COMPARATIVE GENOMICS ASSIGNMENT

Rhodobacteraceae bacterium HIMB11 and Shuttleworthia satelles

Jiangli Gui (r0867437)

Table of Contents

1 Introduction

In this study, a comparative genomics analysis was performed on the *Rhodobacteraceae bacterium HIMB11*, abbreviated as rb, and the *Shuttleworthia satelles (DSM 14600)* species, abbreviated as ss. To identify sets of orthologous genes, the Best Bidirectional Hit (BBH) method was used. In addition, in-paralog and co-orthology relationships were also identified. Two proteins, ERL96417.1 (from the *Rhodobacteraceae bacterium*) and EEP27567.1 (from *Shuttleworthia satelles*), which function as DNA-directed RNA polymerase subunits, were found to have the highest bitscores and were selected for generating phylogenetic tree with 27 other species that were identified using BLASTp and had high similarity to ERL96417.1 protein. The 16s rRNA sequence information for these 29 species was collected to generate a species tree and identify speciation and duplication events. Conservation sites were identified among the 29 aligned protein sequences using both Shannon entropy and Simpson diversity. The nucleotide sequences of these 29 species were retrieved, the same as the previous procedure, and the conservation rate for each site were calculated to identify the promoter regions of the interested gene.

2 Methods

2.1 Dara sources and tools

Data:

The whole genome data for the *Rhodobacteraceae bacterium HIMB11* (GenBank assembly accession: GCA_000472185.1) and *Shuttleworthia satelles DSM 14600* (GenBank assembly accession: GCA_000160115.1) were obtained from the NCBI database (https://www.ncbi.nlm.nih.gov/assembly/).

The protein sequence ERL96417.1 from the *Rhodobacteraceae bacterium HIMB11* was used in a BLASTp search (https://blast.ncbi.nlm.nih.gov/Blast.cgi?PAGE=Proteins) to identify orthologous genes in other species. The protein sequences of the first 27 identified species were collected for multiple sequence alignment. The 16s rRNA sequences for these 29 species (2 interests + 27 identified species) were downloaded from the NCBI database (https://www.ncbi.nlm.nih.gov/refseq/targetedloci/16S_process/) and the SILVA database (https://www.arb-silva.de/). EnsemblBacteria (https://bacteria.ensembl.org/index.html) and BV-BRC (https://www.bv-brc.org/) databases were used to collect the nucleotide sequences of orthology genes.

The accession numbers for the orthologous proteins, 16s rRNA, corresponding strain information (if available), and the genomic sequences, can be found in the appendix.

Software:

The BLAST+ (version 2.7.1-foss-2018a) tool was used on the VSC platform to identify orthologs and in-paralogous for the two interested protein sequences. Python scripts were developed to identify the Best Bidirectional Hit (BBH) between the two species and the in-paralogous. The Shannon entropy and Simpson diversity were also calculated using Python (version 3.8). Multiple sequence alignment and the generation of phylogenetic and species trees were performed using SeaView 5.0 (https://doua.prabi.fr/software/seaview). The FigTree software (version v1.44,

http://tree.bio.ed.ac.uk/software/figtree/) was used for tree visualization and modification.

2.2 Compute orthologs using Best Bidirectional Hits

Files preparation:

Two FAA files for the *Rhodobacteraceae bacterium* and *Shuttleworthia satelles* were input into BLAST+ on the VSC platform. These files were used as the reference sequences (as databases) and the other file was used as the query sequence. To reduce computational cost, the threshold for the e-value was set to 1e-6 to discard unnecessary information. The Smith-Waterman alignment method was also used to determine the similarity between the two genomes (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4094424/). The outfmt 6 format was chosen for the output of BLAST+ as it is a tab-separated values file that allows for the selection of specific columns to include. Only the "qseqid," "sseqid," and "bitscore" columns were kept in the final output.

It is important to note that the Best Bidirectional Hit (BBH) method can only identify bidirectional hits and therefore cannot detect co-orthology relationships that may result from gene duplication events (in-paralogous). To avoid false negative results, a self-BLAST can be performed. However, this approach is not able to detect scenarios c and d. In this study, both the *Rhodobacteraceae bacterium* and *Shuttleworthia satelles* were subjected to a similarity check using strict thresholds (1e-50 e-value[1] and >50% identity[2]) to remove unnecessary information for later analysis.

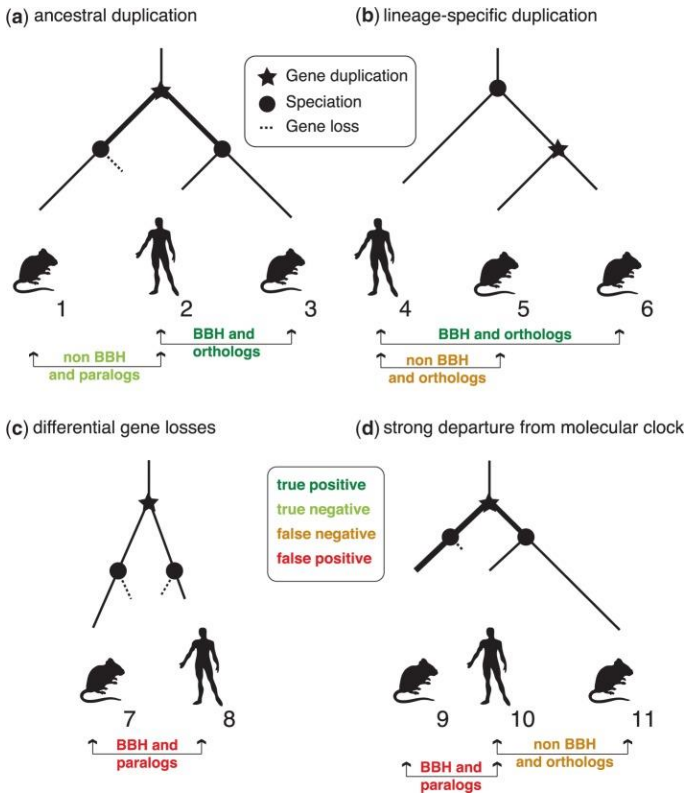


Figure 1. The performance of BBH in the 4 scenarios[3].

