2010

Insights into Genome Functional Organisation through the Analysis of Interaction Networks

Andre Masella
Wilfrid Laurier University

Your file Votre référence
ISBN: 978-0-494-75381-1
Our file Notre référence
ISBN: 978-0-494-75381-1

Insights into Genome Functional Organisation through the Analysis of Interaction Networks

by

Andre Masella

B.A.Sc., University of Waterloo, 2008

A

thesis

submitted to the

Department of Biology,
Faculty of Science

in partial fulfillment of the

requirements for the degree of

Master of Science

in

Integrative Biology

at

Wilfrid Laurier University

Waterloo, Ontario, Canada, 2010

Abstract

Using computational techniques to identify orthology and operon structure, it is possible to find functional interactions between genes, which, together, define the *genetic interactome* These large networks contain information about the relationships between phenotypes in organisms as genes responsible for related abilities are often co-regulated and reassorting of these genes can be detected in the operon structure. However, these networks are too large to analyse by hand In order to practically analyse the networks, a computational tool, `gisql`, was developed and, using this tool, the connectivity patterns in the genetic interactome can be analysed to understand high-level organisation of the genome and to narrow the list of candidate genes for wet lab analysis The many strains of *Escherichia coli* are interesting subjects as there are many sequenced strains and they show highly variable pathogenic abilities. Analysis shows that the pathogenic genes have a strong tendency to connect to genes ubiquitous in the *E coli* pan-genome The Rhizobiales, including *Sinorhizobium meliloti* and *Ochrobactrum anthropi*, are multi-chromosomal eukaryote-associated bacteria and a significant history of horizontal transfer Regions of the pSymB megaplasmid of *S meliloti* which cannot be deleted via transposon-targeted homologous recombination were shown to be significantly more connected to the main chromosome Targets for functional complementation of deletions in pSymB in *S meliloti* using genes from *O anthropi* were identified and unusual connectivity patterns of orthologs were identified Finally, a putative cytokinin receptor in the Rhizobiaceæ, likely involved in symbiosis with plant hosts, was identified Thanks to the flexibility of `gisql`, these analyses were straight-forward and fast to develop

Acknowledgements

I would like to thank Dr Gabriel Moreno-Hagelsieb and Gregory Vey for their help in the lab and my committee members Dr Brian Ingalls and Dr Robin Slawson The Rhizobiales analysis was done in conjunction with John Heil and Dr Trevor Charles with some initial input from Dr Scott Clark Additionally, the work on the putative cytokinin receptor was done in conjunction with John Heil, Dr Trevor Charles, Scott Clemow, and Dr Frédérique Guinel The metagenomic clustering work was done with Michael J Lynch and Eddie Yee Tak Ma and was used as a course project for CS 798 taught by Dr Dan G Brown using data provided by Dr. Josh D Neufeld For the sage advice on how the Java Virtual Machine works, I would like to thank Dan Heidinga (IBM)

I would like to thank Kathy Lam and Danielle Nash for technical and moral support I would also like to thank my parents Finally, I would like to acknowledge and blame Dr Maria Trainer as the voice of (in)sanity that encouraged me to be here

Contents

List of Tables

List of Figures

List of Computer Program Sources

Typographic Conventions

Abbreviations

CDS coding sequence

CMK 4-(cytidine 5′-diphospho)-2-*C*-methylerythritol kinase

CMT 2-*C*-methylerythritol 4-phosphate cytidyl transferase

COG cluster of orthologous genes [1]

DXP deoxyxylulose 5-phosphate

DXR deoxyxylulose 5-phosphate reductoisomerase

DXS deoxyxylulose 5-phosphate synthase

IPP isopentyl diphosphate

JVM Java Virtual Machine

MBPS 1-hydroxy-2-methyl-2-(*E*)-butenyl 4-diphosphate synthase

MECPS 2-*C*-methylerythritol 2,4-cyclodiphosphate synthase

NCBI National Center for Biotechnology Information

RBH reciprocal best hit

Symbols

$\forall$ – Universal quantifier ("for all") $(\forall x)P(x)$ states that $P(x)$ is true for every possible values of x

$\in$ – Membership operator ("is an element of") $x \in S$ states that x is a member of a set S

$\exists$ – Existential quantifier ("there exists") $(\exists x)P(x)$ states exists at least one x such that $P(x)$ is true

$\cap$ – Set-theoretic intersection $x \in (A \cap B)$ only if $x \in A$ and $x \in B$

$\cup$ – Set-theoretic union $x \in (A \cup B)$ if $x \in A$ or $x \in B$

Δ – Genomic deletion

$\vee$ – Boolean union ("or") $x \vee y$ states that either x, y, or both must be true.

$\wedge$ – Boolean intersection ("and"). $x \wedge y$ states that both x and y must be true.

$^\circ$ – Degree in a graph The x° neigbours of a node n are all the nodes with a path to n of length x

Chapter 1

General Introduction

THE OBJECTIVES of this research were to develop and construct software capable of manipulating functional interaction graphs, analyse connectivity patterns in the functional interaction graphs to glean information about high-level organisation of genes and use the interaction networks to find possible candidate genes, responsible for known phenotypes, for further analysis in wet-lab experiments

As model organisms, *Escherichia coli* and the Rhizobiales were selected A great deal of information about the genetics and metabolism of *E coli* is available as it is the standard bacterial model organism In fact, there are more completely-sequenced strains of *E coli* than of any other sequenced organism The Rhizobiales are a diverse group of organisms which include plant symbionts, plant pathogens, and animal pathogens. The diversity of lifestyle and, in the case of plant symbionts, of host specificity, make them interesting models

The major projects include

- the development of `gisql`, a software package to analyse functional interaction networks
- the analysis of *E coli* pathogenicity using connectivity patterns
- connectivity analysis of the pSymB megaplasmid in *Sinorhizobium meliloti*
- preparatory analysis of functional complementation targets in *S meliloti* using *Ochrobactrum anthropi*
- the analysis of the deoxyxylulose 5-phosphate (DXP) pathway in *E coli* as a model for biochemical pathway completion

- determining a candidate receptor in *S. meliloti* for the plant hormone cytokinin as a possible regulator of bacterial genes during symbiosis

1.1 Functional Genetic Interaction Networks

Sequence search-and-alignment algorithms, such as BLAST [2], are computational methods that can find pairs of similar genes between genomes. In prokaryotic genomes, operons are groups of functionally related genes transcribed as a single unit. After identifying orthologs, as explained below, the Nebulon software can computationally find these functional relations by using the operons of multiple organisms [3]. The kinds of functional interactions determined by Nebulon include physical protein-protein interactions, regulatory interactions, membership in a common pathway, or a transporter or chaperone and a substrate-using enzyme.

For a phenotype to exist, there must be an underlying cellular process that maintains it. If another organism lacks this cellular process, it will be unable to express the phenotype even under identical conditions. The cellular process is necessarily made up of chemical reactions mediated by genes and their derivatives (RNA, protein, and enzymatically-produced molecules) and the interactions between these genes, and their derivatives. Nebulon is capable of finding functional relations among protein-coding genes using their operon structures.

Insight can be gained into the phenotypes of two closely related organisms by comparing the Nebulon-derived [3] relations unique to each organism.

1.1.1 Definition of Orthology

Comparison of functional interaction networks requires an "equality" between genes of different organisms, orthology is used to provide this relation. Since detecting orthologs, genes diverging as a consequence of speciation events [4], can be time-consuming if done using phylogenetic trees, researchers have devised shortcuts, or working definitions, such as that of reciprocal best hits (RBHs). Beyond the fact that orthology is not transitive [5], once put together for further analyses, groups of RBHs among several species might reveal that a species contributes more than one gene to the orthology group. Such a situation might also be due to authentic orthologies whereby two genes in one species might be orthologs to single genes in other species because a duplication occurred after the species divergence. In other words, the duplicates are in-paralogs [6] relative to the other species analysed and both genes should be defined as orthologs, or co-orthologs of the homolog in the other species. However, species contributing more than one gene might be due to false positives. For

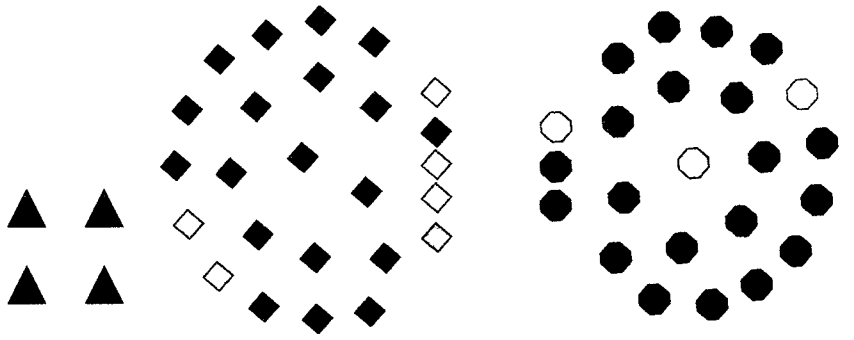
instance, some subgroup of species might contain two versions of a gene, while other species might have lost one copy, and another subgroup of species lost the other copy That is, in the parental species P, there are two paralogs, g and h, and in the offspring A, there is only g_A, while in the offspring B, there is only h_B, even though all the other offspring x, have g_x and h_x This will cause g_A to be linked to h_B and, therefore, link all g_x and h_x into a single group Moreover, in the context of this research, orthology is used to imply functional equivalence, which is not necessarily correct, even if the sequences of the orthologs are very similar

Grouping of orthologs generally employs a very local view (e g , RBHs [7]) or a global view (e g , cluster of orthologous genes (COG) [8]). The disadvantage of the global view is that there is a tendency to group paralogs and the disadvantage of the local view is that, when scaled to include multiple organisms, conflicts are introduced and the results become uncertain

In order to determine orthology by RBHs, the BLAST search algorithm, a heuristic algorithm designed for finding putative homologs, is used Consider the genomes of two organisms, A and B, with a BLAST database created for each genome Given a gene g in genome A, genome B is searched using g as the query and the top scoring result, h, is selected Genome A is then searched using h as the query and the top scoring result, g' is selected If $g = g'$, then g and h are defined to be orthologs This comparison is repeated for all genes in genome A, producing a list of pairs of RBHs between the two genomes [7]

Consider a group of organisms and their respective RBHs and note that there can be several "best" hits if the scores are equal For every gene, a graph is generated for all the genes connected by a reciprocal best hit Using 29 *E. coli* species, in most of the 7039 graphs[1], there was, at most, one gene from each organism However, 819 graphs (about 11 6 %) contained multiple genes from a single organism These ambiguous graphs could define co-orthologs, degenerate groups of genes where the reciprocal best hits have found paralogous genes, or unrelated genes with high sequence similarity mistaken for orthologs Upon inspection, the RBH scores indicated that some graphs should be a single cluster, while others should be split into multiple graphs On the other hand, the graphs that should remain a single cluster appeared to be due to a recent duplication and were generally small. The genes depicted in Figure 1 1(a) are variants of the urease subunit β gene The graphs that required splitting were generally caused by a loss that caused inappropriate pairing of genes across certain organisms For example, Figure 1.1(b) shows a medium-sized group of 50S ribosomal L31 proteins where loss in *Escherichia coli* APEC 01, *Escherichia coli* CFT073, *Escherichia coli* UTI89, *Shigella boydii* CDC3083 94, *Shigella dysenteriæ*, *Shigella flexneri* 2a, *Shigella flexneri* 2a 2457T, and *Shigella sonnei* Ss046 causes in-

[1]Singleton genes are not counted



(a) A small group with a recent duplication

(b) A larger group with two deletions

Figure 1.1: Sample orthologous gene clusters produced by reciprocal best hits where colours indicate organism, node shape indicate the derived grouping, and edges indicate a reciprocal best hit

appropriate pairing The organisms with the loss of a duplicate tended to inappropriately connect with more distantly related genes, but inspection of the phylogeny of the sequences clearly indicates the need for distinct groups The extremely large graphs were generally composed of highly repetitive genes, such as transposable elements The largest group contained 1244 genes and inspection of groups this large is not practical Additionally, this group was a collection of transposon genes, which are not useful for subsequent analyses

For the purposes of the subsequent work, a simple method to group genes was required For each cluster of genes discovered by walking the RBH for the species under investigation, a phylogenetic tree was built The mean and standard deviation of the branch lengths were computed and any branches in the tree greater than two standard deviations above the mean was cut. The connected subset of the resulting tree were considered separate groups of orthologs

1.1.2 Nebulon

The Nebulon system finds putative interactions between genes using genomic context Since operons in prokaryotes tend to carry functionally related genes [9], operons can be used to connect genes by function However, a single biological function may be split into several operons Because operons are unstable across organisms, multiple genomes can be used to recover the uber-operon, a set of genes connected by operons across rearrangements [10] The process begins with a set of genomes First, for each pair of genomes, all the coding sequences (CDSs) are paired with their putative ortholog based on the RBH algorithm The system then at-

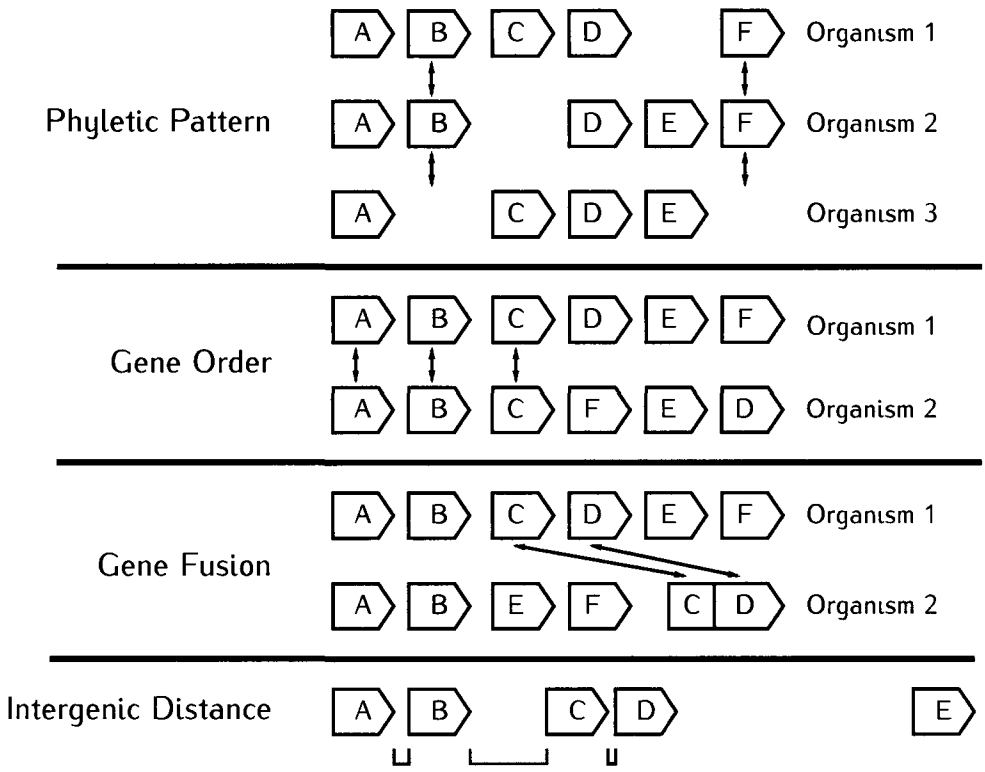


Figure 1.2: Examples of gene linkage patterns used by Nebulon to infer functional interaction

tempts to assign internal links for the genes in each genome [3] Links are determined in four ways phyletic pattern, conservation of gene order, gene fusions, and intergenic distances An example is shown in Figure 1 2

Phyletic pattern assumes that genes working in concert are going to be retained together or lost together. Selective pressure removes useless genes from the genome [11], implying that, if a single gene from a functionally related group is lost, the remaining genes tend to be lost rapidly if they cannot function without the lost gene. In a group of organisms, groups of genes which are all present or all absent are more likely to have related functions [3] In Figure 1 2, genes *B* and *F* are lost in concert in organism 3, indicating they may have related functions in organisms 1 and 2

Conservation of gene order is based on the presence of polycistronic messenger RNA, that is, genes belonging to the same operon. Two neighbouring CDSs in the same direction may be transcribed together (i e , are part of the same operon and, therefore, functionally related). If the genes are transcribed together, the probability of a genome rearrangement which separates the genes yet preserves functionality is low

Conversely, if two unrelated CDSs are neighbouring, then the probability of a genome rearrangement which preserves their individual functionality is much higher In this way, if gene orders are conserved across organisms, the genes are likely part of an operon and, therefore, have related functions [3] In Figure 1 2, genes *A*, *B*, and *C* remain in the same order, indicating they may have related functions

Gene fusions demonstrate that two genes are related because, in another organism, they are a single gene [3] In Figure 1 2, genes *C* and *D* are fused into a single CDS, indicating they very likely to have related functions

Intergenic distances use the amount of non-coding DNA between genes to determine the likelihood of interaction Regulatory elements, such as promoters and transcriptional terminators, are present at the start and end of operons These regulator regions create necessarily larger gaps between CDSs of neighbouring operons than of CDSs within an operon These distances can be used to determine the probability of two genes being part of the same operon and, therefore, the probability of them interacting In Figure 1 2, genes *C* and *D* are very close together, suggesting a very high probability of interaction while genes *D* and *E* are very far apart, suggesting a very low probability of interaction

The power of Nebulon comes from combining external and internal information Internal information comes from within a particular genome For instance, one organism may have a link between gene *A* and gene *B* External information comes from other genomes For instance, another organism may have a link between gene *A* and gene *C* From the perspective of the first organism, the link $A-C$ can be considered external information. Nebulon can then infer that there is an interaction between *A* and *C* in the first organism even though there is no interaction visible in its genome [3]

1.2 *Escherichia coli* Pathogenicity

E coli is a species of bacteria that includes strains that interact commensally with humans as well as those that interact pathogenically Most isolated strains, particularly *Escherichia coli* K12, are not pathogenic The major factors for virulence tend to be clustered into pathogenicity islands, usually on mobile genetic elements, such as transposons [12]. Several different types of pathogenicity mechanisms exist and move between different strains of *E coli* The pathogenicity systems can also recombine such as in one of the most dangerous strains, *Escherichia coli* O157 H7 which has many pathogenic factors on the pO157 plasmid and has the main toxin,

also present in *S dysenteriæ*, in a prophage [12] Approximately 13% of the genome of *E coli* CFT073 consists of pathogenicity islands [13] while other strains have considerably fewer genes *E coli* K12 strains retain part of a type-III secretion system, necessary for injecting proteins into host cells, even though this strain is not pathogenic [14] The complexity of these patterns suggest there is interaction between these pathogenicity genes on a large scale

1.3 The Rhizobiales

The Rhizobiales, a group of α-proteobacteria, are able to interact with eukaryotic organisms, but each species has a different host range and specificity Although the general cellular processes of infection are understood, there is incomplete understanding of host specificity [15]

A number of species, including *S meliloti*, *Rhizobium leguminosarum viciæ* and *Bradyrhizobium japonicum*, form symbiotic relationships with leguminous plants [15] After bacteria and plant exchange chemical signals, the bacteria are able to infect the plant's root cells If the infection is successful, the plant forms nodules to house the bacteria that then differentiate into bacteroids The bacteria fix atmospheric nitrogen into ammonium (NH_4^+), which is absorbed by the plant and used for production of nitrogen-containing organic compounds, the photosynthesis products are converted to dicarboxylates and used as a carbon source by the bacteria These events are initiated by the plant's secretion of a unique flavonoid into the soil Corresponding bacteria in the soil respond to the flavonoids by secreting Nod factors which trigger the root hairs to curl and allow the bacteria to enter [15] However, host specificity is more complicated than flavonoid recognition and Nod factor production alone If the genes responsible for flavonoid recognition and Nod factor production from *B japonicum*, which is hosted by *Glycine max*, are transferred to *S meliloti*, which is hosted by *Medicago sativa*, and the recombinant version placed on *G max*, the recombinant bacteria cause nodule formation, but will be unable to grow inside the nodule or fix nitrogen [15] The precise cellular processes that are responsible for maintaining the bacteria in the plant are not understood Clearly, other physiological processes must be involved in the maintenance of plant nodules

Pisum sativum is nodulated by *R leguminosarum viciæ* and mutant plants have been developed that have unusual nodulation phenotypes *Pisum sativum* R50 (*sym16*), produced by γ-irradiation, accumulates cytokinin more than the wild-type [16]. Although the location of the mutation is known [17], the exact mechanism has not been determined However, the mutant plants display a number of unusual nodulation phenotypes including

reduced nodulation, meandering infection threads, nodule formation with an abnormal number of anticlinal divisions [18] and a variable number of vascular poles [19] Also, there is more tendency for abortion of the nodule

O anthropi is an understudied opportunistic human pathogen Its genome has been recently sequenced but the cellular processes responsible for virulence are not well-understood [20] Putative orthologs to infection genes in other Rhizobiales have been identified, suggesting that these genes are still responsible for infection despite the divergent choice of host The exact function of the *O. anthropi* orthologs has not been characterised This organism is of clinical interest as it is increasingly being identified as a cause of human disease [20], making it an increasingly important target for further study. Phylogenetically, *O anthropi* is closely related to the genus *Brucella*, despite *O anthropi* having a genome of $\sim$ 6 Mbp while members of *Brucella* typically have $\sim$ 3 Mbp genomes The other organisms in the Rhizobiaceæ family are either plant symbionts or plant pathogens

Chapter 2

Overview

DATA USED IN THE ANALYSES were, for the most part, derived from the genome sequences stored in the National Center for Biotechnology Information (NCBI)[21] Interaction networks were prepared by Nebulon and then made available to **gisql**, the software presented in this thesis A prerequisite for the analysis done by **gisql** is the grouping of orthologous genes across species, which varies based on the species used in the particular analysis A flowchart of the analyses is shown in Figure 2 1, blocks with dashed borders were prepared by other members of the lab

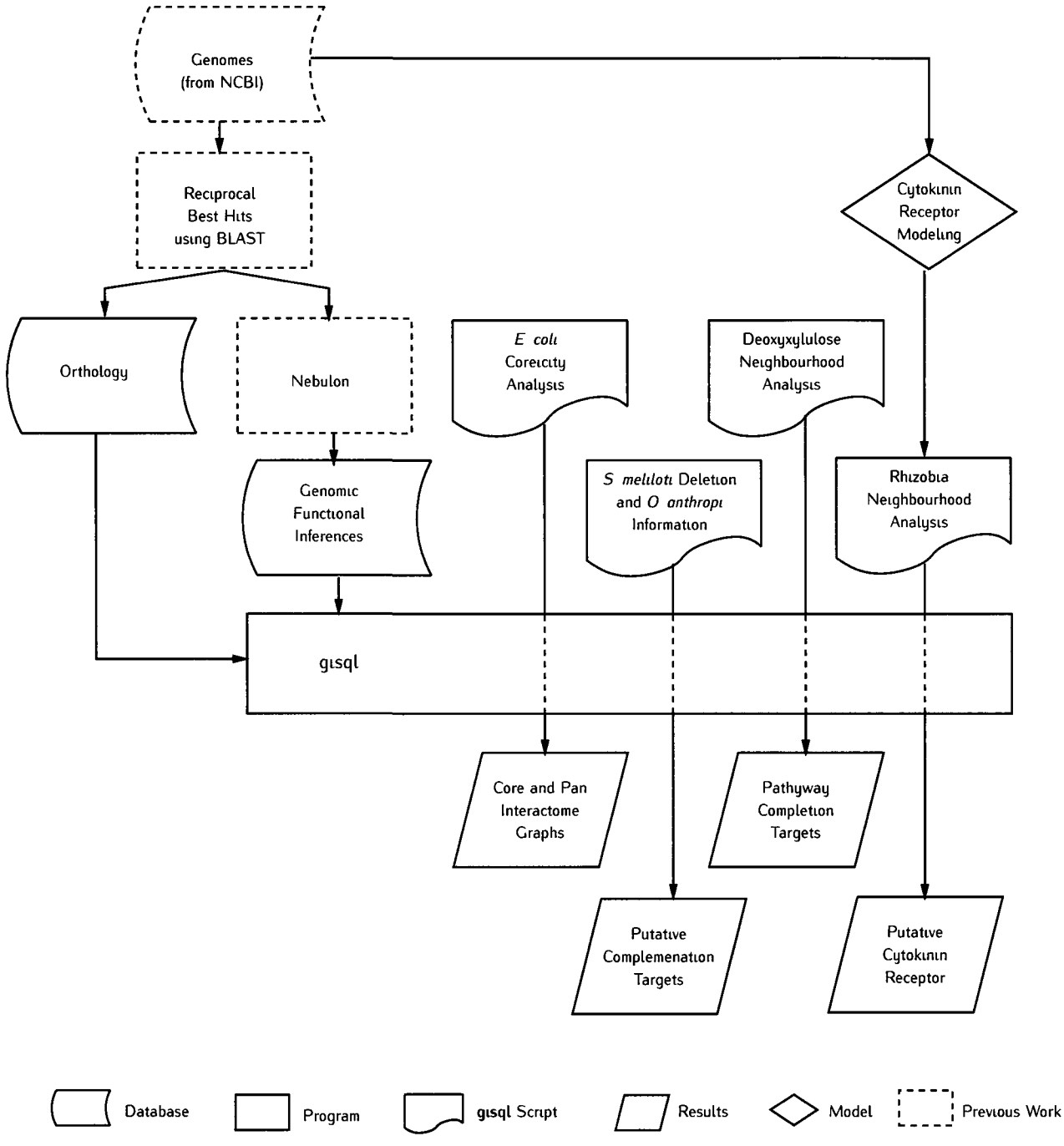


Figure 2.1: Overall method of data processing for projects in this thesis

Chapter 3

The gisql Language

In order to analyse functional interaction networks, a programming language, called `gisql` (the Genetic Interactome Set Query Language), was developed to analyse functional interaction networks using manipulations of fuzzy sets

