INSILICO TOOL FOR DRUG TARGET IDENTIFICATION AND VALIDATION
(In Silico-iTTV)

By

HARIKRISHNA YALAMANCHILI - 031550
PRAVEEN GHATTA - 031559
M. ARVIND - 031551
GARIMA JAJODIA - 031536

MAY-2007

Submitted in partial fulfillment of the Degree of Bachelor of Technology

DEPARTMENT OF BIOTECHNOLOGY AND BIOINFORMATICS
JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY-WAKNAGHAT, SOLAN, H.P., INDIA

CERTIFICATE

This is to certify that the work entitled, "**Insilco tool for drug target identification and validation**" submitted by Mr. Hari Krishna.Y (031550), Mr. Arvind.M (031551), Mr. Praveen.G (031559), Ms Garima.J (031536) in partial fulfillment for the award of degree of Bachelor of Technology in Bioinformatics of Jaypee University of Information Technology has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Naik

Dr. Pradeep Kumar Naik
(Project Coordinator)
Senior Lecturer
Dept. of Bioinformatics and Biotechnology
Jaypee University of Information Technology
Waknaghat, Solan, Himachal Pradesh, India

ACKNOWLEDGMENT

I take this opportunity to extend my gratitude towards our project coordinator Dr Pradeep K Naik who has stood by us during the entire tenure of our project work. He has been like a father figure to us and we appreciate his efforts and his precious time invested in this work. This work has been one of the most important projects in our academic career by far and a lot of exertion has been put into this work. A lot many people have contributed significantly in this project and this space is too little to compose their names, but I would like to thank each one of them for their worthy contribution. One noteworthy person I would like to thank is our classmate Mr. Kunal Jaiswal lacking whose assistance, our work would not have seen the light of success. I extend my best wishes towards him for his future. Also I would like to thank Ms. Somlata, our worthy lab-assistant for providing us the support while we worked on this project.

Praveen

Hari Krishna Yalamanchili

Praveen Ghatta

Garima Jajodia

Arvind Mukundan

Garima Jajodia

TABLE OF CONTENTS

LIST OF FIGURES and TABLES

LIST OF ABBREVIATIONS

BLAST:	Basic Local Alignment Search Tool
DEG:	Database of Essential Genes
DNA:	DeoxyriboNucleic Acid
FASTA:	FAST-All
HGP:	Human Genome Project
iTTV:	Identification of Therapeutic Targets and Validation
KEGG:	Kyoto Encyclopedia of Genes and Genomes
LWP:	Library for WWW in PERL
NCBI:	National Center for Biotechnology Information
PBP:	Penicillin-Binding Proteins
PERL:	Practical Extraction and Report Language
PSSM:	Position Specific Scoring Matrix
RNA:	RiboNucleic Acid
rRNA:	ribosomal Ribonucleic Acid
SQL:	Structured Query Language
T-iDT:	Tool for identification of Drug Target

ABSTRACT

The complete genome sequences of pathogens have provided a plethora of potential drug targets. While these data potentially contain all the determinants of host-pathogen interactions and possible drug targets, computational tools for selecting suitable candidates for further experimental analyses are currently limited. One of the recently adopted strategies is based on a subtractive genomics approach, in which the subtraction dataset between the host and pathogen genome provides information for a set of genes that are likely to be essential to the pathogen but absent in the host. We have used three-way genome comparisons to identify essential genes and their further validation as potential therapeutic targets from bacteria. The tool developed was tested with 3 different pathogen bacteria: *Mycobacterium leprae, Mycobacterium tuberculosis and Bacillus anthracis*. Our approach identified 80, 140 and 14 essential genes respectively for *M. leprae, M. tuberculosis and B. anthracis* that may be considered as potential drug targets. This approach enables rapid potential drug target identification, thereby greatly facilitating the search for new antibiotics. These results underscore the utility of large genomic databases for *in silico* systematic drug target identification in the post-genomic era.

Keywords: Bacterial pathogens, essential genes, Database of Essential Genes (DEG), comparative microbial genomics, subtractive genomics, novel drug targets.

CHAPTER 1

Introduction

The classical progression of the pharmaceutical discovery process goes from drug target to lead compound to drug. The ability to discover novel therapeutic targets for further research is the first critical step in this process. Therefore, researchers are necessarily concerned with this initial aspect of the drug discovery process. In the past, researchers had a tendency to work on a handful of favored genes, often identified in the literature by academic groups, amenable to low-throughput analysis. Thus, a majority of successful drug discovery projects have targeted the relatively small numbers of protein classes that have proved amenable to pharmaceutical development. It is reported that approximately 483 drug targets accounted for nearly all drugs currently on the market (45% G-coupled receptors, 28% enzymes, 5% ion channels, and 2% nuclear receptors) (Drews 1996). However, currently most new drugs that are approved by the regulatory authorities modulate protein targets for which marketed drugs already exist (Zambrowicz et al. 2003). Therefore, one major hurdle for drug development is still the rapid and accurate identification of drug targets with true potential.

The completion of the human genome project in 2003 increased the effort to sequence whole bacterial genomes and as a result ~150 bacterial genomes have been completely sequenced to date (Wheeler et al. 2004). These data provide ample opportunity to fully exploit this treasure trove for the identification and characterization of virulent factors in pathogens, and to identify novel putative targets for therapeutic intervention (Miesel et al. 2003; HGP 2004; Drews 2000; Terstappen 2001). Target identification seeks to identify new targets, normally proteins (or DNA/RNA), whose modulation might inhibit or reverse disease progression. Reliable technologies for addressing target identification and validation are the foundation of successful drug development.

1.1 What are potential drug targets?

Drug targets are membrane or cellular receptors or other molecules that are pivotally involved in disease processes. From a pharmacological viewpoint, a drug target is either inhibited or activated by drug molecules (e.g. small organic molecules, antibodies, therapeutic proteins). Drug molecules can physically attach to a drug target, triggering a cascade of intracellular biochemical reactions, followed by a cellular reaction. Potential drug targets can include genes that are differentially expressed between individuals who are and are not in need of individual is exposed to a drug known to alleviate or exacerbate the symptoms of interest, and genes that are co-expressed with other genes presumed to be involved in the systems and pathways under study. Any gene falling into one of these categories may be a gene for which manipulation of its expression might affect disease or symptom progression (Allison 2002). In summary, good drug targets are potent and specific, that is, they must have strong effects on a specific biological pathway and minimal effects on all other pathways.

1.2 Therapeutic targets for bacterial disease

To understand how antibiotics work and, concomitantly, why they stop being effective requires a brief look at the targets for the main classes of these antibacterial drugs. There are three proven targets for the main antibacterial drugs:

(1) Bacterial cell-wall biosynthesis;

(2) Bacterial protein synthesis; and

(3) Bacterial DNA replication and repair.

1.2.1 Cell-wall biosynthesis

The layer of the bacterial cell wall that confers strength is the peptidoglycan, a meshwork of strands of peptide and glycan that can be covalently crosslinked. The larger the fraction of adjacent peptide strands that are connected in amide linkage by action of a family of transpeptidases, the higher the mechanical strength to osmotic lysis.

Transglycosylases act on the glycan strands to extend the sugar chains by incorporation of new peptidoglycan units from *N*-acetylglucosamine-b-1 4-*N*-acetylmuramyl-pentapeptide-pyrophosphoryl-undecaprenol (lipid II). Bifunctional enzymes containing both transpeptidase and transglycosylase domains are the target sites for the killing of bacteria by the b-lactam-containing penicillin and cephalosporins, which act as pseudosubstrates and acylate the active sites of the transpeptidases (also termed penicillin-binding proteins or PBPs). The ringopened, penicilloylated transpeptidases deacylate very slowly, and so occupy the enzyme active sites, preventing normal crosslinking of peptide chains in the peptidoglycan layer and leaving it mechanically weak and susceptible to lysis on changes in osmotic pressure. In addition to penicillins and cephalosporins, the vancomycin family of glycopeptide antibiotics also targets the peptidoglycan layer in the cell-wall assembly. But rather than targeting the enzymes involved in peptide crosslinking, vancomycin ties up the peptide substrate6 and thereby prevent it from reacting with either the transpeptidases or the transglycosylases.

The net effect is the same: failure to make peptidoglycan crosslinks leads to a weaker wall that predisposes the treated bacteria to a killing lysis of the cellwall layer. The cup-shaped undersurface of the vancomycin antibiotic makes five hydrogen bonds to the D-Ala-D-Ala dipeptide terminus of each uncrosslinked peptidoglycan pentapeptide side chain, which accounts for the high affinity of the antibiotic for its target, both in partially crosslinked walls and in the lipid II intermediate. Because b-lactams and vancomycin work on adjacent steps — substrate and enzyme — they show synergy when used in combination.

1.2.2 ***Protein synthesis***

The RNA and protein machinery of the prokaryotic ribosomes is sufficiently distinct from the analogous eukaryotic machinery that there are many inhibitors of protein synthesis, targeting different steps in ribosome action, with selective antibacterial action. These include such important antibiotics as the macrolides of the erythromycin class7, the tetracyclines8 (which are products of the aromatic polyketide biosynthetic pathways) and the aminoglycosides of which streptomycin was the founding member, supplanted now by later synthetic variants such as kanamycin). Given the large number of molecular steps involved in initiation, elongation and termination of protein assembly by the ribosome, it is not surprising that there would be many steps of binding or catalysis that could be interdicted by these and many other classes of protein-synthesis inhibitors. This multiplicity also indicates that protein synthesis will provide a multifaceted target for new antibiotics and this is the mechanism for the action of oxazolidinones10, one of which has been approved in the United States in the first quarter of 2000.

Box 1

Targets, mode of action and mechanisms of resistance of the main classes of antibacterial drugs

Antibiotic	Target	Mode of action	Resistance mechanism
Cell wall			
β-Lactams	Transpeptidases/transglycosylases (PBPs)	Blockade of crosslinking enzymes in peptidoglycan layer of cell walls	β-Lactamases, PBP mutants
Vancomycin	D-Ala-D-Ala termini of peptidoglycan and of lipid II	Sequestration of substrate required for crosslinking	Reprogramming of D-Ala-D-Ala to D-Ala-D-Lac or D-Ala-D-Ser
Protein synthesis			
Macrolides of the erythromycin class	Peptidyl transferase, centre of the ribosome	Blockade of protein synthesis	rRNA methylation, drug efflux
Tetracyclines	Peptidyl transferase	Blockade of protein synthesis	Drug efflux
Aminoglycosides	Peptidyl transferase	Blockade of protein synthesis	Enzymatic modification of drug
Oxazolidinones	Peptidyl transferase	Blockade of protein synthesis	Unknown
DNA replication/repair			
Fluoroquinolones	DNA gyrase	Blockade of DNA replication	Gyrase mutations to drug resistance

Box 1 Figure Proven targets for antibacterial drugs. Cell-wall biosynthesis at the stage of crosslinking of peptidoglycan peptide strands by transpeptidases and transglycosylases is inhibited by the β-lactam antibiotics (penicillins and cephalosporins). Protein biosynthesis at the ribosome is targeted by several classes of antibiotics, including macrolides, tetracyclines, aminoglycosides and oxazolidinones, which block one or more steps involving rRNA and the proteins of the ribosome at the peptidyl transferase centre. The fluoroquinolone antibiotics interrupt DNA replication by trapping a complex of DNA bound to the enzyme DNA Gyrase, a type II topoisomerase.

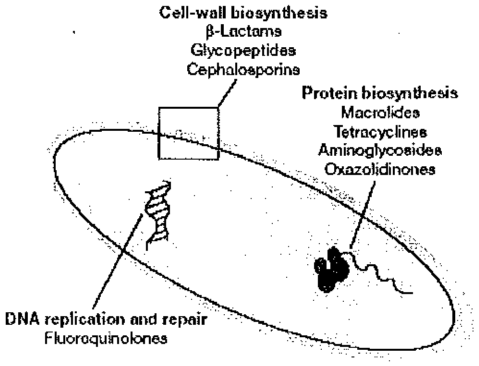


Figure 1.1: Targets, mode of action and mechanisms of resistance of the main classes of antibacterial drugs

1.2.3 DNA replication and repair

The fluoroquinolones, such as ciprofloxacin are synthetic antibiotic structures that kill bacteria by targeting the enzyme DNA gyrase11 (Box 1), the enzyme responsible for uncoiling the intertwined circles of double-stranded bacterial DNA that arise after each round of DNA replication. DNA topoisomerases are classified as type I or type II according to whether transient single-strand breaks (type I) or transient double-strand breaks (type II) are made in the DNA substrate to pass the DNA double helical strands through each other and reduce the linking number (the number of superhelical twists in DNA). Bacterial DNA gyrases are type II topoisomerases and the transient cleavage of both DNA strands involves the reversible attachment of the 5' ends of the cleaved DNA to tyrosyl residues on each of the two GyrA subunits in the active (GyrA)2(GyrB)2 tetramer12. Quinolone antibiotics such as ciprofloxacin are mechanism-based inhibitors of DNA gyrase and act by forming a complex with the enzyme and the doubly cleaved DNA that is covalently tethered to the Gyro subunits11. In the ciprofloxacin complex, the gyrase cannot relegate the cleaved DNA and, as a consequence, double-strand breaks accumulate and ultimately set off the SOS repair system that leads to bacterial cell death. A second type II topoisomerase, known as topoisomerase IV, is also an important target and probably the primary one in *Staphylococcus aureus* infections13. In each of the three main targets viz. cell wall, and protein and DNA biosynthesis, the antibiotics use comparative biochemical differences between prokaryotic machinery and eukaryotic machinery to act selectively. New classes of antibiotics that may work on additional and new targets will have to display equivalent therapeutic indices and efficacy-to-toxicity ratios to gain regulatory approval and widespread acceptance.

1.3 Essential Genes

Essential genes are genes that are indispensable to support cellular life. These genes constitute a minimal gene set required for a living cell. Therefore, the functions encoded by this gene set are essential and could be considered as a foundation of life itself .The definition of the minimal gene set needed to sustain a living cell is of considerable interest not only because it represents a fundamental question in biology, but also because it has much significance in practical use. For example, since most antibiotics target essential cellular processes, essential gene products of microbial cells are promising new targets for antibacterial drugs. The functions encoded by essential genes are considered a foundation of life and therefore are likely to be common to all cells. We have used the DEG database to get the essential genes. Users can BLAST the query sequences against DEG. If homologous genes are found, it is possible that the queried genes are also essential. Users can search for essential genes by their function or name. Users can also browse and extract all the records in DEG. We had extracted all the records in the DEG and have used it to Blast with Human genome and genome of symbiotic organism. Essential gene products comprise excellent targets for antibacterial drugs. Analysis of essential genes could help to answer the question of what are the basic functions necessary to support cellular life. DEG is freely accessible from the website http://tubic.tju.edu.cn/deg/.

1.3.1 *Database of essential genes*

The determination of the minimal gene set for bacteria has only been possible with the advent of the completion of many whole genome sequencing projects and the genome-scale gene inactivation technology. Consequently, essential genes have been determined in a number of different organisms. Essential genes have been determined in Staphylococcus aureus by anantisense RNA technique, in Mycoplasma genitalium by transposon mutagenesis, in Haemophilus influenzae by high-density transposon mutagenesis, in Vibrio cholerae by a mariner-based transposon, in yeast by genetic footprinting , and in M.genitalium and H.influenzae by comparative genomics. We have constructed a Database of Essential Genes (DEG) that contains all the essential genes currently available. These genes include the essential genes identified in the genomes of M.genitalium, H.influenzae , V.cholerae , S.aureus, Escherichia coli and Saccharomyces cerevisiae. The essential genes in the E.coli genome were extracted from the web site http://magpie.genome.wisc.edu/~chris/essential.html, in which the essential genes are collected from a large number of related references. The essential genes in yeast genome were extracted from the yeast genome database http://www.mips.biochem.mpg.de/proj/yeast, which is maintained by the Munich Information Center for Protein Sequences. Each entry of essential genes has a unique DEG identification number, gene reference number, gene function and sequence. All information is stored and operated by using an open-source database management system, MySQL. Users can browse and extract all the records of these entries. In addition, users can also search DEG by gene function or name.

We have installed the BLAST program locally. Therefore, users can BLAST the query sequences against all the essential gene sequences in DEG. One of the applications is the prediction of essential genes based on homologous sequence search against DEG. The functions encoded by essential genes are considered to be generally essential for all cells. It is even believed that some basic functions and principles are common to all cellular life on this planet. Therefore, if the query sequences compared using BLAST have homologous genes in DEG, it is likely that the queried genes are also essential. In addition, by performing the BLAST search against DEG for all the protein-coding genes in a genome, it is possible to define the putative essential genes for the proteomes of newly sequenced genomes. However, caution must be taken in interpreting the BLAST results, since many essential genes are essential only in given growth conditions, such as in rich or minimal medium.

