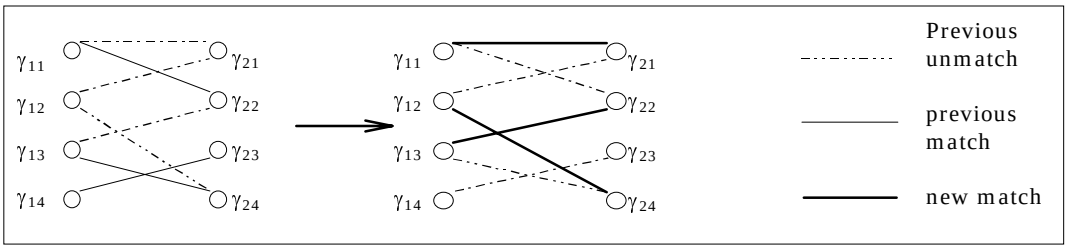

Figure 3. The Hungarian method for bipartite graph matching

Using the Hungarian method, an augmenting path is given by $<(\gamma_{21}, \gamma_{11})\ (\gamma_{11}, \gamma_{22}), (\gamma_{22}, \gamma_{13}), (\gamma_{13}, \gamma_{24})\ (\gamma_{24}, \gamma_{12})>$ such that the sum of weights of the edges (γ_{21}, γ_{11}), (γ_{22}, γ_{13}), and (γ_{24}, γ_{12}) is greater than the sum of the weights of the edges (γ_{11}, γ_{22}) and (γ_{13}, γ_{24}). Adding the odd-numbered edges and deleting the even-numbered edges results in the matching shown on the right of Figure 3. The process is repeated until there are no more augmenting paths. The final matchings are given by the solid edges (γ_{11}, γ_{21}), (γ_{12}, γ_{24}), and (γ_{13}, γ_{22}).

3. OVERALL SCHEME TO IDENTIFY ORTHOLOGUES

In this section, we describe an overall scheme to derive orthologues and corresponding gene-groups. As shown in Figure 4, there are four stages of identifying orthologues, and there are two parallel streams in the BLAST phase. This parallel streams are caused due to the asymmetry of comparison present in the BLAST comparison. We denote the first stream by 'A' and the second stream by 'B' for convenience.

3.1. Stage 1: Gene-pair Filtering by the BLAST Comparison. The input to this stage is a pair of genomes Γ_1 and Γ_2, and the output of this stage is a set of gene-pairs of the form (γ_{1I}, γ_{2J}) or (γ_{2J}, γ_{1I}) where γ_{1I} is a gene in Γ_1 and γ_{2J} is a gene in Γ_2. Every gene ($1 \le I \le$ number of genes in a genome) in one genome is compared with the other complete genome to identify the set of potentially similar genes. In Stage 1A, the genes of Γ_1 are compared against Γ_2, and in Stage 1B, the genes of Γ_2 are compared against Γ_1. A BLAST search process is invoked for every gene search. After the BLAST search, the two similarity scores *high score* and *chance expectation value* are obtained. Those gene comparisons which have a similarity score less than a threshold value (*high-score < 50* and *chance expectation value* $> 10^{-9}$) are pruned. In order to avoid any false negatives, conservative cutoff values are chosen. After identifying the set S_2 of filtered pairs from $\Gamma_2\ \alpha\ \Gamma_1$, the elements in the set S_2 are transposed. The transposed set S_2^T is merged with the set of filtered pairs derived from $\Gamma_1\ \alpha\ \Gamma_2$, and duplicates are removed. This merged set is passed as an input to stage 2.

3.2. Stage 2: Pair-wise Alignment using Dynamic Programming. The input to this stage is a set of filtered gene-pairs (γ_{1I}, γ_{2J}) from Stage 1. The output of this stage is a 9-tuple of the form (γ_{1I}, γ_{2J}, *start*($\pi_1(\gamma_{1I})$), *end*($\pi_1(\gamma_{1I})$), *start*($\pi_2(\gamma_{2J})$), *end*($\pi_2(\gamma_{2J})$), *length of match, maximum length of mismatch, percentage similarity score*). Every gene pair (γ_{1I}, γ_{2J}) (or (γ_{2K}, γ_{1l})) filtered from Stage 1 is aligned using a variant of the Smith-Waterman local alignment provided by the *GCG software package* [5]. The *start* information describes the starting point of a sequence in the gene-pair alignment, and the *end* information describes the relative end point of a sequence in the gene-pair alignment. The *start*, *end*, and *length of match* were used to identify gene-fusion: two consecutive genes fused to form a

single gene. The *length of maximum mismatch* was used to identify false positive matches, since the presence of very long sequences of indels implies the possible presence of artifacts.

3.3. Stage 3: Bipartite Graph Matching. In this stage, comparison of two genomes is modeled as a weighted bipartite graph with genes as nodes, possible matchings as edges, and matching scores after the Smith-Waterman alignment as the weights of the edges. This stage performs the following tasks:

(1) Best matchings between node-pairs are identified, and other matchings are pruned.
(2) Putative corresponding gene-groups are identified and classified into different categories.
(3) Duplicated genes, fused genes, and duplicates of gene-groups are identified.

This stage uses two different algorithms as described in Sections 4, 5 and 6. The first algorithm identifies the putative corresponding gene-groups first by comparing two genomes; sorts the weights of the edges in descending order, and positively biases the weights of the edges in the putative corresponding gene-groups before identifying the best matches. The second algorithm — a variant of the Hungarian method — is based upon maximizing the cumulative sum of the weights of the edges of groups of genes in proximity. Both algorithms use heuristics to positively bias the weights within the putative corresponding gene-groups. Surprisingly, the findings from both algorithms are very similar.

3.4. Stage 4: Identification of Fused Genes. The input to this stage is the output from the Smith-Waterman algorithm. Fused genes are identified using the criterion that two consecutive orthologous genes γ_{1I} and $\gamma_{1(I \pm 1)} \in \Gamma_1$ (γ_{2J} and $\gamma_{2(J \pm 1)} \in \Gamma_2$) match with a gene $\gamma_{2J} \in \Gamma_2$ ($\gamma_{1I} \in \Gamma_1$), the intervals of γ_{1I} α γ_{2J} and $\gamma_{1(I \pm 1)}$ α γ_{2J} (γ_{2J} α γ_{1I} and $\gamma_{2(J \pm 1)}$ α γ_{1I}) are adjacent to each other; and the cumulative sum of the length of γ_{1I} and $\gamma_{1(I+1)}$ (γ_{2J} and $\gamma_{2(J \pm 1)}$) is very close to the length of γ_{2J} (γ_{1I}). It is possible that during fusion, insertion or deletion may occur. In order to take care of alignment variations, length matching is relaxed to fall within a range which is less than half the length of the smallest gene in two genomes.

3.5. Overall System Execution. A prototype software using Prolog, C, and Unix was developed. Each comparison runs the BLAST software two times for self comparison (Γ_1 α Γ_1 and Γ_2 α Γ_2). The result is used to normalize BLAST scores by using the score of self-comparison of genes to represent the score for a 100 percent match. The BLAST software is run two more times to compare genomes (Γ_1 α Γ_2 and Γ_2 α Γ_1). Using the merged set of filtered matching pairs derived from the BLAST comparisons, the Smith-Waterman alignment software is executed once. The resulting similarity scores are used to model the genome matching as a bipartite matching problem, which is solved using one of the two matching algorithms.

For a typical size of 1800 genes X 1600 genes comparison, it takes approximately 7200 seconds for BLAST comparisons on a typical Silicon Graphics machine. For approximately 2000 filtered pairs, it takes approximately 2000 seconds for Smith-Waterman alignment on a typical Silicon Graphics machine. The bipartite graph matching algorithms take approximately 20 seconds to execute. Since the BLAST self-comparison for normalization has to be executed only once for each genome, the performance is improved further to approximately 110 minutes for following genome-pair comparisons.