3.1 Approach

Manipulating functional interaction networks requires a defined set of operations on the graphs Since analyses, initially, were aimed at finding interactions common to certain groups of organisms, the method was to perform basic set operations on the edge and node sets in the graphs. However, the interactions are predictions and each has a score, $s \in [0, 1]$, that defines the confidence for this interaction This scoring system is compatible with fuzzy set theory, where all items have a membership, $\mu \in [0, 1]$, in a fuzzy set To facilitate complex and flexible analyses, the software was designed as a programming language to manipulate fuzzy graphs using set operations specified by the user In addition to basic operations, other types of analyses were included in the language, including measures of fuzzy sets, genomic similarity score, and calculation of gene-based properties

3.1.1 Background on Fuzzy Logic

Axiomatic set theory centres around objects called sets and elements that may be part of a set If a value x is in a set S, then one may write $x \in S$, otherwise $x \notin S$ There is no uncertainty expressible; either x is in S or it is not Fuzzy set theory developed around the need to express inclusion of values which are not precise [22]

The classic example would be temperature given a temperature, is it "warm"? There is no precise definition of warm, but it would be generally accepted that 12 is "less warm" than 20°C. Warmth can be defined as a fuzzy set and, for each temperature, its membership in the set can be defined, membership ranges between 0 and 1 For a fuzzy set A and a value x, the membership of this value in the set is written as Ax Fuzzy logic has analogous operations to all the axiomatic set operators There are three functions that must be defined $\top$, the triangular norm; $\bot$, the triangular co-norm, and $\neg$, the inversion function Any functions may be used provided that

$$\top(a,b) \in [0,1] \quad \forall a,b \in [0,1] \tag{3.1}$$

$$\top(a,b) = \top(b,a) \tag{3.2}$$

$$\top(a,b) \leq \top(c,d) \text{ if } a \leq c \wedge b \leq d \tag{3.3}$$

$$\top(a,\top(b,c)) = \top(\top(a,b),c) \tag{3.4}$$

$$\top(a,1) = a \tag{3.5}$$

$$\bot(a,b) \in [0,1] \quad \forall a,b \in [0,1] \tag{3.6}$$

$$\bot(a,b) = \bot(b,a) \tag{3.7}$$

$$\bot(a,b) \leq \bot(c,d) \text{ if } a \leq c \wedge b \leq d \tag{3.8}$$

$$\bot(a,\bot(b,c)) = \bot(\bot(a,b),c) \tag{3.9}$$

$$\bot(a,0) = a \tag{3.10}$$

$$\bot(a,b) = \neg\top(\neg a,\neg b) \tag{3.11}$$

The most common norms that can be defined are the Godel norms $\top(a,b) = \min(a,b)$, $\bot(a,b) = \max(a,b)$, $\neg x = 1-x$ Given a set of norms, $A \cap B$ is $\top(Ax,Bx)\forall x$, $A \cup B$ is $\bot(Ax,Bx)\forall x$, and $\overline{A}$ is $\neg Ax\forall x$ [23, 24, 22, 25, 26] Since a graph is a set of vertices and a set of edges, **gisql** defines a graph as a fuzzy set of nodes and a fuzzy set of edges Performing an operation on a graph is equivalent to performing the operation on its vertex set and edge set Since the vertices of the interaction graph are genes, **gisql** requires that there be an equivalence of genes across graphs The equivalence of edges extends from the equivalence of vertices This can lead to the "inconsistency" that the membership of an interaction is positive even though one or both of the participating genes have zero membership. There are operations in **gisql** designed to remedy this problem when it arises

It is advantageous to allow it as it permits the user to select nodes given the properties of edges or to select edges given the properties of the nodes

3.1.2 Design Overview

The language is a statically-typed functional language implemented in Java Programs can manipulate fuzzy graphs using set manipulation operators were a notion of orthology, or equivalence of nodes between graphs, is built-in Commands are parsed using an LL(1) recursive-descent parser with limited memorising (i e , a special case of a pack-rat parser [27]), compiled to Java Virtual Machine (JVM) bytecode and hot-loaded into the running JVM Type inference is performed using a derivative of the Hindley-Milner algorithm W [28] Interactomes, objects representing real or computed interaction graphs, are stored and computed in a way that minimises the memory required for manipulation although the entire graph must be loaded in memory The syntax borrows elements from Haskell [29]. The language supports anonymous functions (lambdas), recursive functions, arbitrary interactome processors, lists, and nullable types

While other facilities exist to manipulate graphs, the constraints of dealing with orthology in reasonable way made other tools impractical Moreover, most graph packages either do not have ways to assign multiple weights to edges (one for each interactome) or the methods are very cumbersome to use The major design consideration was to work around a good model for interactomes rather than graphs in general

Compilation Process

Each user is given an environment in which they may define new functions The user can access an environment containing interactomes loaded from an external source (e g., those produced by Nebulon). Compilation begins with the reading of an input command The LL(1) parser, `ca.wlu.gisql.parser.Parser` using syntax-defining objects extending `ca.wlu.gisql.parser.Parseable`, creates an abstract syntax tree of type `ca.wlu.gisql.ast.AstNode`. The parse tree is then semantically analysed and all binding occurs. Once complete, type checking is performed using algorithm W After type checking, a wrapper, `ca.wlu.gisql.util.Rendering` uses OW2 Consortium's ASM [30] to generate Java byte-code which is then loaded into the running JVM and available as a `ca.wlu.gisql.ast.util.GenericFunction`

All types in the system have a matching Java type to which they can be cast Type checking is used to ensure that the generated Java code will indeed have a valid type, however, the JVM must still be informed about these types Functions created during compilation are assigned randomly generated unique identifiers and placed in

classes extending `GenericFunction` and interactome processors created during compilation are placed in classes extending `ca.wlu.gisql.ast.util.UserDefinedInteractome` which implements `Interactome` Although byte-code can be saved to a file, it is not possible to reuse generated byte-code since the byte-code is dependent on information stored in the user's environment Unfortunately, the generated byte-code is permanently loaded into the JVM, meaning that generated classes can become uncollectable, effectively leaking memory, as the JVM assumes that classes are not generated at run time, therefore, there must be some connectivity to a class and no way for a class to become unreferenced. For future versions of the JVM, `java.dyn.AnonymousClassLoader` is being developed to address it for languages, including Scala and Groovy

A novel addition to the language is the subgraph iterator. This iterator allows specification of an arbitrary connected graph and will then find all isomorphisms of the specified graph in an interactome The syntax was inspired by XQuery's FLWOR expression which iterates over matching trees in an XML document [31] The subgraph isomorphism is computed by the VF2 algorithm [32] simplified to work with only connected graphs In this case, any edge with a membership of zero is treated as disconnected and any edge with non-zero membership is treated as connected Since the user can manipulate the edge values, it is possible to select edges with any properties

Type System

The type system resembles Haskell's [29] type system It includes various basic types integral numbers, floating point numbers, generic lists, function types, and type variables It also includes the domain-specific elements genes, interactomes, and memberships Memberships are values appropriate for use in membership functions, that is $\in [0, 1]$ Type classes, which are unrelated to Java's classes, are also available Borrowing from Scala [33], type classes are backed by Java interfaces, however, the user cannot define new type classes nor can they implement type classes It is possible for a developer to register new types and new type classes into the system on the Java side The system also includes nullable types Haskell introduces nullable types using monads, however, this requires a great deal of complexity in the type system as algebraic data types are required [29] It is simpler to treat the nullable type as a special case and modify the inference rules. If α is a type, then $\alpha_\perp$ is a nullable version, that is given a term has some type, α, that term will always return a reference to an object of type α while a term of the type $\alpha_\perp$ will either return a reference to an object of type α or no object The inference rules for application, the only part of the system to change, are as follows:

$$\frac{\Gamma \vdash M \quad \sigma \to \tau \quad \Gamma \vdash N \quad \sigma}{\Gamma \vdash MN \quad \tau} \text{ [App]}$$

$$\frac{\Gamma \vdash M \cdot \sigma_\perp \to \tau \quad \Gamma \vdash N \quad \sigma}{\Gamma \vdash MN \quad \tau} \text{ [Null-App]}$$

$$\frac{\Gamma \vdash M \quad \sigma \to \tau \quad \Gamma \vdash N \cdot \sigma_\perp}{\Gamma \vdash MN \quad \tau_\perp} \text{ [Lifted-App]}$$

When the [Lifted-App] rule is used, a null check must be inserted equivalent to the Java code `(N == null ? null : M(N))`

Graph Storage

Graphs are stored in an SQL database and loaded into main memory upon request A simplified schema is shown in Figure 3 1 Each network exists in the database as a "species" with a unique list of genes identified by 8 byte integers A gene may be part of an orthology group, which marries it to equivalent genes in other species Each interaction is stored as a pair of participating genes and a score The genes of a species may be partitioned into multiple non-overlapping groups This is useful in organisms with multiple replicons, such as *Sinorhizobium meliloti*, which has a main chromosome and two megaplasmids, to allow replicons to be addressed individually.

The user may provide supplementary information about genes, including alternate names and COG identifiers [1] Users may also provide named lists of organisms The analysis in Chapter 4 made use of lists containing pathogenic strains and non-pathogenic strains

3.1.3 Interactome Computation Process

Each interactome object in the system holds information about an interactome graph The system differentiates *species* graphs, ones based on external data, from computed graphs, which are manipulated All genes and interactions, regardless of species, are stored in a single composited graph the ubergraph Processing an interactome is simply looking up the membership of gene or interaction from the ubergraph in an interactome. Set expressions are expressions that manipulate the membership values from other interactomes For instance, given the expression $A \cup B \cap (C \cup D)$ noting that $\cup$ has lower precedence than $\cap$, the system will create a structure as shown in Figure 3.2(a). This structure will be traversed for each gene and each interaction in the system and the final membership value for each gene will be output to the user Additionally, the membership

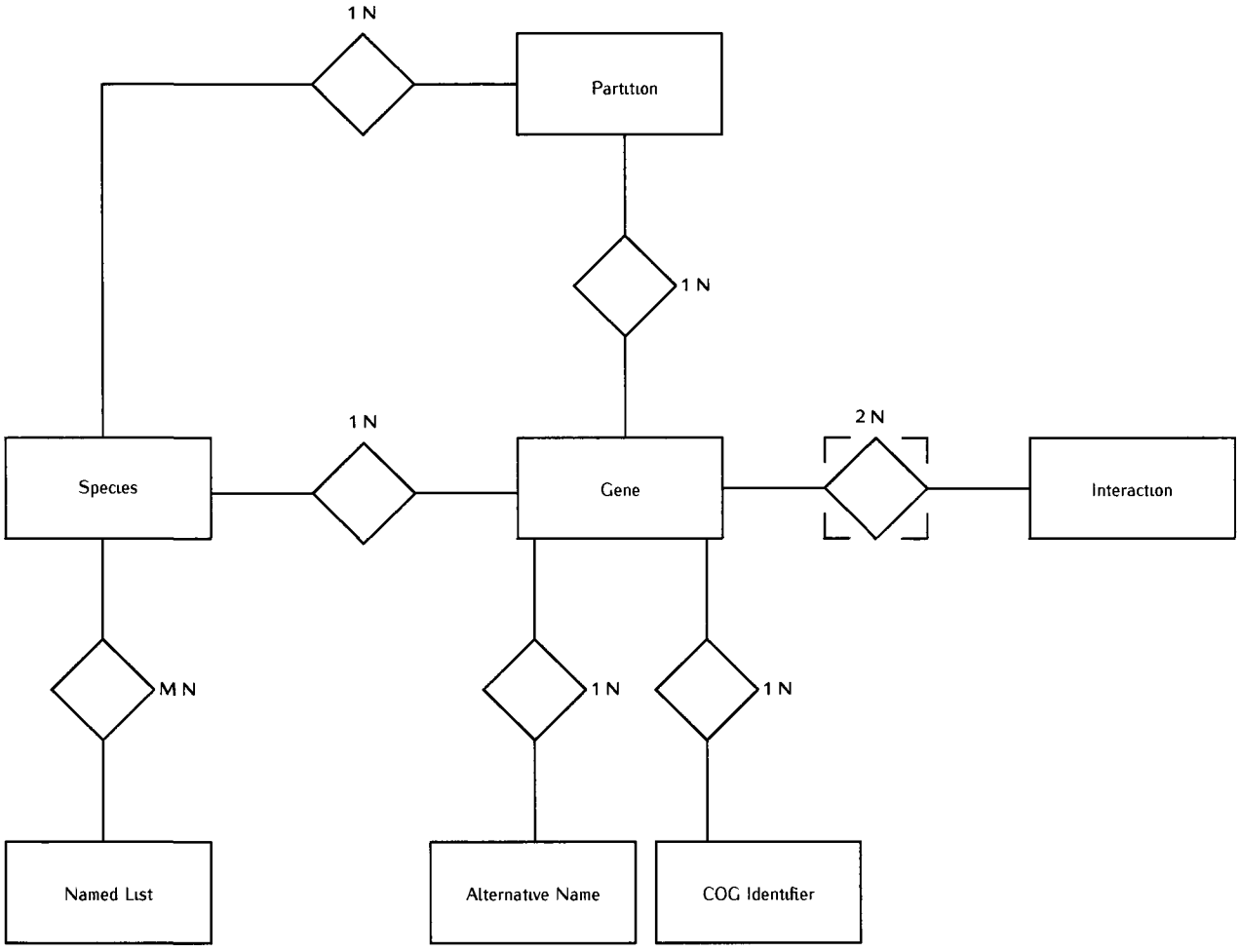


Figure 3.1: Simplified database schema used for interactome storage

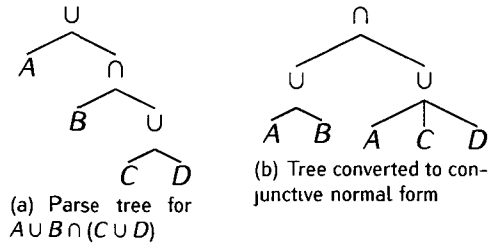


Figure 3.2: Example set operation expressions

of an element may be *missing*; this is a slightly different concept from 0. At the end, the system will compute statistics about all elements, even if their membership is 0, however, it will ignore any which are *missing* There is a `defuzz` operator which will convert any elements with 0 membership to missing and `blanks` which will give a value to any missing items

The basic set operations (i e , union, intersection, difference, symmetric difference, and complement) are treated specially in that the compiler will produce an optimised version of the expression in conjunctive normal form, also called sum of products form. The Figure 3 2(b) shows the conjunctive normal form of Figure 3.2(a) It should be noted that expressions in arbitrary fuzzy logic cannot be converted to conjunctive normal form [34] However, `gisql` uses only the Godel triangular norm, which can be converted to conjunctive normal form There always exists a conjuntive normal form for any logical expression, while frequently containing more terms than the original expression, are usually more managable for complex expressions.

3.1.4 Human Interface

The software is designed such that it can be used as a package by other Java programs However, at present, most users would deal with it directly as the query language is the most brief and flexible interface Three interfaces are provided· a command-line interpreter, a graphical interface, and a Cytoscape plug-in

The command line interface provides the simplest access to the data Programs may be entered via the terminal or from files and the results will be displayed on the terminal The results of interactome computations can be output to disk in formats including Matlab, GML, and GraphML The JLine library provides a terminal interface with command completion and a history buffer The terminal must support full Unicode characters as many are present in the output

The graphical interface, written in Swing, extends the terminal interface in two important ways result data

is presented in tables and the user's environment can be browsed All other functionality of the command-line version is present The output tables are presented to the user as tabs, so the results of many computations can be displayed. Additionally, hyperlinks to the National Center for Biotechnology Information (NCBI)'s genome browser are provided for genes The environment browser allows the user to see all of the objects in the environment, sorted by type, and access more information about interactomes

Cytoscape [35] is a utility to visualise large graphs common in biology and it provides a plug-in architecture that makes it possible to add functionality. A `gisql` plug-in allows the user to redirect the output into Cytoscape's visualisation system, making it possible to manipulate the graph using other Cytoscape tools or simply create pretty pictures

3.2 Discussion

The goals for `gisql` were to manipulate functional interaction graphs and analyse connectivity patterns To this end, it has proven a capable tool The decision to create a programming language allowed for far more complex analysis with shorter, clearer, more expressive programs There was a high initial cost in developing the compiler back-end and debugging the compiler-generated code slowed down the initial usage of the tool In the end, these challenges were well worth the re-usability of the tool

The biggest advantage of a domain-specific programming language is the domain knowledge `gisql` has implicit understanding of genes, fuzzy logic, and the semantically correct way to manipulate them Adding to this framework is a fairly simple matter, meaning that even if far more complex analyses are desired in the future, a developer does not incur the penalty of writing software to read the functional interaction networks More importantly, any tool developed can be combined with the existing suite of tools, allowing for more expressive power

There are, however, several disadvantages to `gisql` as it currently stands the interfacing with Java platform, the need to hold the graph in memory, and the lack of user-definable fuzzy sets

The code generated is run on the JVM, which was designed for Java, a fundamentally different language, requiring several inefficient workarounds, a challenge shared by other JVM-based languages including Jython, Groovy, and Scala Moreover, the code that is produced by `gisql` is not usable from Java, that is, it would not be possible to use a `gisql` program inside of a Java one Moving to the Common Language Runtime environment could reduce the impact of these problems as it was designed with more flexibility in mind

In the current design, the interaction graph must be held in the JVM's memory The *Escherichia coli* interaction graphs require more than a 32-bit address space, necessitating the use of 64-bit processors. A better design would allow the graphs to remain on disk and be loaded incrementally and discarded after they are processed Such an architecture would require a much more sophisticated compiler Indeed, the compilers present in most relational databases are performing similar functions It might be possible to out-source the optimisation task to a relational database, but making such decisions would still be quite difficult

The **gisql** language has no direct concept of a fuzzy set, only a fuzzy graph of gene interactions, making for awkward situations where it would be desirable to manipulate fuzzy sets, not entire interactomes Moreover, it would be useful to have a set of things other than genes and interactions On the surface, designing and implementing such a feature should be relatively straight forward as it would borrow much of the design from the interactome processing infrastructure However, the interactome processing infrastructure assumes that it knows all the items in the universe of discourse In the case of genes, this is a reasonable assumption as all the relevant genes are known, but a fuzzy set of integers would require iterating over all possible integers Undoubtedly, an efficient implementation could be designed with some effort using more mathematical abstractions of fuzzy sets

Overall, the **gisql** software greatly simplified analysis and was well worth the development effort There is no constraint that the functional interaction networks be predicted by Nebulon Indeed, the EcoCyc [36] and KEGG [37, 38, 39] interaction networks were used This means that **gisql** can be expanded to perform analyses on other networks, including protein-protein interaction networks, and to compare networks of different types

It may be possible to extend **gisql** to include other forms of logic For instance, rather than have a fuzzy membership defining graph, it would be possible, with significant alterations, to use a random variable making it possible to manipulate Erdős–Rényi random graphs In general, **gisql** could be a foundation from which to build a more general domain-specific graph manipulation language.

Chapter 4

Escherichia coli Pathogenicity

VIRULENCE FACTORS in *Escherichia coli* show great mobility [12] and so provide an interesting target for analysis of connectivity patterns with the core-genome

4.1 Materials and Methods

The sheer number of sequenced *E coli* strains allows for unique perspective on this group The first analysis done, in Source Listing 4 1 and Source Listing 4.2, tried to ascertain the size of the core and pan-genome and interactome for *E coli* In this analysis, an many subsequent analyses, only the interactions which Nebulon scored at or above 0 9 were included

To determine the prevalence of certain genes in the clade, a measure was devised called *coreicity*, which is calculated as the number of organisms containing an orthologous gene If a gene in one species has an ortholog in a second species, then the coreicity would be 2 This ignores any paralogs, that is, if there are two orthologs in the second species, the coreicity is still 2 A baseline of the distribution of gene coreicities is needed in order to provide context to investigate the coreicities of interactions The program shown in Source Listing 4.3 computes the number of genes in the *E coli* pan-interactome at every given coreicity value The coreicity difference between interactions can be calculated by Source Listing 4 4

The interactions used were inferred by Nebulon and, therefore, may not correctly represent the statistical properties of real interactions Fortunately, the EcoCyc [36] and KEGG [37, 38, 39] databases catalog curated interactions based on biochemical activity These databases only catalogue interactions in *Escherichia coli* K12

Table 4.1: Gene identifiers of known pathogenicity genes in *Escherichia coli*

15829279	15830821	15831408	15833109	15833844	74314950
15829315	15830822	15831409	15833111	15833896	82524596
15829327	15831066	15831410	15833112	15833907	74314926
15829726	15831068	15831480	15833804	15833908	82524572
15830100	15831069	15831481	15833808	15833911	31983586
15830101	15831075	15831483	15833815	15834189	187734515
15830102	15831078	15831681	15833816	15834275	82524591
15830104	15831079	15831968	15833818	15834302	82524590
15830130	15831248	15831969	15833825	15834549	82524589
15830379	15831249	15832739	15833834	31983538	82524588
15830381	15831250	15832740	15833835	157042764	82524581
15830814	15831327	15832741	15833836	82524599	82524583
15830815	15831328	15832742	15833837	31983791	

substr MG1655, so only this strain was used

Genes present as putative orthologs in more than 80% of organisms were defined as *popular* and genes present in less than 20% of organisms as *unpopular* Two modified froms of the analysis in Source Listing 4 4, Source Listing 4 6 and Source Listing 4 5, were modified to insist that one gene was popular or unpopular, respectively

For further analysis, the networks were reduced to the neighbourhood of genes in the interaction graph near known pathogenicity genes[40], shown in Table 4 1 The coreicity analyses above were repeated on the network of genes 2° (second degree, i.e , connected by, at most, two edges) from the known pathogenicity genes The larger this number is, the more the small world effect will take hold and select the entire network If 1° was selected, then the partners of unidentified pathogenicity genes would be missed

Since the distributions are somewhat difficult to analyse visually, Source Listing 4 7 produces data for an interaction heatmap over coreicities This analysis was carried out on the *E coli* pan-interactome

Assaying the presence of the highly pathogenic strain *Escherichia coli* O157 H7, is desirable, but non-trivial as there is no known unique screenable phenotype Current screening techniques for *E coli* O157 H7 are labour-intensive, often requiring multiple steps, and not conclusive, requiring verification by PCR A search was run to look for orthologs common to all *E coli* O157 H7 strains, but missing in all other forms of *E coli*, shown in Source Listing 4 8 If unique biochemical activities of *E coli* O157·H7 could be found, then simple single-step screening or selection could be developed to conclusively identify colonies more rapidly and cheaply.