Another application is that by analyzing all the essential genes in DEG, some principles or regulations could be found to answer the question of what are the basic functions necessary to support cellular life. Those principles could lead to the development of new algorithms to predict essential synthesis, energy production and cell division. Some essential genes, however, are somewhat unexpected, such as Embden-Meyerhof-Parnas pathway genes and a purine biosynthesis gene (1). Analysis of DEG, which has all essential genes among different organisms, could help to classify those `unexpected' essential genes. Currently some essential gene projects are still ongoing and the identification of more essential genes is expected. DEG will be updated periodically to include more entries upon the availability of newly identified essential genes. DEG plans to integrate more information about experimental methods for each entry. In the next version of DEG, they also plan to include the essential genes of vertebrates, such as mouse.

DEG 5.2 contains the essential genes in the following Organisms:

No	Organism	Essential genes	Reference
1	Bacillus subtilis	248	Kobayashi, K. et al., 2003 Essential Bacillus subtilis genes. Proc Natl Acad Sci U S A 100: 4678-4683. [PubMed]
2	Escherichia coli MG1655	619	Gerdes, S.Y. et al., 2003 Experimental determination and system level analysis of essential genes in Escherichia coli MG1655. J Bacteriol. 185(19):5673-84. [PubMed]
3	Haemophilus influenzae	638	Akerley, B.J. et al., 2002 A genome-scale analysis for identification of genes required for growth or survival of Haemophilus influenzae. Proc Natl Acad Sci U S A 99: 966-71. [PubMed]
4	Helicobacter pylori	343	Salama, N.R. et al., 2004 Global transposon mutagenesis and essential gene analysis of Helicobacter pylori. J Bacteriol.186: 7926-7935. [PubMed]
5	Mycobacterium tuberculosis H37Rv	614	Sassetti, C.M. et al., 2003 Genes required for mycobacterial growth defined by high density mutagenesis. Mol Microbiol. 48(1):77-84. [PubMed]
6	Mycoplasma genitalium	381	Hutchison, C.A. et al., 1999 Global transposon mutagenesis and a minimal Mycoplasma genome. Science. 286:2165-2169. [PubMed] Glass, J.I. et al., 2006 Essential genes of a minimal bacterium. Proc Natl Acad Sci U S A 103: 425-30. [PubMed]
7	Saccharomyces cerevisiae	878	http://www.mips.biochem.mpg.de/proj/yeast

8	Salmonella typhimuiium	251	Knuth, K. et al., 2004 Large-scale identification of essential Salmonella genes by trapping lethal insertions. Mol Microbiol. 51(6):1729-44. [PubMed]
9	Staphylococcus aureus	308	Forsyth, R.A. et al., 2002 A genome-wide strategy for the identification of essential genes in Staphylococcus aureus. Mol Microbiol. 43:1387-400. [PubMcd] Ko, K.S. et al., 2006 Screening of Essential Genes in Staphylococcus aureus N315 Using Comparative Genomics and Allelic Replacement Mutagenesis. J. Microbiol. Biotechnol. 16(4):623-632. [PubMed]
10	Streptococcus pneumoniae	243	Thanassi, J.A. et al., 2002 Identification of 113 conserved essential genes using a high-throughput gene disruption system in Streptococcus pneumoniae. Nucleic Acids Res. 30:3152-62. [PubMed] Song, J.H. et al., 2005 Identification of Essential Genes in Streptococcus pneumoniae by Allelic Replacement Mutagenesis. Mol Cells. 19(3):365-74. [PubMed]
11	Vibrio cholerae	5	Judson, N., and J. J. Mekalanos, 2000 TnAraOut, a transposon-based approach to identify and characterize essential bacterial genes. Nat Biotechnol 18: 740-745.

Table 1.1: Detailed information about essential genes in DEG

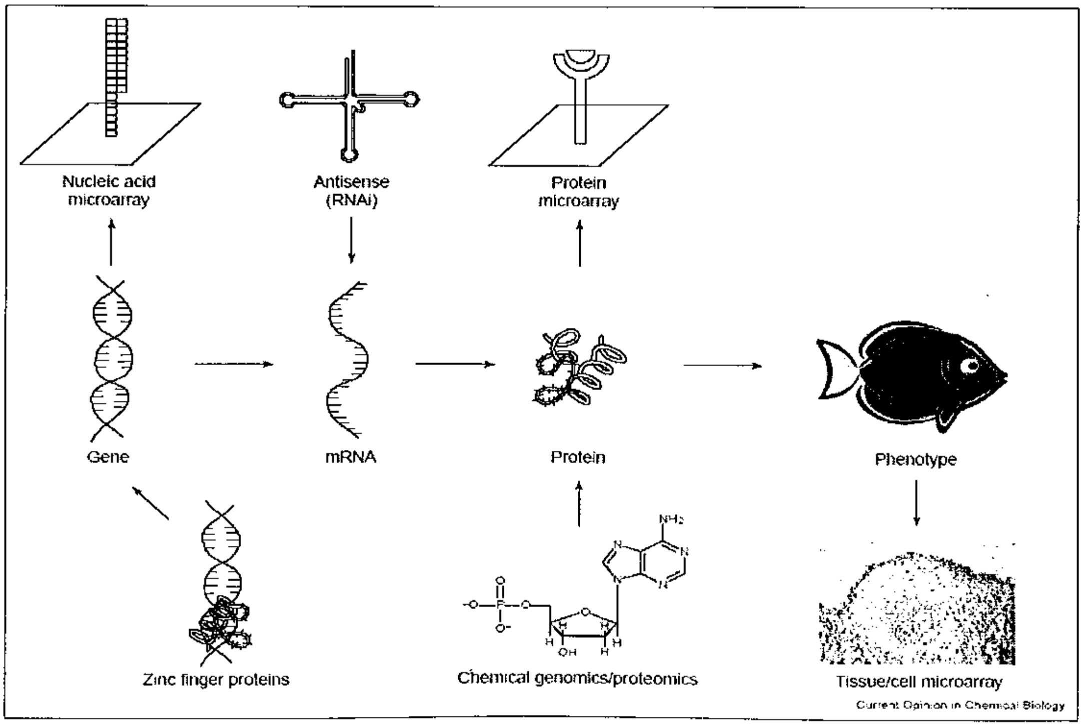
Total number of essential genes present in DEG: **4528**

1.4 High throughput screening of therapeutic target from genome

Traditionally microarrays have been well utilized in genomics/proteomics approaches for gene/protein expression profiling and tissue/cell-scale target identification and validation (Howbrook et al. 2003). Chemical genomics and proteomics are emerging tools for generating phenotype changes, thus leading to target and hit identifications. NMR-based screening, as well as activity-based protein profiling, are trying to meet the requirement of high-throughput target identification (Sem Et Al. 2001). Besides being used high-throughput experiments, bioinformatics can contribute to the processes of target identification and validation by providing functional information of target candidates and positioning information on the biological networks.

1.4.1 *Microarrays*

Target identification seeks to identify new targets, normally proteins (or DNA/RNA), whose modulation might inhibit or reverse disease progression. Current technologies enable researchers to attempt to correlate changes in gene (genomics) and protein (proteomics) expression with human disease, in the hope of finding new targets. Microarrays are a well-utilized tool in both academic and industrial research laboratories. They can be used to assess gene and protein expression (via nucleic acid or protein microarrays) to identify novel targets, and can also be used to validate the found targets at the tissue or cell scale (via tissue or cell microarrays) (Howbrook DN, van der Valk AM, O'Shaughnessy MC, Sarker DK, Baker SC: Developments in microarray technologies. Drug Discov Today 2003, 8:642-651.)



Current Opinion in Chemical Biology

Figure 1.2: Types of Microarray techniques

a) *Nucleic acid microarrays*

Today, nucleic-acid microarrays, which primarily use short oligonucleotides (15–25 nt), long oligonucleotides (50–120 nt) and PCR-amplified cDNAs (100–3000 base pairs) as array elements, are overwhelmingly dominant because of the relatively easy synthesis and the chemical robustness of DNA. Data generated from genome sequencing projects in several organisms has provided the opportunity to build comprehensive maps of transcriptional regulation. Array-based gene expression analysis (immobilized DNA probes hybridizing to RNA or cDNA targets) has enabled parallel monitoring of cellular transcription at the level of the genome. Thus, nucleic-acid microarrays have had a significant impact on our understanding of normal and abnormal cell biochemistry and, thus, on the choice of targets for drug design. In oncology, data generated from high density oligonucleotide microarrays from Affymetrix containing 62 907 probe sets have been analyzed and compared, to identify 97 genes as physiological targets of the retinoblastoma protein pathway, deregulation of which is a hallmark of human cancer (Vernell R, Helin K, Muller H: Identification of Target Genes of the p16INK4A-pRB-E2F Pathway. J Biol Chem 2003, 278:46124-46137). Further characterization of these genes should provide insights into how this pathway controls proliferation, thus providing potential therapeutic targets.

b) *Protein microarrays*

Because most drug targets are proteins, protein and peptide microarrays are set to have an important impact on drug discovery. Protein arrays, an emerging yet very promising technology, are now being used to examine enzyme–substrate, DNA–protein and protein–protein interactions (Zhu H, Snyder M: Protein chip technology. Curr Opin Chem Biol 2003, 7:55-63). By profiling the differential expression of proteins using antibody arrays and correlating the changes to a disease phenotype, putative targets (and biomarkers) to a particular disease can be identified, although to date, such microarrays have not been used to their full potential because of difficulties with the technology. One example using protein microarrays made up of 83 different antibodies enables monitoring alterations of the protein levels in hepatocellular carcinomas (HCCs) and non-neoplastic liver tissue. Further analysis of altered proteins was performed using western blot analysis and tissue microarrays containing 210 HCC specimens and corresponding liver tissue. This approach revealed differential expression between HCC and normal liver of 32 of the 83 proteins examined: 21

of these were up-regulated and 11 down-regulated (Chen GYJ, Uttamchandani M, Zhu Q, Wang G, Yao SQ: Developing a strategy for activity-based detection of enzymes in a protein microarray. ChemBioChem 2003, 4:336-339). Another very interesting example is the use of fluorescent substrate compounds for protein microarrays. Various enzymes immobilized on microarray slides are used to screen fluorescently labeled enzyme substrates, which are small molecules. More importantly, this strategy is based on the mechanism of the reaction between the ligands and proteins, thus demonstrating the approach as an activity-based and high-throughput method. Further application of this method may lie on targeting known drugs or biological active compounds.

c) Tissue and cell microarrays

Almost all published tissue microarray studies have been related to the analysis of tumors. An alternative to the use of whole-tissue specimens is the use of live cell microarrays, which can be used to identify potential drug targets by functionally characterizing large numbers of gene products in cell-based assays. In one example, a tissue microarray was constructed using 210 human HCCs and corresponding normal liver, and followed by immunohistochemical analysis to test the clinical value of the proteins that have been identified.

d) Antisense technology

A key strategy in target validation is to determine what happens, with respect to phenotype and/or the expression of other genes in cells or model organisms, if a gene of interest is either deleted or its activity is inhibited. Gene knockout mimics the activity of a drug that completely inhibits the normal function of the gene's product. A temporary knockout, a so called knockdown, is another popular alternative for real-time analysis of the gene function. Several important strategies for gene knockdown involve the use of specific RNAs and/or RNA or DNA analogues (Scherer LJ, Rossi JJ: Approaches for the sequence specific knockdown of mRNA. Nat Biotechnol 2003, 21:1457-1465).

1.5 In silico prediction of therapeutic target

The availability of genome-scale sequenced data of microbes and the human genome has revolutionized the field of drug-discovery against threatening human pathogens (Lander et al. 2001; Venter et al. 2001). The strategies for drug design and development are progressively shifting from the genetic approach to the genomic approach (Galperin and Koonin 1999). Novel drug targets are required in order to design new defense against antibiotic sensitive pathogens. Comparative genomics and bioinformatics provide new opportunities for finding optimal targets among previously unexplored cellular functions based on an understanding of their related biological processes in bacterial pathogens and their hosts. In general, a target should provide adequate selectivity; yielding a drug which is specific or highly selective against the pathogen with respect to the human host. Moreover, the entire approach is built on the assumption that the potential target must play an essential role in the pathogen's survival and constitute a critical component in its metabolic pathway (Mushegian and Koonin 1996). The above approach to target identification is essentially subtractive because we use a subtraction dataset while comparing the two genomes under consideration (Bruccoleri et al. 1998). The focus is on the complement of the genome of the pathogen that is essential for it but is not present in human. Multiple approaches to locate essential genes in a given organism exist, some of which focus on the concept that essential genes tend to be evolutionarily conserved over species (Itaya 1995; Tatusov et al. 1997; Koonin et al. 1998; Kobayashi et al. 2003). Subtractive genomics has been successfully used by authors to locate novel drug targets in *Pseudomonas aeruginosa* (Sakharkar et al. 2004) and *Helicobacter pylori* (Dutta et al. 2006). The work has been effectively complemented with the compilation of the Database of Essential Genes (DEG) for a number of pathogenic microorganims (Zhang et al. 2004).

There are many *in silico* approaches for finding drug targets in pathogenic bacteria. Bruccoleri et al. (1998) has developed a simple and computational tool that can determine concordances of putative gene products showing sets of proteins conserved across one set of user-specified genomes, but are not present in another set of user-specified genomes, but the availability of this approach as an automated tool is limited. An automated tool, T-iDT developed by Singh et al. (2006) predicts highly conserved genes, which are essential for pathogenic bacteria with no similarities with the host genes as potential drug targets. This and other existing tools use only human genome sequence as a template for comparison against pathogens. However, comparison with the symbiotic organisms living within the human body cannot be ruled out for successful drug development. Fortunately the genome sequences of all these symbiotic bacteria are available and can be used as template for comparison with pathogen bacteria. Although bioinformatics tools and resources can be used to identify putative drug targets, validating these targets is again very essential. Traditionally it requires an understanding of the role of the gene or protein in the disease process and is heavily dependent on laboratory-based work.

Genomics and proteomics technologies have created a paradigm shift in the drug discovery process, with bioinformatics having a key role in the exploitation of genomic, transcriptomic, and proteomic data to gain insights into the molecular mechanisms that underlie disease and to identify potential drug targets. In this work we discuss the current state of the art for some of the bioinformatics approaches to identifying drug targets. It makes use of database of essential genes (DEG) (Ref: http://tubic.tju.edu.cn/deg/) and the subtractive genomic approaches to compare with the pathogen bacteria versus human as well as its symbiotic bacteria, including identifying new members of successful target classes and their functions, predicting disease relevant genes. In addition we use several *in silico* validation strategies to rank the predicted therapeutic targets to be use for drug targeting. The input for the tool is a set of genes or proteins in FASTA format. Our tool predicts both essential gene/protein sets as well as target gene/protein sets in a given data set and further validate them. The tool is stand-alone software, which provides a fast and automated approach for finding drug targets with different stringency levels. Users have the choice to change the stringency level at two stages of the tool, one for the essential genes and the other for homologue genes. The tool was tested on three different bacterial genomes.

Covering issues that range from prescreening target selection to genetic modeling and valuable data integration, the developed online tool "In Silico Identification of Therapeutic Target and Validation" is a self-contained and practical guide to the various computational tools that can accelerate the identification and validation stages of drug target discovery and determine the biological functionality of potential targets more effectively.

OBJECTIVES

Drug target identification involves acquiring a molecular level understanding of a specific disease state and includes analysis of gene sequences, protein structures, protein interactions and metabolic pathways. The ultimate goal of the process is to discover a suitable target whose biological activity can be directly linked to a pathological process. In the age of genomics, discovery of novel drug targets needs to incorporate and integrate different sources of data including gene expression data, gene sequence data, and gene polymorphism data and so on. Many public biological databases are warehousing and providing a great amount of functional information for drug discovery. Yet one of the most important information is the annotation of human genome itself and its associated symbiotic organisms. In addition, the publicly available tools are as important as the data and include algorithms for gene prediction, sequence homology searching, prediction of function and so on. Hence in this study attempts has been made to develop an automated tool for prediction and validation of drug targets from user given genomic sequence for bacterial pathogen with following objectives.

1. To integrating existing data (essential genes) from public databases and piping the tools for gene prediction using BLAST search as subtractive genomic tool.

2. To validate the predicted drug targets using a special scoring scheme based on efficiency of promoters, Shine Dalgarno sequence, essential function for survival of bacteria and drug binding motifs.