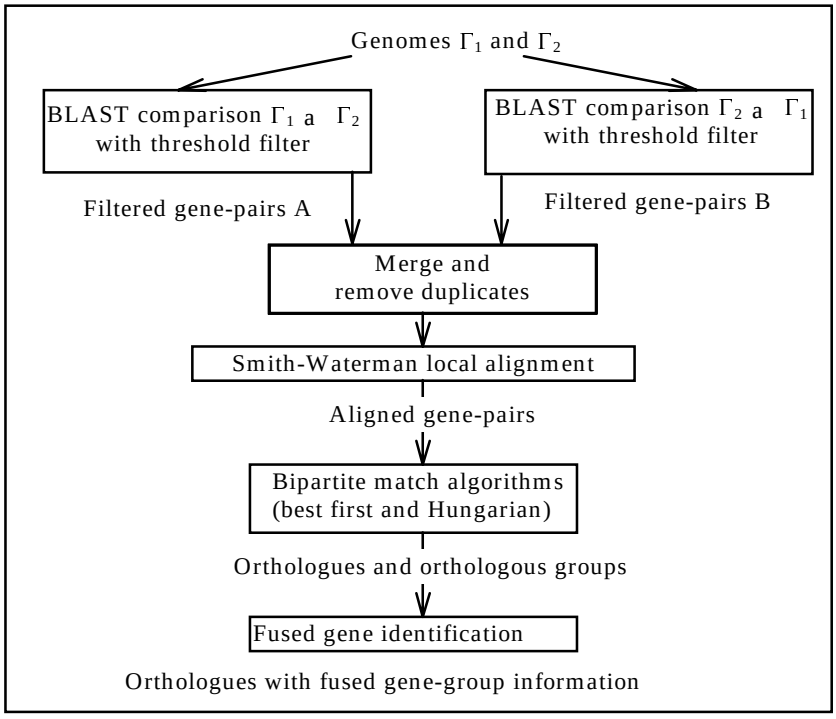

Figure 4. A scheme for the identification of orthologues and gene-groups

4. A TECHNIQUE TO IDENTIFY PUTATIVE GENE-GROUPS

In this section, we describe an algorithm to identify gene-groups in Γ_1 and the corresponding gene-groups in Γ_2. The complexity and proof of correctness of the algorithm has been removed from this paper in the interest of addressing a broad community.

This algorithm is based upon searching for the matchings in a neighborhood of a gene-pair matching $(\gamma_{1I}, \gamma_{2J})$. A seed window size b is used. If any other matching $(\gamma_{1(I+r)}, \gamma_{2(J \pm s)})$ (where r, s ≤ b) is found in the proximity, then the process is repeated from the matching $(\gamma_{1(I+r)}, \gamma_{2(J \pm s)})$. The next search is performed in the proximity (J – b, J + s + b) if the next matching is $(\gamma_{1(I+r)}, \gamma_{2(J+s)})$, or in the proximity (J – s – b, J + b) if the next matching is $(\gamma_{1(I+r)}, \gamma_{2(J-s)})$. Since the proximity keeps changing, variably sized gene-groups are identified. Since one node may have multiple matchings, multiple gene-group matchings are identified. This algorithm is run twice: once by traversing the matching genes in Γ_1 in order, and then traversing the matching genes in Γ_2 in order. This double traversal is needed to derive the information of gene-groups in one genome which match with multiple duplicated gene-groups in another genome. The basic algorithm is given in Figure 5. Many optimized variants of the algorithm are possible. The explanation is as follows.

One of the genomes Γ_1 is traversed from start to end. Let the set of nodes in Γ_1 (or Γ_2) be S_1, and the set of unprocessed weighted edges be S_2. Let the seed window size to identify neighbors in proximity be b. At any point, a weighted edge $(\gamma_{1I}, \gamma_{2J}) \in$ set of remaining edges is picked such that γ_{1I} has the least value of I. S_3 — the neighborhood set of γ_{1I} — is $\{\gamma_{1(I-b)}, \ldots, \gamma_{1(I+b)}\}$ such that $0 \leq I - b$ and $I + b \leq$ number of genes in Γ_1. Let the set of all the nodes in Γ_2 which match with γ_{1I} be S_4. Let the set of nodes in S_4 which share an edge with the nodes in the neighborhood set S_3 be S_5. The set S_5 is identified by selecting the nodes in S_4, one at a time, and taking the intersection of the set of nodes

in Γ_1 connected to the node and the set S_3. If the intersection is non-empty then that node in S_4 is included in S_5. If the set S_5 is non-empty then there is at least one possibly matching gene-group. After detecting the presence of a corresponding gene-group, the nodes in Γ_1 are traversed from the node γ_{1I}, and the putative gene-groups are collected as follows.

To facilitate dynamic alteration of neighboring nodes, a copy of the sets S_3 and S_5 is made in the sets S_7 and S_9 respectively. The set S_8 is used to verify matching edges incident upon the nodes in S_3 during the collection of variably sized gene-groups. If there is a matching node within the range $\gamma_{1(K + r)\ 0 < r \le b,\ K \ge I} \in S_7$ which matches one of nodes $\gamma_{2L} \in S_9$, then there is a group. The set of edges in a neighborhood is collected in S_6; S_7 — the neighborhood set of $\gamma_{1K(K \ge I)} \in S_7$ — is extended dynamically to include the neighbors of $\gamma_{1(K + r)}$; S_9 — the neighborhood set of matching nodes in Γ_2 — is extended dynamically to include the neighbors of γ_{2L}; and all the edges incident on $\gamma_{1(K + r)\ 0 < r \le b \text{ and } K \ge I}$ are included dynamically in the set S_8. Those edges which are traversed once are deleted from S_8, and those nodes in Γ_2 which have been traversed once are deleted from the sets S_5 and S_9. After a matching ($\gamma_{1(K + r)} \in S_7$) α ($\gamma_{2L} \in S_9$) is not found in the neighborhood, S_6 — the current collected set of edges — is closed; $\Pi_1(S_6)$ — the set of source-nodes in S_6 — gives a putative corresponding gene-group in Γ_1, and $\Pi_2(S_6)$ — the set of sink-nodes in S_6 — gives the putative corresponding gene-group in Γ_2. By identifying the direction of mapping, the type of gene-groups {*in-order, reverse, permuted*} is decided, and the putatively matching group (gene-group type, $\Pi_1(S_6)$, $\Pi_2(S_6)$) is stored in S — the set of matching gene-groups. The set of nodes in Γ_2 which have been traversed during the identification of the last group is deleted from the set S_9, the set of traversed edges is deleted from the set S_8.

The process is repeated to identify the next putative corresponding gene-group which starts with γ_{1I}. This is possible since there are duplicates of gene-groups. Dynamically incrementing the neighborhood set of γ_{1I} and the set of matching nodes in Γ_2 and the set of matching edges ensures selection of all the variably sized gene-groups. After finding out all the putative corresponding gene-groups involving γ_{1I} (including singleton groups in the absence of any group of size greater than or equal to 2), the set of edges incident upon γ_{1I} is deleted to avoid reconsideration. The process is repeated by picking up the next edge in S_2 which has the minimum index, until S_2 is empty.

Example 2

Consider Figure 6. There are four matching nodes in genome Γ_1 and six matching nodes in genome Γ_2. Let us further assume that the set S_2 — the set of edges after the Smith-Waterman alignment — is $\{(\gamma_{11}, \gamma_{22}), (\gamma_{11}, \gamma_{26}), (\gamma_{12}, \gamma_{23}), (\gamma_{12}, \gamma_{25}), (\gamma_{13}, \gamma_{24}), (\gamma_{14}, \gamma_{27})\}$. The set S_1 — the set of nodes in Γ_1 — is $\{\gamma_{11}, \gamma_{12}, \gamma_{13}, \gamma_{14}\}$. Let us assume the seed window size to be one.

We start traversing from node γ_{11}. The value of S_3 — the neighborhood set for γ_{11} — is $\{\gamma_{11}, \gamma_{12}\}$. The value of S_4 — the set of nodes in Γ_2 which match with γ_{11} — is $\{\gamma_{22}, \gamma_{26}\}$. The set of neighboring nodes for γ_{22} is $\{\gamma_{21}, \gamma_{22}, \gamma_{23}\}$, and the set of neighboring nodes for γ_{26} is $\{\gamma_{25,} \gamma_{26}, \gamma_{27}\}$. The value of S_5 — the subset of $\{\gamma_{22}, \gamma_{26}\}$ whose neighbors match with the set $\{\gamma_{11}, \gamma_{12}\}$ — is $\{\gamma_{22}, \gamma_{26}\}$. The value of S_9 — the set of all neighboring nodes in the set S_2 — is $\{\gamma_{21}, \gamma_{22}, \gamma_{23}, \gamma_{25,} \gamma_{26}, \gamma_{27}\}$ which is derived by the union of two sets $\{\gamma_{21}, \gamma_{22}, \gamma_{23}\}$ and $\{\gamma_{25,} \gamma_{26}, \gamma_{27}\}$. The value of S_8 — the union of the set of the edges incident upon the nodes γ_{11} and γ_{12} — is $\{(\gamma_{11}, \gamma_{22}), (\gamma_{11}, \gamma_{26}), (\gamma_{12}, \gamma_{23}), (\gamma_{12}, \gamma_{25})\}$. Since the set S_5 is non-empty, putative corresponding gene-groups are present. The traversal from γ_{11} gives the first edge as $(\gamma_{11}, \gamma_{22})$. Since $\gamma_{22} \in S_9$ and the edge $(\gamma_{12}, \gamma_{23}) \in S_8$ and the node $\gamma_{23} \in S_9$, the edge $(\gamma_{11}, \gamma_{22})$ is inserted in the set S_6 and deleted from the set S_8; the node γ_{22} is deleted from the set S_9. The new value of the set S_8 is $\{(\gamma_{11}, \gamma_{26}), (\gamma_{12}, \gamma_{23}), (\gamma_{12}, \gamma_{25}), (\gamma_{13}, \gamma_{24})\}$ which is given by

union of $\{(\gamma_{13}, \gamma_{24})\}$ — the set of edges incident upon γ_{13} — and the set $\{(\gamma_{11}, \gamma_{26}), (\gamma_{12}, \gamma_{23}), (\gamma_{12}, \gamma_{25})\}$. The new value of the set S_9 is $\{\gamma_{21}, \gamma_{23}, \gamma_{24}, \gamma_{25,} \gamma_{26}, \gamma_{27}\}$. The new value of the set S_5 is $\{\gamma_{22}\}$.