Listing 4.1: *Escherichia coli* core-genome and interactome size versus number of included strains

```
map (\x -> (ecoli:slice 1 x):intersectall {0.9}) (range 1 (length ecoli))
```

Listing 4.2: *Escherichia coli* pan-genome and interactome size versus number of included strains

```
map (\x -> (ecoli:slice 1 x):unionall {0.9}) (range 1 (length ecoli))
```

Listing 4.3: Count of genes with specified coreicity (abundance) in *Escherichia coli* pan-genome

```
map (\x -> ((unionall ecoli {0.9}):genecore (\g -> \v -> v:eq x)) : defuzz) (range 1 (length
    ecoli))
```

Listing 4.4: Interaction count versus difference in coreicity (abundance) of interacting genes in the *Escherichia coli* pan-interactome

```
map (\x -> ((unionall ecoli {0.9}):interactioncoreicity (\g1 -> \v1 -> \g2 -> \v2 -> v1:sub v2
     :abs :eq x)) : defuzz) (range 1 (length ecoli))
```

Listing 4.5: Interaction count versus coreicity difference of interacting genes in the *Escherichia coli* pan-interacomte where one gene is present in more than 80% of strains (i e, popular)

```
map (\x -> ((unionall ecoli {0.9}) :interactioncoreicity (\g1 -> \v1 -> \g2 -> \v2 -> (v1:sub
    v2 :abs :eq x) & (v1:le 5 | v2:le 5)) : defuzz) (range 1 (length ecoli))
```

Listing 4.6: Interaction count versus coreicity difference of interacting genes in the *Escherichia coli* pan-interacomte where one gene is present in less than 20% of strains (i e, unpopular)

```
map (\x -> ((unionall ecoli {0.9}) :interactioncoreicity (\g1 -> \v1 -> \g2 -> \v2 -> (v1:sub
    v2 :abs :eq x) & (v1 :ge 23 | v2 :ge 23)) : defuzz) (range 1 (length ecoli))
```

Listing 4.7: Interaction counts for all coreicity combinations in the *Escherichia coli* pan-interactome

```
pool = ecoli:unionall
cnt = map (\x -> pool : genecoreicity (\g -> \v -> v :eq x) :genecount) (range 1 (ecoli:length
    ))
icount x y = pool:interactioncoreicity (\g1 -> \v1 -> \g2 -> \v2 -> (v1:eq x & v2:eq y) | (v2:
    eq x & v1:eq y)) : interactioncount
map (\x -> map (\y -> "\{1}␣\{2}␣\{3}␣\{4}␣\{5}" x y (icount x y) (cnt:index x) (cnt:index y):
    echo) (range 1 x)) (range 1 (length ecoli))
```

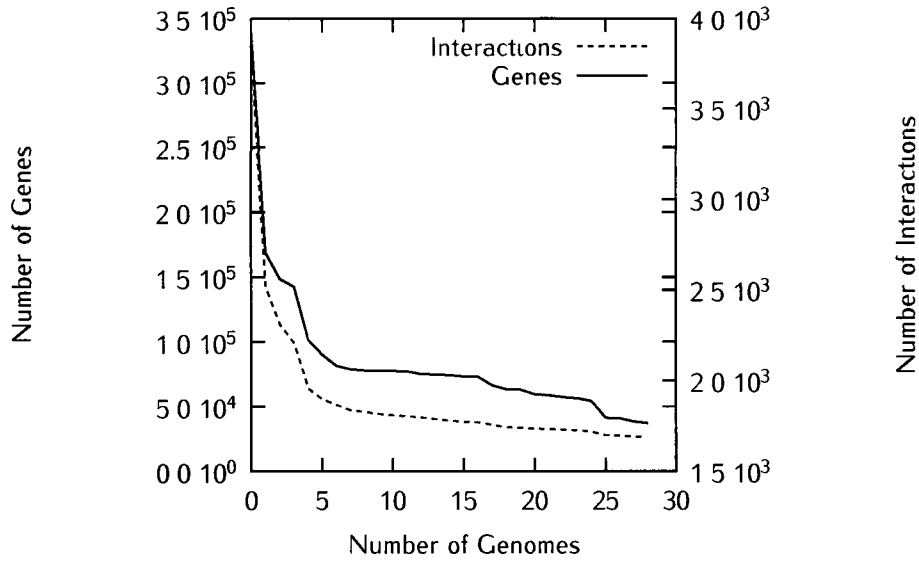


Figure 4.1: *Escherichia coli* core-genome and core-interactome size versus number of included strain genomes

4.2 Results

The first goal was to ascertain the overall properties of the interactome for all the *E. coli* species. The size of the core-genome and core-interactome, shown in Figure 4.1, drops to half the size of a single organism's genome and interactome. The size of the pan-genome and pan-interactome, shown in Figure 4.2, grow quite quickly, but the pan-interactome plateaus despite the pan-genome increasing linearly.

Genes have a tendency to have high coreicity or low coreicity, as shown in the histogram in Figure 4.3. This pattern extends to the interactions, given different coreicities as shown in Figure 4.4 for the *E. coli* strains.

The distribution of interactions in the curated EcoCyc [36] and KEGG [37, 38, 39] databases significantly[1] match the distribution of Nebulon predictions. The distributions are shown in Figure 4.5.

The analysis in Figure 4.4 was repeated for interactions in which at least one gene was popular, shown in Figure 4.6, and where at least one gene is unpopular, shown in Figure 4.7. Again, the analyses presented a similar distribution but both popular and unpopular genes had a strong tendency to pair with popular genes, in general.

For the known pathogenicity network, the analysis of overall coreicity difference is shown in Figure 4.8, the analysis of coreicity difference of popular genes is shown in Figure 4.9, and the analysis for coreicity difference

[1]For EcoCyc, $\chi^2 = 439120.1$ and, for KEGG, $\chi^2 = 58095.91$, in both cases $p < 2.2 \cdot 10^{-16}$

Listing 4.8: Genes common to all *Escherichia coli* O157.H7 strains not found in other *Escherichia coli* strains

```
(([E_coli_O157H7, E_coli_O157H7_EDL933, E_coli_O157_H7_EC4115]:intersectall {0.9} - [
    E_coli_0127_H6_E2348_69, E_coli_536, E_coli_55989, E_coli_APEC_O1, E_coli_CFT073,
    E_coli_C_ATCC8739, E_coli_E24377A, E_coli_ED1a, E_coli_HS, E_coli_IAI1, E_coli_IAI39,
    E_coli_K12_DH10B, E_coli_K12_MG1655, E_coli_K12_W3110, E_coli_S88, E_coli_SE11,
    E_coli_SMS3_5, E_coli_UMN026, E_coli_UTI89, S_boydii_CDC3083_94, S_boydii_Sb227,
    S_dysenteriae, S_flexneri_2a, S_flexneri_2a_2457T, S_flexneri_5_8401, S_sonnei_Ss046]:
    unionall){0.9}) :defuzz
```

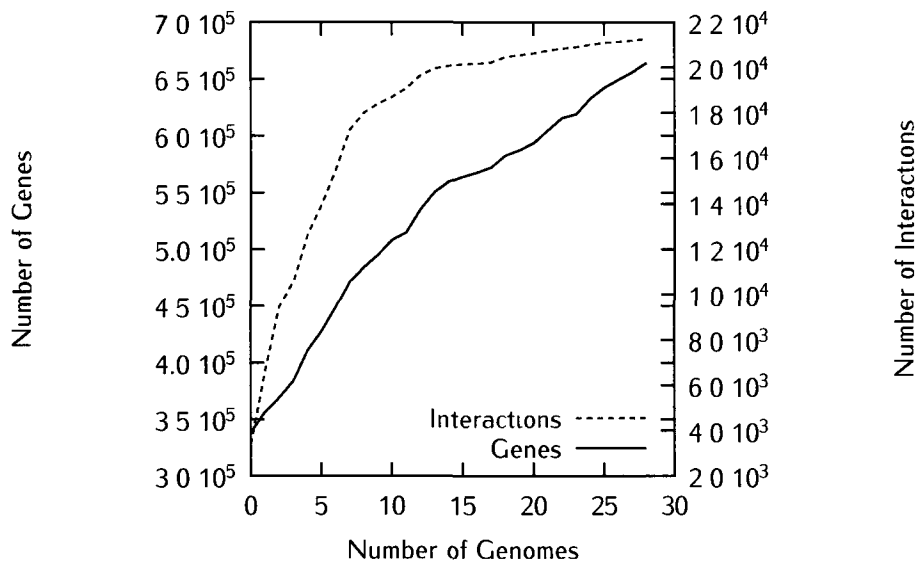


Figure 4.2: *Escherichia coli* pan-genome and pan-interactome size versus number of included strain genomes

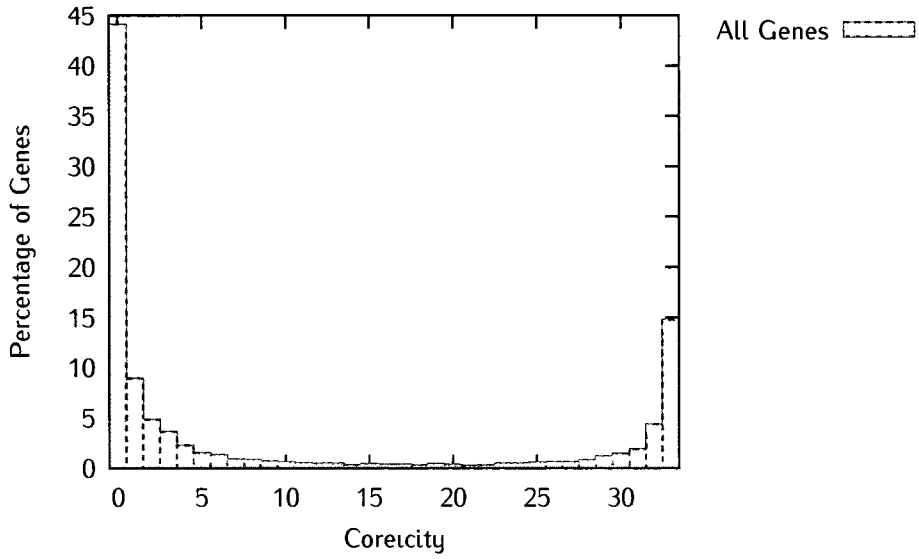


Figure 4.3: Count of genes with specified coreicity (abundance) in *Escherichia coli* pan-genome

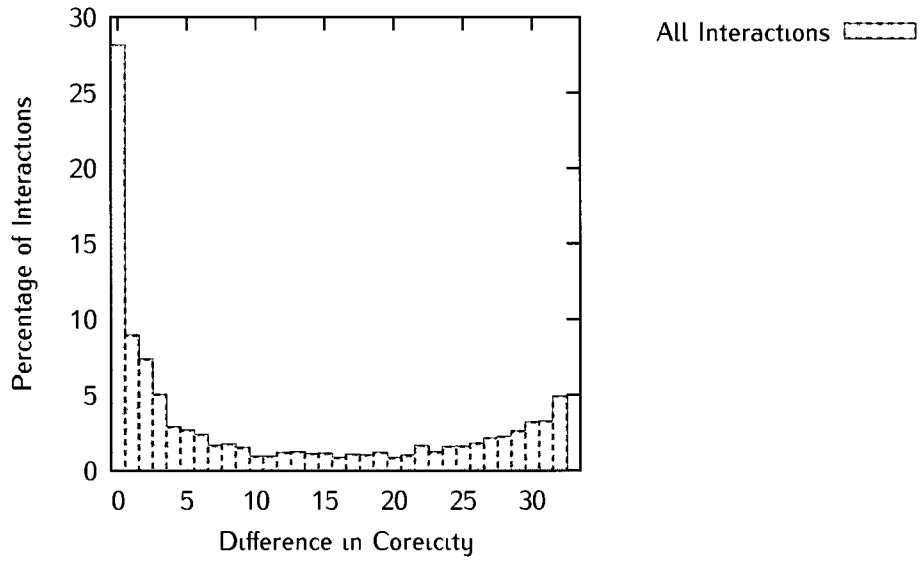


Figure 4.4: Interaction count versus difference in coreicity (abundance) of interacting genes in the *Escherichia coli* pan-interactome

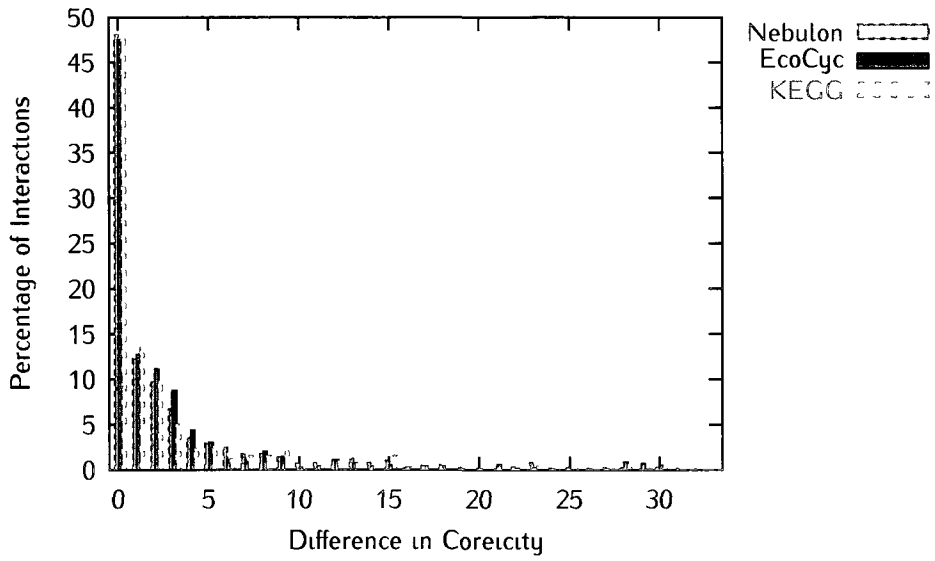


Figure 4.5: Interaction counts versus coreicity differences in *Escherichia coli* K12 substr MG1655 from predicted functional interactions by Nebulon and the curated biochemical EcoCyc and KEGG databases

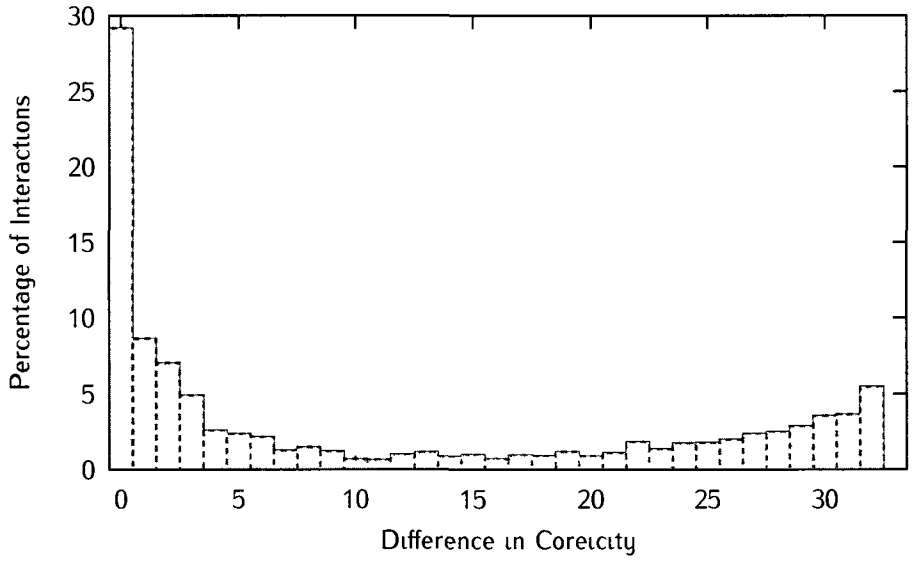


Figure 4.6: *Escherichia coli* interaction abundance as a function of coreicity difference for in which where one gene is present in more than 80% of strains

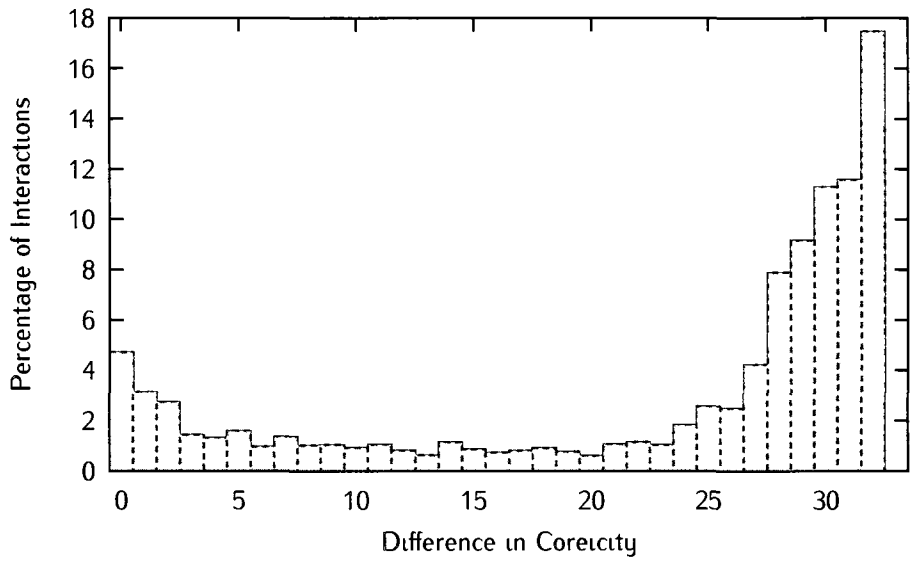


Figure 4.7: *Escherichia coli* interaction abundance as a function of coreicity difference for interactions where one gene is present in less than 20% of strains

of unpopular genes is shown in Figure 4 10 Distributions were similar to the global networks, but there was strong tendency for interactions to pair with unpopular genes In short, genes of high abundance of have more interacting partners than genes of low abundance, but the abudnace of their partners is not biased.

When considering the number of interactions between genes of different coreicities, the distribution is strongly biased, where popular-popular, popular-unpopular, and unpopular-unpopular pairs dominate the landscape by orders of magnitude. Since this closely matches the distribution of genes, a test to look for the density of interactions is necessary A simple measure for a graph with interactions I and genes G is to look at the density defined as

$$d(x,y) = \frac{|I_{C(g_1)=x \wedge C(g_2)=y}|}{\frac{1}{2}|G_{C=x}||G_{C=y}|} 100\% \quad \forall x,y$$

where C is the coreicity The density map, shown in Figure 4.11, indicates that unpopular-unpopular interactions are less prevalent compared to all interactions of popular genes

Unfortunately, no biochemical properties are possessed by *E. coli* O157:H7 that are not possessed by another sequenced strain of *E coli*, despite the organisms being physiologically unique The virulence mechanisms possessed by *E coli* O157·H7 appear to be duplicated elsewhere The only genes present only in *E. coli* O157·H7 were a duplication of the urease pathway and a prophage

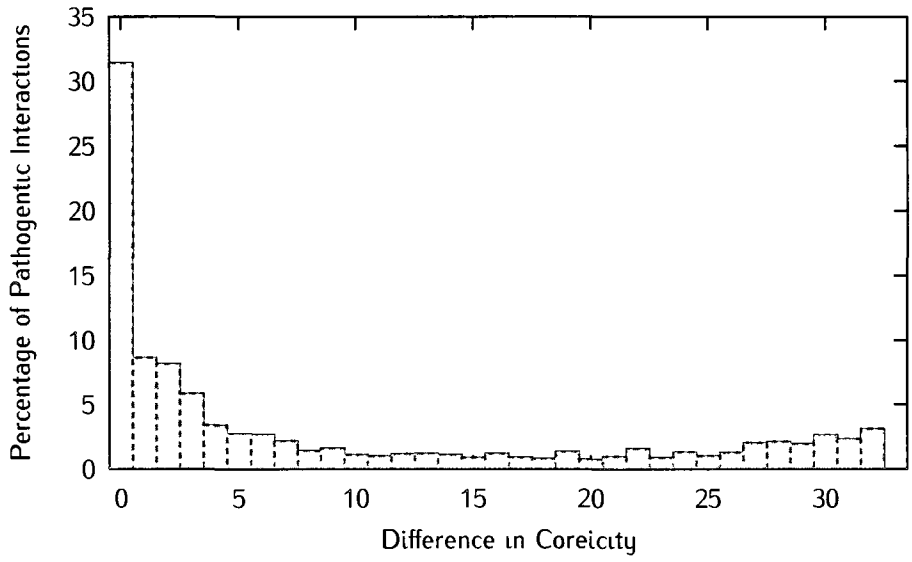


Figure 4.8: *Escherichia coli* interaction count for a coreicity difference for genes in the neighbourhood of known pathogenicity genes

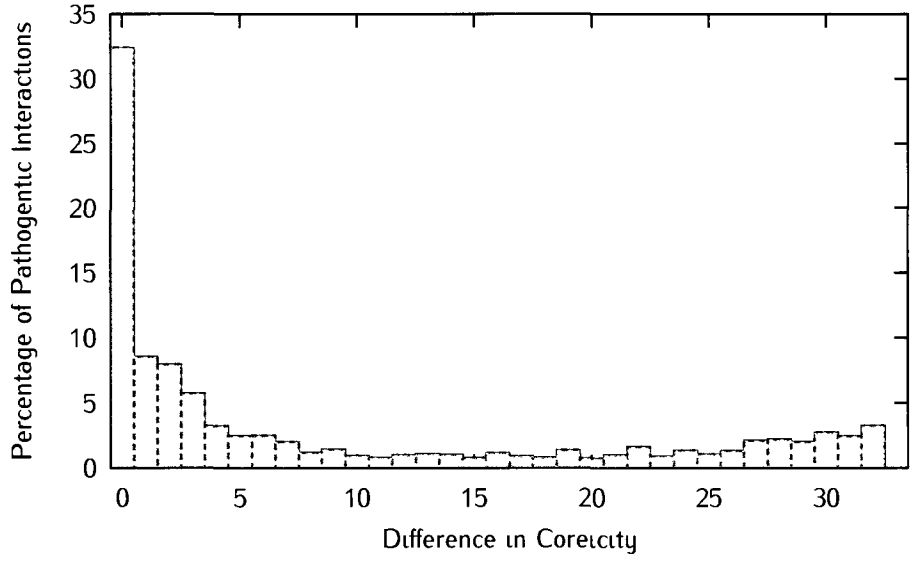


Figure 4.9: *Escherichia coli* interaction abundance as a function of coreicity difference for interactions where one gene is present in more than 80% of strains for genes in the neighbourhood of known pathogenicity genes