3. To test the reliability of developed *in silico* tool for prediction and validation of drug targets in some bacterial pathogen like; *Mycobacterium tuberculosis, Mycobacterium leprae* and *Bacillus anthracis.*

4. To integrate all the modules included in this work together to develop a standalone software package for public uses.

CHAPTER 2

In Silico identification of therapeutic targets from bacterial pathogen

In this work we have developed an automated tool for *in silico* identification of therapeutic target from the user submitted genomic sequence. This is based on the concept of subtractive genomics as explained in the following protocol.

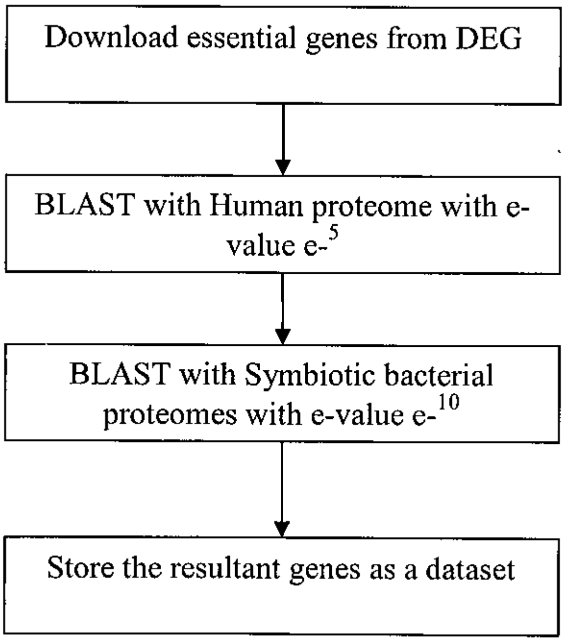
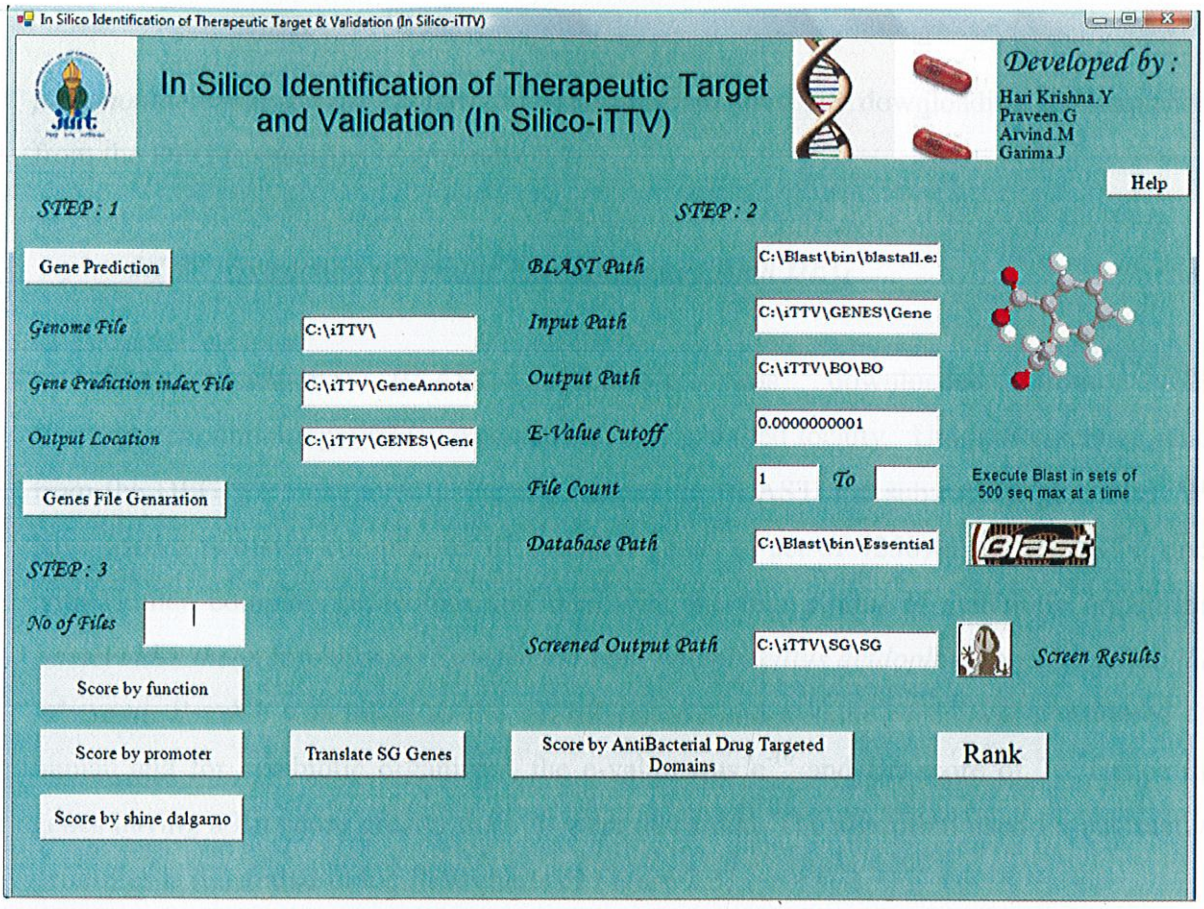

Download essential genes from DEG
BLAST with Human proteome with e-value e-^{5}
BLAST with Symbiotic bacterial proteomes with e-value e-^{10}
Store the resultant genes as a dataset

Figure 2.1: General Protocol for Subtractive Genomics

Development of the tool can be categorized in different phases depending upon the functionality and scheduling. Protocols of the programs used while developing the tool and the complete program itself are described in their respective modules in this chapter.

A snapshot of the tool interface is provided below:


In Silico Identification of Therapeutic Target & Validation (In Silico-iTTV)

In Silico Identification of Therapeutic Target and Validation (In Silico-iTTV)

Developed by :
Hari Krishna.Y
Praveen.G
Arvind.M
Garima.J

STEP : 1

Genome File: C:\iTTV\

Gene Prediction index File: C:\iTTV\GeneAnnota

Output Location: C:\iTTV\GENES\Gen

STEP : 2

BLAST Path: C:\Blast\bin\blastall.e

Input Path: C:\iTTV\GENES\Gene

Output Path: C:\iTTV\BO\BO

E-Value Cutoff: 0.0000000001

File Count: 1 To

Execute Blast in sets of 500 seq max at a time

Database Path: C:\Blast\bin\Essential

Screened Output Path: C:\iTTV\SG\SG

STEP : 3

No of Files

Figure 2.2: Snapshot of tool interface

2.1 Phase I: Extraction of unique essential genes from database of essential genes.

2.1.1 Essential genes of bacterial pathogen

The database of essential genes of bacteria (DEG version 3.2) was downloaded from http://tubic.tju.edu.cn/deg/ using a program, LWP module in PERL and manually compiled to use as a stand-alone database for the BLAST program. It consists of a collection of 4528 essential genes from 11 completely annotated bacterial genomes (*Bacillus subtilis*: 248; *Escherichia coli* (MG1665):619; *Haemophilus influenzae*: 638; *Helicobacter pylori*: 343; *Mycobacterium tuberculosis H37Rv*: 614; *Mycoplasma genitalium*: 381; *Saccharomyces cerevisia*: 878; *Salmonella typhimurium*: 251; *Staphylococcus aureus*: 308; *Streptococcus* *pneumoniae*: 243; *Vibrio cholera*: 5). The program used for downloading the sequences from the DEG is specified in appendix 2.1.

2.1.2 Extraction of unique essential genes from DEG

The standalone BLAST executable was downloaded from NCBI (ftp://ftp.ncbi.nlm.nih.gov/blast/executables/) and installed locally. Uniquee essential genes from the DEG were extracted using the standalone BLAST and subtractive genomics. We have used a two step comparison of essential genes (a set of 4258) using BLASTx, once with Human complete proteomes and then with the four strains of symbiotic organisms: *Bacteroides thetaiotaomicron, Escherichia coli, Lactobacillus acidophilus* and *Lactobacillus johnsoni*i. E-value was taken as the cut-off criteria and a value of e^{-5} was considered for human and for symbiotic organisms, the e-value was e^{-10} and Bit score of 70. Hence the genes having a Bit-score greater than 70 were discarded. The algorithm used for subtractive genomics is illustrated in the flow charts of Figure 2.3.

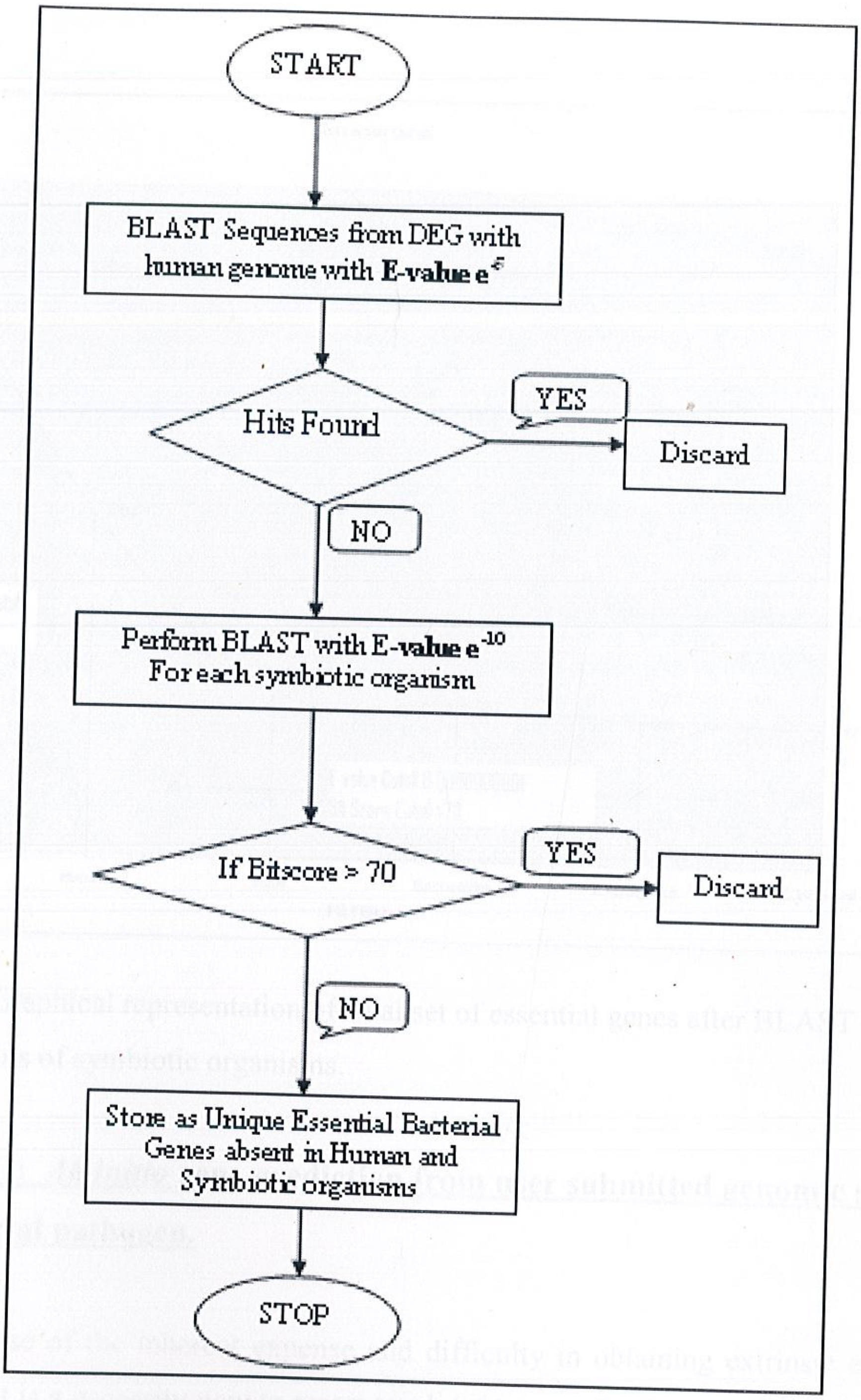

START
BLAST Sequences from DEG with human genome with E-value e^{-5}
Hits Found
YES
Discard
NO
Perform BLAST with E-value e^{-10} For each symbiotic organism
If Bitscore > 70
YES
Discard
NO
Store as Unique Essential Bacterial Genes absent in Human and Symbiotic organisms
STOP

Figure 2.3: Flowchart for finding unique essential bacterial genes non-homolog to human and its symbiotic organisms.

After comparative study with human genomes and its symbiotic organisms, we have extracted a set of 917 unique essential genes (Figure 2.4). These genes are absent in above organisms and thus can be used as an index for predicting essential genes in pathogen bacteria.

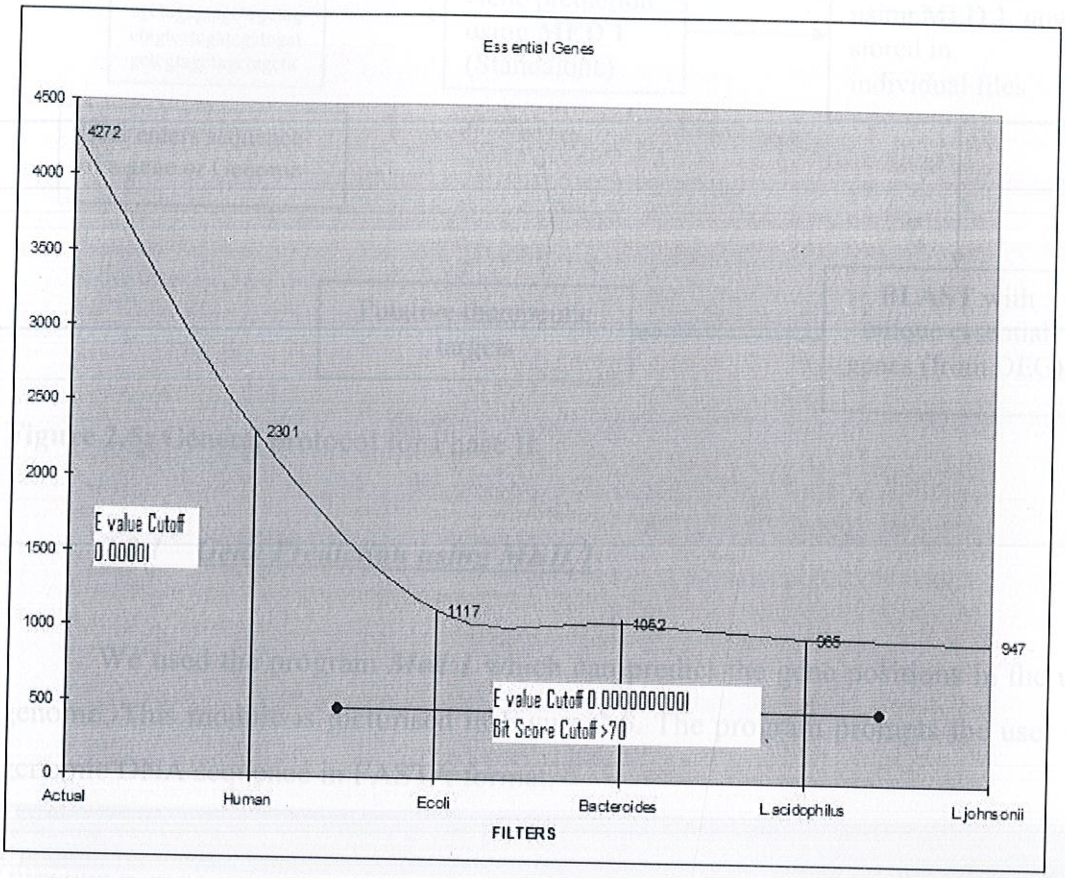
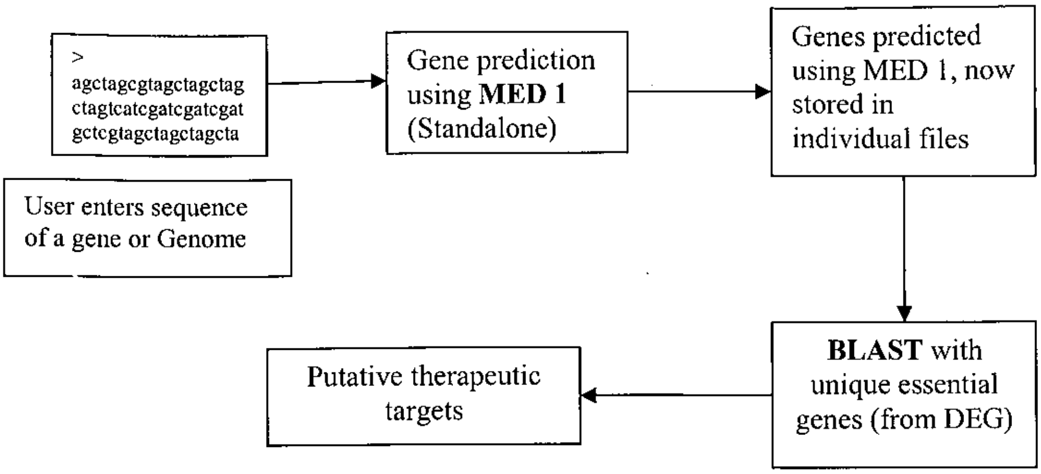


Figure 2.4: Graphical representation of final set of essential genes after BLAST with human and four strains of symbiotic organisms.