The process is repeated with the node γ_{12} which gives the new value of the set S_6 as $\{(\gamma_{11}, \gamma_{22}), (\gamma_{12}, \gamma_{23})\}$, the new value of the set S_8 as $\{(\gamma_{11}, \gamma_{26}), (\gamma_{12}, \gamma_{25}), (\gamma_{13}, \gamma_{24})\}$, the new value of the set S_9 as $\{\gamma_{21}, \gamma_{24}, \gamma_{25,} \gamma_{26}, \gamma_{27}\}$. When the process is repeated with the node γ_{13}, there is no matching of the form $(\gamma_{14}, \gamma_{25})$. The iteration stops after inserting the edge $(\gamma_{13}, \gamma_{24})$ in the set S_6. After this iteration, the value of the set S_6 is $\{(\gamma_{11}, \gamma_{22}), (\gamma_{12}, \gamma_{23}), (\gamma_{13}, \gamma_{24})\}$, the value of the set S_8 is $\{(\gamma_{11}, \gamma_{26}), (\gamma_{12}, \gamma_{25})\}$, the value of the set S_9 is $\{\gamma_{21}, \gamma_{25,} \gamma_{26}, \gamma_{27}\}$, and the value of the set S_5 is $\{\gamma_{22}\}$. It is easy to verify that the group is *in-order*, the set $\Pi_1(S_6)$ gives a putative corresponding gene-group $\{\gamma_{11}, \gamma_{12}, \gamma_{13}\}$ in Γ_1, and the set $\Pi_2(S_6)$ gives a putative corresponding gene-group $\{\gamma_{22}, \gamma_{23}, \gamma_{24}\}$ in Γ_2. The triple (*in-order*, $\{\gamma_{11}, \gamma_{12}, \gamma_{13}\}$, $\{\gamma_{22}, \gamma_{23}, \gamma_{24}\}$) is stored in the set S, and the process is repeated again with γ_{11}. The triple (*reverse*, $\{\gamma_{11}, \gamma_{12}\}$, $\{\gamma_{26}, \gamma_{25}\}$) is identified and inserted in the set S. After inserting the triple, the set S_5 becomes empty. The new value of the set S_2 is $\{(\gamma_{14}, \gamma_{27})\}$. No group is found with the node γ_{14}, and the set S_2 becomes empty in the next iteration. The algorithm terminates with the final value of S as $\{$(*in-order*, $\{\gamma_{11}, \gamma_{12}, \gamma_{13}\}$, $\{\gamma_{22}, \gamma_{23}, \gamma_{24}\}$), (*reverse*, $\{\gamma_{11}, \gamma_{12}\}$, $\{\gamma_{26}, \gamma_{25}\}$)$\}$.

5. THE FIRST TECHNIQUE TO IDENTIFY ORTHOLOGUES

In this section, we describe the first technique to identify putative orthologues. In this scheme, the knowledge about putative corresponding gene-groups is used to bias positively the edges between matching gene-pairs in corresponding gene-groups under the assumption that matching gene-pairs occurring within corresponding gene-groups are more probable to be an orthologue than shuffled genes with similar weights.

The technique uses sorting to arrange the weights of the edges in descending order, and starts marking the node-pairs with highest weight. The node-pairs with highest weight become putative orthologues, and all the edges incident upon those two nodes are removed from further consideration. The process is repeated until no more pairs are left.

5.1 Orthologue Resolution. The gene-pair matchings were divided into four categories: *one-to-one, much-above, preferred,* and *conflict. One-to-one* means that the genes γ_{1I} and γ_{2J} in matching gene-pair $(\gamma_{1I}, \gamma_{2J})$ have no other matching genes. *Much above* means that the weight of $(\gamma_{1I}, \gamma_{2J})$ — one of the edges in the union of the sets of the edges incident upon γ_{1I} and γ_{2J} — stands out compared to all other edges incident upon γ_{1I} or γ_{2J}. We used an *adhoc* criterion that the weight of a *much-above edge* is 20 percent above the weights of other edges incident upon γ_{1I} and γ_{2J}. *Preferred* means that the weight of the highest matching edge is 10 percent above the weights of other edges incident upon γ_{1I} and γ_{2J}, or is inside a putative corresponding gene-group despite having a conflicting weight. *Conflict* means that the weights of all edges incident upon γ_{1I} and γ_{2J} are so close that it is not possible to identify the outstanding matching gene-pair. There is a partial order (under the relationship *orthologues*) between these classes: *one-to-one* ϕ *much above* ϕ *preferred* ϕ *conflict*. Clearly, *one-to-one matching* corresponds to *unique orthologues,* and *much-above matchings* correspond to *clear orthologues*. In the absence of enough number of sequenced genomes, we did not have statistical results to decide upon the cutoff point.

5.2 Algorithm for Orthologues and Corresponding Gene-groups. The similarity scores for gene-groups are first positively biased. The resulting scores are sorted. The classification is done based upon descending order of weights of the edges. The edge $(\gamma_{1I}, \gamma_{2J})$ with highest weight is selected and

classified and all other edges incident on the source-node γ_{1I} and γ_{2J} are discarded. The process is repeated until there are no more edges in the bipartite graph.

Algorithm putative corresponding gene-groups;

Input: 1. A set G of matching gene-pairs from the Smith-Waterman algorithm;
2. A pair of genomes Γ_1 and Γ_2;

Output: 1. A set S of putative gene-groups in Γ_1 (or Γ_2);

{ 1. let the ordered sets of genes in Γ_1 be S_1; $S_2 = G$;
2. while ($S_2 \neq \varnothing$) {
3. let the next edge be $(\gamma_{1I}, \gamma_{2J}) \in S_2$ such that I has the minimum index;
4. S_3 = the neighborhood set of the node $\gamma_{1I} \in S_1$;
5. S_4 = the set of sink-nodes in Γ_2 for the source-node γ_{1I};
6. find out the subset $S_5 \subseteq S_4$ whose neighbors match with nodes in S_3;
7. if ($S_5 \neq \phi$) { % There is at least one putative matching gene-group;
8. $S_7 = S_3$; $S_8 = \bigcup$(sets of edges incident upon the nodes $\in S_3$);
9. $S_9 = \bigcup$(the neighborhood sets of nodes in S_5);
10. while ($S_5 \neq \varnothing$) { % collect single-gene-groups and multi-gene-groups from γ_{1I}
11. $S_6 = \phi$; % get ready to collect the next group
12. pick the next edge $(\gamma_{1K}, \gamma_{2M}) \in S_8$ such that K has the minimum index;
13. while ($\gamma_{1K} \in S_7$) {
14. if ($(\gamma_{1K}, \gamma_{2M}) \in S_8$ && $\gamma_{2M} \in S_9$ && $(\gamma_{1(K+r)\ r \leq b,}\ \gamma_{2L} \in S_9) \in S_8$) {
15. $S_6 = S_6 + (\gamma_{1K}, \gamma_{2M})$; $S_7 = S_7 \cup \{\gamma_{1(K+b)}, \ldots, \gamma_{1(K+r+b)}\}$;
16. $S_5 = S_5 - \gamma_{2M}$; % ensure that same group is not repeated
17. $S_8 = S_8 \cup$ edges incident upon $\{\gamma_{1(K+b),\ \ldots,\ \gamma_{1(K+r+b)}}\} - \{(\gamma_{1K}, \gamma_{2M})\}$; % update the set of edges from the next match
18. K = K + r; % the next matching node with the smallest index
19. $S_9 = S_9 \cup \{\gamma_{2(L-b)}, \ldots, \gamma_{2(L+b)}\} - \{\gamma_{2M}\}$;
20. else $\{S_6 = S_6 + (\gamma_{1K}, \gamma_{2M})$; $S_2 = S_2 - (\gamma_{1K}, \gamma_{2M})$; $S_9 = S_9 - \gamma_{2M}$; $S_5 = S_5 - \gamma_{2M}$;}
21. $S_8 = S_8 - (\gamma_{1K}, \gamma_{2M})$; $S_2 = S_2 - (\gamma_{1K}, \gamma_{2M})$;
}
22. if ($|S_6| \geq 2$) {% a gene-group should have minimum size 2
23. g = group-type(S_6) where g $\in$ {in-order, reverse, permuted};
24. $S = S + (g, \Pi_1(S_6), \Pi_2(S_6))$;}
}
}
}
25. S_{10} = the set of the edges incident upon γ_{1I};
26. $S_2 = S_2 - S_{10}$; % remove γ_{1I} from the future consideration;
}
}