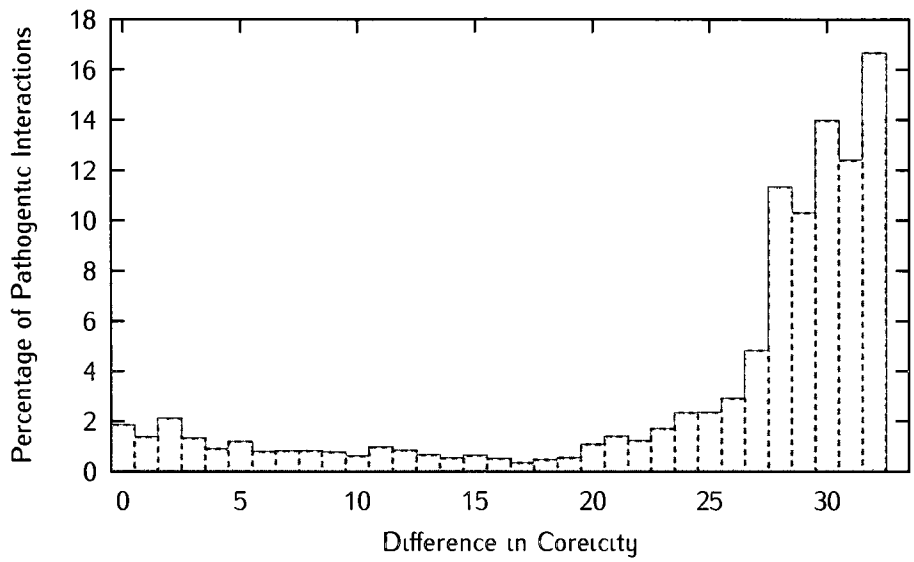


Figure 4.10: *Escherichia coli* interaction abundance as a function of coreicity difference for interactions where one gene is present in less than 20% of strains for genes in the neighbourhood of known pathogenicity genes

4.3 Discussion

The pan and core-interactome of *E coli* reveal information about the relationship between strains, the patterns found in the interaction networks, and the quality of the functional inferences

It is staggering how little overlap there is between the *E coli* strains Given these organisms are classified as the same species, most of their pan-genome is unique to one particular strain or another Even a group with a well-defined physiological and molecular finger print, *E coli* O157 H7, has little genetic commonality among different strains Indeed, the plaguing question of how to define a bacterial species seems even more muddied in a group of organisms designated to be the same strain, the genes unique to that group of organisms lacks the physiological features unique to that group I would imagine virulence mechanisms in *E. coli* to be something like poker hands specific collections of virulence factors give different virulence phenotypes Some combinations are less useful than others, but what might be a poor card in one hand can be valuable in a different hand Given all the virulence factors of *E coli* O157 H7 are present in other strains, yet none of those strains have the same level of pathogenicity, it must be the combination of factors that confer the unique virulence ability of *E. coli* O157 H7

In contrast, there is a clearer picture about the relationship between genes. Genes have a tendency to interact with other genes of similar abundance and genes have a strong tendency to interact with ubiquitous

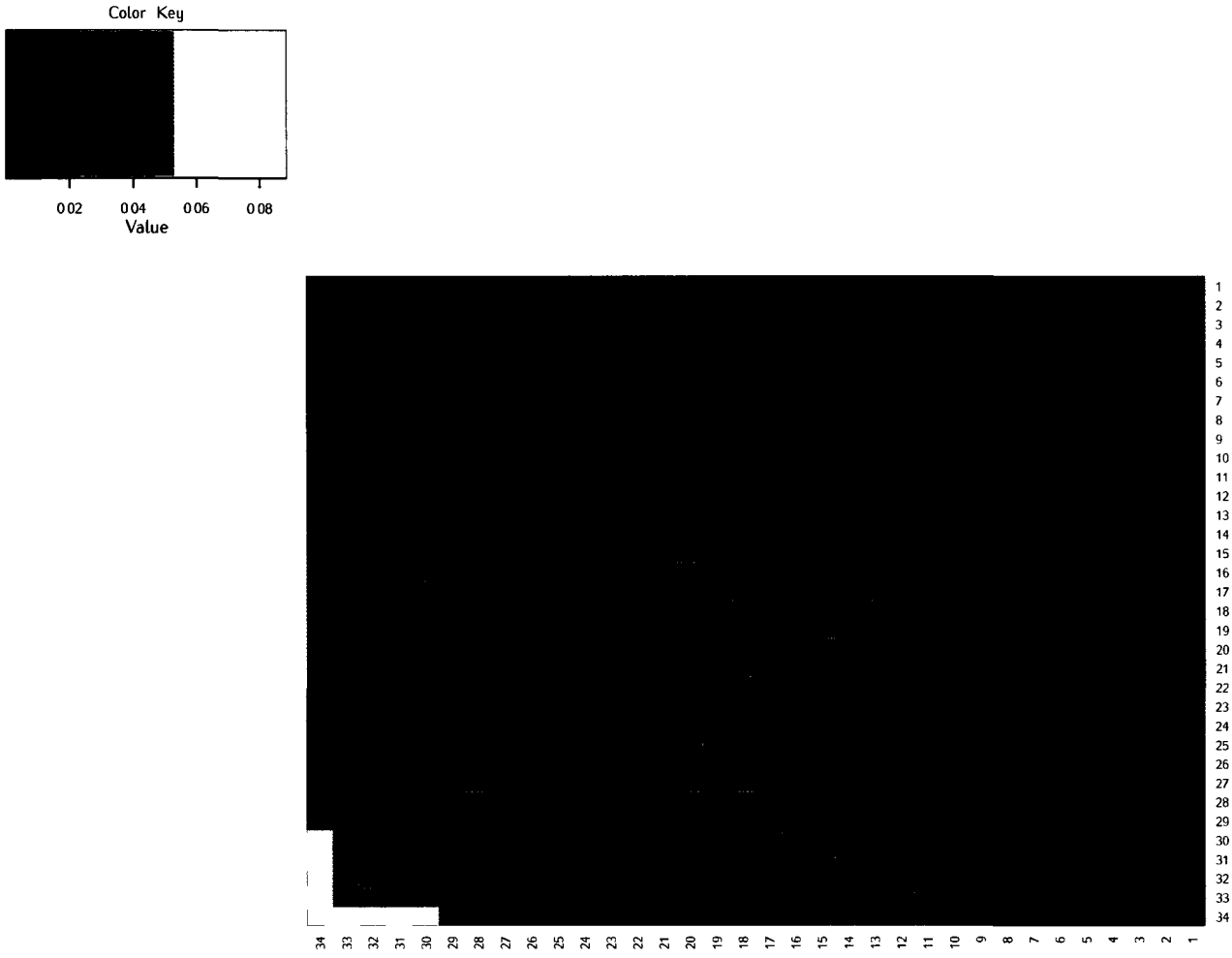


Figure 4.11: Map, as a percentage, of the density of interactions over maximum connectivity in a bipartite graph with the subset of nodes matching the coreicities (x, y)

genes A very simplified explanation of evolution suffices to explain the pattern If the only truly essential genes are the ones for replication, protein synthesis, and basic energy metabolism, genes need to associate as closely with these functions as possible to be conserved As sexual reproduction is not possible, the best mode for genes to propagate is horizontal gene transfer, which generally duplicates contiguous pieces of DNA If a gene is transferred to a new host, it must be functional (and useful) to avoid being eliminated, as non-functional genes are eliminated [11], and being functional with the aid of activities of genes "guaranteed" to be in the new host is a favourable strategy

Finally, comparisons between the similarity between the distributions of interactions in EcoCyc [36], KEGG [37, 38, 39], and Nebulon indicates an interesting, though circular, measure of quality:

- the Nebulon predictions are of reasonable quality given that they match the biochemical databases
- the curated databases are less biased by the interests of the research community than expected given they match the theoretically unbiased predictions of Nebulon

Clearly, both statements cannot be, though there is no obvious resolution At the very least, whatever biases exist in Nebulon match the biases that exist in curated biochemical databases, in terms of the distribution of interactions between genes of varying abundance, and given that they *should* have different sources of bias, it seems more likely that the bias is very small

Chapter 5

Functional Complementation in *Sinorhizobium meliloti*

FUNCTIONAL COMPLEMENTATION has been a long-standing method in bacterial genetics to determine the functions of genes. Deletions of the pSymB megaplasmid in *Sinorhizobium meliloti*, previously created by transposon mutagenesis, contained screenable phenotypes [41] for which complementary genes from *Ochrobactrum anthropi* were sought. Moreover, several regions of the pSymB megaplasmid are essential and cannot be deleted [41]. The connectivity profile of genes in these regions were analysed.

5.1 Materials and Methods

S. meliloti has a chromosome and two megaplasmids, pSymA and pSymB. Charles and Finan created deletions of the pSymB megaplasmid [41]. Since *O. anthropi* is closely related, homologs from *O. anthropi* were going to be used by John Heil to functionally complement the deletions in *S. meliloti*. Since the deletions are quite large, analysis to find all the orthologs of all relevant genes was necessary.

Although there are many deleted regions, shown in Figure 5.1, the primary region of interest is the ΔG373 which deletes several screenable phenotypes, shown in Table 5.1. There are also three regions, ∇1, ∇2, and ∇3, which appear to be essential as no viable deletions of these regions could be made. The locations of these regions are shown in Table 5.2. Although some genes in these regions have been identified as essential, further

analysis was done to determine if the properties of other genes in these regions are different than the properties of genes from the deletable regions of pSymB

The genes in ΔG373, ∇1, ∇2, and ∇3 are shown in Table 5.3, Table 5.4, Table 5.5, and Table 5.6, respectively

The first analysis involved determining if the genes in ∇1, ∇2, and ∇3 were more connected to the chromosome and pSymA than other genes on pSymB, shown in Source Listing 5.1, which uses the function defined in Source Listing 5.2. For functional complementation analysis, screenable phenotypes found in the ΔG373 region and the involved genes are shown in Table 5.1 Interestingly, *O. anthropi* does not appear to contain the gene *thiC*, used in thiamine biosynthesis Therefore, neighbourhood analyses were done on the genes for each screenable phenotype to determine which biochemically-complementary genes in *O. anthropi* could be identified even if they were non-homologous, shown in Source Listing 5.3. As further analysis, the coding sequences (CDSs) of a chromosome or megaplasmid are broken into fixed-sized windows and the number of genes on a target chromosome or megaplasmid interacting with the window are counted.

5.2 Results

Analysis of pSymB in *S. meliloti* yields several interesting results. First, the regions of pSymB which cannot be deleted are hot spots for genes which interact with the chromosome and pSymA Second, *O. anthropi* has a gene that is well connected to protochatecuate metabolism which is not present in *S. meliloti* Finally, while *O. anthropi* is lacking an ortholog to a gene in the *S. meliloti* thiamine biosynthesis pathway, no non-orthologous equivalent gene is identified.

The regions ∇1, ∇2, and ∇3 each have an essential feature preventing deletion. For instance, ∇1 contains the origin of replication [42]. These regions appear to be hotspots for activities involving genes on the main chromosome. In Table 5.8, the number of interactions between these regions and the main chromosome is listed. These regions account for 71% of the interactions between the main chromosome and pSymB and 62% of the interactions between pSymA and pSymB. These regions contain fewer genes than the remainder of pSymB, shown in Table 5.7 When the number of interactions is scaled by the maximum number of possible interactions, the bias is even more evident, these regions have twice the connectivity to the chromosome and pSymA than the remainder of pSymB, shown in Table 5.8

For each replicon, the chromosome and the two megaplasmids, pSymA and pSymB, the number of interacting genes over small windows of the chromosome were calculated and are shown in Figure 5.2 There are distinct

Table 5.1: Screenable phenotypes in ΔG373 in *Sinorhizobium meliloti*

Gene	Name	Accession	GI
	Melibiose Non-utilisation		
agaL2	melibiase	NP_438102.1	16265310
agaL1	melibiase	NP_438107.1	16265315
	C4-dicarboxylate Non-utilisation		
dctA	transporter	NP_438063.1	16265271
dctB	sensor	NP_438064.1	16265272
dctD	transcriptional regulator	NP_438065.1	16265273
	Lactose Non-utilisation		
lacE	transporter	NP_436541.1	16263749
lacF	permease	NP_436542.1	16263750
lacG	permease	NP_436543.1	16263751
lacZ1	β-galactosidase	NP_436544.1	16263752
lacK1	ATP-binding transporter	NP_436545.1	16263753
	Protocatechuate Non-utilisation		
pcaB	3-carboxy-*cis,cis*-muconate cycloisomerase	NP_438027.1	16265235
pcaG	protocatechuate 3,4-dioxygenase α-chain	NP_438028.1	16265236
pcaH	protocatechuate 3,4-dioxygenase β-chain	NP_438029.1	16265237
pcaC	γ-carboxymuconolactone decarboxylase	NP_438030.1	16265238
pcaD	β-ketoadipate enol-lactone hydrolase	NP_438031.1	16265239
pcaQ	transcriptional activator	NP_438032.1	16265240
	Branched Amino-acid Non-utilisation		
bhbA	methylmalonyl-CoA mutase	NP_437989.1	16265197
	Thiamine Auxotrophy		
thiC	thiamine biosynthesis	NP_438067.1	16265275
thiO	thiamine oxidoreductase	NP_438068.1	16265276
thiG	thiazole synthase	NP_438069.1	16265277
thiE	thiamine-phosphate pyrophosphorylase	NP_438070.1	16265278

Table 5.2: Regions of interest on pSymB in *Sinorhizobium meliloti*

Region	Start	End
ΔG373	1452882	26409
∇1	26409	106128
∇2	735511	770089
∇3	1177742	1371104

Table 5.3: Gene identifiers of ΔG373 genes in *Sinorhizobium meliloti*

16263749	16265124	16265164	16265203	16265242	16265281
16263750	16265125	16265165	16265204	16265243	16265282
16263751	16265126	16265166	16265205	195970018	16265283
16263752	16265127	16265167	16265206	16265245	16265284
16263753	16265128	195970024	16265207	16265246	16265285
16263754	16265129	16265169	16265208	16265247	16265286
16263755	16265130	16265170	16265209	16265248	16265287
16263756	16265131	16265171	16265210	16265249	16265288
16263757	16265132	16265172	16265211	16265250	16265289
16263758	16265133	16265173	16265212	16265251	16265290
16263759	16265134	16265174	16265213	16265252	16265291
16263760	16265135	16265175	16265214	16265253	16265292
16263761	16265136	16265176	16265215	16265254	16265293
16263762	16265137	16265177	16265216	16265255	16265294
16263763	16265138	16265178	16265217	16265256	16265295
16263764	16265139	16265179	195970022	16265257	195970017
16263765	16265140	16265180	195970021	16265258	16265297
16263766	16265141	16265181	16265220	16265259	16265298
16263767	16265142	16265182	195970020	16265260	16265299
16263768	16265143	16265183	195970019	16265261	16265300
16263769	16265144	16265184	16265223	16265262	16265301
16263770	16265145	16265185	16265224	16265263	195970016
16265107	16265146	16265186	16265225	16265264	16265303
16265108	16265147	16265187	16265226	16265265	16265304
16265109	16265149	16265188	16265227	16265266	16265305
16265110	16265150	16265189	16265228	16265267	16265306
16265111	16265151	16265190	16265229	16265268	16265307
16265112	16265152	16265191	16265230	16265269	16265308
16265113	16265153	16265192	16265231	16265270	16265309
16265114	16265154	16265193	16265232	16265271	16265310
16265115	16265155	16265194	16265233	16265272	16265311
16265116	16265156	16265195	16265234	16265273	16265312
16265117	16265157	16265196	16265235	16265274	16265313
16265118	16265158	16265197	16265236	16265275	16265314
16265119	16265159	195970023	16265237	16265276	16265315
16265120	16265160	16265199	16265238	16265277	16265316
16265121	16265161	16265200	16265239	16265278	16265317
16265122	16265162	16265201	16265240	16265279	16265318
16265123	16265163	16265202	16265241	16265280	

Table 5.4: Gene identifiers of $\nabla 1$ genes in *Sinorhizobium meliloti*

16263771	16263784	16263797	16263809	16263822	16263835
16263772	16263785	16263798	16263810	16263823	16263836
16263773	16263786	16263799	16263811	16263824	16263837
16263774	16263787	16263800	16263812	16263825	16263838
16263775	16263788	16263801	16263813	16263826	16263839
16263776	16263789	16263802	16263814	16263827	16263840
16263777	16263790	16263803	16263815	195970069	16263841
16263778	16263791	16263804	16263816	16263829	16263842
16263779	16263792	16263805	16263817	16263830	16263843
16263780	16263793	16263806	16263818	16263831	16263844
16263781	16263794	16263807	16263819	16263832	16263845
16263782	16263795	16263808	16263820	16263833	
16263783	16263796	195970076	16263821	16263834	

Table 5.5: Gene identifiers of $\nabla 2$ genes in *Sinorhizobium meliloti*

16264421	16264427	16264433	16264439	195970043	16264452
16264422	16264428	16264434	195970044	16264447	16264453
195970046	16264429	16264435	16264441	16264448	16264454
16264424	16264430	16264436	16264442	16264449	16264455
16264425	195970045	16264437	16264443	16264450	16264456
16264426	16264432	16264438	16264444	16264451	16264457

Listing 5.1: Calculation of functional connectivity between the chromosome and pSymA to $\nabla 1$, $\nabla 2$, $\nabla 3$, and the whole of pSymB

```
do S_meliloti
u1 = (map gi [from "u1.gi"]):flatten
u2 = (map gi [from "u2.gi"]):flatten
u3 = (map gi [from "u3.gi"]):flatten
remainder = foldl except pSymB [u1,u2,u3] :genesof
whole = pSymB:genesof
map (\c -> ((c & pSymB):interactioncount) * (map (\g -> (\i -> i:interactioncount*i:genecount)
    (c :near 1 g)) [u1,u2,u3,remainder, whole])) [chromosome, pSymA]
```

Listing 5.2: Function to produce a position-based count of the number of genes on target chromosome interacting with a source chromosome, divided into windows

```
connectivity window target gilist = map (\g -> target :cut 0.9 :near 1 (map gi (gilist:slice g
    (g:add window)):flatten):genecount :echo) (1:range (gilist:length:div window))
```

Table 5.6: Gene identifiers of ∇3 genes in *Sinorhizobium meliloti*

16264822	16264855	195970031	16264921	16264954	16264987
16264823	16264856	16264889	16264922	16264955	16264988
16264824	16264857	16264890	16264923	16264956	16264989
16264825	16264858	16264891	16264924	16264957	16264990
16264826	16264859	16264892	16264925	16264958	16264991
16264827	16264860	16264893	16264926	16264959	16264992
16264828	195970033	16264894	16264927	16264960	16264993
16264829	16264862	16264895	16264928	16264961	16264994
16264830	16264863	16264896	16264929	16264962	16264995
16264831	16264864	16264897	16264930	16264963	16264996
16264832	16264865	16264898	16264931	16264964	16264997
16264833	16264866	16264899	16264932	16264965	16264998
16264834	16264867	16264900	16264933	16264966	16264999
16264835	16264868	16264901	16264934	16264967	16265000
16264836	16264869	16264902	16264935	16264968	16265001
16264837	16264870	16264903	16264936	16264969	16265002
16264838	16264871	16264904	16264937	16264970	16265003
16264839	16264872	16264905	16264938	16264971	16265004
16264840	16264873	16264906	16264939	16264972	16265005
16264841	16264874	16264907	16264940	16264973	16265006
16264842	16264875	16264908	16264941	16264974	16265007
16264843	16264876	16264909	16264942	16264975	16265008
16264844	16264877	16264910	16264943	16264976	16265009
16264845	195970032	16264911	16264944	16264977	16265010
16264846	16264879	16264912	16264945	16264978	16265011
16264847	16264880	16264913	16264946	16264979	16265012
16264848	16264881	16264914	16264947	16264980	16265013
16264849	16264882	16264915	16264948	16264981	195970030
16264850	16264883	16264916	16264949	16264982	16265015
16264851	16264884	16264917	16264950	16264983	16265016
16264852	16264885	16264918	16264951	195970074	16265017
16264853	16264886	16264919	16264952	16264985	195970029
16264854	16264887	16264920	16264953	16264986	

Listing 5.3: Identification of gene differences between *Ochrobactrum anthropi* and *Sinorhizobium meliloti* in the neighbourhoods surrounding screenable phenotypes

```
do S_meliloti
screen = map (\label*l -> label*((map gi l):flatten)) [from "screen.lst"]
corepct l = foldl (\c->\g -> c:add (if g:coreicity :eq 1 then 1 else 0)) 0 l * (l:length)

info g = map (\l->l:genesof :corepct) [(S_meliloti :near 1 g), (O_anthropi_ATCC49188 :near 1 g
    ), (S_meliloti & O_anthropi_ATCC49188) :near 1 g]

map (\label*l -> label*l:corepct*l:info*((foldr (\g->\i->i& O_anthropi_ATCC49188:near 1 [g])
    O_anthropi_ATCC49188 l - S_meliloti):genecount)) screen
```

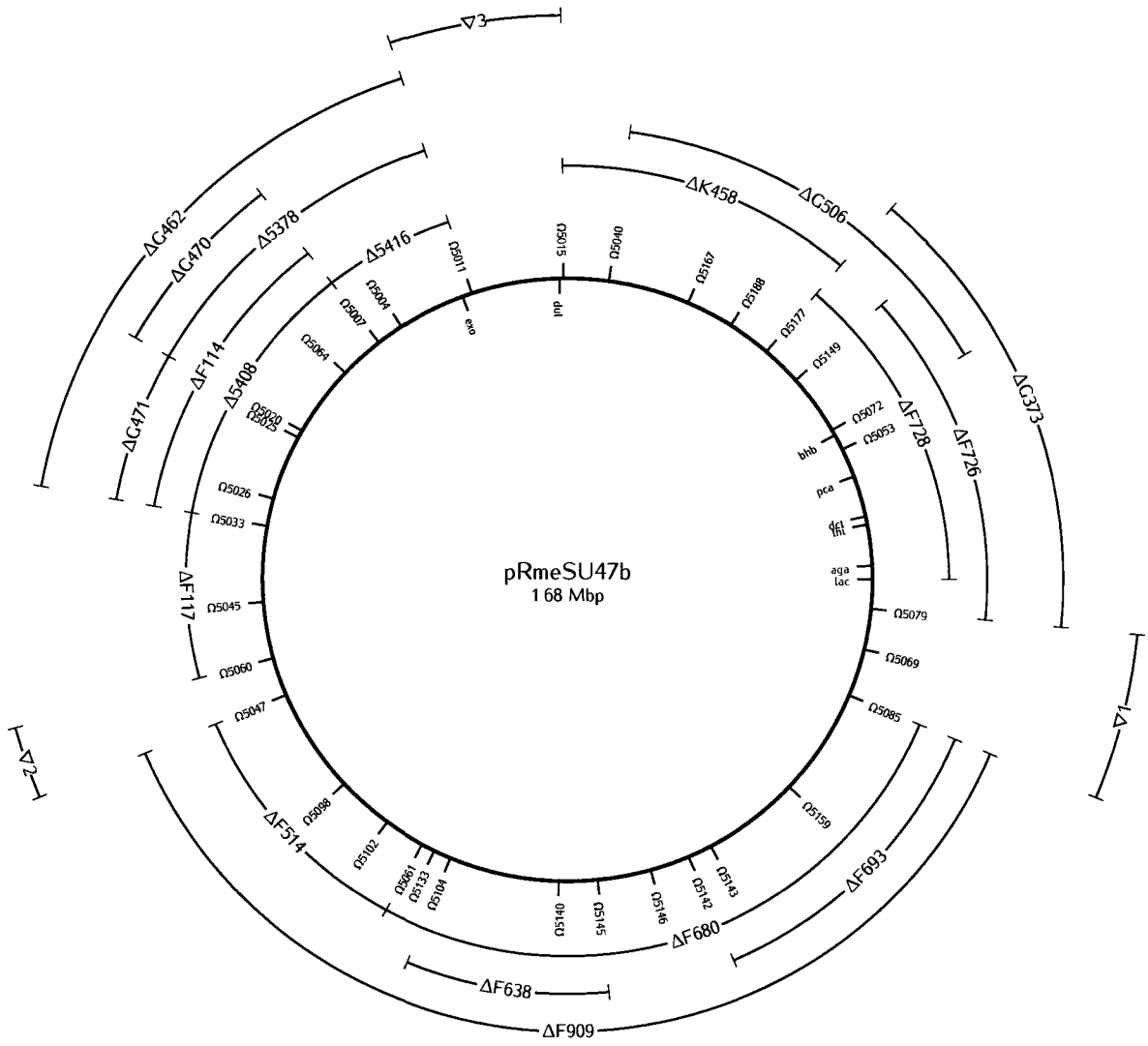


Figure 5.1: Map of pRmeSU47b, a derivative of pSymB from *Sinorhizobium meliloti* with inserted transposons (labelled Ω*x*), screenable phenotypes in ΔG373, and constructed deletions shown [41, 42]