2.2 Phase II: *Ab initio* gene prediction from user submitted genomic sequence of bacterial pathogen.

Because of the inherent expense and difficulty in obtaining extrinsic evidence for many genes, it is a necessity now to resort to *ab initio* gene finding, in which genomic DNA sequence alone is systematically searched for certain distinctive, signs of protein-coding genes. *Ab initio* gene finding might be more accurately characterized as gene *prediction*, since extrinsic evidence is generally required to conclusively establish that a putative gene is functional. The general protocol for Phase II can be depicted using the following flowchart:

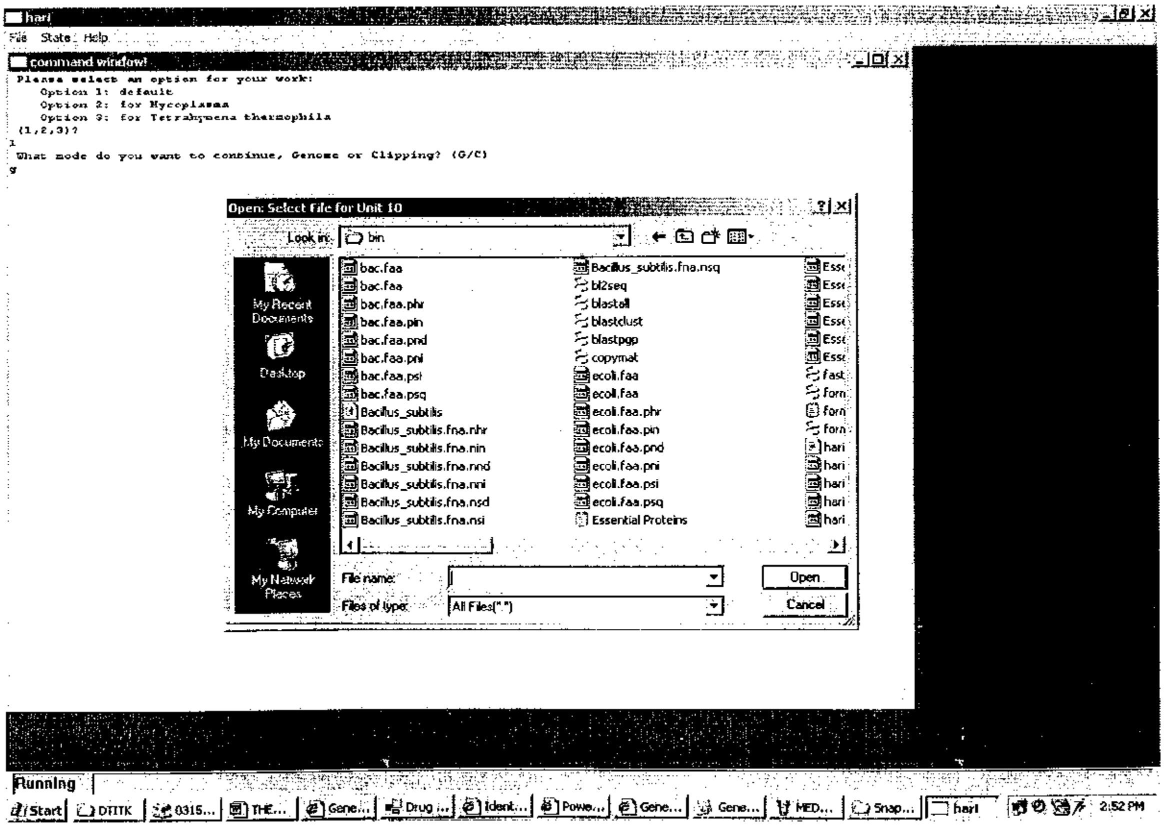


User enters sequence of a gene or Genome
Gene prediction using MED 1 (Standalone)
Genes predicted using MED 1, now stored in individual files
BLAST with unique essential genes (from DEG)
Putative therapeutic targets

Figure 2.5: General protocol for Phase II

2.2.1 Gene Prediction using MED 1

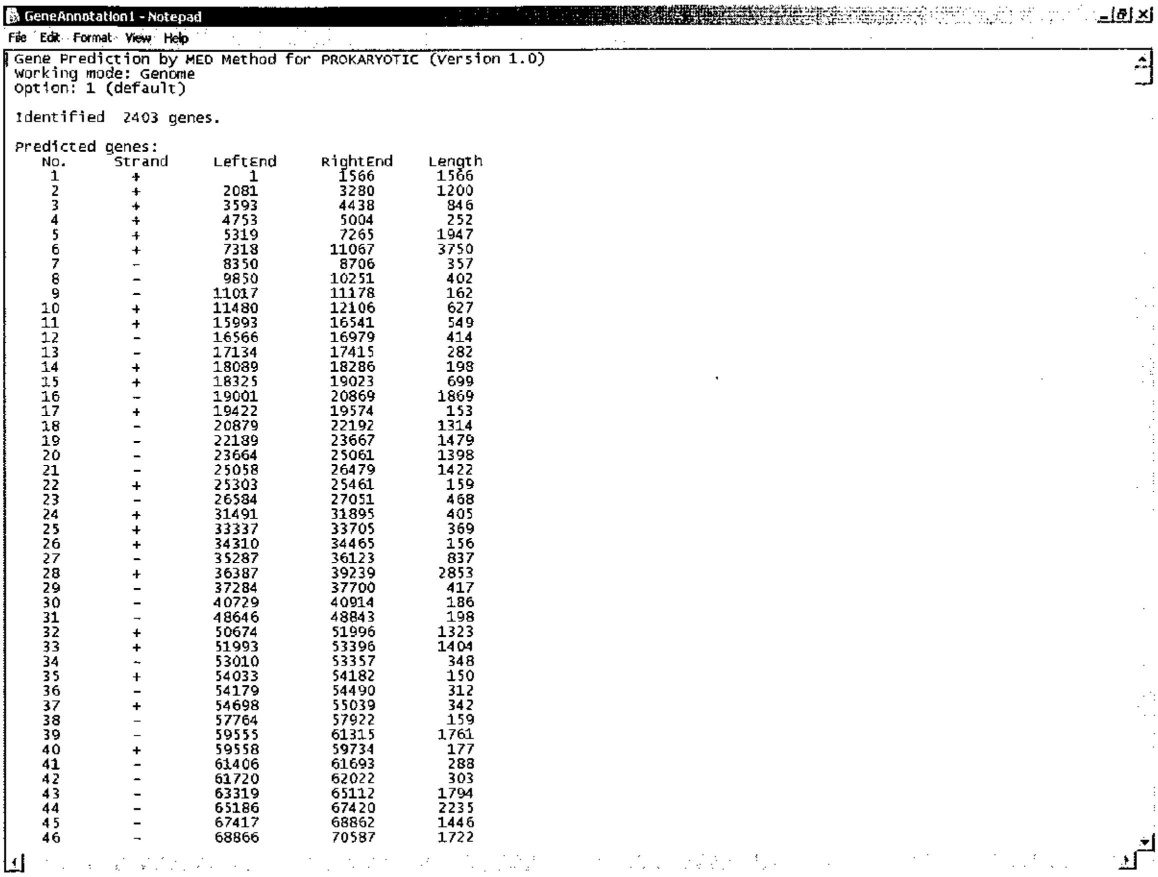
We used the program ***Med 1*** which can predict the gene positions in the user given genome. This module is picturised in Figure 2.6. The program prompts the user to input a genomic DNA sequence in FASTA format.



```
 Please select an option for your work:
     Option 1: default
     Option 2: for Mycoplasma
     Option 3: for Tetrahymena thermophila
 (1,2,3)?
1
 What mode do you want to continue, Genome or Clipping? (G/C)
g
```

Figure 2.6: Input Window to Gene prediction program MED 1.

Archaeal and bacterial genes typically comprise uninterrupted stretches of DNA between a start codon (usually ATG, but in a minority of genes, GTG, TTG, or CTG) and a stop codon (TAA, TGA, or TAG; alternative genetic codes of certain bacteria, such as mycoplasmas, have only two stop codons). MED 1 generally searches for these start and stop codons and give their index or positions along with the length of the gene as the output. It also gives the strand information from which the gene is predicted i.e. + or – strand. Finally, the output is stored in a file GeneAnnotation1.txt (Figure 2.7).



```
Gene Prediction by MED Method for PROKARYOTIC (Version 1.0)
Working mode: Genome
Option: 1 (default)

Identified  2403 genes.

Predicted genes:
    No.      Strand       LeftEnd       RightEnd     Length
     1         +              1          1566        1566
     2         +           2081          3280        1200
     3         +           3593          4438         846
     4         +           4753          5004         252
     5         +           5319          7265        1947
     6         +           7318         11067        3750
     7         -           8350          8706         357
     8         -           9850         10251         402
     9         -          11017         11178         162
    10         +          11480         12106         627
    11         +          15993         16541         549
    12         -          16566         16979         414
    13         -          17134         17415         282
    14         +          18089         18286         198
    15         +          18325         19023         699
    16         -          19001         20869        1869
    17         +          19422         19574         153
    18         -          20879         22192        1314
    19         -          22189         23667        1479
    20         -          23664         25061        1398
    21         -          25058         26479        1422
    22         +          25303         25461         159
    23         -          26584         27051         468
    24         +          31491         31895         405
    25         +          33337         33705         369
    26         +          34310         34465         156
    27         -          35287         36123         837
    28         +          36387         39239        2853
    29         -          37284         37700         417
    30         -          40729         40914         186
    31         -          48646         48843         198
    32         +          50674         51996        1323
    33         +          51993         53396        1404
    34         -          53010         53357         348
    35         +          54033         54182         150
    36         -          54179         54490         312
    37         +          54698         55039         342
    38         -          57764         57922         159
    39         -          59555         61315        1761
    40         +          59558         59734         177
    41         -          61406         61693         288
    42         -          61720         62022         303
    43         -          63319         65112        1794
    44         -          65186         67420        2235
    45         -          67417         68862        1446
    46         -          68866         70587        1722
```

Figure 2.7: Output of the Gene prediction program MED 1, results in GeneAnnotation1.txt

2.2.2 Extraction of genes sequence from GeneAnnotation1.txt file

We have written a separate program in Perl for extracting the genes sequences from the user given genomic sequence using GeneAnnotation index file. After complete execution of the program, we obtained individual gene files. This program has been specified in appendix 2.2.

2.2.3 *Prediction of putative drug targets using standalone BLAST*

The unique essential genes from the user given bacterial pathogen can be obtained by comparing with the unique essentail genes obtained from phase 1. Here we have used tThe BLASTx program which picks up each of the genes predicted using MED1 in the previous step and BLAST with the unique essential genes dataset prepared during Phase I. A cut off value of e^{-10} was used to filter out unique essential genes which are present only in the given bacterial pathogen. But the user can change the E-value cut off as desire.

2.2.4 *Parsing of BLAST output*

Once the BLAST has been performed, the number of output files will be equal to the number of input gene files. But whole BLAST output will not be useful to the user. To parse the BLAST output, another perl program was written (which is specified in appendix 2.3) which selects only those BLAST output files which have high similarity to the essential genes dataset. In order to achieve this, we have specified a bit score of 100, below which, the genes will be discarded. Hence, we will be left with only those genes which have high similarity with the essential dataset and these can be the potential targets as they are absent in both human and symbiotic bacteria.

The entire algorithm implemented in phase II is explained in flowchart below (Figure 2.8)

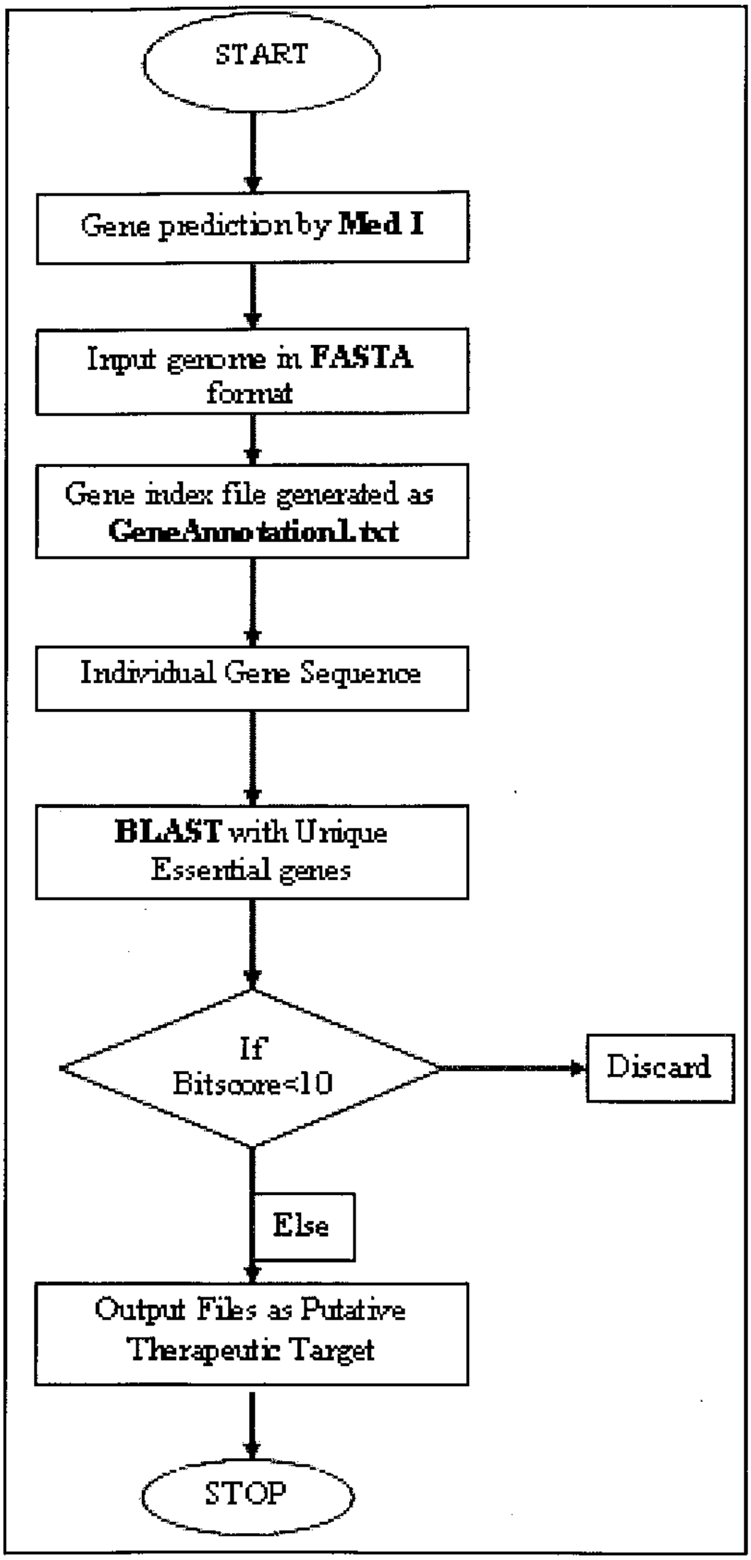

START
Gene prediction by Med I
Input genome in FASTA format
Gene index file generated as GeneAnnotation1.txt
Individual Gene Sequence
BLAST with Unique Essential genes
If Bitscore<10
Discard
Else
Output Files as Putative Therapeutic Target
STOP

Figure 2.8: Flowchart for *ab initio* prediction of genes and finding essential genes from user giving input sequence using Med I and subtractive genomics.

2.3 Phase III: Validation of putative drug targets.

Predicted drug targets in phase II is validated further by using a special scoring system accomplished in four steps: scoring on the basis of function, scoring based on strong promoter and weak promoter, scoring based on consensus Shine Dalgarno sequence and scoring based on presence of drug binding domain in the predicted drug targets. The general protocol adopted for validation is given below (Figure 2.9) an explained in detail further.