The commands used to generate genome comparisons and in paralogs gene pair files can be found in the appendix.

Calculate BBH in python:
After performing BLAST searches and filtering the results, the protein IDs with the highest bitscores were selected for each protein ID in one species, forming a pair. These two pair groups were considered the Best Bidirectional Hit (BBH) only if the bidirectional relationship held in both pair groups.

Identity in-paralogous groups:
Based on the results of the self-BLAST search with very strict selection criteria, the paired protein IDs were clustered using the modularity maximization algorithm (https://en.wikipedia.org/wiki/Modularity_(networks)). This approach was used because it is possible for more than two proteins to be in-paralogous.

Identity co-orthologous groups:
It is not necessarily the case that all protein IDs in-paralogous occur in BBH groups, so only those in BBH were used to enrich the co-orthology information by searching the protein IDs in BBH groups. Those protein IDs that were not in the BBH group but were in-paralogous with the protein ID being searched were selected to generate new pairs. It is important to note that the *Rhodobacteraceae bacterium* protein IDs were typically stored as the keys in a dictionary, while the *Shuttleworthia satelles* protein IDs were stored as the values. For the dictionary of co-orthology groups formed from the ss proteins, the key-value pairs were reversed because a dictionary cannot contain the same key more than once and the old value for the paired key would be replaced by the new value, which could cause problems for multiple in-paralog relationships.

2.3 Check orthology with a phylogenetic tree

Orthologs selection:
The pair with the highest bitscore in the BBH groups was ERL96417.1 and EEP27567.1. Both of these proteins are DNA-directed RNA polymerases that function in RNA synthesis from a DNA template. The sequence of ERL96417.1 was loaded for the BLASTp search. The top 27 species with the highest similarity to the protein sequence of the Rhodobacteraceae bacterium (rs) were collected for multiple sequence alignment.

Phylogenetic tree:
The Clustalo algorithm was used for multiple sequence alignment. Given that the sequences are highly similar, the BLOSUM62 substitution matrix was used to generate a phylogenetic tree. This matrix assigns higher scores to the alignment of amino acids that are similar in chemical properties and evolutionary conservation, based on the maximum likelihood method with 100 times bootstrap.

Species tree:
The 16s rRNA sequence information for the previous 29 species was collected from the NCBI and SILVA databases and grouped into a single file. The Clustalo algorithm was then used for multiple sequence alignment. Since these are nuclear sequences, the General Time Reversible (GTR) matrix was used to construct a species tree based on the maximum likelihood method with 100 times bootstrap.

Identify Speciation and Duplication events:
Based on the bootstrap value and the reconciliation between phoylogenetic tree and species tree, the speciation and duplication events were annotated on the phylogenetic tree.

2.4 Identify functional regions

The Bio.AlignIO package in Python was used to read the sequence alignment results, which were then converted into a matrix with each element storing a letter or a gap. Both the Simpson Diversity and Shannon Entropy methods were used to measure the conservation of homologous sequences, and the results were plotted to better visualize the conserved regions.

Simpson Diversity (https://geographyfieldwork.com/Simpson'sDiversityIndex.htm):

$$D = 1 - (\sum n(n-1)/N(N-1))$$

(n = # of individuals of a single amino acids; N = # of individuals in total frequencies)

Shannon Entropy (https://www.sciencedirect.com/science/article/pii/B9780128201251000142):

$$S(X) = -\sum_{i=1}^{N} p(x_i)\log_2(p(x_i))$$

($p(x_i)$ is the probability of obtaining the value)

Both the Simpson Diversity and Shannon Entropy scores were calculated at each site. Those with a zero score were considered conserved sites. The Protein Variability Server (http://imed.med.ucm.es/PVS/) and the NCBI Conserved Domains database (https://www.ncbi.nlm.nih.gov/Structure/cdd/wrpsb.cgi) was used to identify conserved regions.

2.5 Identify promoter regions

The corresponding nucleotide sequences of these orthologous proteins from 29 species were found in the EnsemblBacteria and BV-BRC databases. They were integrated for multiple sequence alignment. Using the same procedure as the functional region identification, the conservation rate of each site was calculated and plotted in figures to better visualize the distribution of potential promoter regions, which are expected to be highly conserved.

3 Results

3.1 Orthologs

The *Rhodobacteraceae bacterium HIMB11* has 3183 protein-coding genes, while the *Shuttleworthia satelles DSM 14600* has 1854 protein-coding genes. Among these genes, 564 gene pairs were identified using the Best Bidirectional Hit (BBH) method. To further enrich the co-orthology information, the in-paralog genes of the *Rhodobacteraceae bacterium HIMB11* and ss were collected and matched against the BBH sets. During this process, it was observed that in the BBH set, the pairs (ERL98362.1, EEP27403.1) and (ERL98677.1, EEP27674.1) were found, while the in-paralog analysis revealed that the protein ERL96553.1 was in-paralogy with both ERL98677.1 and ERL98362.1. However, there was no in-paralogy information between any of these proteins (ERL98677.1, ERL96553.1, or ERL98362.1) and EEP27674.1, but all three were in-paralogous with EEP27403.1. Upon checking the original in-paralog pairs in the *Rhodobacteraceae bacterium HIMB11*, it was found that the identity between EEP27674.1 and EEP27403.1 was 48%, which is almost there but excluded by our filtering before but they could still be considered in-paralogous. However, it was not useful to add this pair back into the in-paralog groups, as the dictionary structure cannot have duplicate keys, even with different values.