Table 5.7: Number of canonical genes in selected regions of the *Sinorhizobium meliloti* genome

Region	Genes
Main Chromosome	3359
pSymA	1290
pSymB	1569
∇1	76
∇2	36
∇3	243
Remaining pSymB	1214

Table 5.8: Interactions between regions of pSymB against the chromosome and pSymA in *Sinorhizobium meliloti*

		pSymB Region				
		∇1	∇2	∇3	Remainder	Total
Other Region		**Interactions**				
	Chromosome	8992	4973	26070	66866	71196
	pSymA	274	343	779	4362	5279
		Interaction Density ($10^{-4}\frac{\text{interactions}}{\text{gene}^2}$)				
	Chromosome	352	411	319	127	135
	pSymA	28	74	25	28	26
Other Region		**Connecting Genes**				
	Chromosome	554	451	1084	2247	2337
	pSymA	114	110	232	653	704
		Connecting Gene Density ($10^{-4}\frac{1}{\text{gene}}$)				
	Chromosome	22	37	13	6	4
	pSymA	12	25	7	4	3

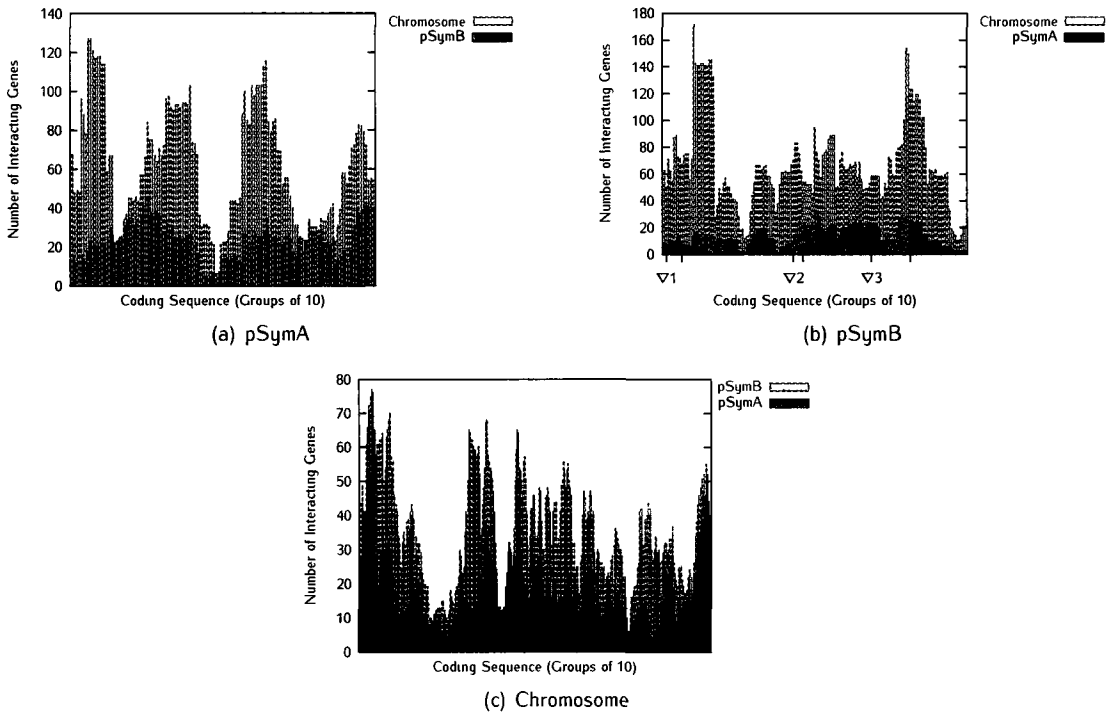


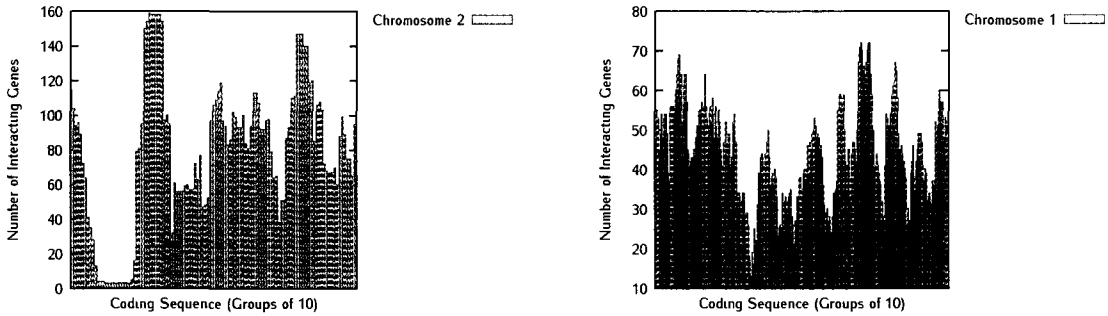
Figure 5.2: Counts of interacting genes for groups of 10 adjacent genes in *Sinorhizobium meliloti*

patterns of heavily interacting regions and there are different levels of interaction with other replicons in some regions For comparison, this was repeated on *Vibrio cholerae* O395, shown in Figure 5 3, and *Burkholderia cenocepacia* MC0-3, shown in Figure 5 4[1]

For the functional targets shown in Table 5 1, analysis shows unexpected conservation and interaction patterns, including genes where fuzzy membership is non-zero In Table 5 9, for each gene group, the number of identified genes in *S meliloti* is shown as well as the number of genes in *O anthropi* that do not have an ortholog, as specified by reciprocal best hit (RBH) For each organism, the number genes in the union of 1° neighbours for all the genes in the group is shown as well as the number of those genes which have orthologs Next, the number of genes which are found in the union of the neighbourhoods for each organism Finally, the number of genes which are in the genes intersection of the 1° neighbourhood of each cluster in *O anthropi* which have no ortholog in *S meliloti* are shown

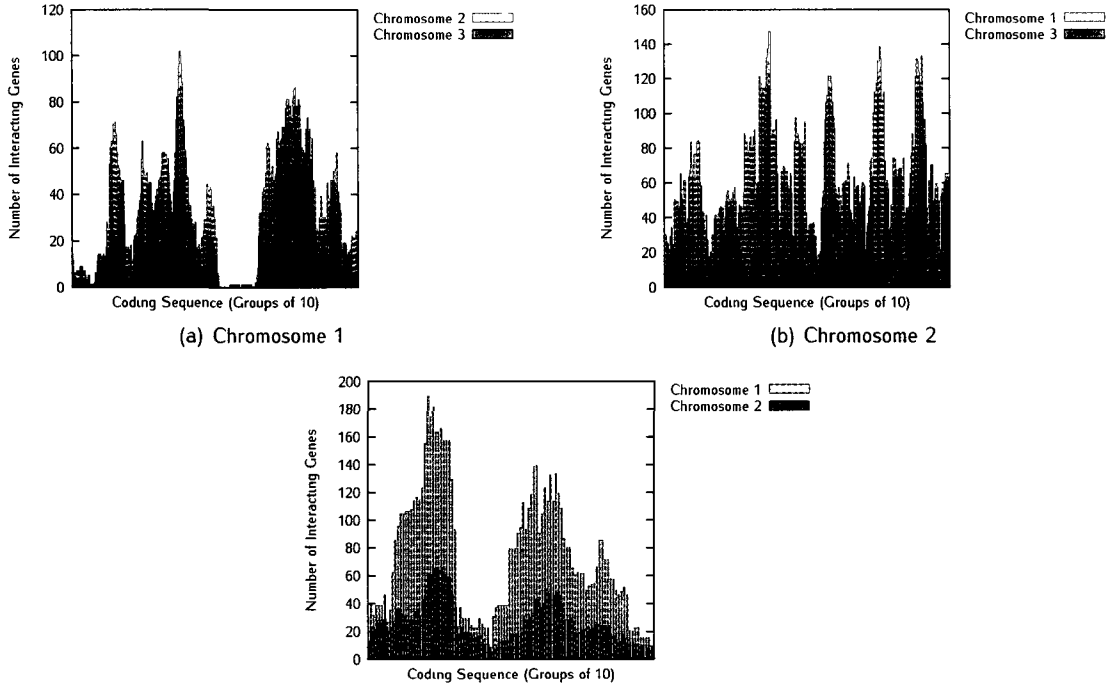
For the *aga* group, neither protein was conserved, making the analysis pointless The *dct* group, which is necessary for nodulation in *S meliloti*, has much higher conservation with the majority of genes and interactions

[1]These organisms were selected based on [43]



(a) Chromosome 1 (b) Chromosome 2

Figure 5.3: Counts of interacting genes for groups of 10 adjacent genes in *Vibrio cholerae* O395



(a) Chromosome 1 (b) Chromosome 2

(c) Chromosome 3

Figure 5.4: Counts of interacting genes for groups of 10 adjacent genes in *Burkholderia cenocepacia* MC0-3

Table 5.9: Comparison of functional complementation targets between *Sinorhizobium meliloti* and *Ochrobactrum anthropi*

Gene Cluster	aga	dct	lac	pca	bhb	thi
Number of Genes	2	3	5	6	1	4
Unmatched Genes	2	0	4	0	0	1
***S. meliloti* Neighbours**	36	163	102	126	47	174
S. meliloti* Neighbours Not Conserved in *O. anthropi	36	36	55	34	5	43
***O. anthropi* Neighbours**	0	183	42	133	51	134
O. anthropi* Neighbours Not Conserved in *S. meliloti	0	23	2	30	8	26
Conserved Neighbourhood	0	100	15	59	38	87
***O. anthropi* Specific**	0	0	0	1	8	0

conserved in both organisms For the *lac* group, one of the proteins was conserved even though the biochemical function of the group is not present in *O anthropi* In the *pca* group which metabolises protochatecuate, there was high overlap of the pathway, though there was one gene which was, in *O. anthropi*, connected to all the genes in the group, but has no equivalent in *S meliloti* There are eight such genes connected to *bhb*, though this is more likely as, while the single gene will have several non-conserved genes, the intersection will be much smaller as there is a lower probability of any particular gene being conected to every gene of interest Finally, for the *thi* group, one gene in the pathway is not conserved This pathway is responsible for thiamine biosynthesis Since *O anthropi* is not a thiamine auxotroph, the biochemical step performed by the missing gene must be carried out by an alternate, non-homologous enzyme, likely one of the 26 well-connected genes unique to *O anthropi*

5.3 Discussion

Analysis of the deletions of pSymB from *S meliloti* and the possible complementation targets in *O anthropi* yields interesting results There is a high-level organisation of pSymB where functions interacting with the main chromosome cluster into hotspots, although the reason is unknown Strangely, the *lac* cluster has conserved genes which should not be functional in *O anthropi*. The *dct* cluster, which is important for carbon update during symbiosis in *S meliloti*, is well-conserved and the *pca* cluster is well-conserved, but has an additional *O anthropi*-specific gene of interest Finally, the *thi* cluster has an unclear conservation pattern that requires complementation experiments to resolve

The high density of interactions between some regions of pSymB and the rest of the genome in *S meliloti*

point to a more complex organisation of the genome If essential genes were transferred to pSymB, there is no reason for the regions surrounding them to contain more genes interacting with the rest of the genome The density suggest that these genes are under some selective pressure to cluster, the pressure is likely weak, but any clustering is an indication of selective pressure, otherwise genes or operons would drift apart just as quickly [44] Recent work looking at mutation rates on secondary chromosomes, suggests that the replication process for secondary chromosomes allows the cell cycle to influence the activity of genes based on their position [43] The clustering of these genes may be a similar effect It is worth noting that $\nabla 2$ and $\nabla 3$ are approximately the same distance from $\nabla 1$, which contains the origin of replication, consistent with that hypothesis Future work should focus on determining the forces that shape these patterns of this interaction clustering. Ideally, if more is known about the function of gene clusters in the *V cholerae* O395 and *B cenocepacia* MC0-3 genomes, it should be possible to determine if regions of like-functioning genes cluster There is evidence that evolutionary pressure places genes in locations on chromosomes to control their dosage [43] and it might be possible to compare the locations of genes, based on general functions, across different organisms and correlate this with the amount of cross-replicon interaction

Unfortunately, analysis of the *O anthropi* orthologs for the screenable deletions yielded some surprising results Given *O anthropi* is unable to grow on melibiose [45], the lack of *aga* genes is unremarkable *O anthropi* cannot grow on lactose [45], making the conservation of a gene in the *lac* cluster surprising, even if the other genes are not conserved Given *O anthropi* cannot use lactose as a carbon source, the *lac* cluster is non-functional and the genes it should be purged by selection It is possible that this gene is involved in α-galactosidase activity, mediated by the *aga* cluster, as lactose is cleaved into galactose As the *bhb* cluster consists of a single gene, the results may not be as clear since overlapping provides some reenforcement of the connectivity patterns That being said, there is generally good conservation of neighbouring genes Similarly, the *dct* group has relatively good conservation of neighbouring genes, the conserved gene interactions cover more than two thirds of the total neighbours in either organism. Since *dct* is an important component in metabolism during infection in *S meliloti*, it suggests that *O anthropi* is also using *dct* metabolism during infection despite the difference in host During symbiosis, the plant provides dicarboxylates in large quantity to the bacteria as a carbon source, it seems unlikely that the host of a pathogenic organism would be so generous The *pca* cluster has good conservation, similar to *dct*, even though protochatecuate metabolism is not known to be involved in infection. Interestingly, a single gene is present in *O anthropi*, but is not present in *S meliloti*, that connects to all the genes in the *pca* cluster This gene probably deserves further investigation to determine what additional

abilities it could be providing to *O. anthropi*, likely ability to utilise other carbon sources by degradation to protochatechuate Finally, the *thi* group remains a mystery as one gene is not conserved in *O anthropi*, though *O. anthropi* is not a thiamine auxotroph. Furthermore, if there were a well-connected gene specific to *O anthropi*, it would suggest the existence of a non-orthologous functionally equivalent gene Complementation of *S meliloti* ΔG373 mutant with the *O. anthropi* cassette of *thi* genes would indicate that the *O. anthropi* orthologs have different biochemical abilities or the non-conserved gene is unnecessary, if the complementation works, or that *O anthropi* has an alternate gene performing an equivalent function if complementation fails

Chapter 6

Pathway Completion

MANY BIOCHEMICAL PATHWAYS are only partially known. Often, some enzymes in the pathway are unidentified. Using the deoxyxylulose 5-phosphate (DXP) pathway as a model, functional interaction networks could provide candidates for the unknown genes in the pathway.

6.1 Background

In many cases, most of a biosynthetic pathway will be identified, but the remaining genes will prove difficult to find. Genes in a pathway have a tendency to collect into operons, but since operons are unstable, a crucial gene may not be part of the operon containing the bulk of the biosynthesis pathway. As an example, the terminal gene in the DXP pathway was the last to be discovered [46].

As a number of products are synthesised from the precursor isopentyl diphosphate (IPP), including commercially-relevant ones, such as squalene, a compound used as an adjuvant in vaccines and a moisturiser in cosmetics, determining the final gene in this pathway was of interest for optimising strains for production. In bacteria and archaea, synthesis is carried out using the DXP pathway which uses pyruvate and glyeraldehyde-3-phosphate as substrate, shown in Figure 6.1. The use of IPP is fairly ubiquitous and the genes in the DXP pathway are conserved [46, 47, 48]. As most research in the area focuses on *Escherichia coli*, it was a useful model.

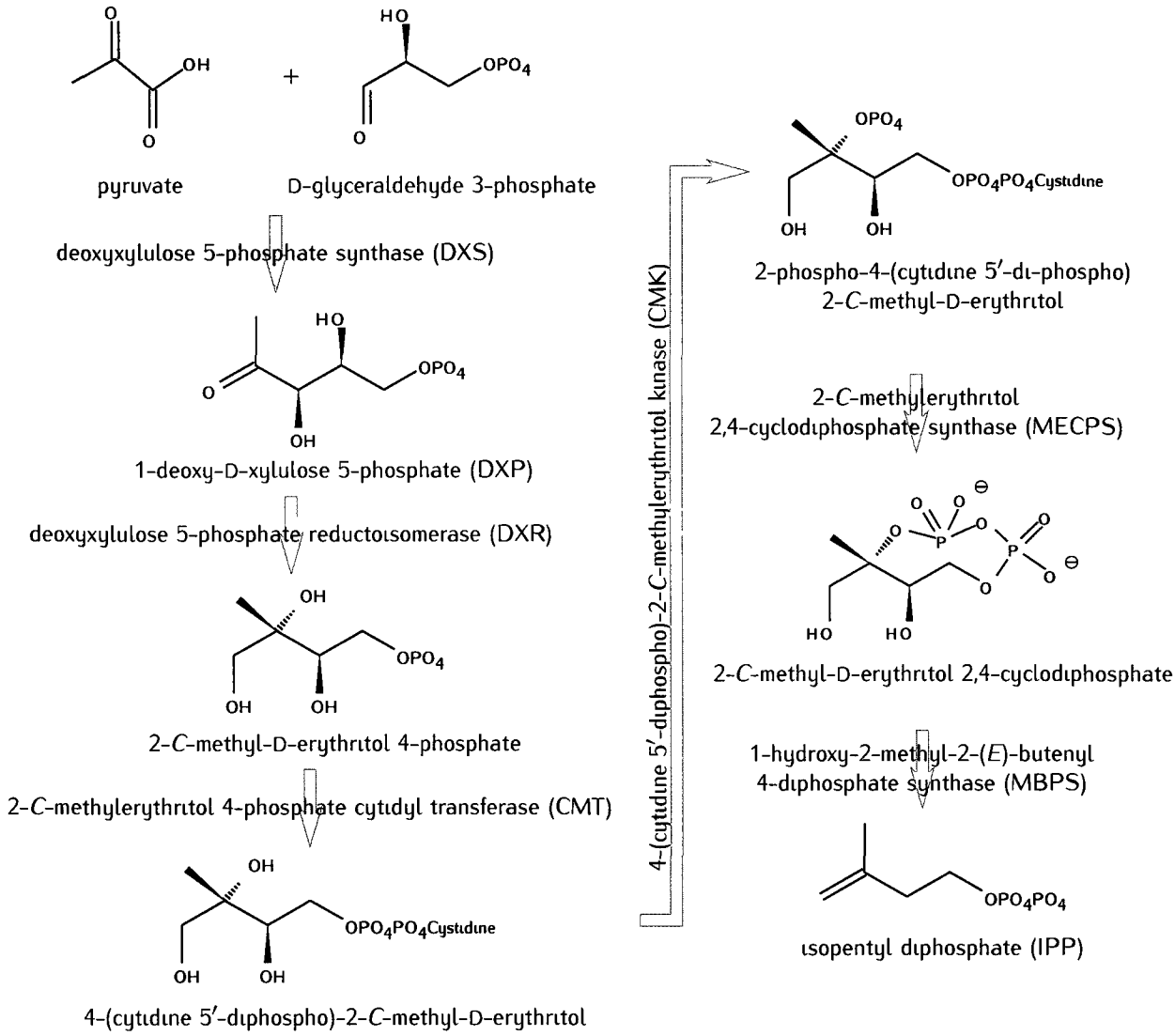


Figure 6.1: Conserved bacterial biosynthesis pathway to produce isopentyl biphosphate via deoxyxylulose 5-phosphate [48]

Table 6.1: Deoxyxylulose 5-phosphate biosynthesis genes [46, 47, 48]

Enzyme	Accession	GI
deoxyxylulose 5-phosphate synthase (DXS)	NP_414954	16128405
deoxyxylulose 5-phosphate reductoisomerase (DXR)	NP_414715	16128166
2-*C*-methylerythritol 4-phosphate cytidyl transferase (CMT)	NP_417227	16130654
4-(cytidine 5'-diphospho)-2-*C*-methylerythritol kinase (CMK)	NP_415726	16129171
2-*C*-methylerythritol 2,4-cyclodiphosphate synthase (MECPS)	NP_417226	16130653
	NP_311627	15832854
1-hydroxy-2-methyl-2-(*E*)-butenyl 4-diphosphate synthase (MBPS)	NP_417010	16130440

6.2 Materials and Methods

E. coli was selected as the model for the DXP pathway analysis Given the pathway is highly conserved, the choice of organism is not expected to be critical to analysis The known genes in the pathway, shown in Figure 6.1, were identified and are shown, in the same order as the figure, in Table 6 1 [46, 47, 48].