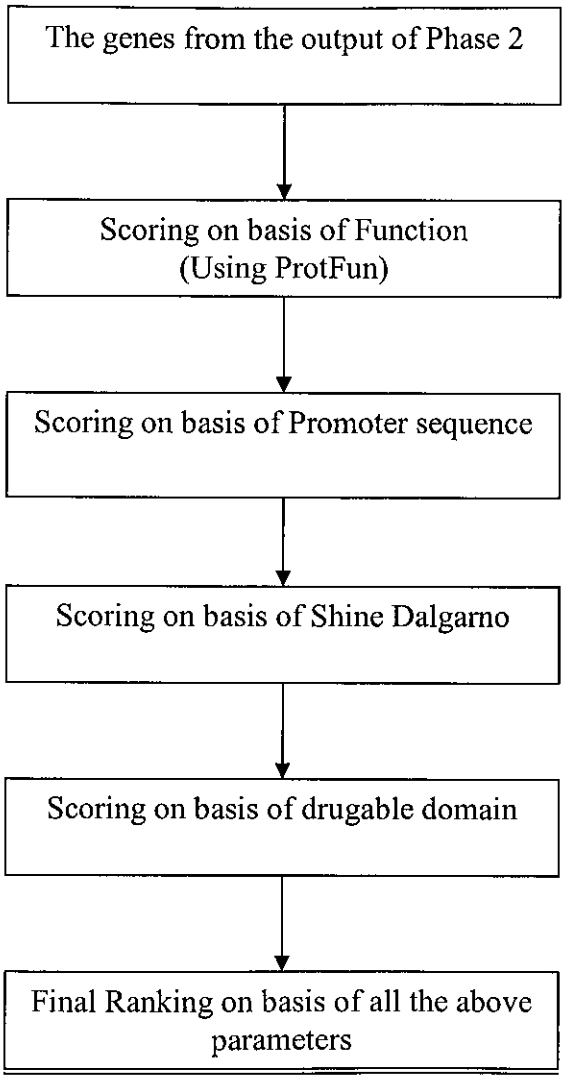

The genes from the output of Phase 2
Scoring on basis of Function (Using ProtFun)
Scoring on basis of Promoter sequence
Scoring on basis of Shine Dalgarno
Scoring on basis of drugable domain
Final Ranking on basis of all the above parameters

Figure 2.9: The general protocol adopted for validation

2.3.1 Scoring of predicted targets on the basis of predicted function

The functions of all the unique essential genes were predicted using ProtFun version 2.2 (http://www.ncbi.nlm.nih.gov/BLAST/tutorial/Altschul-1.html). The program was downloaded from http://www.cbs.dtu.dk/services/ProtFun/, manually compiled and installed locally. *Ab initio* prediction of protein functions from protein sequence is based on cellular role, enzyme class and Gene Ontology category. It uses large number of features including post-translational information and localization aspects of the protein; hence the prediction is more reliable. All the unique essential genes (a set of 917) are classified into 12 different functional categories based on the cellular roles (Figure 2.10).

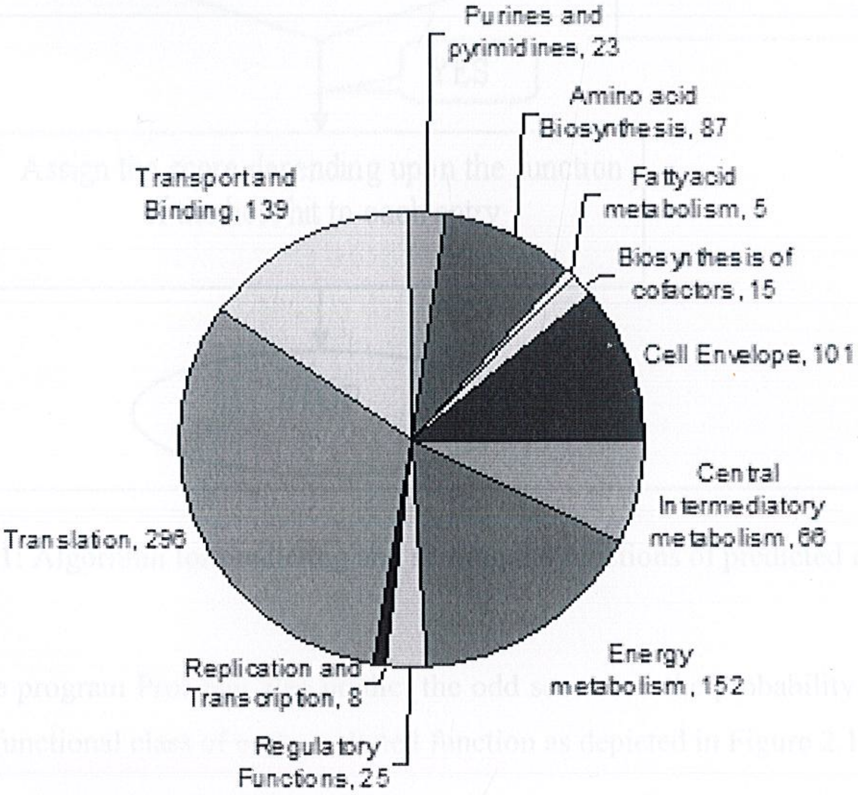


Figure 2.10: Unique essential genes (a set of 917) classified into 12 different functional categories based on the cellular roles

The program for predicting and scoring the functions of predicted essential genes is done using a program coded in Perl and the algorithm used is depicted in flow charts in Figure 2.11.

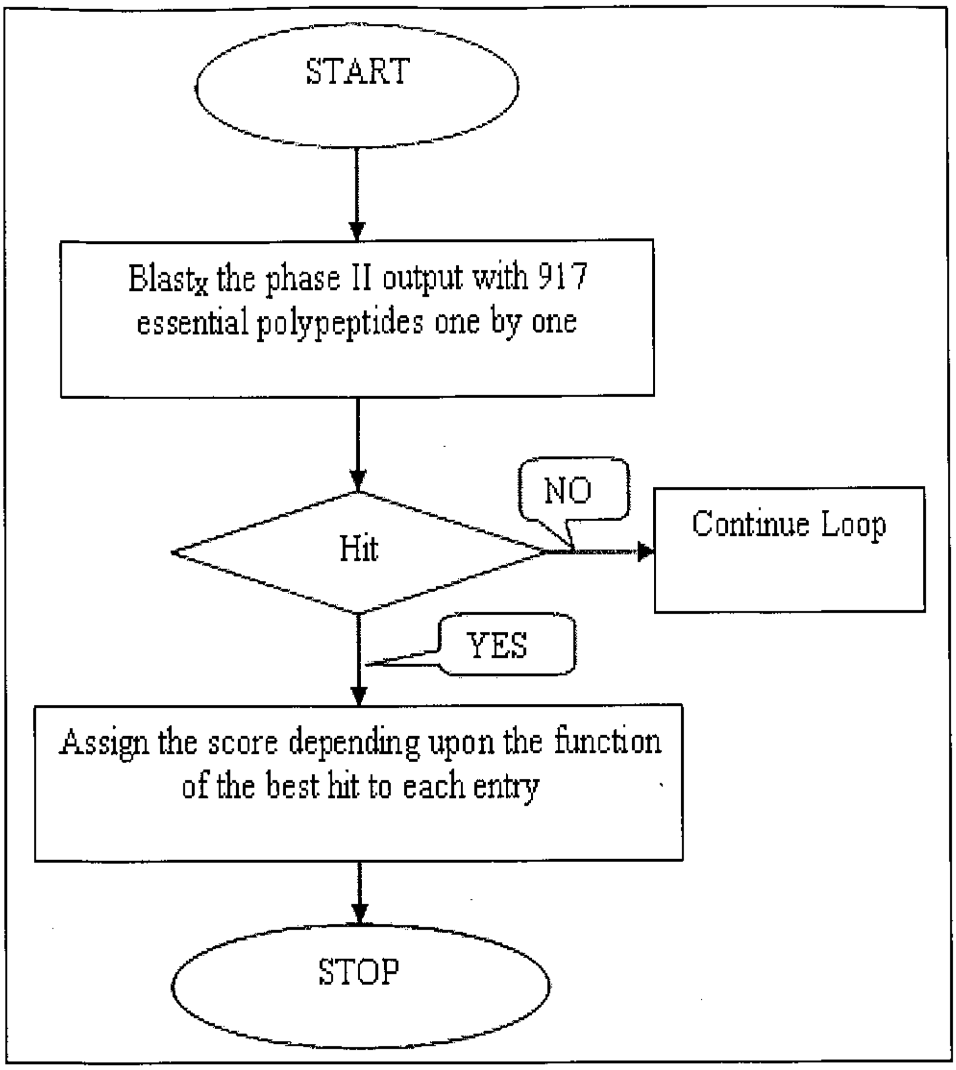
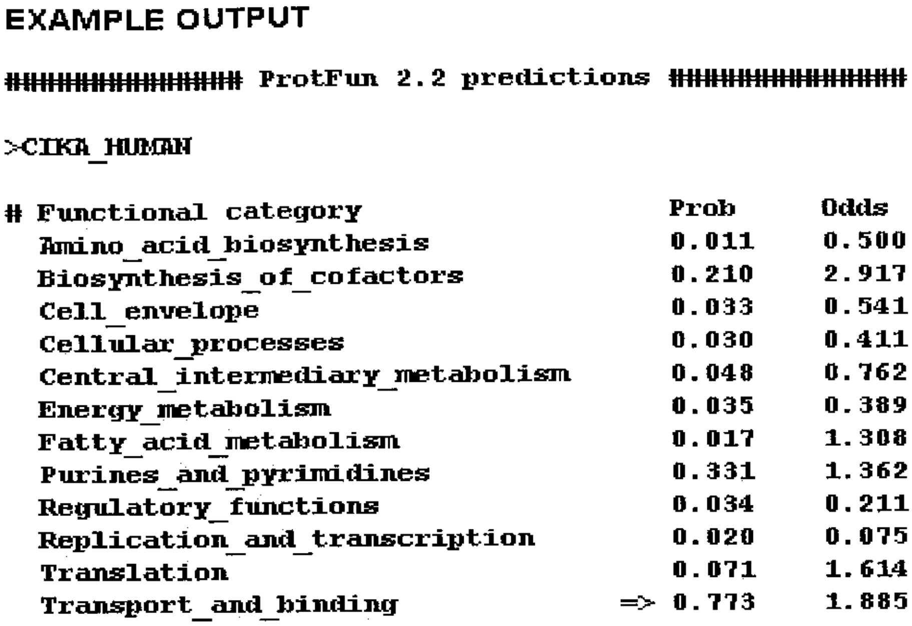

START
Blast$_X$ the phase II output with 917 essential polypeptides one by one
NO
Hit
Continue Loop
YES
Assign the score depending upon the function of the best hit to each entry
STOP

Figure 2.11: Algorithm for predicting and scoring the functions of predicted essential genes

The program Prot Fun also predict the odd score and the probability belonging to a particular functional class of each predicted function as depicted in Figure 2.12.



```
EXAMPLE OUTPUT

################ ProtFun 2.2 predictions ################

>CIKA_HUMAN

# Functional category                          Prob      Odds
  Amino_acid_biosynthesis                      0.011     0.500
  Biosynthesis_of_cofactors                    0.210     2.917
  Cell_envelope                                0.033     0.541
  Cellular_processes                           0.030     0.411
  Central_intermediary_metabolism              0.048     0.762
  Energy_metabolism                            0.035     0.389
  Fatty_acid_metabolism                        0.017     1.308
  Purines_and_pyrimidines                      0.331     1.362
  Regulatory_functions                         0.034     0.211
  Replication_and_transcription                0.020     0.075
  Translation                                  0.071     1.614
  Transport_and_binding                     => 0.773     1.885
```

Figure 2.12: Output of the ProtFun 2.2

However, we have adopted a different scoring method to score each predicted function. The score of each predicted function of putative drug target is assigned based on the score of all 917 unique essentail genes. Initially the score to each unique essential gene is computed by comparing with the percentage of occurrence of individual functions in bacteria (Figure 2.13)

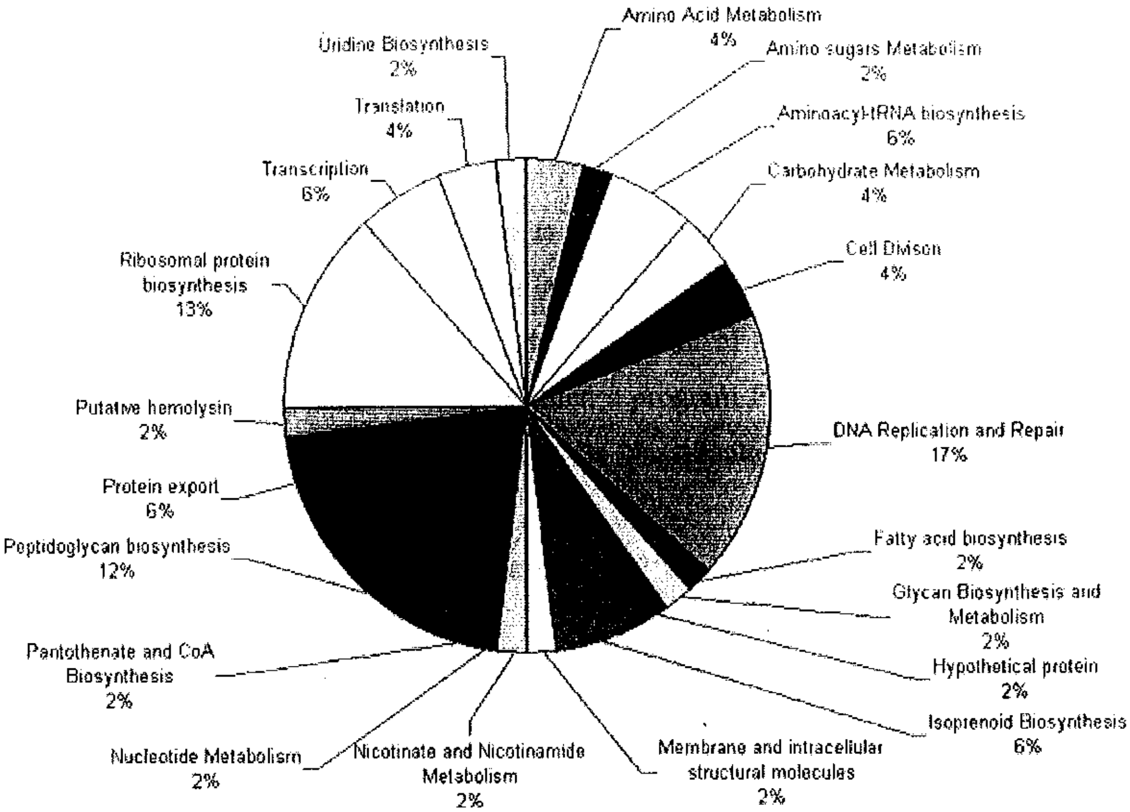


Figure 2.13: Functions of essential genes in bacteria (in percentage).

[**Ref:** In Silico Biology 6, 0045 (2006); ©2006, Bioinformation Systems e.V (T-iDT: Tool for identification of drug target in bacteria and validation by Mycobacterium tuberculosis)]

The program for scoring the genes on basis of functions is coded in Perl and is specified in appendix 2.4.

2.3.2 Scoring of predicted drug targets based on consensus promoter sequence

In prokaryotes, the promoter consists of two short sequences at -10 and -35 position upstream from the transcription start site. The sequence at -10 is called the Pribnow box and usually consists of the six nucleotides TATAAT, whereas at -35 position it consist of another hexamer TTGACA, which are absolutely essential to start transcription in prokaryotes.

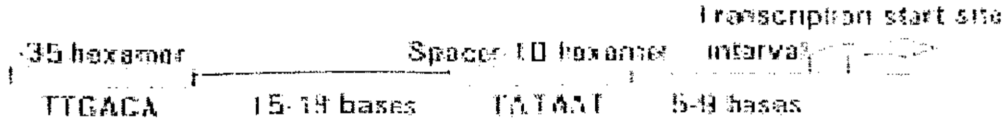


Figure 2.14: Promoter sequence

The rate of transcription of genes is depends on strong, average and weak promoter sequence (classification is based on similarity with the mentioned promoter sequence). A gene with low expression is insignificant for survival of the bacteria and is generally not preferable for drug targeting. We have written a program for predicting promoter sequence of essential genes using PSSM matrix and its classification as strong, average and weak promoter.