A similar phenomenon was observed for another four proteins. It was found in the BBH set that the pairs (ERL98203.1, EEP28013.1) and (ERL99007.1, EEP28227.1) were present, while EEP28013.1 and EEP28227.1 were in-paralogous with 68% identity. Due to the limitations of the dictionary structure, some co-orthology information was lost.
As a result of excluding these IDs from the BBH, 11 additional co-orthologous groups (one protein only had one paralog) were found. The protein IDs of BBH and co-orthologous can be found in the attachment.
These two special scenarios were illustrated in figures to better understand their relationships.

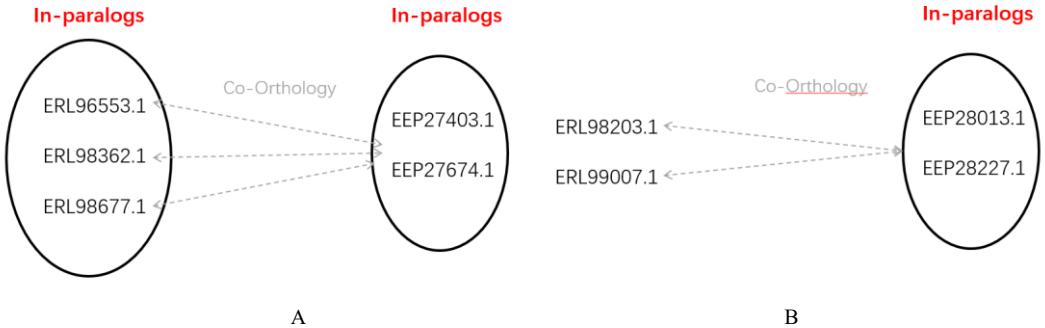


Figure 2. Complicated co-orthologous relationships. (A) protein 'ERL98677.1', 'ERL96553.1', 'ERL98362.1' are in-paralogous from *Rhodobacteraceae bacterium HIMB11*, and 'EEP27403.1' and 'EEP27674.1' proteins are in-paralogous from *Shuttleworthia satelles DSM 14600*. Each protein is co-orthologous with all the proteins (listed in the pictures) in another species (B) protein 'EEP28227.1', 'EEP28013.1' are in-paralogous from *Shuttleworthia satelles DSM 14600*, while 'ERL98203.1' and 'ERL99007.1' have no in-paralogous relationships. Either 'ERL98203.1' or 'ERL99007.1' are co-orthologous with these two proteins from *Shuttleworthia satelles DSM 14600*

3.2 Phylogenetic and species tree with speciation and duplication identification

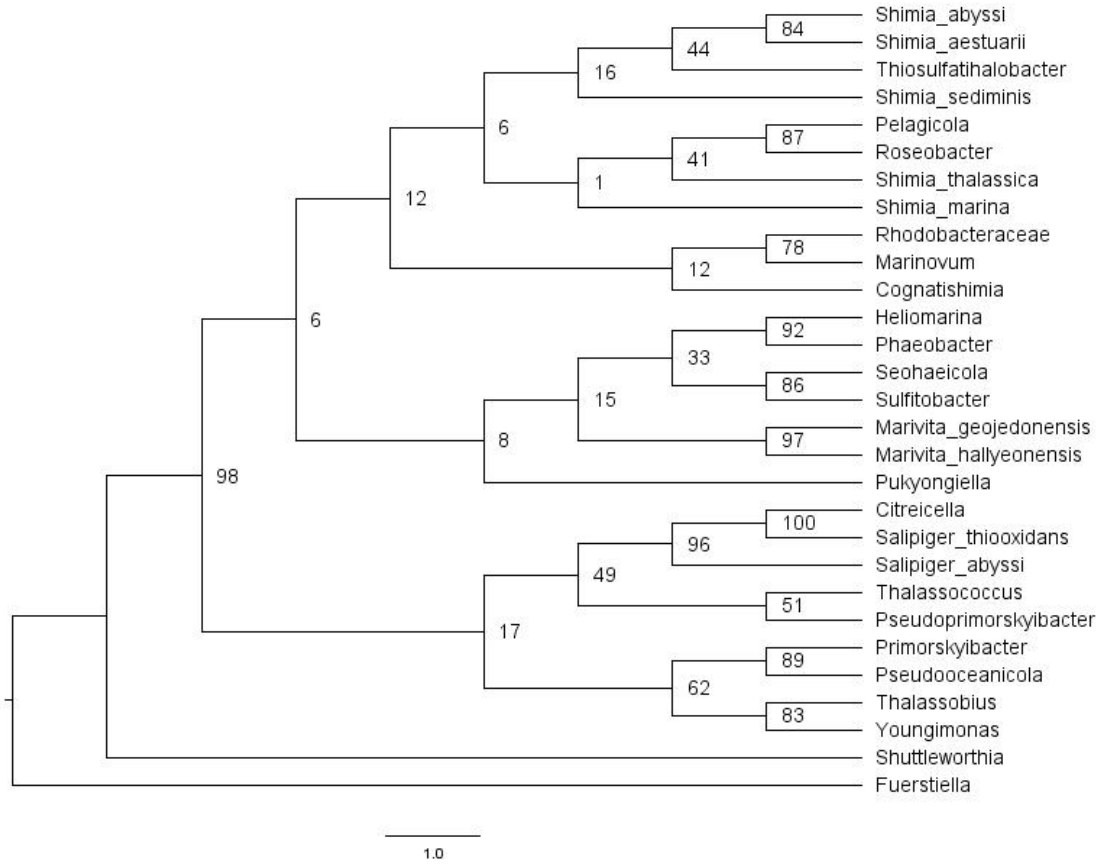


Figure 3. the species tree of these 29 species, the full name, and corresponding accession number for the 16s rRNA sequences can be found in the appendix.