To test the efficacy of the method, the final gene in the pathway, MBPS, was eliminated from the list of known genes and the remaining genes were used to find it From these genes, the 1° neighbours were isolated and then the intersection of the neighbours was computed, as shown in Source Listing 6 1 These genes were then compared against the eliminated gene

6.3 Results

Analysis of overlapping neigbhourhoods of known genes in the DXP pathway yielded 61 candidate genes However, the target gene was not present in this set Futher analysis reveals that CMT does not have the target gene in its neighbourhood This shows incompleteness in the functional inferencing done by Nebulon The number of overlapping genes are shown in Table 6 2

6.4 Discussion

Since many pathways have unknown enzymes performing some steps, using the functional inferences to determine the missing enzymes seems like a valuable tool However, even highly conserved pathways have sufficient unreliability to make the method unusable. It is more concerning that the method yields a reasonable number of results, 61 candidate genes in the case of the DXP pathway If no genes were found, it would be conceivable to

Listing 6.1: Query to find unknown deoxyxylulose pathway genes using known genes

```
map do ecoli
(map (\x -> ecoli:unionall : near 1 (gi x):genecoreicity (\g -> \c -> c :gt 30)) [from "
    dxpknown.gi"]):intersectall
```

Table 6.2: Overlapping genes in the neighbourhoods of the deoxyxylulose pathway

Gene Finds Target?	DXS Yes	DXR Yes	CMT Yes	CMK No	MECPS Yes	Total
	■	■	■	■	■	61
	■	■	■	■		34
	■	■	■		■	29
	■	■	■			33
	■	■		■	■	11
	■	■		■		4
	■	■			■	7
	■	■				9
	■		■	■	■	54
	■		■	■		56
	■		■		■	49
	■		■			96
	■			■	■	22
	■			■		7
	■				■	17
	■					31
		■	■	■	■	4
		■	■	■		15
		■	■		■	8
		■	■			23
		■		■	■	35
		■		■		44
		■			■	38
		■				114
			■	■	■	11
			■	■		20
			■		■	17
			■			65
				■	■	91
				■		170
					■	194

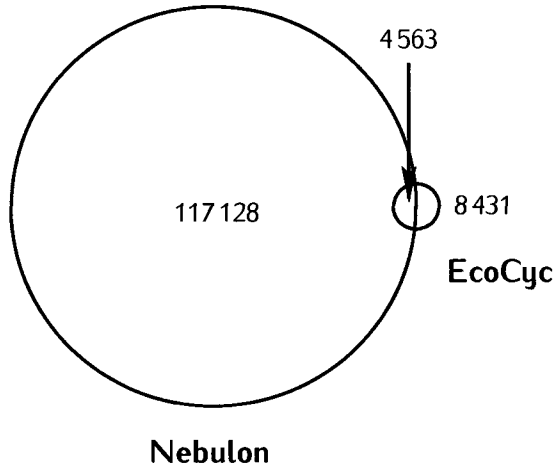


Figure 6.2: Comparison of predicted gene interaction networks from Nebulon and the curated EcoCyc database [36]

expand the search parameters to include more genes. However, since a reasonable number of candidate genes are found, there would be no obvious reason to expand the search parameters

Part of the issue is the general lack of coverage of the predictions A comparison of the predicted interactions from Nebulon versus the curated EcoCyc database [36] shows that 4563 of 8431 interactions (54 122 %) are captured by Nebulon, shown in Figure 6 2 This suggests that, despite the large number of interactions predicted by Nebulon, more are still needed to gain better insight into the biochemistry of the cell For comparison, the Nebulon predictions for *Bacillus subtilis* str 168 were compared against a network derived from a computational metabolic model [49] and, again, the overlap is quite small even though both predicted networks are roughly the same size, this is shown in Figure 6.3 The metabolic network was converted by assuming that proteins sharing common reactants and or products, were interacting, ignoring certain ubiquitous reactants, such as ATP and water

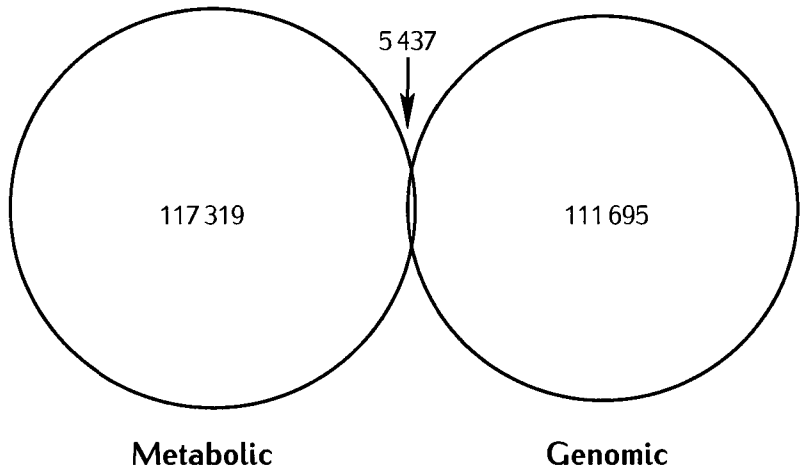


Figure 6.3: Comparison of predicted gene interaction networks from two sources: genomic context (Nebulon) and metabolic reaction inference [49]

Chapter 7

Cytokinin Receptors in the Rhizobiaceæ

BACTERIA engaging in symbiosis with leguminous plants have complex signalling interactions with their hosts. Based on unusual nodule organogenesis in *Pisum sativum* R50 (*sym16*), which accumulates cytokinin [16], the possibility of bacterial response to the plant hormone was investigated

7.1 Materials and Methods

Dr Guinel hypothesised that the rhizobia in symbiosis with *Pisum sativum* may have the ability to communicate via the plant hormone cytokinin as cytokinin mutants have unusual nodulation phenotypes The bacteria could synthesise cytokinin, degrade cytokinin, or sense cytokinin There are two evolutionary hypotheses that the protein responsible for degrading, making, or sensing cytokinin was formed from an existing prokaryotic gene or that the protein was horizontally transferred from the plant to the bacteria. The second hypothesis is testable using sequence-based methods, as performed below

Given a protein is conserved among all the Fabales, some ancestral Fabales genes could be transferred to a common ancestor in the Rhizobiales Therefore, the logical search pattern is to use orthologous proteins from the extant Fabales lineages to build a model of the protein that was likely transferred This model can then be used to search for matching proteins in the Rhizobiales

Starting with the cytokinin oxidase[1] from *Arabidopsis thaliana*, a model was built using Ψ-BLAST [2] against matching cytokinin oxidase genes in the Fabales This model was then used to search the Rhizobiales for

[1]Accession NP_565455, GI 18399056

matching proteins The best match had 60% sequence identity to the model and, unfortunately, the matching region was a FAD binding domain[2], not the cytokinin binding domain[3] A model was built of the cytokinin binding domain, but no matches were found in the Rhizobiales

The cytokinin receptor in *Lotus japonicus* has been identified lotus histidine kinase 1 (LHK1)[4] Initially, a Ψ-BLAST model was built using the whole protein. Many high quality hits were found, but only of the histidine kinase effector domain. The receptor, a CHASE domain[5], was isolated and a model built using the Fabales. This model was then used to search the Rhizobiales A high quality match was found to the receptor domain in a *Sinorhizobium meliloti* protein. When this protein was used to search other Rhizobiales, the search results had a bimodal distribution: the best matches had > 90% sequence identity between each other; the next best match was < 50% identity These highly conserved genes were present only in the monophyletic fast fermenting clade [15], a group containing both *Rhizobium leguminosarum viciæ* and *S meliloti* A recent phylogeny [50] is shown in Figure 7 1 with the fast fermenting clade emphasised, not all members of the Rhizobiales are included The matching proteins all had a second conserved domain a LuxR-like DNA binding domain Lux is a bacterial quorum sensing system where LuxI creates small diffusible molecules that activate LuxR which initiates gene transcription at a specific DNA sequence known as the *lux* box The structure of this protein suggests that it is a fusion protein of the CHASE receptor and the LuxR effector This evidence alone suggests that this protein is a cytokinin receptor, although more analysis yields further evidence

Using `gisql`, the neighbourhood of genes connected to this protein was analysed for any connection to known plant-symbiosis systems The gene is in the same operon as a putative ethylene receptor, ethylene being another plant hormone whose production is stimulated by cytokinin Since these two receptors are in the same operon, they likely have a related function and this is reinforced given they both seem to sense plant hormones Again, this putative ethylene receptor is highly conserved among the fast fermenting group, as is the gene order. The neighbourhood of the putative receptor includes several transporters and secretion systems, both are systems typically involved with symbiosis A putative heavy metal uptake system was connected and the uptake of iron is important during symbiosis [15] The neighbourhood also included a σ factor A σ factor is a DNA binding protein which recognises promoters and recruits RNA polymerase to begin transcription Expressing a new σ factor indicates that this gene is involved in physiological processes capable of causing

[2]Accession cl10516, PSSM ID 158898
[3]PSSM ID 150064
[4]Accession ABI48271, GI 113911570
[5]Accession cl01369, PSSM ID 154357

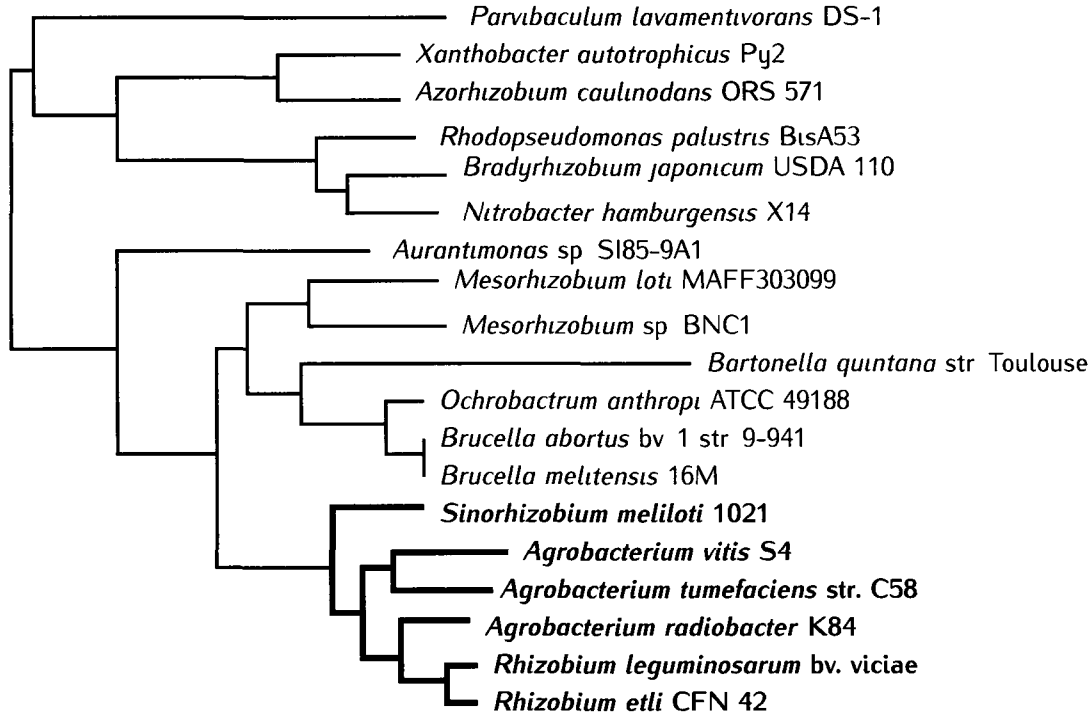


Figure 7.1: Phylogeny of Rhizobiales with the monophyletic fast-fermenting strains bold, which all contain the putative cytokinin receptor [50, 15]

Table 7.1: Putative cytokinin receptors and ethylene receptors

Gene		*S. meliloti* 1021	*R. leguminosarum viciæ* 3841
Putative Cytokinin Receptor	Name	SMa0206	pRL120282
	GI	14523174	4398113
	Locus	113822	299040
	Strand	+	-
Ethylene Histidine Kinase Receptor	Name	SMa0204	pRL120283
	GI	1235333	4398114
	Locus	112578	299776
	Strand	+	-

massive shifts in the transcription profile of the organism, again consistent with plant symbiosis Unfortunately, the Nebulon software does not likely provide accurate information about the genetic network surrounding σ factors as the mechanism of connectivity is not detectable by the methods described in Section 1 1.2, so no further insight can be gained

To test if this gene is involved in nodule organogenesis, bacteria lacking the putative receptor would be placed on host plants and nodule organogenesis observed The organisms of choice were *Sinorhizobium meliloti* 1021 which nodulates *Medicago sativa* and *Medicago truncatula*, and *Rhizobium leguminosarum viciæ* 3841 which nodulates *P sativum*. First, knockouts needed to be made both the putative cytokinin receptor and the putative ethylene receptor would be removed and replaced with a reporter construct The putative receptor genes in these organisms are shown in Table 7 1. The knockout system was designed such that the putative cytokinin receptor and ethylene receptor would be cleanly deleted and a BioBrick® coding sequence to be inserted under control of the native promoter, as a reporter The knockout also includes a cloning site for an Ω fragment [51] selection marker to be inserted The destination vector is pJQ200SK [52] as this vector allows homologous recombination with the selectable marker *sacB* The designed fragments are shown in Figure 7 2

Knockouts will be constructed using the method shown in Figure 7 3 The BioBrick® reporter used will be BBa_E0040, a green fluorescent protein derived from the jellyfish *Æqueora victoria*[6] Continued work by Dr Guinel and Dr Charles will construct these knockouts and determine if these knockouts will show unusual nodule organogenesis patterns on their respective hosts

[6]SwissProt P42212

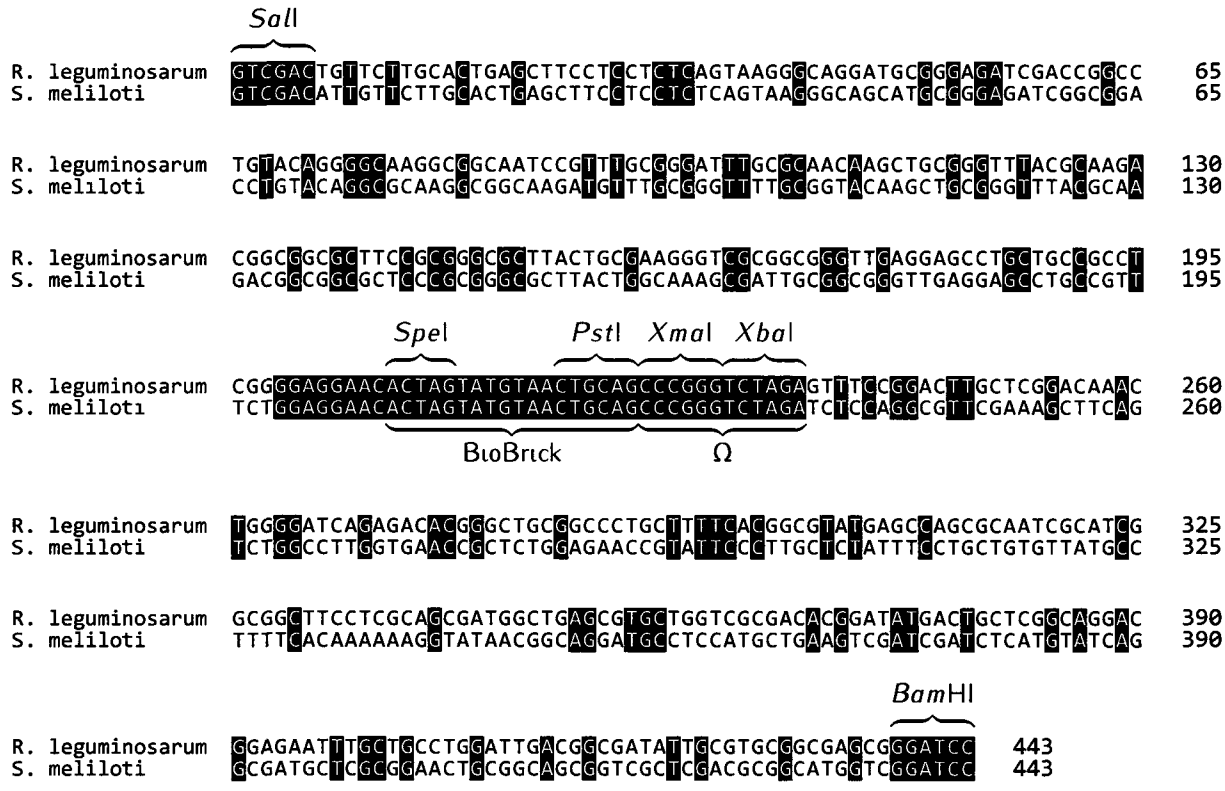


Figure 7.2: Knockout cassettes synthesised for deletion of putative cytokinin receptor and ethylene receptor in model organsisms

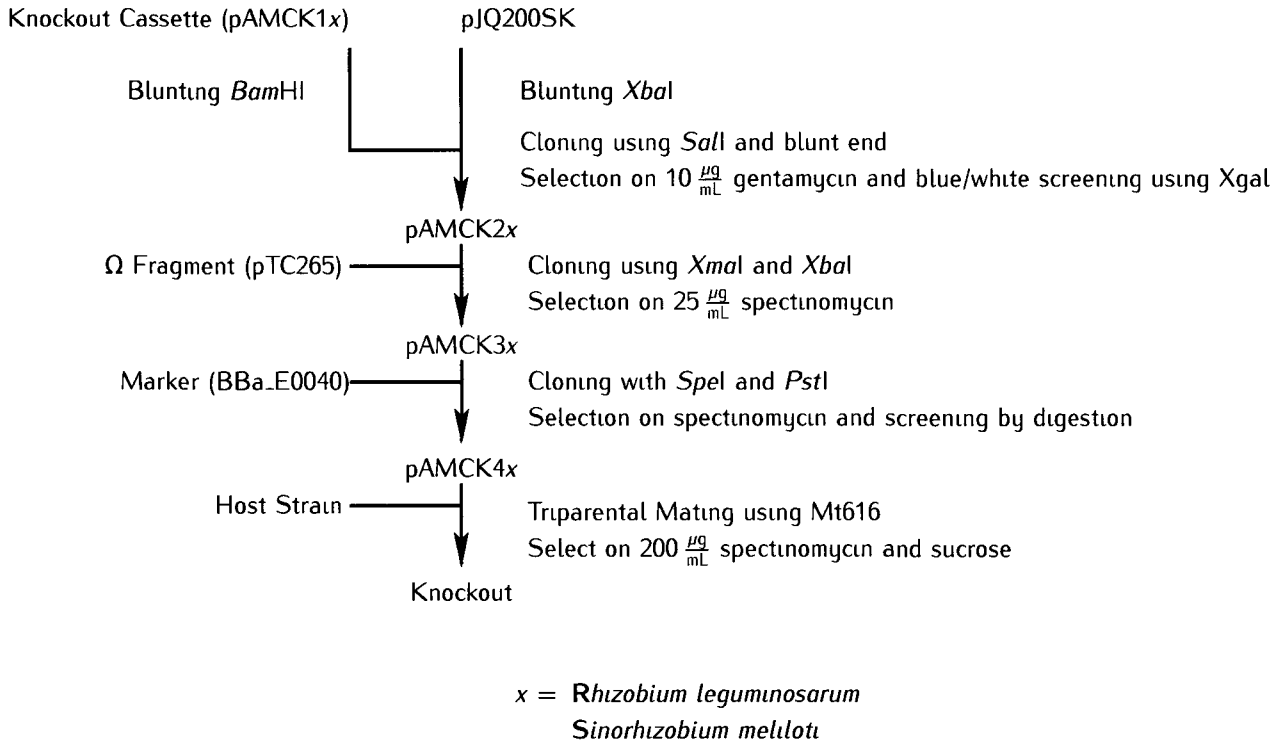


Figure 7.3: Overview of knockout cloning and recombination procedure

7.2 Results

Based on the available evidence, there is likely a cytokinin receptor in the Rhizobiaceæ This cannot be confirmed without future work However, there are many encouraging signs Under the assumption that cytokinin reception was transferred from the host plant, the protein found is ideal as it has a cytokinin-sensitive domain that is similar to a family of well-conserved cytokinin receptors in the plant This protein is well conserved over a monophyletic group of bacteria which all engage in symbiotic or parasitic relationships with plants. The predicted genetic interactions of this protein in *S meliloti* are all in systems only indirectly connected with symbiosis While it would be ideal to have a more direct connection to symbiosis-related genes, it is likely that a protein directly regulating nitrogen fixation would have been discovered earlier. Finally, the conserved putative ethylene receptor upstream is another indicator of plant interaction as ethylene is a plant hormone There is little else, from a bioinformatics perspective, that can shed light on the function of this protein

In order to determine if the protein is a cytokinin receptor, two knockouts *S meliloti* ΔSMa0204–SMa0206 and *R leguminosarum viciæ* 3841 ΔpRL120283–pRL120282 will be created The activity of the promoter driving expression of these genes can be monitored as a green fluorescent protein reporter was inserted. The *S. meliloti* ΔSMa0204–SMa0206 will be tested on *M sativa* and *R leguminosarum viciæ* 3841 ΔpRL120283–pRL120282 will be tested on the wild-type *Pisum sativum* Sparkle and the cytokinin-retaining *P sativum* R50 and the nodule organogenesis observed If cytokinin signalling between the bacteria and host is an important part of nodule organogenesis, the mutant *P sativum* R50, given it accumulates cytokinin, should have a different nodulation pattern than the wild-type. Wild-type bacteria have different nodulation phenotypes between the wild-type and mutant hosts, it seems likely that some of the nodulation patterns seen with wild-type bacteria on cytokinin-retaining mutants, including delayed nodule formation and nodule abortion could be seen However, if the bacteria fail to sense cytokinin, the nodulation phenotypes in the plant driven by signalling from the bacteria should be identical regardless of the level of cytokinin in the host plant

7.3 Discussion

Based on the evolutionary evidence collected, the most parsimonious explanation of the evolution of the putative cytokinin receptor is shown in Figure 7 4 Without more knowledge of the evolution of symbiosis it would be difficult to suggest whether the cytokinin receptor was transferred before or after the symbiotic association

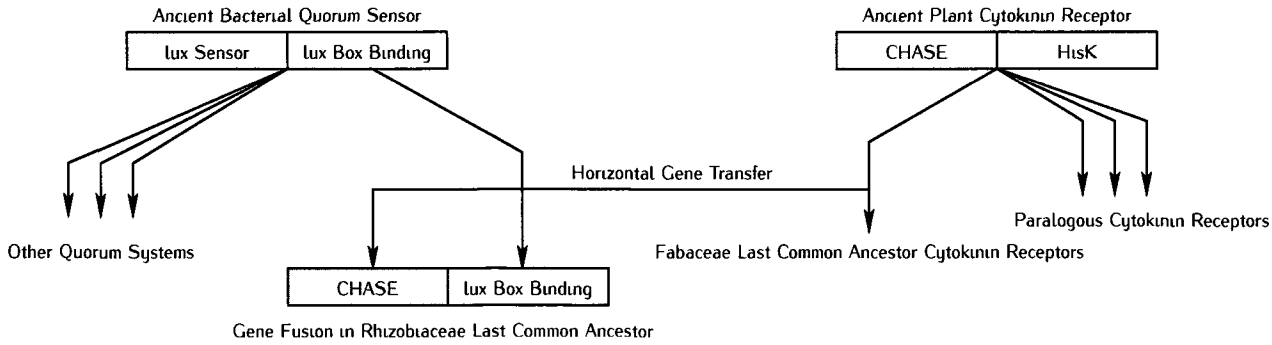


Figure 7.4: Probable evolutionary history of putative cytokinin receptor in fast-fermenting Rhizobiaceae by horizontal gene transfer of cytokinin-binding CHASE domain from host plant

between legumes and Rhizobiaceæ began, though gene transfer between organisms in symbiotic or parasitic relations are frequent The Rhizobiaceæ, particularly *S meliloti*, is quite dependent on its host and is even killed by its host during nodule senescence, whereas *Bradyrhizobium japonicum* is able to escape the nodule, this further highlights the tight relationship between the host plants and the Rhizobiaceæ common ancestor Moreover, there are several paralogs of the cytokinin receptors in plants, including the Fabaceæ, and several lux-like quorum sensing systems in most bacteria, including at least five in *S meliloti*, giving ample raw material for such a gene fusion to form. It would be extremely difficult, and probably not worthwhile, to determine which of these copies was ancestor to the fused gene

Only further analysis will reveal if this gene is indeed a cytokinin receptor This would require determining

1. if the bacteria lacking the putative receptor have unusual or failed nodule organogenesis
2. the binding domain of the receptor
3. if binding to cognate DNA varies based on the presence of cytokinin
4. the role of the neighbouring ethylene receptor in binding or activation

If it is a receptor and the binding domain is determined, then the physiological processes controlled by this gene can be determined

Chapter 8

Conclusion

DEVELOPMENT of **gisql** provided a very efficient and flexible way to analyse the model systems previously discussed The simple, composable functions allowed analysis of a wide variety of organisms and genetic patterns Using **gisql**, definite connectivity patterns in the pathogenicity mechanisms of *Escherichia coli* were revealed and, in a very different way, the connectivity patterns in between the pSymB megaplasmid and the chromosome in *Sinorhizobium meliloti* were also revealed Assistance in wet-lab projects was also provided by finding additional information for functional complementation of *S meliloti* using genes from *Ochrobactrum anthropi* Additionally, the feasibility of using functional interaction networks was assessed using the well-conserved deoxyxylulose 5-phosphate (DXP) pathway by removing a gene from the pathway and attemping to recover it by computational analysis, though this turned out to be impractical. Using other methods in addition to **gisql**, a putative cytokinin receptor conserved in the Rhizobiaceæ was found which likely plays a role in regulation of the bacteria when engaging in interaction with a host plant

Functional interaction networks have provided several insights into the evolutionary history of organisms Analyses suggest higher level genome organisation in some cases and, yet, contradicts the usual conception of a biochemical pathway At any rate, using **gisql** to manipulate functional interaction networks can provide clues to assist in wet-lab biology

8.1 Recommendations

While there are several additions to the `gisql` language that would be beneficial, adding an optimiser capable of using the database in a way that does not require loading all interactions into main memory is the most needed Such an addition would significantly improve performance and allow manipulations of larger interaction networks

The analysis of the regions for which deletions cannot be made in *S meliloti* needs further analysis to determine if there is a pattern to find essential regions It seems that, with more examples, a clear statistical pattern could emerge and be used to predict essential genes in multi-chromosomal systems In *O. anthropi*, two genes identified, the conserved *lac* gene and the unknown gene well-connected to the protochatechuate gene, should be investigated further The gene in the *lac* operon may be involved in α-galactoside metabolism and this could be verified by a simple knockout The unknown gene connected to protochatechuate metabolism is likely involved in metabolism a different protochatechuate-like precursor Further sequence analysis may find a homolog with some known biochemical activity

The initial attempt at pathway discovery, using the DXP pathway, is not necessarily an indication that the method is flawed, only that it does not work for certain pathways To understand the mode of failure, the analysis should be repeated on a large number of known pathways of different types, including catabolic versus anabolic pathways, pathways at difference distances from core metabolism, and pathways with different levels of conservation That analysis could lead to an understanding of when the method is appropriate or, at least, a better understanding of why it does not work as expected

For the putative cytokinin receptor, the first step is to make knockouts and allow them to nodulate their respective hosts The element bound by the lux-like binding domain should be determined as well as the sensitivity of the protein to cytokinin Additionally, the sensitivity of the putative ethylene receptor to ethylene must be determined and the proteins it phosphorylates

Summary

PREDICTED FUNCTIONAL INTERACTION NETWORKS have provided insight into the connectivity patterns of *Escherichia coli* pathogenicity genes, the organisation of the pSymB megaplasmid in *Sinorhizobium meliloti*, potential gene complementation targets for *S. meliloti* using *Ochrobactrum anthropi*, methods for finding unknown genes in biosynthetic pathways, and a putative cytokinin receptor in the plant-associated Rhizobiaceæ.