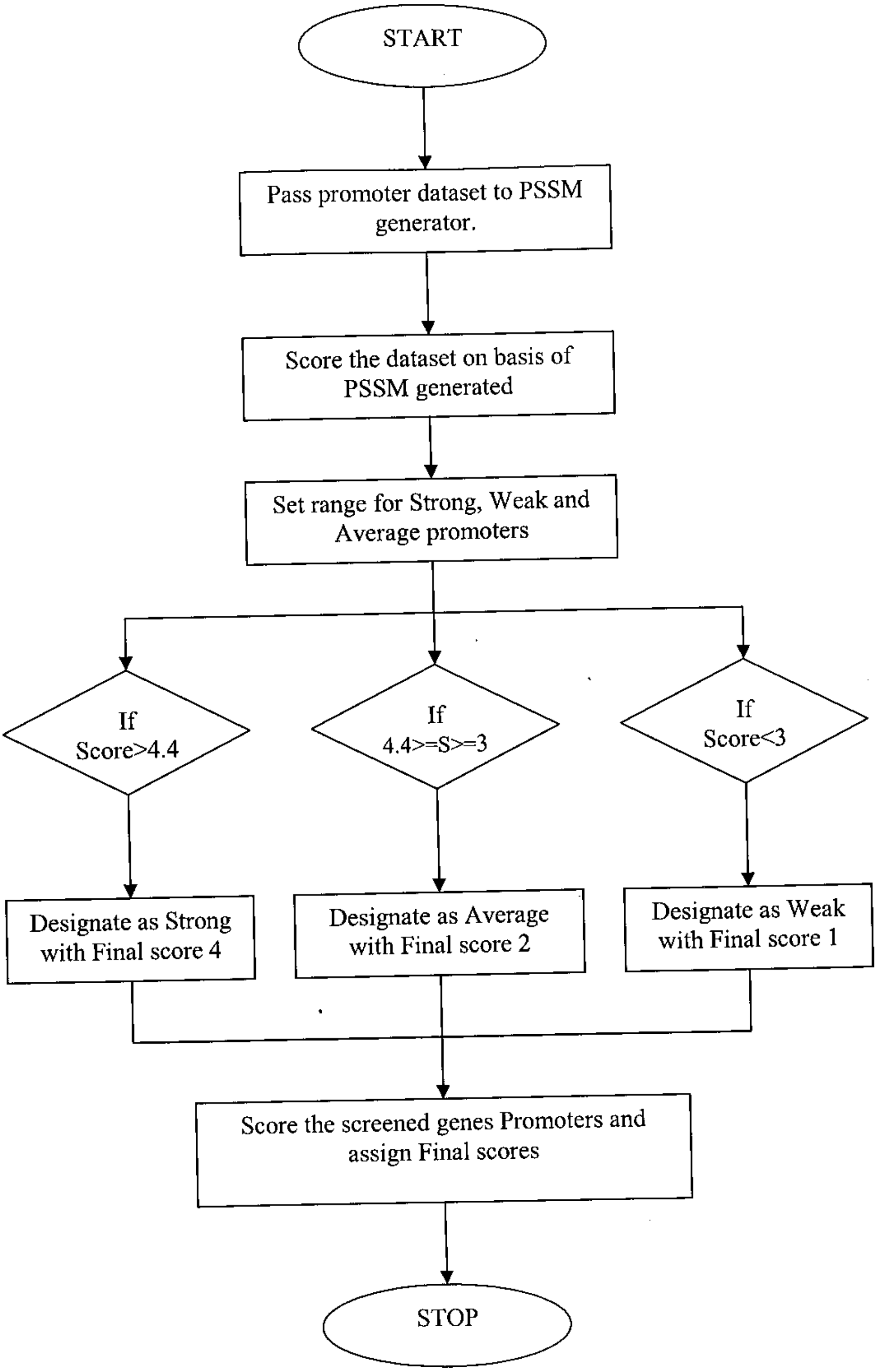
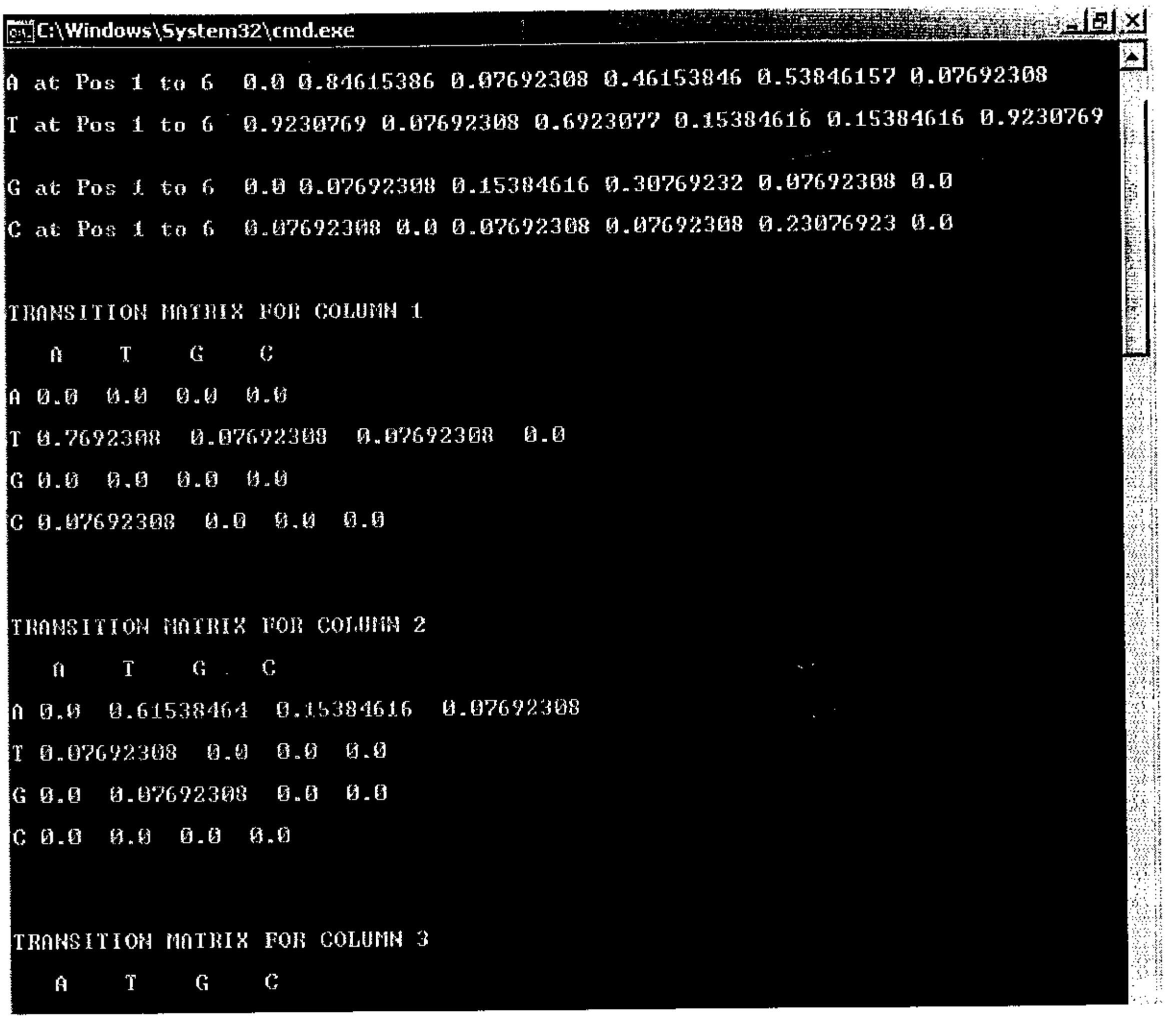

START
Pass promoter dataset to PSSM generator.
Score the dataset on basis of PSSM generated
Set range for Strong, Weak and Average promoters
If Score>4.4
If 4.4>=S>=3
If Score<3
Designate as Strong with Final score 4
Designate as Average with Final score 2
Designate as Weak with Final score 1
Score the screened genes Promoters and assign Final scores
STOP

Figure 2.15: Flowchart for predicting strong, average and weak promoters of the predicted essential genes and their scoring.

The program for scoring the training set of promoters is specified in appendix 2.5(a) and the program for scoring the genes on basis of Promoter sequence is written in Perl and is specified in appendix 2.5(b). The PSSM matrix obtained from the PSSM generator program script is mentioned below (Figure 2.16).



```
A at Pos 1 to 6  0.0 0.84615386 0.07692308 0.46153846 0.53846157 0.07692308
T at Pos 1 to 6  0.9230769 0.07692308 0.6923077 0.15384616 0.15384616 0.9230769

G at Pos 1 to 6  0.0 0.07692308 0.15384616 0.30769232 0.07692308 0.0
C at Pos 1 to 6  0.07692308 0.0 0.07692308 0.07692308 0.23076923 0.0


TRANSITION MATRIX FOR COLUMN 1
   A    T    G    C
A 0.0  0.0  0.0  0.0
T 0.7692308  0.07692308  0.07692308  0.0
G 0.0  0.0  0.0  0.0
C 0.07692308  0.0  0.0  0.0


TRANSITION MATRIX FOR COLUMN 2
   A    T    G    C
A 0.0  0.61538464  0.15384616  0.07692308
T 0.07692308  0.0  0.0  0.0
G 0.0  0.07692308  0.0  0.0
C 0.0  0.0  0.0  0.0


TRANSITION MATRIX FOR COLUMN 3
   A    T    G    C
```

Figure 2.16: Output of the promoter score generator using self-designed PSSM matrix

An Example of Scoring Function:

TATAAT

Score = Occurrence Probability of T in column 1

+ Emission Probability of A from T in column 1

+ Occurrence Probability of A in column 2

+ Emission Probability of T from A in column 2.......till T from A in column 5

2.3.3 Scoring on basis of Shine-Dalgarno sequence

The Shine-Dalgarno sequence (+10, Ribosome Binding Site) “AGGAGGU” is the signal for initiation of protein biosynthesis in bacterial mRNA (Shine and Dalgarno, 1974). It is located 5' of the initiation codon AUG, and consists primarily, but not exclusively, of purines. The complementary sequence (ACCUCCU), rich in pyrimidines, is called the Anti-Shine-Dalgarno Sequence and is located at the 3' end of the 16S rRNA in the ribosome (http://www.ambion.com/techlib/append/rbs_requirements.html). Mutations in the Shine-Dalgarno sequence can reduce translation (Shine and Dalgarno, 1974). The 3'-terminal sequence of Escherichia coli 16S ribosomal RNA:Complementarity to nonsense triplets and ribosome binding sites. Proc.Nat.Acad.Sci.USA 71:1342-1346). This reduction is due to a reduced mRNA-ribosome pairing efficiency, as evidenced by the fact that complementary mutations in the Anti-Shine-Dalgarno Sequence can restore translation. This sequence usually locates 4-9 nucleotides 5' of the initiator AUG of many mRNAs. The procedure used to identify and scoring the drug targets on the basis of this Shine Dalgarno sequence is very much similar to that of scoring system used for promoter sequences.

The program for scoring the genes on basis of Shine Dalgarno sequence is written in Perl and is specified in appendix 2.6.

2.3.4 Scoring the drug targets on the basis of predicted drug binding motifs

The essential genes identified were translated into protein sequences and used for prediction of drug binding motifs within it. Functional motifs are the fundamental units of tertiary structure, defining their function on the whole and are often conserved in a protein family. The training set consisting of the drug binding motifs commonly present among the antibacterial and antibiotic drug target protein sequences were extracted from the Drug Bank (http://redpoll.pharmacy.ualberta.ca/drugbank/) and used to train the model.

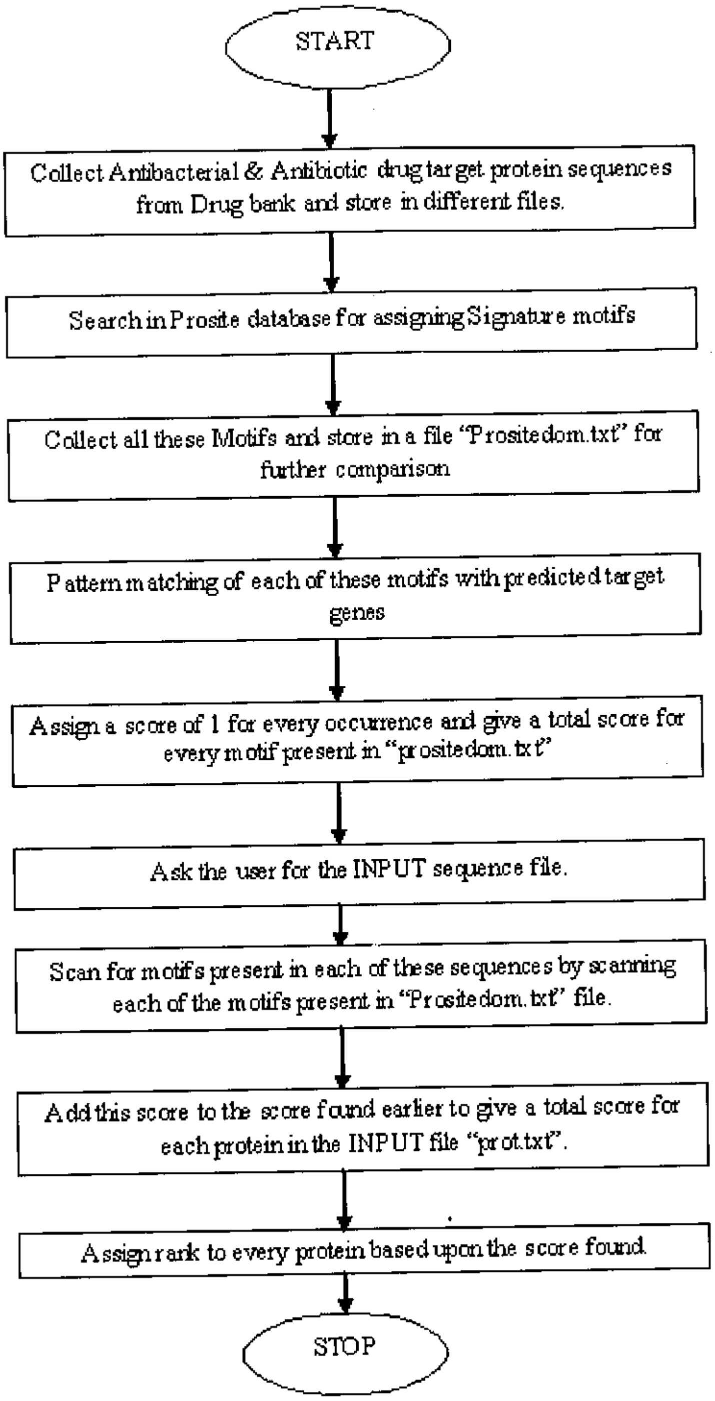
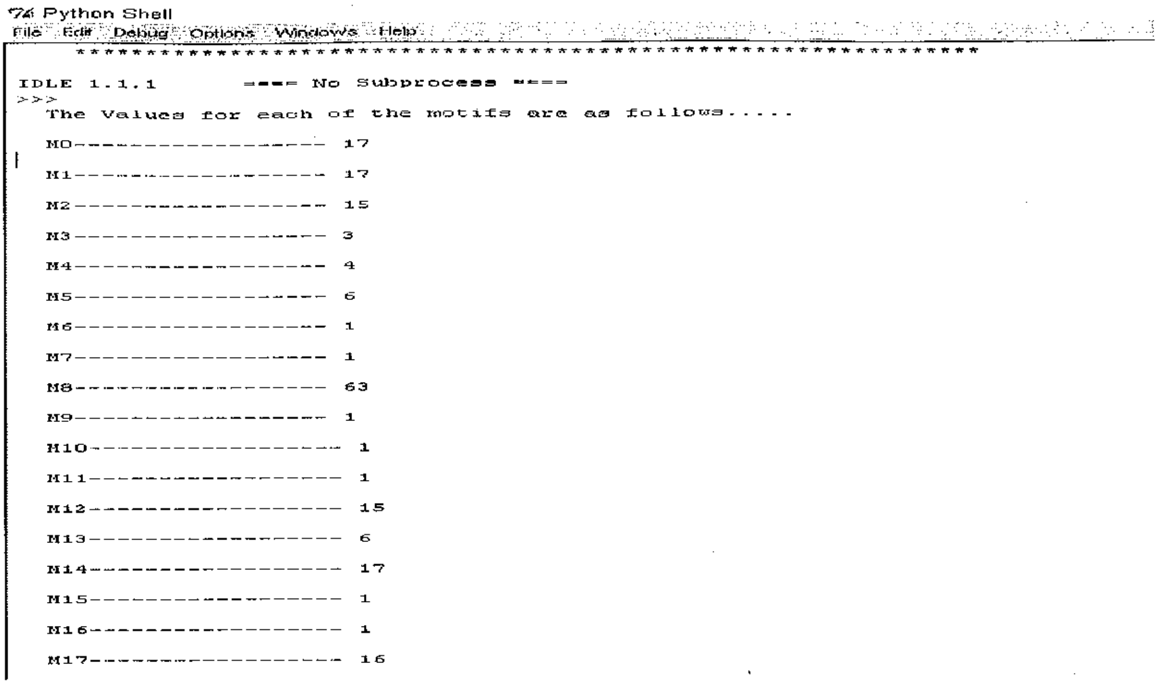

START
Collect Antibacterial & Antibiotic drug target protein sequences from Drug bank and store in different files.
Search in Prosite database for assigning Signature motifs
Collect all these Motifs and store in a file "Prositedom.txt" for further comparison
Pattern matching of each of these motifs with predicted target genes
Assign a score of 1 for every occurrence and give a total score for every motif present in "prositedom.txt"
Ask the user for the INPUT sequence file.
Scan for motifs present in each of these sequences by scanning each of the motifs present in "Prositedom.txt" file.
Add this score to the score found earlier to give a total score for each protein in the INPUT file "prot.txt".
Assign rank to every protein based upon the score found.
STOP

Figure 2.17: Flowchart for predicting drug binding motifs in the predicted essential target proteins and scoring.