Figure 5. An algorithm to identify putative matching gene-groups

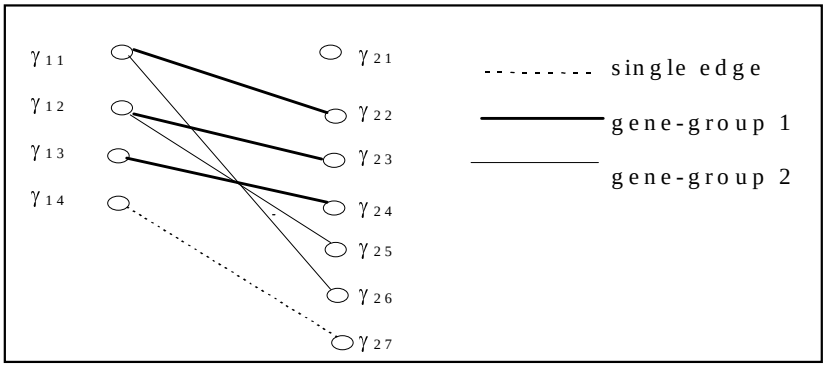

Figure 6. Identifying putative gene-groups in $\Gamma_1 \alpha \Gamma_2$

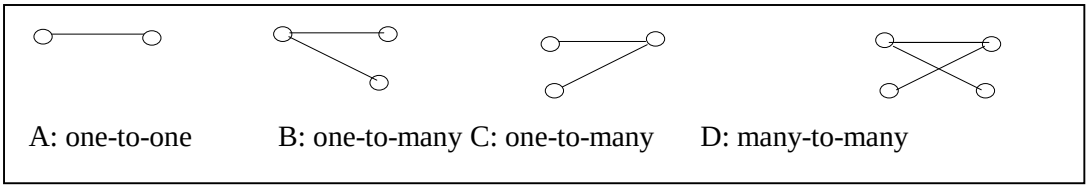

Figure 7. Different classification of edges incident upon matching nodes

The edge groups between two matching nodes $\gamma_{1I} \in \Gamma_1$ and $\gamma_{2J} \in \Gamma_2$ can be classified into three major groups (see Figure 7) as follows: *one-to-one*, *one-to-many*, and *many-to-many*. *One-to-one* means that degree of both γ_{1I} and γ_{2J} is one. *One-to-many* means that either γ_{1I} or γ_{2J} has degree greater than one. *One-to-many* can be further classified based upon whether the node with degree one is γ_{1I}, or the node with the degree one is γ_{2J}. *Many-to-many* means that both γ_{1I} and γ_{2J} have degree greater than one. The *one-to-one* edges are *unique* orthologues, and they are removed from the consideration first. After picking the edges in descending order of the weights, each case is handled separately. In each case, we check whether the highest weight edge is *much-above*, *preferred*, or *conflicting*. For *one-to-many*, the edge with second highest weight from the set of edges incident upon γ_{2J} (or γ_{1I}) is picked. For *many-to-many* edges, the second highest weights from the set of edges incident upon γ_{1I} and the set of edges incident upon γ_{2J} are picked up. The highest weight is compared against the second highest weights[4] to classify the orthologues. A formal algorithm is described in Figure 8.

The scheme works well except in some cases: it can not identify orthologues in the presence of paralogues which give many-to-many matchings, or when the similarity scores are so close that it is not possible to distinguish the orthologues. In such cases there is a clear conflict, and the putative orthologues can be resolved only by biological reasoning or by understanding the role of proteins in the metabolic pathways. The values of the bias-factors for *much-above*, *group-biasing* etc. used in this algorithm are adhoc due to the current lack of sequenced genomes needed for statistics.

Example 3

Consider Figure 9. After the Smith-Waterman alignment, the value of S_1 — the set of weighted edges in the bipartite graph — is {(γ_{11}, γ_{22}, *79*), (γ_{12}, γ_{21}, *56*), (γ_{12}, γ_{23}, *84*), (γ_{13}, γ_{24}, *49*), (γ_{13}, γ_{26}, *38*), (γ_{14}, γ_{21}, *46*), (γ_{15}, γ_{25}, *36*), (γ_{15}, γ_{27}, *33*), (γ_{16}, γ_{25}, *38*), (γ_{16}, γ_{27}, *39*)}. The value of S_2 — the set of gene-groups — is {(*reverse*, (γ_{11}, γ_{12}), (γ_{22}, γ_{21})), (*permuted*, (γ_{11}, γ_{12}, γ_{13}, γ_{14}), (γ_{22}, γ_{23}, γ_{24}, γ_{21})), (*in-order*, (γ_{15}, γ_{16}), (γ_{25}, γ_{27})), (*reverse*, (γ_{12}, γ_{16}), (γ_{27}, γ_{25}))}. After sorting, the value of S_3 — the set of

[4] Second highest weights from γ_{1I} and γ_{2J} in the case of many-to-many

edges sorted by weight — becomes {(γ_{12}, γ_{23}, *84*), (γ_{11}, γ_{22}, *79*), (γ_{12}, γ_{21}, *56*), (γ_{13}, γ_{24}, *49*), (γ_{14}, γ_{21}, *46*), (γ_{13}, γ_{26}, *38*), (γ_{15}, γ_{25}, *36*), (γ_{15}, γ_{27}, *33*)}.

After positively biasing (by 20 percent) the weights of the edges inside the putative corresponding gene-groups, the value of S_4 — the set of positively biased edges — becomes {(γ_{11}, γ_{22}, *95*), (γ_{12}, γ_{21}, *67*), (γ_{12}, γ_{23}, *100*), (γ_{13}, γ_{24}, *59*), (γ_{13}, γ_{26}, *38*), (γ_{14}, γ_{21}, *55*), (γ_{15}, γ_{25}, *44*), (γ_{15}, γ_{27}, *40*), (γ_{16}, γ_{25}, *46*), (γ_{16}, γ_{27}, *47*)}. In the first pass, {(γ_{11}, γ_{22})} — the set of one-to-one matchings — is identified and removed. Remaining classifications are done next.

During the first iteration, the edge (γ_{12}, γ_{23}) is classified as *much above orthologue* since the weight *100* is greater than 1.2×67. The weighted edges (γ_{12}, γ_{23}, *100*) and (γ_{12}, γ_{21}, *67*) are deleted from the set S_4, and the process of classification is repeated. The edge (γ_{13}, γ_{24}) is classified *much-above* since the weight *59* is greater than *38* — the weight of the edge (γ_{13}, γ_{26}). After the deletion of the edge (γ_{12}, γ_{21}), the weighted edge (γ_{14}, γ_{21}) becomes *much-above*. Note that this choice of classification for (γ_{14}, γ_{21}) is made under the assumption to identify the maximum number of orthologues. It is possible that γ_{21} is a mutated duplication of γ_{23}, and γ_{14} and γ_{21} are less similar. Handling such cases of duplications is a limitation of any automated matching technique, and needs biological reasoning. There is a many-to-many matching between the nodes γ_{15}, γ_{16}, γ_{25}, and γ_{27}. This occurs because γ_{15} and γ_{17} are putative paralogues and γ_{25} and γ_{27} are putative paralogues. The resolution of orthologues in such cases is very difficult, and must be performed by biological reasoning or by the correct knowledge of metabolic pathways.

6. APPLICATION OF THE HUNGARIAN METHOD FOR BIPARTITE MATCHING

The first technique was based upon finding the putative corresponding gene-groups, and then identifying the orthologues. In order to confirm the results derived from the previous technique, and to establish confidence in our results, we used a different technique for bipartite graph matching. The second technique applies the Hungarian method to find a maximal matching in the weighted bipartite graph. Two genomes form a bipartite graph with edges between nodes that have passed the BLAST filter. Similarity score after the Smith-Waterman alignment provides the weight of an edge. A maximal matching for a weighted graph is a set of weighted edges with disjoint endpoints such that the sum of their weights is maximal. The algorithm given in Figure 10 clusters edges in a bipartite graph iteratively by positively biasing the grouped genes such that the cumulative sum of the weights of the edges is locally maximum. A single orthologue is treated as a cluster of size one.