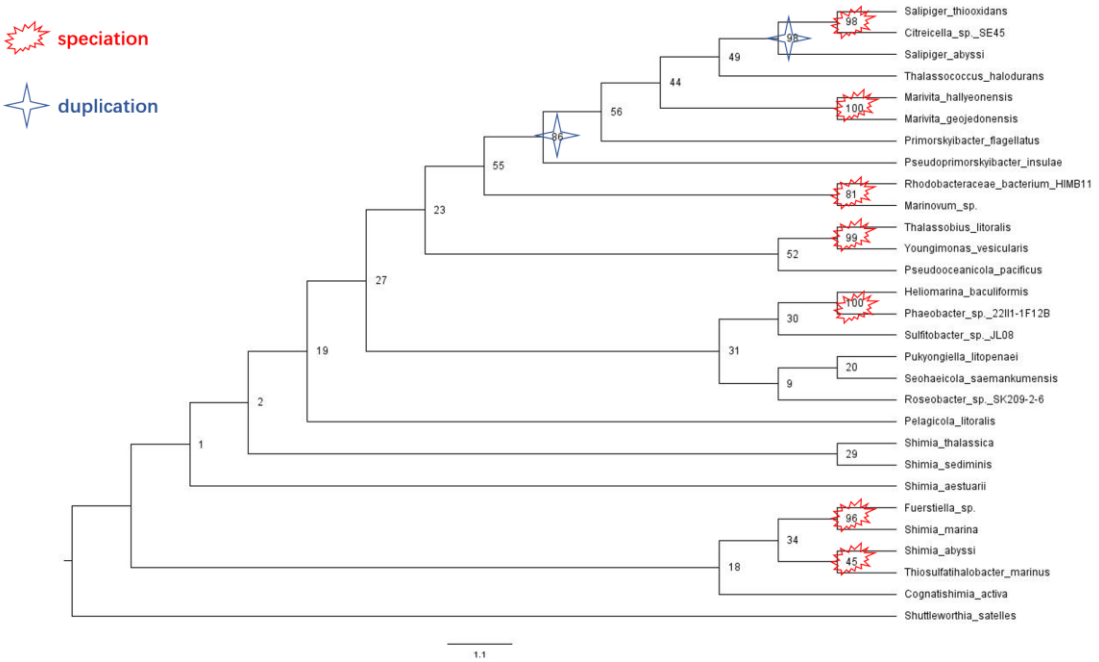


Figure 4. the phylogenetic tree of these 29 species with annotated speciation or duplication events.

The bootstrap value indicates the confidence of the tree topology, and in this case, only values higher than 80% were considered to be strongly supported and meaningful. Speciation events were marked with a red explosion, while duplication events were marked with a blue star.

3.3 Conserved regions identification

The multiple sequence alignments results were transferred into a matrix, and both Simpson Diversity and Shannon Entropy were applied for each column, the conservation rated versus each location were plotted below:

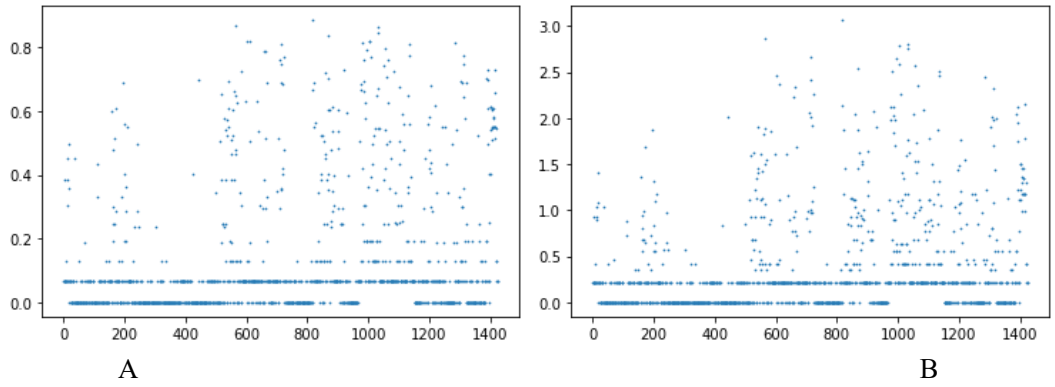
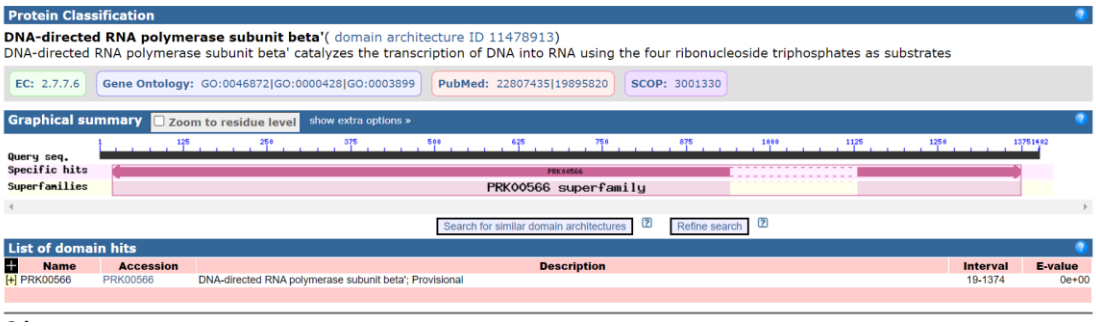