Each drug binding motif is assigned a score on the basis of the number of occurrence among the drug target sequences. For scoring the drug binding motif present in the predicted drug targets, a program has been written in python code (Figure 2.14). The sore given the program is represented in Figure 2.18.

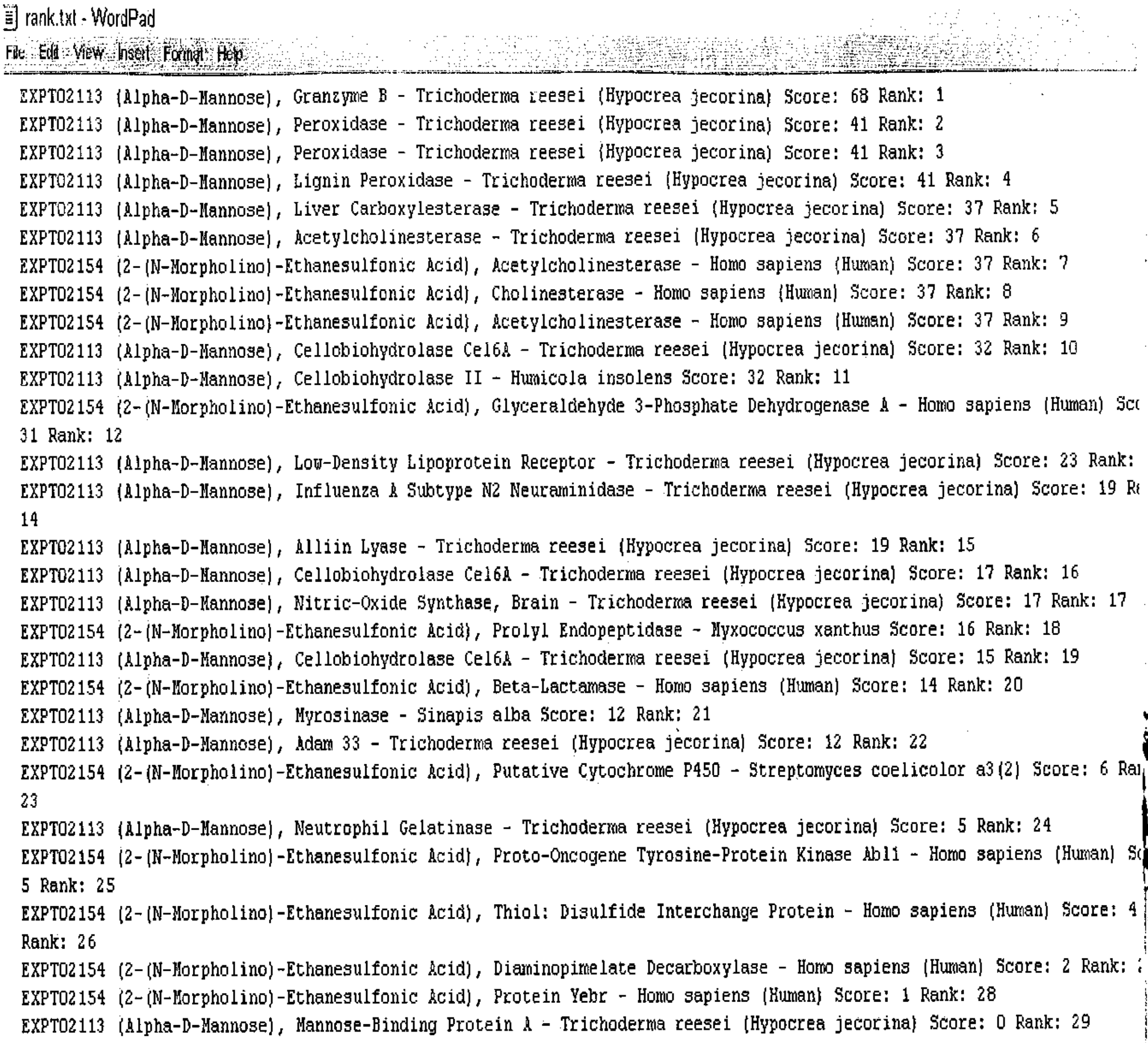


```
    ***************************************************************************

IDLE 1.1.1        ==== No Subprocess ====
>>>
  The Values for each of the motifs are as follows.....
  M0------------------- 17
  M1------------------- 17
  M2------------------- 15
  M3------------------- 3
  M4------------------- 4
  M5------------------- 6
  M6------------------- 1
  M7------------------- 1
  M8------------------- 63
  M9------------------- 1
  M10------------------- 1
  M11------------------- 1
  M12------------------- 15
  M13------------------- 6
  M14------------------- 17
  M15------------------- 1
  M16------------------- 1
  M17------------------- 16
```

Figure 2.18: Scores for each motif found: SAMPLE OUTPUT

The final scoring of the predicted and validated drug targets is based on the sum of all the score coming out of the entire four validation step as discussed above and shown in figure 2.19.



```
EXPTO2113 (Alpha-D-Mannose), Granzyme B - Trichoderma reesei (Hypocrea jecorina) Score: 68 Rank: 1
EXPTO2113 (Alpha-D-Mannose), Peroxidase - Trichoderma reesei (Hypocrea jecorina) Score: 41 Rank: 2
EXPTO2113 (Alpha-D-Mannose), Peroxidase - Trichoderma reesei (Hypocrea jecorina) Score: 41 Rank: 3
EXPTO2113 (Alpha-D-Mannose), Lignin Peroxidase - Trichoderma reesei (Hypocrea jecorina) Score: 41 Rank: 4
EXPTO2113 (Alpha-D-Mannose), Liver Carboxylesterase - Trichoderma reesei (Hypocrea jecorina) Score: 37 Rank: 5
EXPTO2113 (Alpha-D-Mannose), Acetylcholinesterase - Trichoderma reesei (Hypocrea jecorina) Score: 37 Rank: 6
EXPTO2154 (2-(N-Morpholino)-Ethanesulfonic Acid), Acetylcholinesterase - Homo sapiens (Human) Score: 37 Rank: 7
EXPTO2154 (2-(N-Morpholino)-Ethanesulfonic Acid), Cholinesterase - Homo sapiens (Human) Score: 37 Rank: 8
EXPTO2154 (2-(N-Morpholino)-Ethanesulfonic Acid), Acetylcholinesterase - Homo sapiens (Human) Score: 37 Rank: 9
EXPTO2113 (Alpha-D-Mannose), Cellobiohydrolase Cel6A - Trichoderma reesei (Hypocrea jecorina) Score: 32 Rank: 10
EXPTO2113 (Alpha-D-Mannose), Cellobiohydrolase II - Humicola insolens Score: 32 Rank: 11
EXPTO2154 (2-(N-Morpholino)-Ethanesulfonic Acid), Glyceraldehyde 3-Phosphate Dehydrogenase A - Homo sapiens (Human) Sc
31 Rank: 12
EXPTO2113 (Alpha-D-Mannose), Low-Density Lipoprotein Receptor - Trichoderma reesei (Hypocrea jecorina) Score: 23 Rank:
EXPTO2113 (Alpha-D-Mannose), Influenza A Subtype N2 Neuraminidase - Trichoderma reesei (Hypocrea jecorina) Score: 19 R
14
EXPTO2113 (Alpha-D-Mannose), Alliin Lyase - Trichoderma reesei (Hypocrea jecorina) Score: 19 Rank: 15
EXPTO2113 (Alpha-D-Mannose), Cellobiohydrolase Cel6A - Trichoderma reesei (Hypocrea jecorina) Score: 17 Rank: 16
EXPTO2113 (Alpha-D-Mannose), Nitric-Oxide Synthase, Brain - Trichoderma reesei (Hypocrea jecorina) Score: 17 Rank: 17
EXPTO2154 (2-(N-Morpholino)-Ethanesulfonic Acid), Prolyl Endopeptidase - Myxococcus xanthus Score: 16 Rank: 18
EXPTO2113 (Alpha-D-Mannose), Cellobiohydrolase Cel6A - Trichoderma reesei (Hypocrea jecorina) Score: 15 Rank: 19
EXPTO2154 (2-(N-Morpholino)-Ethanesulfonic Acid), Beta-Lactamase - Homo sapiens (Human) Score: 14 Rank: 20
EXPTO2113 (Alpha-D-Mannose), Myrosinase - Sinapis alba Score: 12 Rank: 21
EXPTO2113 (Alpha-D-Mannose), Adam 33 - Trichoderma reesei (Hypocrea jecorina) Score: 12 Rank: 22
EXPTO2154 (2-(N-Morpholino)-Ethanesulfonic Acid), Putative Cytochrome P450 - Streptomyces coelicolor a3(2) Score: 6 Ra
23
EXPTO2113 (Alpha-D-Mannose), Neutrophil Gelatinase - Trichoderma reesei (Hypocrea jecorina) Score: 5 Rank: 24
EXPTO2154 (2-(N-Morpholino)-Ethanesulfonic Acid), Proto-Oncogene Tyrosine-Protein Kinase Abl1 - Homo sapiens (Human) S
5 Rank: 25
EXPTO2154 (2-(N-Morpholino)-Ethanesulfonic Acid), Thiol: Disulfide Interchange Protein - Homo sapiens (Human) Score: 4
Rank: 26
EXPTO2154 (2-(N-Morpholino)-Ethanesulfonic Acid), Diaminopimelate Decarboxylase - Homo sapiens (Human) Score: 2 Rank:
EXPTO2154 (2-(N-Morpholino)-Ethanesulfonic Acid), Protein Yebr - Homo sapiens (Human) Score: 1 Rank: 28
EXPTO2113 (Alpha-D-Mannose), Mannose-Binding Protein A - Trichoderma reesei (Hypocrea jecorina) Score: 0 Rank: 29
```

Figure 2.19: Final output of ranking of proteins on basis of domains

CHAPTER 3

Results and Discussions

Although experimental and computational methods has been previously employed for the identification of probable drug targets by predicting only the essential genes in pathogen bacteria (Dutta et al., 2006; Sakharkar et al., 2004). These works are confined to selective bacterial species. Moreover, they simply compare the bacterial genome sequences with human genome only to discard those essential genes which are homologous to human being. However, comparison with the symbiotic bacterial genome sequences present within human body cannot be ruled out for predicting the potential therapeutic targets. Symbiotic bacteria such as E. coli, Lactobacillus etc. reside inside the human system and perform various important and helpful functions. There presence is indispensable for survival of human beings. Hence in order to preserve them from harmful side effects of a drug, we have chosen to eliminate those essential genes of pathogen bacteria which share close similarity with these symbiotic organisms. Also the available tools didn't use any validation techniques to evaluate the best therapeutic target in a bacterial species. This is most essential for drug discovery process as within a single genome it is not the single therapeutic target present. The developed automated tools take into consideration all these aspects and to our knowledge no such tool is available till now.

The tool "In Silico-iTTV" (In Silico Identification of Therapeutic Target and Validation) is software that finds potential drug targets in a given set of genes/proteins. The input submitted to the tool is a set of gene sequences or protein sequences in FASTA format. The tool works on two basic ideas for finding drug targets. The target genes should be essential to the concerned pathogenic bacteria, i.e. any disruption in the functioning of those genes will lead to bacterial death. All such essential genes can be potential drug targets but including those genes whose products have sequence similarities with any human protein and its symbiotic bacterial species may lead to drug reactions with the host and, thus, to toxic effects. Therefore, the tool excludes those essential genes having sequence similarities with the human and its symbiotic bacterial genes and considers only unique essential genes which present only in the pathogen bacteria. It uses the standalone BLAST to filter out the unique essential genes from the DEG after subtracting from the human and its symbiotic bacterial genomes. We identified 917 unique essential genes in bacterial species. All these 917 unique essential genes were classified into 12 valuable functional categories necessary for the bacteria to survive using ProtFun. These numbers are in good agreement with the findings of Jacobs et al., 2003, who reported 300-400 essential genes. Nonetheless, since gene disruption data are not available for all the genes in all the pathogens, this approach makes it possible to hazard a "first-order guess" for the probability that any untested gene is essential and may be a probable drug target which needs further validation.

The developed tool was tested on three different bacteria viz. Mycobacterium tuberculosis, Mycobacterium leprae and Bacillus anthracis. The files containing all the genes in a genome (.ffn file) and all the proteins encoded by a genome (.faa file) were downloaded from the NCBI ftp site (ftp://ftp.ncbi.nlm.nih.gov/genomes/Bacteria/). The genome sequences were used to predict initially the number of genes in the entire genome using Med 1 and the reliability of prediction was validated by comparing with the reported number of genes from literature. The accuracy of gene prediction was varies for different bacteria. In case of Bacillus anthracis the accuracy of gene prediction was highest (99.54 %) followed by Mycobacterium tuberculosis (97.25 %) and the least accuracy was in Mycobacterium leprae (88. 93 %). Overall, it revealed a good prediction and can be used for annotating newly sequenced bacterial genome. These predicted genes were then compared with the set of unique essential gene using BLASTX to predict the number of putative therapeutic targets using default parameters (E-value = e-10). Also the user can change the E-value cutoff. The results were tabulated and are presented in table below.

The tool predicted 355 essential genes and 140 potential drug target genes at default parameters (E-value = 1e-10, program = blastx) in Mycobacterium tuberculosis. In Mycobacterium leprae the number of essential genes and potential target genes at default parameters was 276 and 80 respectively. However, in Bacillus anthracis the predicted number of essential genes is 55 and potential target genes are 14 using same default parameters

For further validation the protein product or function of predicted target genes was determined and all the genes were categorized on the basis of their valuable functions like; metabolic pathways, transport phenomena, cell envelope, protein synthetic machinery etc. in which they are involved. The results are shown as percentage distribution of essential genes and potential target genes in Fig. 3.1, Fig. 3.2 and Fig. 3.3 respectively. The final ranking of potential target genes based on their preferences was done using four scoring schemes as mentioned in the methodology. The results are represented in Tables below. It is revealed that in bacteria Mycobacterium tuberculosis the potential target genes having the function Translation are ranked as best and are predicted to be best therapeutic target for drug design. In bacteria Mycobacterium leprae ranked 1 & 2 target genes having function Translation are predicted as the best therapeutic target, whereas in Bacillus anthracis the target genes with function Translation are ranked as 1 & 2 and are predicted to be good therapeutic targets for drug design.

As anticipated, many of the candidate genes predicted by our tool for drug target encode proteins for the basic survival mechanisms of the bacterium. The list of potential drug targets encoded in microbial genomes includes genes involved in translation, transcription, DNA replication, repair, outer-membrane proteins, permeases, enzymes of intermediary metabolism, host-interaction factors, and many more. For example, the fatty acid synthesis pathway appears attractive for drug discovery because a few known antibacterial compounds target enzymes of this pathway. Our approach identified 11 genes from this group. This is in accordance with Payne et al., 2001; Heath et al., 2002. Most of the predicted target genes belong to the DNA replication and repair pathway, which is one of the most essential pathways. The tool predicted 7 genes from ribosomal protein biosynthesis pathway and 2 genes from translation in general. All these 9 genes are involved in protein translation metabolism hence are necessary for the bacterial survival. 6 predicted genes were involved in peptidoglycan synthesis, which is a major component of the bacterial cell wall. Any disruption of these genes will lead to disruption in the cell wall biosynthesis resulting in bacterial cell death. It also predicts 3 genes for isoprenoid biosynthesis, the non-mevalonate pathway of isoprenoid biosynthesis has already been identified as potential pathway for antimicrobial drug development [Dubey, 2002].