At the conclusion of the maximal matching phase, we have a matching between the genes in Γ_1 and Γ_2 such that no node is matched more than once and the weights between matched genes are high on average. The next stage is a grouping algorithm which initially treats each matched pair as a gene-group of size one. Adjacent gene-groups are merged iteratively if the regions where the gene-groups appear in Γ_1 and Γ_2 are sufficiently close.

We merge gene-groups which overlap within a seed window size b. A matched gene-group {(γ_{1K}, γ_{2J}), …, (γ_{1K+r}, γ_{2L})}, is considered to have an extended range pair ([*minimum*{*K*, …, *K* + *r*} – *b*, *maximum*{*K*, …, *K* + *r*} + *b*], [*minimum*{*J*, …, *L*} – *b*, *maximum*{*J*, …, *L*}+ *b*]). This shows the regions in Γ_1 and Γ_2, respectively, where the matched genes appear. Two gene-groups with extended range pair ([l_{11}, u_{11}], [l_{21}, u_{21}]} and ([l_{12}, u_{12}], [l_{22}, u_{22}]) are considered to overlap, if both the ranges [l_{11} – b, u_{11} + b] and [l_{12}, u_{12}] overlap and the ranges [l_{21} – b, u_{21} + b] and [l_{22}, u_{22}] overlap.

Algorithm orthologues;

Input: 1. A set S_1 of gene-pairs after the Smith-Waterman alignment;
2. A set S_2 of the union of set of weighted edges in the putative corresponding gene-groups derived from $\Gamma_1 \alpha \Gamma_2$ and the transpose of the set of weighted edges in putative corresponding gene-groups derived from $\Gamma_2 \alpha \Gamma_1$;
3. *Much-above* factor c, *group bias factor* f, and *preferred factor* p;

Output: 1. A set O of triples (γ_{1I}, γ_{2J}, classification);

{

1. S_3 = the set of weighted edges in S_1 sorted in descending order by weight;
2. O =∅; G =∅; S_4 = set of gene-pairs of the form $\gamma_{1I} \alpha \gamma_{2J}$ in S_2; S_5 = ∅;
3. ∀ ((γ_{1I}, γ_{2J}, weight) ∈ S_3) {% positively bias the weights of grouped edges
4. if (($\gamma_{1I} \alpha \gamma_{2J}$) ∈ S_4) $S_5 = S_5 + (\gamma_{1I}, \gamma_{2J}, f * weight)$; else $S_5 = S_5 + (\gamma_{1I}, \gamma_{2J}, weight)$;}
5. ∀ (γ_{1I}, γ_{2J}, weight) ∈ S_5 { % identify one-to-one matches first
6. if (((degree(γ_{1I}) = = 1) && (degree(γ_{2J}) = = 1))) { % it is a unique orthologue
7. O = O + (γ_{1I}, γ_{2J}, *unique*); $S_5 = S_5 - (\gamma_{1I}, \gamma_{2J}, weight)$;}
8. while ($S_5 \neq \emptyset$) { % find the best edge classification
9. pick the next edge (γ_{1I}, γ_{2J}, $weight_1$) ∈ S_5 in descending order by weight;
10. S_6 = set of the edges incident upon γ_{1I}; S_7 = set of the edges incident upon γ_{2J};
11. if (degree(γ_{1I}) = = 1 && degree(γ_{2J}) = = 1) O = O ∪ (γ_{1I}, γ_{2J}, *clear*);
12. else if (degree(γ_{2J}) = = 1) { % handle the case when γ_{1I} has two or more matches;
13. pick (γ_{1I}, γ_{2L}, $weight_2$) ∈ S_6 such that $weight_2$ is the second highest in S_6;
14. if ($weight_1 \geq c * weight_2$) O = O ∪ ($\gamma_{1I}$, γ_{2J}, *clear*);
15. else if ($weight_1 \geq p * weight_2$) O = O ∪ ($\gamma_{1I}$, γ_{2J}, *preferred*);
16. else {
17. S_8 = subset of S_6 such that $weight_1 \leq p *$ (minimum of all weights in S_8);
18. S_9 = set of triples with every edge of S_8 marked as conflict; O = O ∪ S_9;
}
19. else if (degree(γ_{1I}) = = 1 { % handle the case when γ_{2J} has two or more matches;
20. let (γ_{1K}, γ_{2J}, $weight_2$) ∈ S_7 such that $weight_2$ is the second highest in S_7;
21. if ($weight_1 \geq c * weight_2$) O = O ∪ ($\gamma_{1I}$, γ_{2J}, *clear*);
22. else if ($weight_1 \geq p * weight_2$) O = O ∪ ($\gamma_{1I}$, γ_{2J}, *preferred*);
23. else {
24. S_8 = subset of S_7 such that $weight_1 \leq p *$ (minimum of all weights in S_8);
25. S_9 = set of triples with every edge of S_8 marked as conflict; O = O ∪ S_9;
}
26. else { % both γ_{1I} and γ_{2J} have two or more matches;
27. let (γ_{1I}, γ_{2L}, $weight_2$) ∈ S_6 such that w_2 is the second highest weight in S_6;
28. let (γ_{1K}, γ_{2J}, $weight_3$) ∈ S_7 such that w_3 is the second highest weight in S_7;
29. if (($weight_1 \geq c * weight_2$) && ($weight_1 \geq c * weight_3$)) O = O ∪ ($\gamma_{1I}$, γ_{2J}, *clear*);
30. else if (($weight_1 \geq p * weight_2$) && ($weight_1 \geq p * weight_3$)) O = O ∪ ($\gamma_{1I}$, γ_{2J}, *preferred*);
31. else {
32. S_8 = subset of S_7 such that $weight_1 \leq p *$ (minimum of all weights in S_8);
33. S_9 = set of triples with every edge of S_8 marked as conflict; O = O ∪ S_9;
}
}
34. $S_5 = S_5 - S_6 - S_7$; % remove other edges incident upon these nodes from consideration;}

Figure 8. An algorithm to identify orthologues using first technique

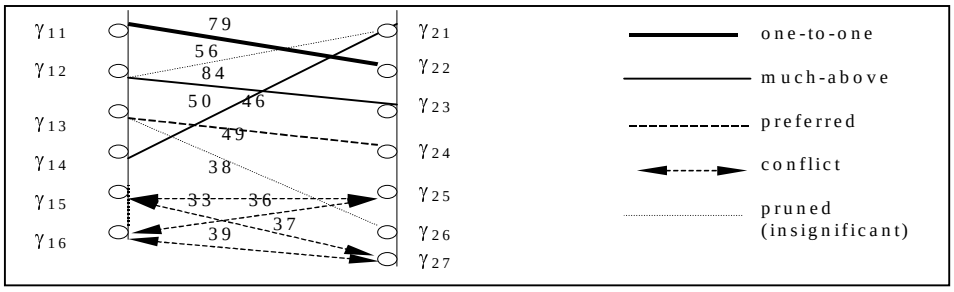

Figure 9. Bipartite matching of various types of edges for orthologues

The algorithm repeatedly searches for overlapping gene-groups, which are merged to form a larger gene-group. The search continues until none of the remaining gene-groups overlaps. Note that the

process is confluent, in the sense that it does not matter in which order the search for overlapping gene-groups is performed — the same overall result will be discovered. The time-complexity of the calculation of the final gene-groups is $O(N^2)$ where N is the number of nodes.

The next stage of the technique uses the computed gene-groups of size greater than *minsize* — minimum size of the group — to alter the weights of edges in the initial graph. Each edge ($\gamma_{1(K+r)}$, γ_{2M}) which, if considered as a singleton gene-group, would overlap with a gene-group of size *minsize* or greater if the calculated gene-groups are positively biased. Note that this will increase all the weights of edges in gene-groups with size greater than *minsize*. The weights of other edges are negatively biased.

We could perform the same process repeatedly until the calculated matching does not change. Indeed, we experimented with this idea. The calculated gene-groups did not change significantly in each iteration in the process, but it was not stable. This occurred because the maximal matching algorithm will always match a gene if possible, and hence some of the edges in the maximal matching have low similarity scores and are unlikely to represent useful biological information. These parts of the matching altered in each iteration.

Finally the maximal matching and gene-group calculations are repeated with these new edge weights. The justification for the new calculation is that the original maximal matching may not have chosen to match two genes that are part of a gene-group because there was an equally likely match candidate that was not part of the gene-group. By penalizing the matches which are inside a putative gene-group in the recalculation, the maximal matching will build larger gene-groups.

Example 4

Consider the diagram illustrated to the left of Figure 11. Suppose a maximal matching is given by the solid lines, and the remaining edges are shown as dashed lines. The first two matching edges are considered as singleton gene-groups. Given a seed window size of one, the extended range pairs are illustrated by vertical lines. Since both ranges overlap they are merged into a single group. The result of the final calculation is shown to the right. Supposing that *minsize* is *two*, there are two gene-groups of size *two* or greater, illustrated with different shadings. Every dashed edge and each solid edge that is part of a gene-group of size *two* or greater is positively biased by the bias-factor. The bias-factor is subtracted from other dashed edges and solid edges which form a singleton gene-group.