Using `gisql`, a computational tool designed to allow powerful and flexible manipulations of the functional interaction networks from multiple organisms, connectivity patterns of genes and genes associated with certain phenotypes can be extracted. This tool provides an efficient means to pose questions about the abilities of organisms based on the organisation of their genes and to compare interactions between different methods of obtaining them.

In *E. coli*, there is a strong tendency for genes to interact with genes abundant in the pan-genome. This tendency is particularly true for genes of low abundance. Also, pathogenicity genes have a tendency to interact not only with high-abundance genes, but some low-abundance genes, usually other pathogenicity genes. Nebulon's predicted interactions [3] between genes found at different abundances in the *E. coli* pan-genome exhibit similar connectivity patterns as the interactions described in the curated biochemical databases EcoCyc [36] and KEGG [37, 38, 39].

In *S. meliloti*, previous transposon-mediated deletions of the megaplasmid pSymB yielded three regions for which deletions could not be made [41]. These three regions show significantly enriched connectivity to the main chrosome and the pSymA megaplasmid compared with the remainder of pSymB. There are several screenable phenotypes in the deletions of pSymB and complementation targets from *O. anthropi* were identified. The carbon utilisation systems for lactose and melibiose, do not have orthologs in *O. anthropi* while the carbon utilisation systems for C4-dicarboxylate and protochatechuate are well conserved and a gene unique to *O. anthropi* was found that was well connected to the orthologs responsible for protochatechuate metabolism. Finally, one of

the thiamine biosynthesis genes lacks an ortholog even though *O anthropi* is not a thiamine auxotroph; no candidate for a biochemically equivalent gene was found

Using the pathway which produced isopentyl diphosphate via deoxyxylulose 5-phosphate as a model, a method was tested to determine if it is possible to find a missing gene in a biosynthetic pathway given most of the pathway is known When tested with the model pathway, less the final gene, the method found 61 candidate genes, none of them the correct gene. The correct gene was not predicted to be connected to a gene in the pathway, excluding it from the candidate list, and, given the number of candidates, it seems unreasonable to relax the search conditions This analysis should be expanded to more pathways to determine if this is a typical case

A putative receptor for the plant hormone cytokinin was found in the plant symbionts, the Rhizobiaceæ The candidate gene possesses a binding domain matching the cytokinin binding sensing domain in their hosts, this domain is coupled to a bacterial transcription factor DNA binding domain The gene is in the same operon as a putative ethylene receptor, ethylene is also a plant hormone These two genes, in the same order, are conserved with extremely high sequence similarity across all the Rhizobiaceæ and the genes predicted to interact with this gene suggest activities that would be involved with plant symbiosis

Literature Cited

[1] R L Tatusov, D A Natale, I V Garkavtsev, T A Tatusova, U. T Shankavaram, B S. Rao, B Kiryutin, M Y. Galperin, N D. Fedorova, and E V Koonin, "The COG database new developments in phylogenetic classification of proteins from complete genomes," *Nucleic Acids Research*, vol 29, no 1, pp 22–8, 2001

[2] S. F Altschul, T. L Madden, A A Schaffer, J. Zhang, Z Zhang, W Miller, and D J Lipman, "Gapped BLAST and PSI-BLAST a new generation of protein database search programs," *Nucleic Acids Research*, vol 25, no 17, pp 3389–3402, 1997

[3] S C Janga, J Collado-Vides, and G Moreno-Hagelsieb, "Nebulon: a system for the inference of functional relationships of gene products from the rearrangement of predicted operons," *Nucleic Acids Research*, vol 33, no 8, pp 2521–30, 2005

[4] W M Fitch, "Homology. a personal view on some of the problems," *Trends in Genetics*, vol 16, no 5, pp 227–231, 2000

[5] A M Altenhoff and C Dessimoz, "Phylogenetic and functional assessment of orthologs inference projects and methods," *PLoS Computational Biology*, vol 5, no 1, p e1000262, 2009

[6] E L L Sonnhammer and E V Koonin, "Orthology, paralogy and proposed classification for paralog subtypes," *Trends in Genet ics*, vol. 18, no 12, pp 619–20, 2002

[7] T Hulsen, M A Huynen, J de Vlieg, and P M Groenen, "Benchmarking ortholog identification methods using functional genomics data," *Genome Biology*, vol. 7, no R31, 2006

[8] R L Tatusov, E. V Koonin, and D. J Lipman, "A genomic perspective on protein families," *Science*, vol 278, no 5338, pp 631–637, 1997

[9] R Overbeek, M. Fonstein, M D'Souza, G D Pusch, and N Maltsev, "The use of gene clusters to infer functional coupling," *Proceedings of the National Academy of Science of the United States of America*, vol 96, no March, pp. 2896–2901, 1999

[10] W C. Lathe, B Snel, and P. Bork, "Gene context conservation of a higher order than operons," *Trends in Biochemical Sciences*, vol. 25, pp. 476–479, 2000

[11] J G. Lawrence, R W Hendrix, and S. Casjens, "Where are the pseudogenes in bacterial genomes?," *Trends in Microbiology*, vol 9, no 11, 2001

[12] M A. Croxen and B. B. Finlay, "Molecular mechanisms of *Escherichia coli* pathogenicity," *Nature Reviews Microbiology*, vol 8, no 1, pp 26–38, 2010

[13] A L Lloyd, D A Rasko, and H L. T Mobley, "Defining genomic islands and uropathogen-specific genes in uropathogenic *Escherichia coli*," *Journal of Bacteriology*, vol. 189, no. 9, pp 3532–3546, 2007

[14] C-P Ren, R R Chaudhuri, A Fivian, C M Bailey, M Antonio, W M Barnes, and M J Pallen, "The ETT2 gene cluster, encoding a second type III secretion system from *Escherichia coli*, is present in the majority of strains but has undergone widespread mutational attrition," *Journal of Bacteriology*, vol 186, no. 11, pp 3547–3560, 2004

[15] A K H P Spaink and P J Hooykaas, eds , *The Rhizobiaceae. Molecular Biology of Model Plant-Associated Bacteria* Springer, 1 ed , 1998

[16] B. J Ferguson, E M. Wiebe, R J. N. Emery, and F C Guinel, "Cytokinin accumulation and an altered ethylene response mediate the pleiotropic phenotype of the pea nodulation mutant R50 (*sym16*)," *Canadian Journal of Botany*, vol. 8, no 83, pp 989–1000, 2005

[17] T. H N Ellis and S J. Poyser, "An integrated and comparative view of pea genetic and cytogenetic maps," *New Phytologist*, vol 153, no 1, pp 17–25, 2002

[18] F C Guinel and L. L Sloetjes, "Ethylene is involved in the nodulation phenotype of *Pisum sativum* R50 (*sym16*), a pleiotropic mutant that nodulates poorly and has pale green leaves," *Journal of Experimental Botany*, vol 51, no 346, pp 885–894, 2000

[19] A. N Pepper, A. P. Morse, and F C Guinel, "Abnormal Root and Nodule Vasculature in R50 (*sym16*), a Pea Nodulation Mutant which Accumulates Cytokinins," *Annals of Botany*, vol 99, no 4, pp 765–776, 2007

[20] D. Alnor, N Frimodt-Møller, F Espersen, and W Frederiksen, "Infections with the unusual human pathogens *Agrobacterium* species and *Ochrobactrum anthropi*," *Clinical Infectious Diseases*, vol. 18, no 6, pp. 914–20, 1994

[21] *The NCBI Handbook* Bethesda, Maryland, October 2002

[22] V. Novak, *Fuzzy Sets and their Applications.* Taylor & Francis, 1989

[23] P Bernays, *Axiomatic Set Theory* Dover Publications, 1991

[24] T. Mitsuishi, K Wasaki, and Y Shidama, "Basic Properties of Fuzzy Set Operation and Membership Functions," *Journal of Formalized Mathematics*, vol. 12, 2003

[25] G J Klir, U S Clair, and B. Yuan, *Fuzzy Set Theory. Foundations and Applications* Prentice Hall PTR, 1997

[26] M Wygralak, *Cardinalities of Fuzzy Sets (Studies in Fuzziness and Soft Computing)* Springer, 2003

[27] B Ford, "Packrat Parsing Simple, Powerful, Lazy, Linear Time," in *Proceedings of the 2002 International Conference on Functional Programming*, October 2002

[28] R Milner, "A theory of type polymorphism in programming," *Journal of Computer and System Sciences*, vol 17, pp 348–375, 1978

[29] S Peyton Jones, *Haskell 98 Language and Libraries· the Revised Report* Cambridge University Press, January 2003 http //www w3 org/TR/2007/REC-xquery-20070123/.

[30] R L E. Bruneton and T Coupaye, *ASM a code manipulation tool to implement adaptable systems* Grenoble, France, November 2002

[31] D Chamberlin, J Robie, D Florescu, S Boag, J Siméon, and M F Fernández, "XQuery 1 0· An XML Query Language," tech rep , January 2007. http //www.w3 org/TR/2007/REC-xquery-20070123/

[32] L P Cordella, P Foggia, C Sansone, and M Vento, "An improved algorithm for matching large graphs," in *In 3rd IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition, Cuen*, pp. 149–159, 2001

[33] M. Odersky, *The Scala Language Specification* Lausanne, Switzerland, 2.8 ed., July 2010.

[34] V Novák, I Perfilieva, and J Mockor, *Mathematical Principles of Fuzzy Logic (The Springer International Series in Engineering and Computer Science)* Springer, 1999

[35] P Shannon, A Markiel, O Ozier, N S Baliga, J. T Wang, D Ramage, N Amin, B Schwikowski, and T Ideker, "Cytoscape. a software environment for integrated models of biomolecular interaction networks," *Genome Research*, vol. 13, no 11, pp. 2498–2504, 2003

[36] P D Karp, M Riley, M Saier, I T Paulsen, J Collado-Vides, S M Paley, A Pellegrini-Toole, C Bonavides, and S Gama-Castro, "The EcoCyc Database," *Nucleic Acids Research*, vol 30, no 1, pp 56–58, 2002

[37] M. Kanehisa, S. Goto, M. Furumichi, M Tanabe, and M Hirakawa, "KEGG for representation and analysis of molecular networks involving diseases and drugs," *Nucleic Acids Research*, vol 38, no Database issue, pp D355–360, 2010

[38] M Kanehisa, S Goto, M Hattori, K F Aoki-Kinoshita, M Itoh, S Kawashima, T Katayama, M Araki, and M Hirakawa, "From genomics to chemical genomics new developments in KEGG," *Nucleic Acids Research*, vol 34, no Database issue, pp. D354–357, 2006

[39] M. Kanehisa, S Goto, M. Furumichi, M Tanabe, and M Hirakawa, "KEGG for representation and analysis of molecular networks involving diseases and drugs," *Nucleic Acids Research*, vol 38, no Database issue, pp D355–360, 2010

[40] A Karimpour-Fard, "Collected known pathogenicity genes in *E. coli*"

[41] T C Charles and T M Finan, "Analysis of a 1600-kilobase *Rhizobium meliloti* megaplasmid using defined deletions generated in vivo," *Genetics*, vol 127, no 1, pp. 5–20, 1991

[42] S Lehman, J Cheng, G Golding, and T Finan, "Locating the Precise Endpoints of Deletions in the pSymB megaplasmid of *Sinorhizobium meliloti*," (McMaster University, Hamilton, Ontario, Canada), May 2003

[43] V S Cooper, S H Vohr, S C Wrocklage, and P J Hatcher, "Why genes evolve faster on secondary chromosomes in bacteria," *PLoS Computational Biology*, vol 6, no 4, p e1000732, 2010

[44] G Fang, E P C. Rocha, and A Danchin, "Persistence drives gene clustering in bacterial genomes," *BMC Genomics*, vol 9, p. 4, 2008

[45] B. Holmes, M R Popoff, M Kiredjian, and K Kersters, "Ochrobactrum," in *Bergey's Manual® of Systematic Bacteriology* (D J Brenner, N R Krieg, G M Garrity, J T Staley, D R Boone, P Vos, M Goodfellow, F. A Rainey, and K.-H Schleifer, eds.), pp 389–392, Springer US, 2005.

[46] B Altincicek, A K. Kollas, S Sanderbrand, J Wiesner, M Hintz, E. Beck, and H Jomaa, "GcpE is involved in the 2-*C*-methyl-D-erythritol 4-phosphate pathway of isoprenoid biosynthesis in *Escherichia coli*," *Journal of Bacteriology*, vol 183, no 8, pp. 2411–6, 2001

[47] S Hecht, W Eisenreich, P Adam, S Amslinger, K Kis, A Bacher, D Arigoni, and F Rohdich, "Studies on the nonmevalonate pathway to terpenes the role of the GcpE (IspG) protein," *Proceedings of the National Academy of Science of the United States of America*, vol 98, no 26, pp 14837–42, 2001

[48] B M Lange, T Rujan, W Martin, and R Croteau, "Isoprenoid biosynthesis the evolution of two ancient and distinct pathways across genomes," *Proceedings of the National Academy of Sciences of the United States of America*, vol 97, no 24, pp 13172–13177, 2000

[49] C S Henry, J F Zinner, M P Cohoon, and R L Stevens, "iBsu1103 a new genome-scale metabolic model of Bacillus subtilis based on SEED annotations," *Genome Biology*, vol 10, no 6, p R69, 2009

[50] J T Foster, S M Beckstrom-Sternberg, T Pearson, J S Beckstrom-Sternberg, P S G Chain, F F Roberto, J. Hnath, T Brettin, and P Keim, "Whole-genome-based phylogeny and divergence of the genus *Brucella*," *Journal of Bacteriology*, vol. 191, no 8, pp 2864–2870, 2009

[51] R D Pridmore, "New and versatile cloning vectors with kanamycin-resistance marker," *Gene*, vol 56, no. 2-3, pp 309–312, 1987

[52] J Quandt and M. F Hynes, "Versatile suicide vectors which allow direct selection for gene replacement in gram-negative bacteria," *Gene*, vol 127, no. 1, pp 15–21, 1993

[53] J F Petrosino, S Highlander, R A Luna, R A Gibbs, and J Versalovic, "Metagenomic pyrosequencing and microbial identification," *Clinical Chemistry*, vol 55, no 5, pp 856–866, 2009

[54] Q Wang, G M Garrity, J M Tiedje, and J R Cole, "Naive Bayesian classifier for rapid assignment of rRNA sequences into the new bacterial taxonomy," *Applied Environmental Microbiology*, vol 73, no 16, pp 5261–5267, 2007

[55] J R Cole, Q Wang, E Cardenas, J Fish, B Chai, R J Farris, A. S Kulam-Syed-Mohideen, D M. McGarrell, T Marsh, G. M Garrity, and J M Tiedje, "The Ribosomal Database Project improved alignments and new tools for rRNA analysis," *Nucleic Acids Research*, vol 37, no. Database issue, pp D141–145, 2009.

[56] J. R Cole, B Chai, R J Farris, Q Wang, A S Kulam-Syed-Mohideen, D M McGarrell, A M Bandela, E. Cardenas, G M Garrity, and J. M Tiedje, "The ribosomal database project (RDP-II)· introducing myRDP space and quality controlled public data," *Nucleic Acids Research*, vol 35, no Database issue, pp D169–172, 2007

[57] W Li and A Godzik, "Cd-hit a fast program for clustering and comparing large sets of protein or nucleotide sequences," *Bioinformatics*, vol 22, no 13, pp 1658–1659, 2006

[58] E P Nawrocki, D L Kolbe, and S R Eddy, "Infernal 1 0 inference of RNA alignments," *Bioinformatics*, vol 25, no 10, pp 1335–1337, 2009.

Appendix A

Reference Guide for gisql

The **gisql** langauge is similar to other functional languages Programs may be entered directly or run from a file There must be a database containing all the information about the species that will be used in analysis The language assumes the database is static during execution

A.1 Configuring the Database

A separate project **pg-nebulon** provides tools to load Nebulon output in a PostgreSQL database, for use by **gisql** The database holds a number of *species*, each representing one interaction network Each species must have a name which is a valid Java identifier with the additional constraint of lacking dollar signs For each species, genes may be defined and genes may be associated in to arbitrary groups The groups as used to group genes into orthologous clusters Genes may also have attached cluster of orthologous genes (COG) or Gene Ontology labels Separately, a list of interactions between genes is present in the database Genes may only interact with other genes of the same species

A.2 Manipulating Interactomes

From the console, all the interactomes available can be displayed by typing **ls()** In fact, this will show all defined variables Typing the name of an interactome and pressing enter will cause it to be loaded from the database and the statistics about its interactions are presented as follows

```
# A
# Gene fuzziness: 0.0
# Interaction fuzziness: 0.0
# Gene histogram: 0 0 0 0 0 0 0 0 0 6
# Interaction histogram: 0 0 0 0 0 0 0 0 0 5
```

Fuzziness is a quantity that measures the non-binariness of membership values in a fuzzy set A defined as $\sum_x 1 - |2Ax - 1|$ The histogram is a count of the number of interactions in 10 bins of width 0 1 Interactomes can be manipulated as fuzzy sets using union, intersection, symmetric difference, difference, and the residuum. For instance `A & (B ^ C)` will compute the intersection of A with the symmetric difference of B and C

A common operation on fuzzy sets is the α-cut, where memberships below a certain value are made zero. To remove any values less than 0 9 from a set A, the syntax is `A {0.9}`.

The interactions discovered can be written to a file using the syntax `A @ "filename"` By default, the interactions are printed in plain text, but other formats are available and may be specified before the file name

There are a large number of functions that can manipulate interactomes Typing `help` will list them and a brief description

A.3 General Features of the Language

The language also supports many more general features General number manipulation, strings, boolean and fuzzy logic, and function definition The command `typeof` will display the type of an expression An `if` ***condition*** `then` ***truepart*** `else` ***falsepart*** expression is included to allow decision logic In scripts, a command may span multiple lines by starting the subsequence lines with white space

A.3.1 Variables and Functions

Variables may be defined using the syntax `variable = value` and functions may be defined using the syntax `function param = expression` where any number of parameters can be defined Recursive functions are supported by the language Anonymous functions are also available using the syntax `\var -> expression` [1]

[1] The \ symbol is also used for set difference

A special syntax is introduced to allow functions to be written more linearly, with fewer parentheses Instead of `f x y`, a function may be written `x :f y` This is convenient for deeply nested expressions, as `x :a :b 5 :c` is shorter than `(c (b (a x)5))`

There are two kinds of strings in the language, regular strings, and formatting strings The formatting strings work much like C's `printf` formatter. If a literal string contains placeholders of the form $\{x\}$, where x is a natural number or a name, then the string will become a formatter, a function which takes as many arguments are needed and converts produces a formatted string as output For instance, `"foo"` is simply a string, while `"The␣value␣is␣\{1}."` is a function taking a single argument and returning a string and `"The␣value␣is␣\{a}."` is equivalent to `"The␣value␣is␣\{1}."a`

A.3.2 Lists

Homogeneous lists are included in the language Lists can be specified literally using the syntax `[1, 2, 3]` and the empty list has the obvious syntax `[]` Lists can be concatenated using the syntax `list1 + list2` Typical list manipulation functions are available including **`map`**, **`zip`**, **`foldl`**, and **`foldr`** The function **`flatten`** converts a nested list input into a single output list There are two sorting functions, one which sorts item that have natural order and one which uses an arbitrary comparator The function **`index`** will return the n^{th} item from a list and **`length`** will return the number of items in a list There is also a **`slice`** function that will create a sublist given a specific range. Note that indices are always 1-based and negative indices refer to the n^{th} last item in a list.

A.3.3 Numbers and Logic

The language supports three distinct kinds of numbers integers, type `number`, floating point numbers, type `real`, membership values, type `membership` These numbers can be compared using the **`eq`**, **`ne`**, **`lt`**, **`le`**, **`gt`**, and **`ge`** operations Integers and floating point numbers can also be manipulated using **`add`**, **`sub`**, **`mul`**, and **`div`** Integers also support **`mod`** and floating point numbers support **`exp`**, **`ln`**, and **`sqrt`** Integers and floating point numbers can be converted using **`n2r`** and **`r2n`** The largest integral value possible is defined in the constant **`iinf`** The function **`range`** will create a list of numbers over a certain range

The boolean logic is available and the fuzzy set operations also work for boolean values That is **`true & false`** will compute logical AND. Fuzzy logic is available using membership values The previous statement has the fuzzy equivalent `1.0 & 0.0`

A.3.4 Manipulating the Environment

All currently defined variables and known interactomes are in the user's current environment The names of the known variables can be see by typing `ls()`. The defined variables can also be discarded using the **`clear()`** command The **`echo`** command can be used to display information, this is done by default in interactive mode, but not when running from a file

An external file can be processed using the **`run`** command or **`import`**, the semantics are slightly different an imported file must be specified with a fixed string and the definitions will be recognised in subsequent scripts, while a file that is run will not get the same binding privileges, but the file name can be computed When at the console, the difference is unnoticeable, but, when writing scripts, the semantics will be noticeable A list of interactome expressions can be read from another file using the **`read`** command Normally, output is directed to standard output It can be redirected to a file using the **`outputfile`** command, standard output can be recovered by redirecting output to `"-"` Computing interactomes, especially when the output is already being directed to a file, can create unnecessary output, the **`do`** command will process an interactome, but discard and output statistics The default format of output can be specified with the **`format`** command

A.3.5 Nullable Types

Types in the system must have values Types can be lifted to include a missing or null value The lack of a value is the constant **`missing`**. A nullable value can be converted back to types with defined values using the **`otherwise`** function Functions are automatically lifted if necessary That is, a function of type $\alpha \to \beta$ given an argument of type $\alpha_\perp$, will be automatically converted to a function that will return $\beta_\perp$.