Figure 5. Simpson Diversity / Shannon Entropy score versus location for 29 protein sequences. A: Simpson Diversity B: Shannon Entropy

Other than the protein from *Shuttleworthia satelles DSM 14600*, the length of the rest of the proteins is approximately 1400bp. Based on Figure 5, it is evident that most sites of this protein are highly conserved overall. They are conserved at the beginning, almost from the very beginning to around 1000, and also at the end, from around 1160 to 1440. There is a flexible region between these two conserved regions.
Sites with zero Simpson Diversity/Shannon Entropy values account for 41.40% of the whole protein.



The sequences were input into the NCBI conserved domains database and all of them matched the PRK0056 superfamily, which is responsible for catalyzing the transcription of DNA into RNA. The pattern is similar to the two conservation rate plots.

With the help of the Protein Variability Server (PVS), two highly conserved regions were identified from 19 to 156 and from 242 to 442, with the following sequences:

Number	Start	End	Sequence
1	38	110	G E I K K P E T I N Y R T F K P E R D G L F C A R I F G P I K D Y E C L C G K Y K R M K Y R G V V C E K C G V E V T L Q K V R R E R M G H I E L A
2	306	424	V D A L F D N G R R G R V I T G A N K R P L K S L S D M L K G K Q G R F R Q N L L G K R V D F S G R S V I V T G P E L K L H Q C G L P K K M A L E L F K P F I Y S R L E A K G L S S T V K Q A K K L V E K E R P E V W D I L D E V I R E H P V

Table 1. The conserved fragments with a threshold <= 0.2

It was noted in the Uniprot protein database that positions 72, 74, 87, 90, 811, 885, 892, and 895 are Zn^{2+} binding sites, while 463, 465, and 467 are Mg^{2+} binding sites. Magnesium and zinc ions play important roles in the activity of DNA-directed RNA polymerase[4,5]. $Mg(^{2+})$ ions help to stabilize the structure of the enzyme and are required for its activity. $Zn(^{2+})$ ions also play a structural role and are involved in the catalytic mechanism of the enzyme.

3.4 Promotor regions identification

With the help of the Uniprot protein, EnsemblBacteria, and BV-BRC databases, 26 nucleotide sequences for coding these orthologous proteins were found, with 3 failures. Except for the gene from *Shuttleworthia satelles DSM 14600*, which is called rpsS, the rest of the 25 genes have the same name, rpoC. The significant difference between *Shuttleworthia satelles DSM 14600* and the others is also apparent in the length of the genes. The former has only 282 bps, while the others have more than 4000 bps.

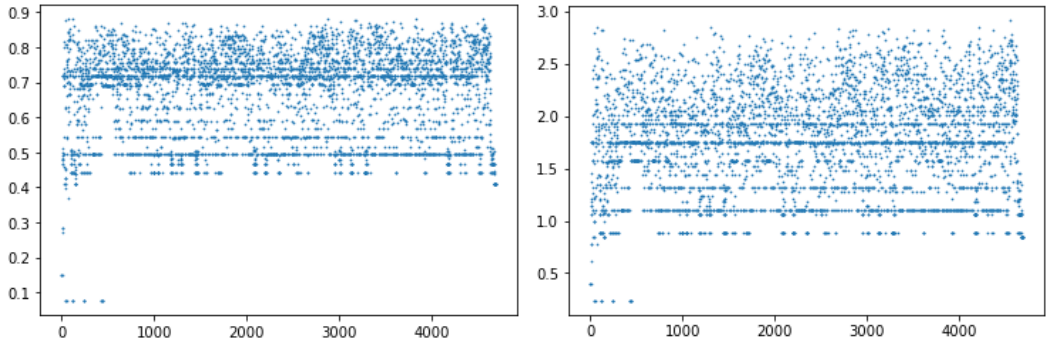


Figure 6. Simpson Diversity / Shannon Entropy score versus location for 26 nucleotide sequences. A: Simpson Diversity B: Shannon Entropy

The same procedure was applied to these 26 sequences using a Python script that calculates the conservation rate of each site. It is evident that these sequences are more divergent. The conservation rate of the percentage of sites is 18.68% using the Simpson Diversity method with a threshold of 0.5 and 4.79% using the Shannon Entropy method with a threshold of 1.0. Since the gene rpsS shows significant differences from the others, it was excluded and the remaining 25 sequences underwent the previous analysis again.

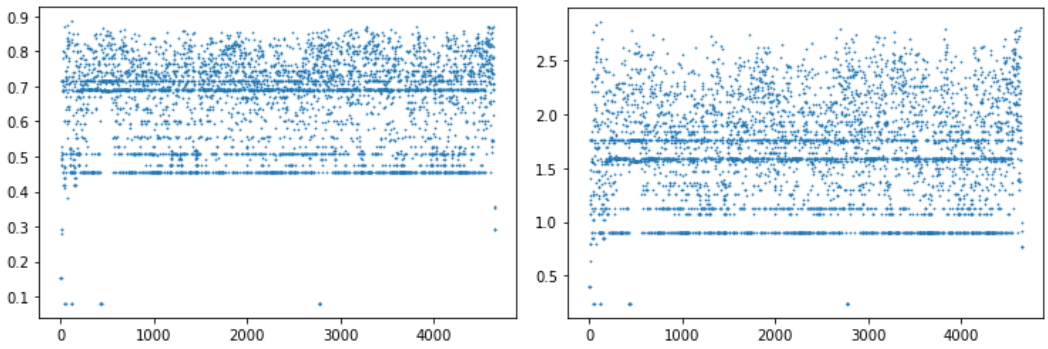


Figure 7. Simpson Diversity / Shannon Entropy score versus location for 25 nucleotide sequences. A: Simpson Diversity B: Shannon Entropy

However, no significant improvement was found with a conservation rate of 17.97% using the Simpson Diversity method and 14.25% using the Shannon Entropy method. Therefore, no promoter regions were found using this method.