Algorithm Hungarian-variant-orthologues;

Input:
1. G – a set of weighted edges in bipartite graph;
2. A maximal matching S_1 from the Hungarian method;
3. A seed window size b;
4. A bias-factor f to positively bias the genes in a putative corresponding gene-group;
5. A predetermined number of iterations N;

Output: 1. A set S of grouped orthologues (including single element groups);

{ Iteration-index = 1;
while (iteration-index ≤ N){ % repeat a reasonable number of times
1. $S_2 = \varnothing$; S =∅; % S_2 is the initial set of single element groups
2. ∀ (pairs (γ_{1K}, γ_{2M}) ∈ S_1) { % Initialize
3. $S_2 = S_2 + ([K, K], [M, M], \{(\gamma_{1K}, \gamma_{2M})\})$;
4. ∀ ($[l_{11}, u_{11}]$, $[l_{21}, u_{21}]$, G_1) ∈ S_2 {
5. $S_2 = S_2 - \{([l_{11}, u_{11}], [l_{21}, u_{21}], G_1)\}$;
6. ∀ ($[l_{12}, u_{12}]$, $[l_{22}, u_{22}]$, G_2) ∈ S_2 { % start merging neighboring groups
7. if ((($l_{11} - b \le l_{12} \le u_{11} + b$) || ($l_{11} - b \le u_{12} \le u_{11} + b$)) &&
(($l_{21} - b \le l_{22} \le u_{21} + b$) || ($l_{21} - b \le u_{22} \le u_{21} + b$)) {
% ranges overlap so merge
8. $S_2 = S_2 - \{([l_{12}, u_{12}], [l_{22}, u_{22}], G_2)\}$;
9. l_{11} = minimum(l_{11}, l_{12}); u_{11} = maximum(u_{11}, u_{12});
10. l_{21} = minimum(l_{21}, l_{22}); u_{22} = maximum(u_{21}, u_{22}); % range of merged groups
11. $G_1 = G_1 + G_2$;
}
}
}
12. S3 =∅; % S3 is new graph with biased edges
13. ∀ (γ_{1I}, γ_{2J}, weight) ∈ G {
14.if ∃ (($[l_{11}, u_{11}]$, $[l_{21}, u_{21}]$, G_1) ∈ S
where (($l_{11} - b \le I \le u_{11} + b$) && ($l_{21} - b \le J \le u_{21} + b$)))
15. weight = weight * (1 + f);
16. else weight = weight * (1 – f);
17. $S_3 = S_3 + \{(\gamma_{1I}, \gamma_{2J}, weight)\}$;
}
18. S_1 = maximal matching on graph S_3;
19. else I = I + 1;
}

Figure 10. An algorithm for the Hungarian method variant to identify orthologues and gene-groups

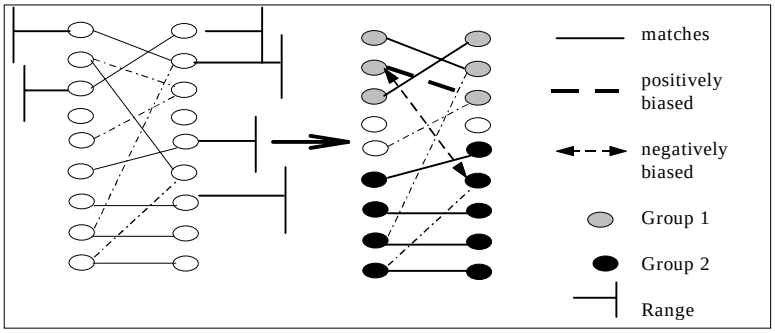

Figure 11. Merging gene-groups in the Hungarian variant method

7. IDENTIFYING FUSED GENES

A fused gene-group has two genes in one genome which match with different portions of a gene in the other genome as shown in Figure 12.

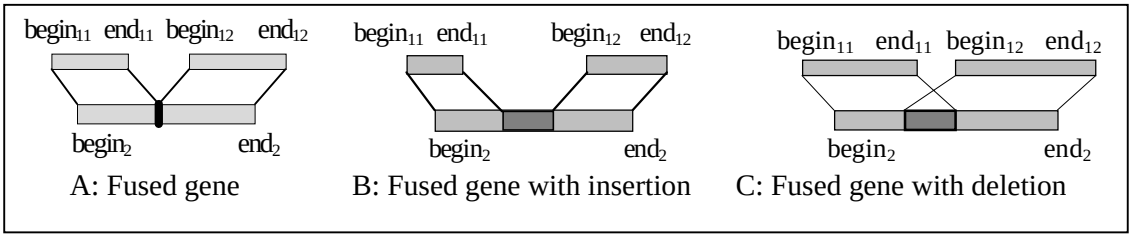

Figure 12. Types of fused genes

Let us assume that two consecutive genes γ_{1I} and $\gamma_{1(I+1)}$ fuse to form a new fused gene γ_{2J}. The following set of rules (1A) to (1D) is used to identify exact gene-fusion (see Figure 12A). The set of rules (2A) to (2D) is used to identify fused genes with insertions (see Figure 12B). The set of rules (3A) to (3D) is used to identify fused genes with deletions (see Figure 12C). The rationale for ε and δ is that there may be some mismatch allowed due to imprecision involved in the sequence alignment process or sequencing errors. Equation 1D is derivable from the application of 1A, 1B, and 1C; Equation 2D is derivable from the application of Equations 2A, 2B, and 2C; and Equation 3C is derivable from the application of Equations 3A, 3B, and 3C .

$$length(\gamma_{1I}) + length(\gamma_{1(I+1)}) - \varepsilon \le length(\gamma_{2J}) \le length(\gamma_{1I}) + length(\gamma_{1(I+1)}) + \varepsilon \quad (1A)$$

$$begin(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J})) - \delta \le begin(\pi_2(\gamma_{1I} \leftrightarrow \gamma_{2J})) \le begin(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J})) + \delta \quad (1B)$$

$$end(\pi_1(\gamma_{1(I+1)} \leftrightarrow \gamma_{2J})) - \delta \le end(\pi_2(\gamma_{1I} \leftrightarrow \gamma_{2J})) \le end(\pi_1(\gamma_{1(I+1)} \leftrightarrow \gamma_{2J})) + \delta \quad (1C)$$

$$end(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J}) - 2\delta - \varepsilon < begin(\pi_1(\gamma_{1(I+1)} \leftrightarrow \gamma_{2J})) \le end(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J})) + 2\delta + \varepsilon \quad (1D)$$

$$length(\gamma_{1I}) + length(\gamma_{1(I+1)}) + \varepsilon < length(\gamma_{2J}) \quad (2A)$$

$$begin(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J})) - \delta \le begin(\pi_2(\gamma_{1I} \leftrightarrow \gamma_{2J})) \le begin(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J})) + \delta \quad (2B)$$

$$end(\pi_1(\gamma_{1(I+1)} \leftrightarrow \gamma_{2J})) - \delta \le end(\pi_2(\gamma_{1I} \leftrightarrow \gamma_{2J})) \le end(\pi_1(\gamma_{1(I+1)} \leftrightarrow \gamma_{2J})) + \delta \quad (2C)$$

$$end(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J}) + 2\delta + \varepsilon < begin(\pi_1(\gamma_{1(I+1)} \leftrightarrow \gamma_{2J})) \quad (2D)$$

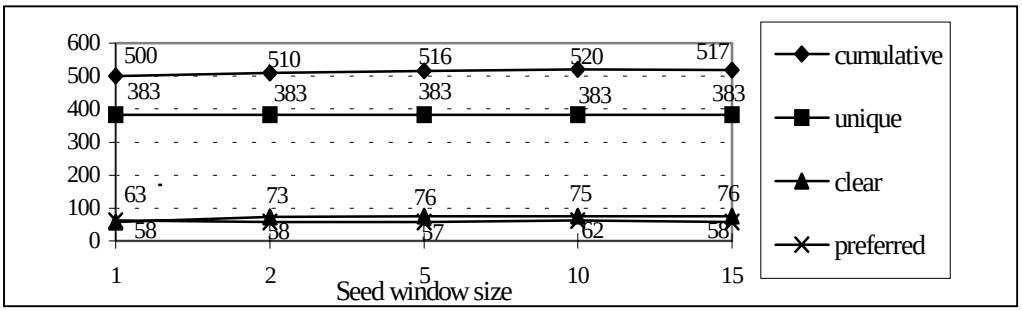

Figure 13 : Effect of seed window size on number of orthologues and corresponding gene-groups

$$length(\gamma_{1I}) + length(\gamma_{1(I+1)}) - \varepsilon > length(\gamma_{2J}) \quad (3A)$$

$$begin(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J})) - \delta \le begin(\pi_2(\gamma_{1I} \leftrightarrow \gamma_{2J})) \le begin(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J})) + \delta \quad (3B)$$

$$end(\pi_1(\gamma_{1(I+1)} \leftrightarrow \gamma_{2J})) - \delta \le end(\pi_2(\gamma_{1I} \leftrightarrow \gamma_{2J})) \le end(\pi_1(\gamma_{1(I+1)} \leftrightarrow \gamma_{2J})) + \delta \quad (3C)$$

$$end(\pi_1(\gamma_{1I} \leftrightarrow \gamma_{2J}) - 2\delta - \varepsilon > begin(\pi_1(\gamma_{1(I+1)} \leftrightarrow \gamma_{2J})) \quad (3D)$$