A.3.6 Graph Iteration

In many programming languages, it is possible to iterate over each item in a list Notably, PHP add the ability to iterate over key-value pairs in a mapping SQL's `SELECT` command allows iteration over rows in a table XQuery's for-let-where-order-return expression iterates over "twigs", that is, subtrees in a tree By extension, since the primary focus of **`gisql`** is graphs, an expression is provided that allows iteration over isomorphic subgraphs in an interactome

The syntax is `for` G `in` E `where` C `return` V where G is a specification of the subgraph to match, E is the source interactome to search, C is an optional boolean condition to eliminate unwanted matches, and V

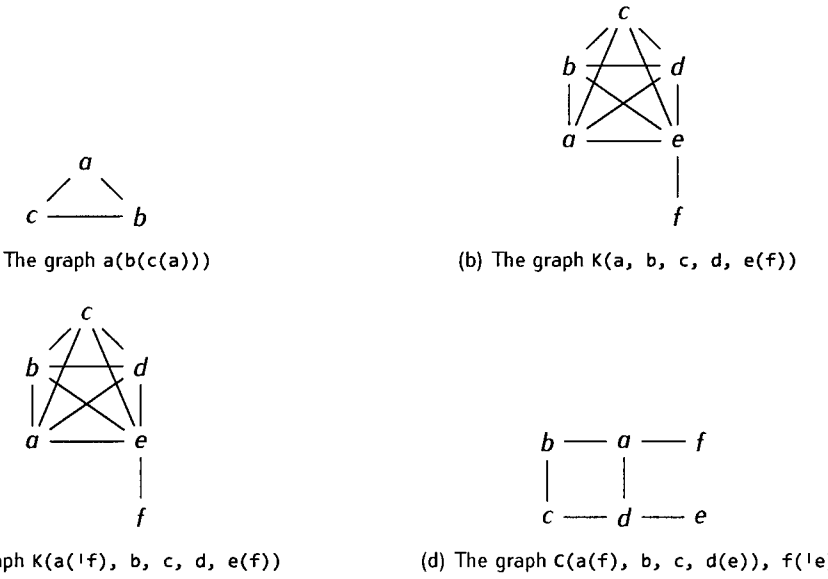


Figure A.1: Subgraph specification examples with required edges shown as solid lines and edges which must be absent shown as dotted lines

is the return value Subgraphs are specified using nested bracketed expressions to indicate connectivity. Each name specified becomes a node that must be matched in the subgraph When a node is followed by a list of nodes, all these nodes are connected to the enclosing node The use of C specifies a cycle graph and K specifies a complete graph If any edges are not specified, then they search will not care if they are present or absent in the underlying graph If they must be absent, a node prefixed with ! will indicate an absent edge Examples are shown in Figure A 1 The worst-case effiency of the search is $O(k)$ space complexity and $O(n^k)$ time complexity where k is the number of nodes in the subgraph and n is the number of nodes in the source interactome

A.3.7 User Defined Interactomes

Users can define their own custom graph manipulations using the command `interactome given` *sources*, `{ unknown =` U`; gene` g `=` G`; interaction` $g1$ $g2$ `=` I`}` The *sources* are a comma-separated list of the names given to any interactomes that must be provided, if one were to redefine union, one would need two sources· left and right Three pieces of information are required· the membership of a gene, the membership of an interaction, and the membership of an unknown gene or interaction, specified in G, I, and U, respectively. In each of G, I, and U, the sources will be take on the corresponding values in the source interactomes g, $g1$,

and $g2$ are the genes involved

For example, a function which excludes a particular gene and its interactions would be as follows

```
exclude badgene = interactome given source { unknown = source; gene g = if g :eq badgene then
    0.0 else source; interaction g1 g2 = if g1 :eq badgene then 0.0 else if g2 :eq badgene
    then 0.0 else source}
```

A.4 Interaction Graph Manipulation

There a number of functions to manipulate interactomes In order to provide certain set operations, an interactome containing no genes or interactions is available, **null**, and an interactome containing all genes and interactions known to the systems, **universe**

For convenience, the intersection and union of lists of interactomes can be computed using **intersectall** and **unionall**, respectively

A.4.1 Statistics on Interaction Graphs

Statistics on the genes or interactions are available. The number of items with non-zero membership is available via **genecount** and **interactioncount** The cardinality of fuzzy set, $\sum_x Ax$, is available via **genecard** and **interactioncard** The fuzziness of the set, $\sum_x 1 - |2Ax - 1|$, is available via **genefuzz** and **interactionfuzz**

Additionally, the genomic similarity score, $\frac{|A \cap B|}{|A| \wedge |B|} \forall x \in genes$, can be computed using **gss** and a list of interactomes can be sorted in clusters, using some cutoff to divide the clusters, using **gsscluster** If the clusters are degenerate, no clusters will be returned.

The **jaccardcore** calculates the Jaccard score and assigns it to the membership·

$$J_A\langle x, y\rangle = \frac{\sum_{l \in \text{leaves}(A)} \lceil lx \rceil \wedge \lceil ly \rceil}{\sum_{l \in \text{leaves}(A)} \lceil lx \rceil \vee \lceil lx \rceil} \forall \langle x, y\rangle \in interactions$$

where A is an interactome expression and leaves(A) is the individual species used in A

A.4.2 Membership Manipulation

There are several functions to manipulate the memberships of genes and interactions in interactomes. The **defuzz** function will round all interactions with membership > 0 to 1 and mark all others as missing The **blanks**

function does the reverse, it fills in all missing values with some specified membership The **avgblanks** calculates the blanks as the average membership over all species and uses this as the value

Since membership between genes and interactions are not necessarily treated the same by all functions, it is possible to be in a state where an interaction is present but the genes involved are not The **snap** function will delete any interactions for which the genes are not present The **orphans** function will delete any genes which are present but do not have any interactions that are present

A.4.3 Gene Manipulation

All the genes present in an interactome can be converted to a list using the **genesof** function If the gene identifiers are known, genes can be retrieved using the **gi** function If names of genes are present in the database, genes can be retrieved by regular expression using the **findgenes** function Specific genes can be removed from an interactome using the **except** function An interactome can be limited to genes connected to a specified list of genes using the **near** function

A.4.4 Coreicity

There are two functions that allow filtering based on coreicity The function **genecoreicity** will select any genes for which a supplied function, given the coreicity of the gene, returns true, any interactions will be retained if one of their genes is retained **interactioncoreicity** allows filtering of interactions based on the coreicity of the two genes

A.4.5 Output

Interactions can be used as characters for the construction of phylogenies The **phylip** command will produce a file compatible with the PHYLIP package from a list of interactomes A mapping of names is returned

Venn diagrams of interactions or genes can also be produced using the **venn** command The output is a list of the sizes of the sections with a binary number indicating which interactomes are present

Appendix B

Metagenomic Taxonomic Classification

This project, completed for CS 798, was an analysis of metagenomic data provided by Dr J Neufeld and Michael J Lynch The goal was to classify unknown 16S ribosomal subunit sequences from metagenomic libraries constructed by Dr Neufeld's students

B.1 Background

Traditionally, the genomes of organisms were sequenced by first isolating the organism in pure culture, then sequencing the clones of a single organism However, not all organisms can be grown in pure culture, including obligate symbionts and organisms with complex nutritional requirements, and so metagenomics emerged as a way to sequence the genomes of organisms in an environmental sample The microbial make up of an environment can be determined by sequencing the 16S ribosomal subunit as it has highly conserved regions, from which to begin sequencing, and variable regions, which give a phylogenetic fingerprint [53]

The latest advancement in sequencing technology is pyrosequencing. The traditional Sanger method, also known as dideoxy-chain termination, uses fluorescently-labelled dideoxynucleosides to terminate the growing chain at a location where a specific base was incorporated The labelled chains are run on a gel and reading the fluorescent markers gives the correct sequence of bases in order. The limiting factor in this method is the speed at which fragments migrate through the gel In pyrosequencing, flashes of light are released as bases are incorporated into a copy of the DNA As DNA polymerase catalyses the replication of DNA, it releases energetic phosphates in the process and additional enzymes use this energy to release a flash of light This

allows pyrosequencing to be considerably faster than traditional sequencing as DNA polymerase can incorporate bases very quickly. The major limitations of pyrosequencing include fidelity and length. Pyrosequencing reads can contain many truncations and missing bases or residues where the pulse of light does not conform to the expected profile Moreover, the accuracy falls off with length, so there is comparatively short range for accurate pyrosequencing [53]

Even with these limitations, pyrosequencing is valuable for analysis of 16S ribosomal subunits from metagenomic samples The regions of interest in the ribosomal subunit are sufficiently short that the range of pyrosequencing is not an issue. The major benefit to using pyrosequencing is the vast quantities of data it produces for the cost Even though there are errors, the increase in total data greatly exceeds the increase in errors.

The primary goal of current metagenomic research is to create phylogenetic profiles of ecosystems in order to make comparisons of microbial communities Dr. Neufeld, his lab, and Michael Lynch are analysing the third variable region of the 16S ribosomal subunit in order to classify organisms in the metagenomic sample The sequences are assigned taxonomic identifiers by a Bayesian k-nearest neighbour classifier developed by the Ribosome Database Project [54] Some of the sequences fail to classify, or only classify at high level taxon identifiers (e g , "bacteria") Since metagenomics is a field devoted to novelty, it is desirable to know if these sequences are from unknown organisms or simply poor classification effort and where they should be placed in the accepted bacterial "tree of life"

B.2 Materials and Methods

The current work in Dr Neufeld's lab is centred around adapting existing tools from the Ribosome Database Project [55, 56], designed to handle data produced by Roche 454® sequencers, to work with Illumina SOLEXA® sequence data, which produces greater sequence volume Sequences are preprocessed and classified, sequences exiting classification with low quality classification, either due to low score or shallow taxonomic labels, should be reclassified more throughly Statistics about the type of sequences missed by the Bayesian classifier should be collected

B.2.1 Current Pipeline

Dr. Neufeld and Michael Lynch are currently adapting the pipeline used by the Ribosome Database Project [55, 56, 54] There are two obstacles· the Illumina SOLEXA® data formats and quality statistics differ from the

Roche 454® formats and the Illumina SOLEXA® produces enough data that the 32-bit tools will exceed their address space Adaption of the pipeline to handle Illumina SOLEXA® data has been completed by Michael Lynch. Tool improvement is happening on an as-needed basis in conjunction with the original RDP developers. The complete process is as follows.

DNA Extraction – Samples are collected from soil, water, or human ecosystems DNA is purified from particulate and other cell components

16S Amplification – Polymerase chain reaction is used to amplify variable region 3 from the 16S ribosomal subunit. Universal primers are used that can amplify the region from most prokaryotic organisms Some nanoarcheota are excluded in this step

Sequencing – Amplified DNA is sequenced using the Illumina SOLEXA® sequencing platform

Assembly – Forward and reverse reads are assembled using the quality information to discard reads of low quality Reads are not meant to be assembled into contigs, as is often done when assembling DNA for analysis of genes In this case, there are two primers a forward primer and a reverse primer, and the reads created by these primers should match Reads which do not have a matching sequence are sequencing errors, and can be discarded

Sequence Cleanup – Sequences are then analysed further for validity Reads shorter than 100 nucleotides, even if correctly assembled, are discarded as they are unlikely to be ribosomal sequences This cutoff is arbitrary and it is possible that these shorter sequences are ribosomal genes that have undergone selective pressure to be reduced, as it is often the case in obligate intercellular parasites and self-replicating organelles. Some sequences may be concatomers of the primers used for amplification. The sequences that seem to be composed of only primer sequences are discarded

Clustering – Sequences are then clustered into groups using CD-HIT [57] This is not a phylogenetic clustering The clustering is meant to reduce the total number of sequences that must be processed downstream and remove single-base pyrosequencing errors. By clustering similar sequences into groups, low abundance sequences which are nearly identical to high abundance sequences are, effectively, discarded These sequences are likely pyrosequencing errors

Classification – Sequences are assigned taxonomic labels, with varying confidences, by the RDP Bayesian

Table B.1: Taxonomic depth of sequences in sample dataset

Taxon	Depth	Count
Life	1	361
Domain	2	6149
Phylum	3	2267
Class	4	4484
Order	5	1356
Family	6	2629
Genus	7	5884
Species	8	1140
Strain	9	613

rRNA Classifier [54] The classifier assigns taxonomic labels from a set of curated sequences with an established taxonomy based on unaligned sequences

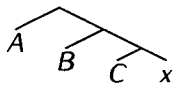
Since the classifier has an error rate of approximately 5% on well-characterised data [54], establishing the error rate on unknown data is required

B.2.2 Taxonomic Coverage

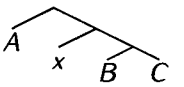
Analysis was performed on sequences from soil in Alert, Nunavut, Canada The numbers of sequences classified to different taxonomic depths with a confidence greater than 50% are shown in Table B 1 It is worth noting that 361 sequences, or 1 4%, of the data are not identified as bacterial or archeal sequences Approximately one quarter of the sequences are classified to only the domain level (i e , bacterial or archeal), providing little information Since classification down to the class level is not very informative, 53 3% of sequences do not have meaningful taxonomic annotation

B.2.3 Small Tree Approach

The initial approach for resolving the poorly defined sequences is the use of similar sequences with known taxonomies to infer the taxonomy of the query sequence Naturally, this method could be used for the entire data set, however, it is necessarily more computationally expensive than the Bayesian classifier The central idea of the method is to build a database of known sequences, then use BLAST2 [2] to find matching sequences from the National Center for Biotechnology Information (NCBI) with well-known taxonomic labels The fetched sequences can then be used to build a phylogenetic tree Since the internal nodes of the tree represent extinct



(a) An uninformative tree



(b) A less informative tree

Figure B.1: Examples of trees and their usefulness in the small tree approach to inferring phylogeny of unknown metagenomic sequences, where x is the unknown sequence

taxa, each one can be assumed to be at least the last common ancestor of its child nodes and so the intersections of the taxonomic labels of the child nodes should apply to the internal node In this way, taxonomic labels of known sequences can be attached to the leaves of the tree and propagated to the root of the tree, ignoring the query sequence The unknown sequence must have, at least, the taxonomic labels of its parent By this method, the taxonomic labels can be inferred for the unknown sequence

The selection of BLAST results is important. Since labels must be propagated toward the root, an out-group must be selected to root the tree A natural idea would be to use the top BLAST hits as part of the in group and a hit that is of sufficiently poor score to act as an out-group If few results are available, it may become impossible to gain meaningful insight For instance, if there are fewer than three results, the result will likely be uninformative regardless of the choice of root For three results, as shown in Figure B 1, the results, at best, are not significantly different from assuming the taxonomic labels of the best BLAST hit Moreover, in the worst case, the best results provide worse information about the placement of the unknown sequence as the taxonomy of the BLAST results becomes more divergent, which seems likely given there are few hits

Since these sequences belong to an RNA with a reasonably conserved secondary structure, sequences can be aligned to a structural model of the 16S ribosomal subunit. Since building a phylogenetic tree from a multiple sequence alignment is the preferred method for short sequences, the quality of the alignment has a significant impact on the constructed tree Fortunately, each sequence can be aligned independently to the structural model, eliminating the need for multiple sequence alignment The Ribosome Database Project provides the structural model of the 16S ribosomal subunit used for the applications in the suite [55] This model is formatted for use by Infernal, an RNA sequence aligner based on a covariance model [58]

B.2.4 Known Sequence Database

The Ribosome Database Project provides two groups of sequences: those used for the construction of the classifier, and the complete collection of NCBI sequences The selection of the reference database is difficult

Table B.2: Metagenomic sequence abundance

Taxon	Depth	Total	Zero Hits	Uncultured Matches	Unmatched
Life	1	331	328	3	100.0%
Domain	2	5070	572	4498	100.0%
Phylum	3	1898	10	1887	99.947%
Class	4	3777	0	3777	100.0%

either way. Since the metagenomic analysis is meant to find novel organisms, selecting the broadest possible database would be the obvious choice. The NCBI collection represents the most inclusive database, however, many of the sequences in the database are not well-taxonomically labelled as the database contains sequences from unknown, uncultured organisms. The set used to build the classifier is a quarter the size of the NCBI collection, however, the sequences are all labelled to the species level. The essential problem with this database is the bias. By nature, all sequences in this database come from cultured organisms and, therefore, cannot reflect the taxonomic diversity in the metagenome, as this is the point of engaging in metagenomics!

Therefore, the reference database was built from the model-aligned collection of NCBI sequences using BLAST's `formatdb` [2], which ignores the gapping characters included in the aligned sequences.

B.3 Results

The first step in analysis was determining the candidate known sequences for tree generation for each metagenomic sequence. Using BLAST [2], against the NCBI-derived aligned ribosomal database [54], produced no usable results. Shown in Table B.2, the sequences did not match with any known organisms.

Clearly, these results prevent continued analysis. There is insufficient data to build any trees. Moreover, this is an indication that unknown sequences are from wholly unknown groups of organisms. However, these results indicate that the Bayseian classifier is doing quite well; a small consolation.

B.4 Discussion

This method seems unhelpful in determining the taxonomic labels of unknown sequences. The ultimate problem is a lack of information about uncultured bacteria, but it is obviously necessary that taxonomy be inferred without the benefits of culture-based information.

It seems reasonable to assume that the sequences that do not match existing organisms are from large clades of the tree of life that have no cultivable members Therefore, these sequences must be placed in the tree using other sequences from uncultured organisms. Given the issue of abundance, it is probably reasonable to pool several metagenomes together Even without alignment, there are far too many sequences for efficient tree construction Unknown sequences would have to be clustered into gross groups, based on sequence identity and smaller trees constructed from closely related sequences. From each of these groups, representatives would have to be used to guess where the roots of the small trees should be grafted to the accepted phylogeny of bacteria. It is undesirable to simply graft every unknown sequence on to the large tree As with cultured organisms, uncultured organisms should be isolated into taxonomic units

Grouping into taxonomic units would have to be approximated from sequences in existing taxonomic units Given an arbitrary cluster of unknown sequences, a profile of the alignment scores could be built The goal would then be to use the statistics for those alignment scores to find similar groups of cultured organisms and the accepted taxonomic boundaries in the cultured set could be applied to the uncultured set Effectively, if a group of metagenomic sequences had the same divergence as a group of sequences from cultured organisms comprising a genus, then it follows that the uncultured group should be considered a genus This only works if taxonomic units have relatively uniform distribution of sequence alignment scores The validity of this assumption could be determined by looking at the distribution of the desired statistical properties of taxonomic units of cultured organisms Indeed, this might be a way to select the most meaningful statistical measures

Unfortunately, this approach would be extremely computationally expensive Given the huge number of taxonomic units in the cultured organism database, and that any statistics would likely require pair-wise analysis, the computational cost of analysing the clusters would be very high A more efficient approach might be a recursive Bayesian classifier After classifying the 16S sequences in a metagenome, the unknown sequences could be fed back into the classifiers corpus to produce a classifier capable of recognising certain unknown sequences While this method would not provide a way to place these items correctly in the tree of life, it would allow similar sequences to be clustered in future metagenomes, providing, at least, a more informed way to cluster sequences

Appendix C

Sequences

The sequences of all vectors and DNA fragments from other sources are provided, if known

C.1 BBa_E0040 Fragment

Green fluorescent protein in a BioBrick®-standard vector from the Registry of Standard Biological Parts

```
BBa_E0040  ATGCGTAAAGGAGAAGAACTTTTCACTGGAGTTGTCCCAATTCTTGTTGAATTAGATGGTGATGT   65

BBa_E0040  TAATGGGCACAAATTTTCTGTCAGTGGAGAGGGTGAAGGTGATGCAACATACGGAAAACTTACCC  130

BBa_E0040  TTAAATTTATTTGCACTACTGGAAAACTACCTGTTCCATGGCCAACACTTGTCACTACTTTCGGT  195

BBa_E0040  TATGGTGTTCAATGCTTTGCGAGATACCCAGATCATATGAAACAGCATGACTTTTTCAAGAGTGC  260

BBa_E0040  CATGCCCGAAGGTTATGTACAGGAAAGAACTATATTTTTCAAAGATGACGGGAACTACAAGACAC  325

BBa_E0040  GTGCTGAAGTCAAGTTTGAAGGTGATACCCTTGTTAATAGAATCGAGTTAAAAGGTATTGATTTT  390

BBa_E0040  AAAGAAGATGGAAACATTCTTGGACACAAATTGGAATACAACTATAACTCACACAATGTATACAT  455

BBa_E0040  CATGGCAGACAAACAAAAGAATGGAATCAAAGTTAACTTCAAAATTAGACACAACATTGAAGATG  520

BBa_E0040  GAAGCGTTCAACTAGCAGACCATTATCAACAAAATACTCCAATTGGCGATGGCCCTGTCCTTTTA  585
```

```
BBa_E0040  CCAGACAACCATTACCTGTCCACACAATCTGCCCTTTCGAAAGATCCCAACGAAAAGAGAGACCA   650

BBa_E0040  CATGGTCCTTCTTGAGTTTGTAACAGCTGCTGGGATTACACATGGCATGGATGAACTATACAAAT   715

BBa_E0040  AATAA   720
```

C.2 Omega Fragment

The spectinomycin/streptomycin marker with omega cloning ends [51].

```
Spc/Sm Ω  CTAGATTATTTGCCGACTACCTTGGTGATCTCGCCTTTCACGTAGTGGACAAATTCTTCCAACTG    65

Spc/Sm Ω  ATCTGCGCGCGAGGCCAAGCGATCTTCTTCTTGTCCAAGATAAGCCTGTCTAGCTTCAAGTATGA   130

Spc/Sm Ω  CGGGCTGATACTGGGCCGGCAGGCGCTCCATTGCCCAGTCGGCAGCGACATCCTTCGGCGCGATT   195

Spc/Sm Ω  TTGCCGGTTACTGCGCTGTACCAAATGCGGGACAACGTAAGCACTACATTTCGCTCATCGCCAGC   260

Spc/Sm Ω  CCAGTCGGGCGGCGAGTTCCATAGCGTTAAGGTTTCATTTAGCGCCTCAAATAGATCCTGTTCAG   325

Spc/Sm Ω  GAACCGGATCAAAGAGTTCCTCCGCCGCTGGACCTACCAAGGCAACGCTATGTTCTCTTGCTTTT   390

Spc/Sm Ω  GTCAGCAAGATAGCCAGATCAATGTCGATCGTGGCTGGCTCGAAGATACCTGCAAGAATGTCATT   455

Spc/Sm Ω  GCGCTGCCATTCTCCAAATTGCAGTTCGCGCTTAGCTGGATAACGCCACGGAATGATGTCGTCGT   520

Spc/Sm Ω  GCACAACAATGGTGACTTCTACAGCGCGGAGAATCTCGCTCTCTCCAGGGGAAGCCGAAGTTTCC   585

Spc/Sm Ω  AAAAGGTCGTTGATCAAAGCTCGCCGCGTTGTTTCATCAAGCCTTACGGTCACCGTAACCAGCAA   650

Spc/Sm Ω  ATCAATATCACTGTGTGGCTTCAGGCCGCCATCCACTGCGGAGCCGTACAAATGTACGGCCAGCA   715

Spc/Sm Ω  ACGTCGGTTCGAGATGGCGCTCGATGACGCCAACTACCTCTGATAGTTGAGTCGATACTTCGGCG   780

Spc/Sm Ω  ATCACCGCTTCCCTCATGATGTTTAACTTTGTTTTAGGGCGACTGCCCTGCTGCGTAACATCGTT   845
```

```
Spc/Sm Ω  GCTGCTCCATAACATCAAACATCGACCCACGGCGTAACGCGCTTGCTGCTTGGATGCCCGAGGCA   910

Spc/Sm Ω  TAGACTGTACCCCAAAAAAACAGTCATAACAAGCCATGAAAACCGCCACTGCGCCGTTACCACCG   975

Spc/Sm Ω  CTGCGTTCGGTCAAGGTTCTGGACCAGTTGCGTGAGCGCATACGCTACTTGCATTACAGCTTACG  1040

Spc/Sm Ω  AACCGAACAGGCTTATGTCCACTGGGTTCGTGCCTTCATCCGTTTCCACGGTGTGCGTCACCCGG  1105

Spc/Sm Ω  CAACCTTGGGCAGCAGCGAAGTCGAGGCATTTCTGTCCTGGCTGGCGAACGAGCGCAAGGTTTCG  1170

Spc/Sm Ω  GTCTCCACGCATCGTCAGGCATTGGCGGCCTTGCTGTTCTTCTACGGCAAGGTGCTGTGCACGGA  1235

Spc/Sm Ω  TCTGCCCTGGCTTCAGGAGATCGGAAGACCTCGGCCGTCGCGGCGCTTGCCGGTGGTGCTGACCC  1300

Spc/Sm Ω  CGGATGAAGTGGTTCGCATCCTCGGTTTTCTGGAAGGCGAGCATCGTTTGTTCGCCCAGCTTCTG  1365

Spc/Sm Ω  TATGGAACGGGCATGCCCCCAACTGAGAGAACTCAAAGGTTACCCCAGTTGGGGCCCGGG  1425
```