Conclusion

To summarize, this comparative genomics study identified 564 pairs of orthologous proteins and 11 additional co-orthologous groups between *Rhodobacteraceae bacterium HIMB11* and *Shuttleworthia satelles DSM 14600*. The phylogenetic tree and species tree were built using these proteins and additional species were found via BLASTp, and speciation and duplication events were annotated on the phylogenetic tree. Further analysis of the DNA-dependent RNA polymerase proteins revealed two large conserved regions, which were confirmed to belong to the PRK0056 superfamily by the NCBI database. However, no highly conserved regions were found in the nucleotide sequences coding for these proteins.

Limitations and future work:

It is possible that using a different data structure, such as a list of tuples or using tuples as keys in a dictionary, could better suit the needs of storing complicated co-orthologous relationships. However, it is important to carefully consider the benefits and drawbacks of each data structure before making a decision.

It is also possible that the lack of conserved regions in the promoter region identification is due to incorrect input data or an incomplete dataset. It may be helpful to try different methods for obtaining nucleotide sequence data or to consider using additional resources to identify potential promoter regions. It is also important to carefully consider the limitations of the methods and tools used in the analysis and to carefully evaluate the results obtained.

Reference

(1) Ambrosino, L.; Ruggieri, V.; Bostan, H.; Miralto, M.; Vitulo, N.; Zouine, M.; Barone, A.; Bouzayen, M.; Frusciante, L.; Pezzotti, M.; Valle, G.; Chiusano, M. Multilevel Comparative Bioinformatics to Investigate Evolutionary Relationships and Specificities in Gene Annotations: An Example for Tomato and Grapevine. *BMC Bioinformatics* **2018**, *19*. https://doi.org/10.1186/s12859-018-2420-y.

(2) Pérez-Bercoff, Å.; Makino, T.; McLysaght, A. Duplicability of Self-Interacting Human Genes. *BMC Evol. Biol.* **2010**, *10* (1), 1–8.

(3) Dalquen, D. A.; Dessimoz, C. Bidirectional Best Hits Miss Many Orthologs in Duplication-Rich Clades Such as Plants and Animals. *Genome Biol. Evol.* **2013**, *5* (10), 1800–1806.

(4) Nelson, D. L.; Lehninger, A. L.; Cox, M. M. *Lehninger Principles of Biochemistry*; Macmillan, 2008.

(5) Alberts, B.; Johnson, A.; Lewis, J.; Morgan, D.; Raff, M.; Roberts, K.; Walter, P. Molecular Biology of The Cell. New York: Garland Science, 2014.

Appendix

Meta information for 29 species with name, accession number for interested proteins, 16s rRNA, specific strain, protein corresponding gene, and its accession number for nucleotide sequences.

	Species name	Orthology proteins	16S ribosomal RNA	Strain	gene name	gene accession
2	**Rhodobacteraceae bacterium HIMB11**	ERL96417.1	AVDB01000023.451.1903		rpoC	AVDB01000016
3	**Shuttleworthia satelles DSM 14600**	EEP27243.1	ACIP02000001.771359.772881		rpsS	ACIP02000010
4	Marinovum sp.	MBT4872449.1	NZJQ01000038.21173.21628	strain SI074_bin43	rpoC	JABJIL010000255
5	Thiosulfatihalobacter marinus	WP_197916664.1	NR_181537.1	strain GL-11-2	rpoC	JAECSA010000002
6	Shimia aestuarii	WP_243611561.1	AY442178.1.1386		rpoC	SAMN04488042_11619
7	Thalassobius litoralis	WP_142492825.1	NR_113362.1	strain KU5D5		
8	Thalassococcus halodurans	WP_103911467.1	DQ397336.1.1383		rpoc	SAMN04488045_3179
9	Pseudoprimorskyibacter insulae	WP_108884440.1	NR_144598.1	strain SSK3-2	rpoC	PRI8871_00330
10	Youngimonas vesicularis	WP_136338428.1	NR_134231.1	strain CC-AMW-E	rpoC	E7681_06400
11	Shimia thalassica	WP_058309627.1	NR_170514.1	strain CECT 7735 (R47)	rpoC	PH7735_00374
12	Salipiger thiooxidans	WP_206890740.1	NR_043058.1	strain CHLG 1	rpoC	SAMN04488105_1346
13	Shimia abyssi	WP_106610597.1	LC057677.1.1404		rpoC	CLV88_1297
14	Citreicella sp. SE45	EEX16655.1	ACNW01000104.72954.74395		rpoC	NZ_GG704598
15	Fuerstiella sp.	MCP4784058.1	NR_179021.1	Fuerstiella marisgermanici strain NH11	rpoC	Fuma_05726
16	Marivita hallyeonensis	WP_072780222.1	JF260872.1.1384		rpoC	SAMN05443551_0134
17	Marivita geojedonensis	WP_085641389.1	JN885198.1.1386		rpoC	MGEO_19280
18	Pseudooceanicola pacificus	WP_160383877.1	NR_174321.1	strain 216_PA32_1		
19	Sulfitobacter sp. JL08	WP_114868846.1	CP025815.289790.291257		rpoC	C1J05_02275
20	Cognatishimia activa	WP_265019690.1	OM267642.1	strain 2012CJ37-3	rpoC	TA5114_00953
21	Shimia sediminis	WP_127114788.1	NR_180011.1	strain ZQ172		
22	Shimia marina	WP_058239835.1	NR_043300.1	strain CL-TA03	rpoC	SHM7688_02071
23	Seohaeicola saemankumensis	WP_224990809.1	NR_044437.1	strain SD-15	rpoC	JAINWI010000030
24	Pelagicola litoralis	WP_138014979.1	NR_044158.1	strain CL-ES2	rpoC	FAP39_03445
25	Heliomarina baculiformis	WP_238373288.1	NR_181734.1	strain T40-3	rpoC	JAIMIC010000021
26	Primorskyibacter flagellatus	WP_084352001.1	NR_134108.1	strain DY470	rpoC	SAMN06295998_103436
27	Roseobacter sp. SK209-2-6	WP_008204005.1	AAYC01000001.206374.207813		rpoC	BPES01000011
28	Phaeobacter sp. 22II1-1F12B	WP_088635876.1	AQQP01000053.396.1864		rpoC	CP104100
29	Salipiger abyssi	WP_076696773.1	NR_148263.1	strain JLT2014	rpoC	Ga0080574_TMP1550
30	Pukyongiella litopenaei	WP_106472438.1	NR_173672.1	strain SH-1	rpoC	C6Y53_10660
31						
32					DNA-directed RNA polymerase subunit beta'	