8. RESULTS AND DISCUSSION

Five genomes — partial *E. coli*, *H. influenzae*, partial *B. subtilis*, *M. genitalium*, and *M. jannaschii*, were compared pair-wise. These genomes were among the first to be sequenced. *E. coli* and *H. influenzae* belong to the gram-negative group of bacteria; *M. genitalium* and *B. subtilis* belong to the gram-positive group of bacteria; and *M. jannaschii* is an archaeon. We compared 1304 genes of *E. coli*[5] versus 1680 genes of *H. influenzae*, 1680 genes of *H. influenzae* versus 1852 genes of *B. subtilis*, 1852 genes of *B. subtilis* versus 1735 genes of *M. jannaschii*, 1852 genes of *B. subtilis* versus 472 genes of *M. genitalium*, and 472 genes of *M. genitalium* versus 1735 genes of *M. jannaschii*. *E. coli* and *H. influenzae* are phylogenetically very close, *H. influenzae* and *B. subtilis* are somewhere in-between, and *B. subtilis* and *M. jannaschii* are further away in evolution. The results were repeated with different seed window sizes for neighborhood proximity to see the variations in groupings. The results have been summarized in the following subsections.

8.1 Effect of Seed Window Size on Orthologues and Gene-groups. Figure 12 shows the effect of changing seed window size (for neighborhood proximity) on the number of orthologues. This experiment was run by comparing 1304 genes of *E. coli* and 1680 genes of *H. influenzae*. The graph shows that the number of orthologues saturates after the seed the window size of ten. The variation in the number of corresponding gene-groups and orthologues is less than 4 percent, which is well within tolerance limits, suggesting that the seed window size does not have a significant effect on the identification of orthologues or gene-groups. This is to be expected since our scheme takes care of variably sized groups dynamically. The only contributing effect is that the choice of larger window size marginally increases the number of grouped genes. As the genes inside the gene-groups are positively biased, some of the marginally shuffled genes in the proximity become *clear orthologues* and some of them become *preferred orthologues*.

[5] for historical reasons. Our first experiment was performed on 1304 genes of curate E. Coli present in EMBL database in 1995.

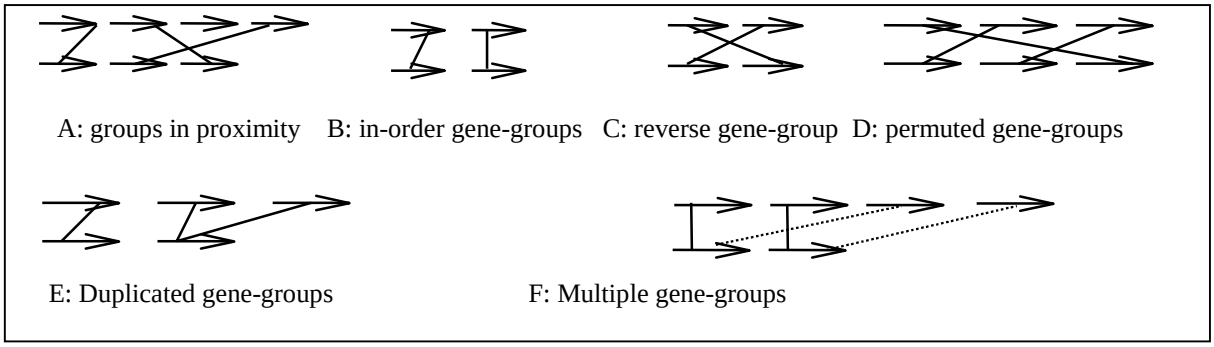

Figure 14. Different classification of corresponding gene-groups

8.2 Gene-group Classification and Findings. Using cross-species comparison, gene-groups have been classified into different categories: *in-order, reverse, permuted, fused, multiple*, and *duplicated*. *Fused gene-groups* can be further subclassified as *exact fusion*, *fusion with deletion*, and *fusion with insertion*. An *in-order* gene-group has the corresponding gene-groups in the same sequence as shown in Fig 14B. A *reverse-order* gene-group has the matching genes in one of the groups in ascending order, and the corresponding genes in the other group in descending order, as shown in Figure 14C. A *permuted gene-group* (see Figure 14D) has at least one matching gene-pair which does not preserve any order. A *duplicated group* has at least one gene in at least one of the groups which has been duplicated in the corresponding gene-group of another genome (see Figure 14E). A *multiple gene-group* has a subset of gene-group duplicated at more than one location in a genome (see Figure 14F). Multiple gene-groups include paralogous gene-groups which can not be clearly distinguished without use of the knowledge of metabolic pathways and biological reasoning.

8.3 A Comparative Study of Orthologues and Corresponding Gene-groups. Table 1 shows a statistical study of different types of corresponding gene-groups (in $\Gamma_1 \alpha \Gamma_2$): *single* denotes unique corresponding gene-groups, *multiple I* denotes multiple duplicated gene-groups in Γ_2 corresponding to a gene-group in Γ_1, *multiple II* denotes multiple duplicated gene-groups in Γ_1 corresponding to a gene-group in Γ_2, *duplicate I* denotes the gene-groups which have one or more duplicated genes in the corresponding gene-group in Γ_2 for a gene-group in Γ_1, *duplicate II* denotes one or more duplicated genes in the corresponding gene-group in Γ_1 for a gene-group in Γ_2, *fused I* denotes the gene-pairs in Γ_1 which are fused in Γ_2, and *fused II* denotes the gene-pairs in Γ_2 which are fused in Γ_1.

We counted the total number of orthologues as the sum of orthologues in three categories, namely, *unique, clear, and preferred*; *conflicting orthologues* need to be resolved by biological reasoning, and are not accounted for in our statistics. Under this criterion, 39.6 percent of *E. coli* genes (516 out of 1304 genes) are orthologous with *H. influenzae* genes, 24.5 percent of *B. subtilis* genes (453 out of 1852 genes) are orthologous with *H. influenzae* genes, 38.2 percent of *M. genitalium* genes (180 out of 472 genes) are orthologous with *B. subtilis* genes, 17 percent of *M. genitalium* (80 out of 472 genes) genes are orthologous with *M. jannaschii* genes. In addition, 24 percent of *E. coli* genes (313 out of 1304 genes) have unresolved matchings with *H. influenzae* genes, 8.8 percent of *B. subtilis* genes (163 out of 1852 genes) have unresolved matchings with *H. influenzae* genes, 5.9 percent of *M. genitalium* genes (28 out of 472 genes) have unresolved matchings with *B. subtilis* genes, 1.5 percent of *B. subtilis* genes (29 out of 1852 genes) have unresolved matchings with *M. jannaschii* genes, and 5.9 percent of *M. genitalium* genes (28 out of 472 genes) have unresolved

Table 1. Corresponding gene-groups in various genomes

	Partial EC - HI	Partial BS - HI	Partial BS - MJ	Partial BS - MG	MG - MJ
Single	88	60	22	18	11
Multiple I	82	45	12	22	9
Multiple II	63	50	14	18	7
Duplicated I	170	90	21	38	38
Duplicated II	193	92	21	48	37
Fused I	4	3	0	0	0
Fused II	0	0	0	0	0

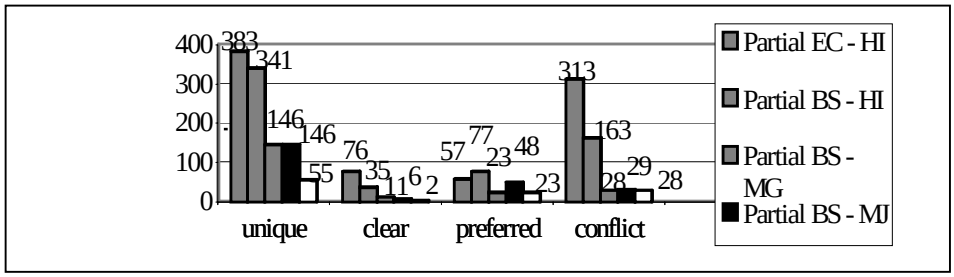

Figure 15. A comparative study of orthologues in multiple genomes

matchings with *M. jannaschii* genes. The phylogenetically close microbes (Olsen et. al, 1994), such as *E. coli* versus *H. influenzae* and *B. subtilis* versus *M. genitalium*, share more orthologues compared to microbes further away such as *B. subtilis* versus *H. influenzae*. Evolutionary distant genome pairs (Olsen et. al, 1994) such as *B. subtilis* versus *M. jannaschii* or *M. genitalium* versus *M. jannaschii*, have a much smaller percentage of orthologous genes.