Command to run BLAST+ in VSC to generate the files for orthology identification

```
module load BLAST+/2.7.1-foss-2018a

# upload two protein sequnce files into working directory, rename them into
'rb_protein.faa' (Rhodobacteraceae bacterium HIMB11) and 'ss_protein.faa'
(Shuttleworthia satelles)

### make the rb as the database
makeblastdb -in rb_protein.faa -dbtype prot

blastp -num_threads 2 -evalue 1e-6 -use_sw_tback -query ss_protein.faa -db
rb_protein.faa -outfmt "6 qseqid sseqid bitscore" -out ss_rb.txt

blastp -num_threads 2 -evalue 1e-50 -use_sw_tback -query rb_protein.faa -db
rb_protein.faa -outfmt "6 qseqid sseqid pident" -out rb_rb.txt

awk '$3 < 100 && $3 > 50 {print$1,$2}' rb_rb.txt > rb_inP.txt

### make the ss as the database
makeblastdb -in ss_protein.faa -dbtype prot
blastp -num_threads 2 -evalue 1e-6 -use_sw_tback -query rb_protein.faa -db
ss_protein.faa -outfmt "6 qseqid sseqid bitscore" -out rb_ss.txt

blastp -num_threads 2 -evalue 1e-50 -use_sw_tback -query ss_protein.faa -db
ss_protein.faa -outfmt "6 qseqid sseqid pident" -out ss_ss.txt
awk '$3 < 100 && $3 > 50 {print$1,$2}' ss_ss.txt > ss_inP.txt
```

'getBBH.py' script to find the orthology by BBH and also the in-paralogs and co-orthologs

```
import os
import networkx as nx
from networkx.algorithms import community
import matplotlib.pyplot as plt
import csv

print(os.getcwd())
os.chdir("D:/KU Leuven/Third semester/Comparative and Regulatory Genomics/Comparative genomics/Assignment")

def getPairs(filename):
    rows = []
    with open(filename, 'r') as f:
        for line in f:
            rows.append(line.strip())
    pairs = {}

    # Iterate over the rows of data
    for row in rows:
        # Split the row into columns
        cols = row.split()
        # Extract the values for the first and second columns and the bit score
        id1 = cols[0]
        id2 = cols[1]
        bitscore = float(cols[2])
        # Check if the unique ID is already in the dictionary
        if id1 in pairs:
            # If it is, check if the current bit score is higher
            if bitscore > pairs[id1][1]:
                # If it is, update the pair for the unique ID
                pairs[id1] = (id2, bitscore)
        else:
            # If the unique ID is not in the dictionary, add the current pair
            pairs[id1] = (id2, bitscore)

    return pairs

def get_BBH(pair1, pair2):
    bbh = {}
    rb_ss_pairs = pair1
    ss_rb_pairs = pair2
```

```
    for pairA in rb_ss_pairs:
        for pairB in ss_rb_pairs:
            if pairA == ss_rb_pairs[pairB][0]:
                if pairB == rb_ss_pairs[pairA][0]:
                    bbh[pairA] = pairB

    return bbh

def getInparalogs(filename):

    # Load the graph from a file
    G = nx.read_edgelist(filename)

    # Compute the clusters using the modularity maximization algorithm
    clusters = community.greedy_modularity_communities(G)

    inParalogs_groups = []

    # Transfer frozensets into normal sets
    for cluster in clusters:
        group = set(cluster)
        inParalogs_groups.append(group)

    return inParalogs_groups


def findCoOrthologs(inParalogs_groups, bbh):
    # Create a new dictionary to store the co-orthology information
    co_orthologs = {}

    # Iterate over the sets in the list
    for group in inParalogs_groups:
        # Iterate over the elements in the set
        for element in group:
            if element.startswith("ERL"):
                # Check if the element is a key in the bbh dictionary
                if element in bbh:
                    co_orthologs[element] = bbh[element]
                    # Iterate over the remaining elements in the group
                    for other_element in group:
                        if other_element != element:
                            co_orthologs[other_element] = bbh[element]
            elif element.startswith("EEP"):
                # Check if the element is a value in the bbh dictionary
```

```
                    # Normally we used "ERL" as key and "EEP" as value,
                    # While in this case, "EEP" are in-paralogs, so one "ERL"
                    # can have multiple values, while is not capable for the
                    # dictionary structure, so we reverse the key-value
                    # relationship here
                    for key, value in bbh.items():
                        if element == value:
                            co_orthologs[element] = key
                            # Iterate over the remaining elements in the group
                            for other_element in group:
                                if other_element != element:
                                    co_orthologs[other_element] = key

    return co_orthologs

def output(filename, dictionary):
    # create a file "co_orthology.tsv" to store co-orthologs information
    with open(filename, "w", newline="") as f:
        writer = csv.writer(f, delimiter="\t")
        for key, value in dictionary.items():
            writer.writerow([key, value])
    return

rb_ss_pairs = getPairs('rb_ss.txt')
ss_rb_pairs = getPairs('ss_rb.txt')

bbh = get_BBH(rb_ss_pairs, ss_rb_pairs)

# After checking the results in in-paralogs and co-orthologs, remove those special cases
del bbh['ERL98362.1']
del bbh['ERL98677.1']
del bbh['ERL99007.1']
del bbh['ERL98203.1']
print(len(bbh))

output("BBH.tsv", bbh)


rb_inParalogs = getInparalogs('rb_inP.txt')
ss_inParalogs = getInparalogs('ss_inP.txt')

rb_co_orthologs = findCoOrthologs(rb_inParalogs, bbh)
ss_co_orthologs = findCoOrthologs(ss_inParalogs, bbh)
```

```
co_orthologs = rb_co_orthologs.copy()
co_orthologs.update(ss_co_orthologs)

for key, value in co_orthologs.items():
    if key in rb_co_orthologs and key in ss_co_orthologs:
        # Key exists in both dictionaries, check if the values are the same
        if rb_co_orthologs[key] == ss_co_orthologs[key]:
            print(f'{key} has the same value in both dictionaries')
        else:
            print(f'{key} has different values in the two dictionaries')

output("co_orthology.tsv", co_orthologs)

co_orthologs_groups = getInparalogs("co_orthology.tsv")

# Two special cases are listed here, they are not included in the both BBH and co-orthology result files
# {'ERL98677.1', 'ERL96553.1', 'ERL98362.1'; 'EEP27403.1', 'EEP27674.1'}
# {'ERL98203.1', 'ERL99007.1'; 'EEP28227.1', 'EEP28013.1'}
```

'Conversion.py' script to transfer the multiple alignment sequences into a matrix with each element in the matrix storing one letter or a gap. Then the conservation rate of each column will be calculated by both Simpson Diversity and Shannon Entropy methods. Results also are visualized in scatter plots with a calculated percentage of conservation sites compared to the whole sequence.

```
import os
import numpy as np
from Bio import AlignIO
import matplotlib.pyplot as plt
import math
from collections import Counter


print(os.getcwd())
os.chdir("D:/KU Leuven/Third semester/Comparative and Regulatory Genomics/Comparative genomics/Assignment")

def read_File(filename,file_type):
    # Read the MSA file
    alignment = AlignIO.read(filename,file_type)

    # Get the number of sequences and the length of the alignment
    num_seqs = len(alignment)
    aln_len = alignment.get_alignment_length()

    # Create a matrix to store the data
    matrix = np.empty((num_seqs, aln_len), dtype=np.object)

    # Iterate over the rows and columns of the matrix
    for i, record in enumerate(alignment):
        for j in range(aln_len):
            matrix[i, j] = record.seq[j]

    return matrix

def get_Conservation(matrix):
    Simpson_Diversity = []
    Shannon_Entropy = []
    for i in range(len(matrix[0])):
        column = (matrix[:, i])
        # Count the number of occurrences of each category
        counts = Counter(column)
```

```
            N = len(column)
            D = 1 - sum(counts[x] * (counts[x] - 1) / (N * (N - 1)) for x in counts)
            H = -1 * sum(counts[x]/N * math.log2(counts[x]/N) for x in counts)
            Simpson_Diversity.append(D)
            Shannon_Entropy.append(H)
    return Simpson_Diversity, Shannon_Entropy

def get_Conservation_rate(list,threshold):
    count = 0
    for e in list:
        if e <= threshold:
            count += 1
    # print(count)
    conser_rate = count/len(list)
    # print(conser_rate)
    conser_rate = '%.2f%%' % (conser_rate * 100)
    return conser_rate


def visualization(sd,se,threshold1, threshold2):
    # plot
    x = np.array(range(0,len(sd)))
    plt.figure()
    plt.scatter(x, sd, s=0.5)
    plt.figure()
    plt.scatter(x, se, s=0.5)

    conser_rate = get_Conservation_rate(sd, threshold1)
    print("Consevation rate calculated by Simpson Diversity method", conser_rate)

    conser_rate = get_Conservation_rate(se, threshold2)
    print("Consevation rate calculated by Shannon Entropy method", conser_rate)

    return


# MSA for 29 protein sequences
matrix1 = read_File('29sequences.txt', 'fasta')
sd1, se1 = get_Conservation(matrix1)
visualization(sd1,se1,0,0)

# MSA for 26 nucleotide sequences for corresponding proteins
matrix2 = read_File('26promoter.txt', 'fasta')
sd2, se2 = get_Conservation(matrix2)
```

```
visualization(sd2,se2,0.5,1.0)

# MSA for 25 nucleotide sequences for corresponding proteins without Shuttleworthia satelles DSM
14600
matrix3 = read_File('25promoter.txt', 'fasta')
sd3, se3 = get_Conservation(matrix3)
visualization(sd3,se3,0.5,1.0)
```