The statistics in Table I show that as the phylogenetic distance increases, the number of corresponding gene-groups decreases. *M. jannaschii* shows the least similarity. Phylogenetically close genomes such as *E. coli* and *H. influenzae* share the maximum number of corresponding gene-groups, including a very large number of duplicated gene-groups, suggesting duplication as a common mechanism of evolution. The normal size of gene-groups is 2 to 5 with the maximum size as large as 27 in case of rRNA proteins between *E. coli* and *H. influenzae*. Fused genes are present in genome-pairs *E. coli* and *H. influenzae*, and *H. influenzae* and *B. subtilis*.

Orthologues statistics in Table I and Figure 16 shows that phylogenetically close genome pairs tend to have a higher percentage of orthologues in the corresponding gene-groups while phylogenetically distant genome pairs have a lower percentage of orthologues in the corresponding gene-groups, suggesting more shuffling as the phylogenetic distance increases. For phylogenetically close genomes *E. coli* and *H. influenzae*, the similarity scores of the *unique orthologues* varied from 49 to 98 with a average around 76. For phylogenetically distant genome pairs the similarity scores of *unique orthologues* varied from 45 to 70 with an average score of 55.

A comparison of orthologues of *H. influenzae*, *B. subtilis*, and *M. jannaschii* suggested that there are 104 orthologues[6] which are common in all three. There were 94 orthologues shared between *B.*

[6] Sum of number of unique orthologues + number of clear orthologues + number of preferred orthologues

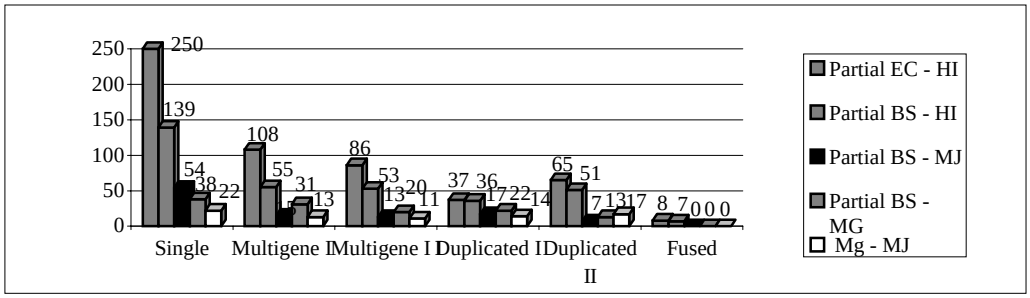

Figure 16. A comparative study of orthologues genes in corresponding gene-groups

subtilis and *M. jannaschii* which were not orthologous in *B. subtilis* and *H. influenzae*, and there were 349 orthologues shared between *B. subtilis* and *H. influenzae* which were not orthologous in *B. subtilis* and *M. jannaschii*. Comparing with grouped-orthologues in all three suggests that significant shuffling takes place in all three genomes.

A comparison of the orthologues of *M. genitalium*, *B. subtilis*, and *M. jannaschii* suggested that 44 orthologues are common in all three. There are 136 orthologues shared between *B. subtilis* and *M. genitalium* which do not occur in *M. jannaschii*. There are 154 orthologues shared between *B. subtilis* and *M. jannaschii* which do not occur in *M. genitalium*. There are 36 orthologues shared between *M. genitalium* and *M. jannaschii* which are absent in *B. subtilis*.

Out of 104 common orthologues shared by *H. influenzae* and *B. subtilis* and *M. jannaschii*, 25 orthologues were also found in *M. genitalium*. With the knowledge that *M. genitalium* is a very small genome, we conclude that these 25 genes are relatively conserved. A detailed comparative study of differences and conserved orthologues is outside the scope of this paper, and will be presented separately.

9. CONCLUSION

In this paper, we have presented a technique to model pair-wise comparison of two genomes as a bipartite graph matching problem. We have presented two different approaches and algorithms to identify the best-matching edges and gene-groups. The first approach finds the putative gene-groups first and then positively biases the edges between putative gene-groups to identify different classes of orthologues and corresponding gene-groups. The second approach is a variation of the well-known Hungarian method to identify the gene-groups and orthologues iteratively until no better groups can be found. The second approach also uses heuristics to positively bias the edges starting within a group. Both the approaches give very similar results.

The study of pair-wise comparisons of five genomes suggests that evolutionary distant organisms share less numbers of common orthologous genes and corresponding gene-groups. Duplication is a common mechanism for evolution. Evolutionary close genomes share a high percentage of duplicated genes and some fused genes, while evolutionary distant genomes share a smaller percentage of duplicated genes and have no fused genes. As the evolutionary distance increases, the shuffling of orthologous genes increases. In addition to conserved paralogues, there are around 25 orthologues which are present in all five of the genomes.

ACKNOWLEDGEMENTS

This research was initiated during the first author's visit at EMBL during Fall 1995. The first author acknowledges lively discussions with the colleagues at EMBL, especially with Chris Sander and his group during Fall 1995.

REFERENCES

1. Alberts, B., D. Bray, J. Lewis, M. Raff, K. Roberts, and J. D. Watson (1983), "Molecular Biology of THE CELL," Publishers: Garland Publishing, Inc.
2. Almgren, J., J. Anderson, S. Anderson et. al. (1995) "Sicstus 3 Prolog Manual," Swedish Institute of Computer Science
3. Altschul, S. F., W. Gish, W. Miller, E. W. Myers, D. J. Lipman (1990), "Basic Alignment Search Tools," J. Mol. Biol., 215: 403-410
4. Bult, C. J., O. White, G. J. Olsen et. al. (1996), "Complete Genome Sequence of the Methanogenic Archaeon, Methanococcus jannaschii," Science, 273: 1067–1044
5. Devereux, Haberli, and Smithies (1984), "A Comprehensive Set of Sequence Analysis Program for Vax," Nucleic Acid Research, 12(1): 387–395
6. Fitch, W. M. (1970) "Distinguishing Homologous from Analogous Proteins," Systematic Zoology, pp. 99 – 113
7. Fleischmann, R. D., M. D. Adams, O. White et. al. (1995) , "Whole-Genome Random Sequencing and Assembly of Haemophilus influenzae Rd," Science, 269: 496–512
8. Fraser, C. M, J. D. Gocayne, O. White et. al. (1995), "The Minimal Gene Complement of Mycoplasma Genitalium," Science, 270: 397–403
9. Koonin, E. V., R. L. Tatusov, and K. E. Rudd (1995), "Sequence Similarity Analysis of Escherichia coli Proteins: Functional and Evolutionary Implications," Proc. of Natl. Acad. Sci., USA, 92: 11921–11925
10. Koonin, E. V., A. R. Mushegian, M. Y. Galperin, and D. R. Walker (1997), "Common and Distinctive Features of Archeal and Bacterial Genomes Revealed by Computer Analysis of Protein Sequences," to appear in Molecular Microbiology
11. Labedan, B. and M. Riley (1995), "Widespread Protein Sequence Similarities: Origin of E. coli Genes," Journal of Bacteriology, 177: 1585–1588.
12. Mushegian, A. R. and E. V. Koonin (1996), "A Minimal Gene Set for the Cellular Life Derived by Comparison of Bacterial Genomes," Proc. Natl. Acad. Sci. USA, 93: 10268 – 10273
13. Olsen, J., C. R. Woese, and R. Overbeek (1994), "The Winds of Evolutionary Change: Breathing New Life into Microbiology," Journal of Bacteriology, Vol. 176, 1:1–6
14. Papadimitrou, C. H., and K. Steiglitz (1982) "Combinatorial Optimization: Algorithm and Complexity," Publisher: Prentice Hall
15. Setubal, J. and J. Meidanis (1997), "Introduction to Computational Biology," PWS Publishing Company
16. Tatusov, R. L., A. Mushegian, P. Bork et. al., (1996), "Metabolism and Evolution of Haemophilius Influenzae Deduced From a Whole-Genome Comparison with Escherichia Coli," Current Biology, Vol. 6, 3: 279–291
17. Waterman, M. S. (1984), "General Methods for Sequence Comparison," Bull. Math. Biol., 46: 473–500
18. Waterman, M. S.(1995), "Introduction to Computational Biology: Maps, Sequences, and Genomes," Publishers: Chapman & Hall