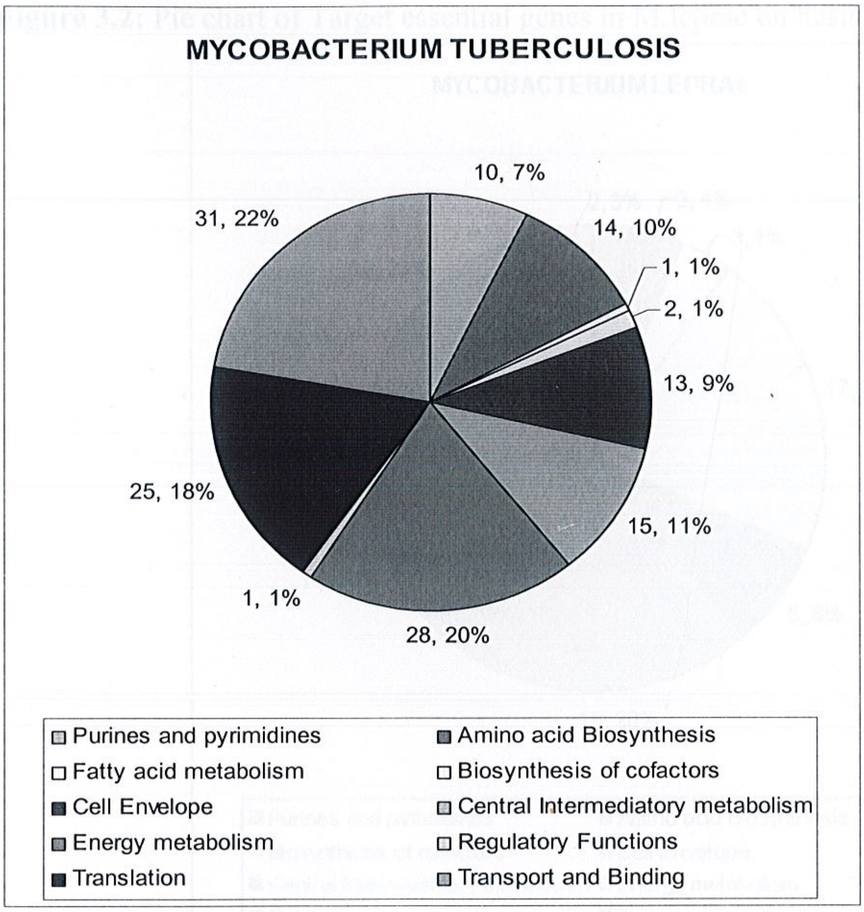
Previous comparative analyses of complete genomes revealed that most of the pathogens have drastically diminished biosynthetic capabilities compared to their free-living relatives [Fraser et al., 1995; Lewis, 1999]. Instead, these organisms depend on their hosts to provide essential nutrients such as amino acids, nucleotides, and vitamins. Thus, transport systems for these nutrients are generally well-conserved and easily identifiable [Clayton et al., 1997]. Analysis of metabolic pathways allows one to predict which substrates cannot be produced inside the cells and therefore need to be transported. This renders bacterial transport proteins that do not have human homologs as attractive drug targets [Smith et al., 2004]. Our approach identified the category of "transport of small molecules" as the main one for drug target identification (9%). Translation, post-translational modification and degradation were next on the list. Our approach did not identify any genes involved in chemotaxis as essential, in agreement with previous experimental data [Jacobs et al., 2003]. The complete list of essential P. aeruginosa genes generated by this methodology with hyperlinks to DEG is available online as supplementary data. This example thus illustrates the use of this approach to identify essential genes in pathogens that may be considered as drug targets with more confidence.

Our Results are displayed below with corresponding tables and pie-charts.

Mycobacterium tuberculosis

	DTITK (Med v1.0)	Literature	Accuracy
Total no. of genes	3819	3927	97.25 %
Essential genes	355		
Targets predicted	140		

Figure 3.1: Pie chart of Target essential genes in M.tuberculosis on basis of function.

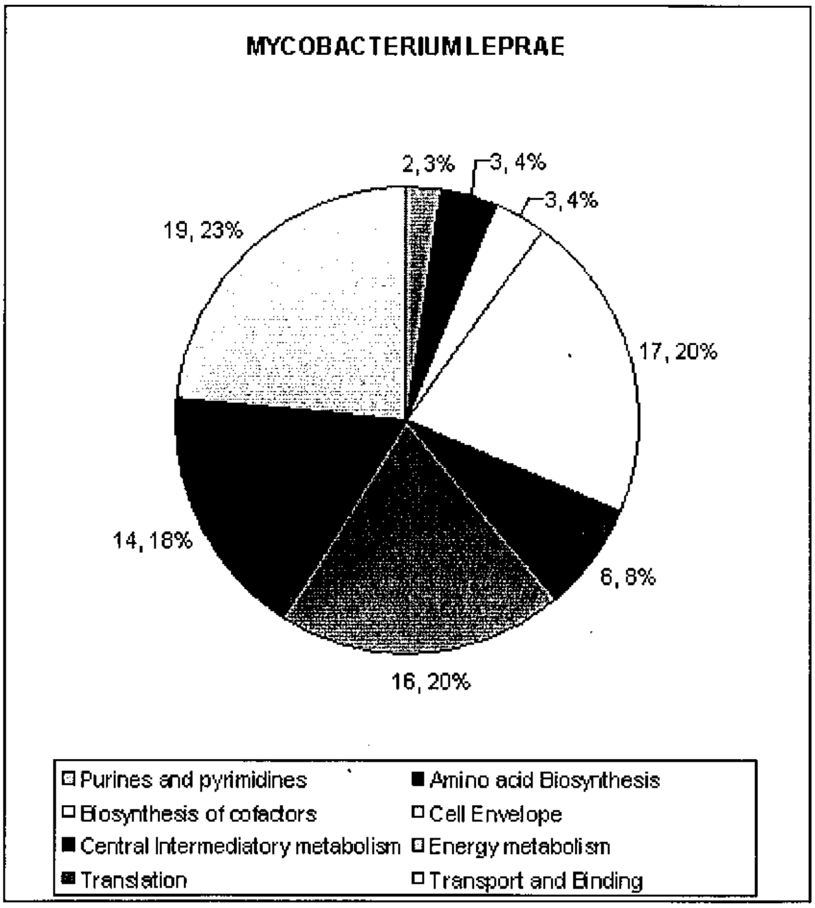


Function	Count	Score Range	Ranks
Purines and Pyrimidines	10	4 to 10	10 to 13
Amino acid Biosynthesis	14	4 to 10	10 to 13
Fatty acid metabolism	1	2	14
Biosynthesis of cofactors	2	6 to 8	11,12
Cell Envelope	13	16 to 20	5 to7
Central Intermediatory metabolism	15	10 to 14	8 to10
Energy metabolism	28	6 to 10	10 to 12
Regulatory Functions	1	6	12
Translation	25	23 to 29	1 to 4
Transport and Binding	31	6 to 12	9 to 12

Mycobacterium leprae

	DTITK (Med v1.0)	Literature	Accuracy
Total no. of genes	2403	2702	88.93 %
Essential genes	276		
Targets predicted	80		

Figure 3.2: Pie chart of Target essential genes in M.leprae on basis of function.

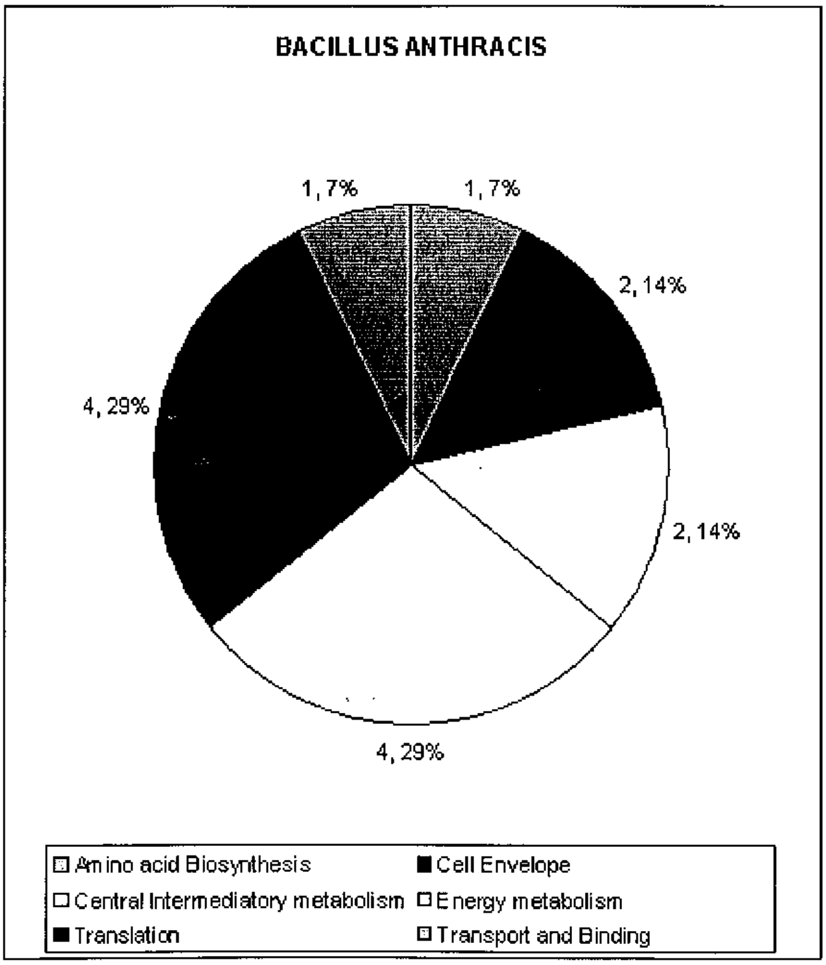


Function	Count	Score Range	Ranks
Purines and Pyrimidines	2	6 to 8	10 to 11
Amino acid Biosynthesis	3	6 to 10	9 to 11
Biosynthesis of cofactors	3	6 to 8	10 to11
Cell Envelope	17	16 to 20	4 to 6
Central Intermediatory metabolism	6	10 to 14	7 to 9
Energy metabolism	16	6 to 10	9 to 11
Translation	14	23 to 27	1 to 3
Transport and Binding	19	6 to 12	8 to 11

Bacillus anthracis

	DTITK (Med v1.0)	Literature	Accuracy
Total no. of genes	5263	5287	99.54%
Essential genes	55		
Targets predicted	14		

Figure 3.3: Pie chart of Target essential genes in B.anthracis on basis of function.



Function	Count	Score Range	Ranks
Amino acid Biosynthesis	1	6	8
Cell Envelope	2	16 to 18	3 to 4
Central Intermediatory metabolism	2	10 to 12	5 to 6
Energy metabolism	4	6 to 10	6 to 8
Translation	4	23 to 25	1 to 2
Transport and Binding	1	8	7

CONCLUSION

For the first time, the availability of complete genome sequences of many bacterial species is facilitating many computational approaches. The complete definition of all gene products by gene identification tools exemplified here is just the first step. The data presented here demonstrates that stepwise prioritization of genome open reading frames using simple biological criteria can be an effective way of rapidly reducing the number of genes of interest to an experimentally manageable number. This process is an efficient way for enriching potential target genes, and for identifying those that are critical for normal cell function. The generation of a comprehensive essential gene list will allow an accelerated genetic dissection of traits such as metabolic flexibility and inherent drug resistance that render *P. aeruginosa* such a tenacious pathogen. Such a strategy will enable us to locate critical pathways and steps in pathogenesis; to target these steps by designing new drugs; and to inhibit the infectious agent of interest with new antimicrobial agents.

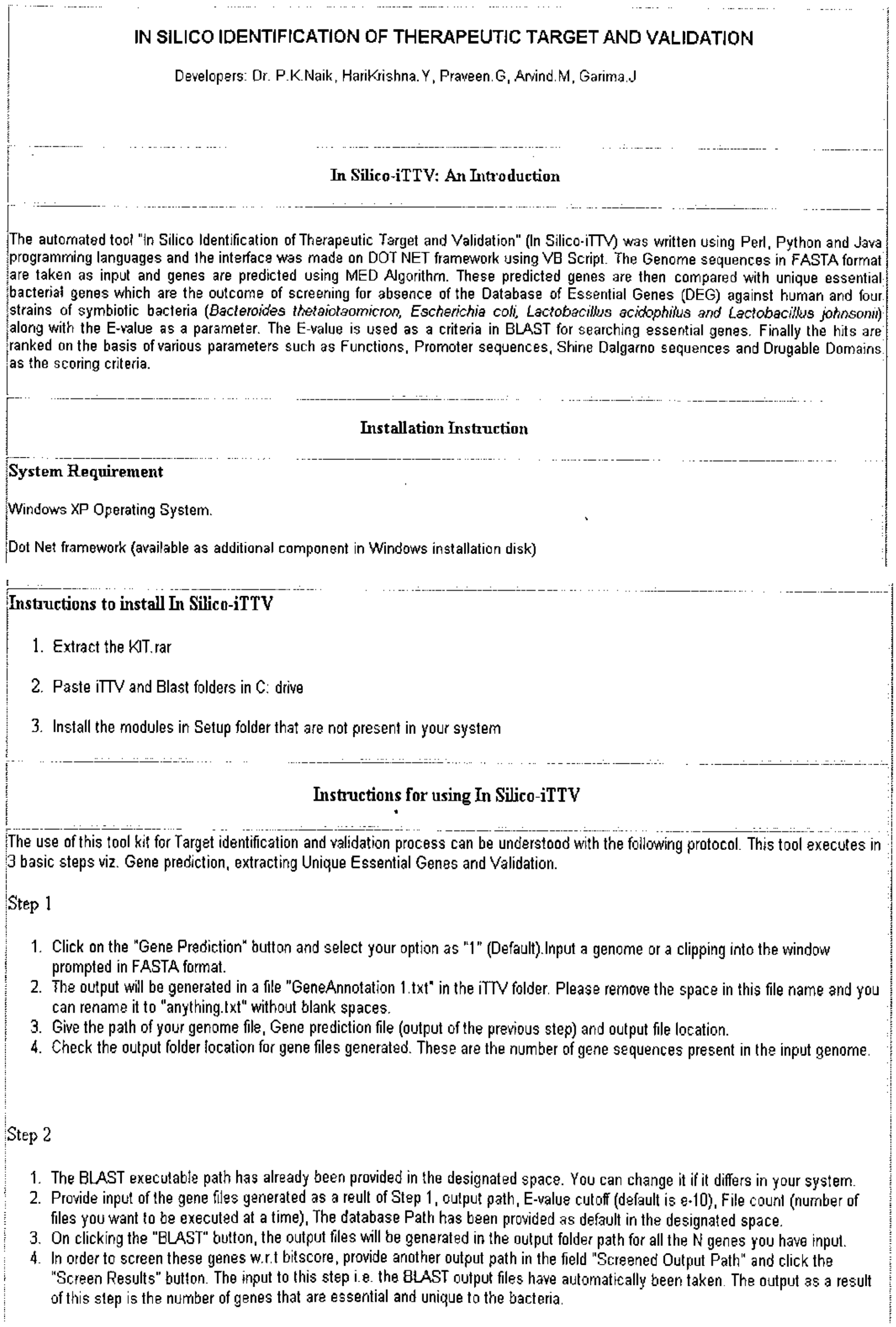
Availability of the tool

The automated tool "In Silico Identification of Therapeutic Target and Validation" (In Silico-iTTV) was written using Perl, Python and Java programming languages and the interface was made on DOT NET framework using VB Script. The Genome sequences in FASTA format are taken as input and genes are predicted using MED Algorithm. The standalone package is uploaded on the University Web server "In Silico-iTTV" and available at http://www.juit.ac.in/iTTV.html (Figure 3.4).

Jaypee University of Information Technology

(Approved by UGC Under Section 2f of UGC Act)

Department of Biotechnology and Bioinformatics



IN SILICO IDENTIFICATION OF THERAPEUTIC TARGET AND VALIDATION

Developers: Dr. P.K.Naik, HariKrishna.Y, Praveen.G, Arvind.M, Garima.J

In Silico-iTTV: An Introduction

The automated tool "In Silico Identification of Therapeutic Target and Validation" (In Silico-iTTV) was written using Perl, Python and Java programming languages and the interface was made on DOT NET framework using VB Script. The Genome sequences in FASTA format are taken as input and genes are predicted using MED Algorithm. These predicted genes are then compared with unique essential bacterial genes which are the outcome of screening for absence of the Database of Essential Genes (DEG) against human and four strains of symbiotic bacteria (*Bacteroides thetaiotaomicron, Escherichia coli, Lactobacillus acidophilus and Lactobacillus johnsonii*) along with the E-value as a parameter. The E-value is used as a criteria in BLAST for searching essential genes. Finally the hits are ranked on the basis of various parameters such as Functions, Promoter sequences, Shine Dalgarno sequences and Drugable Domains as the scoring criteria.

Installation Instruction

System Requirement

Windows XP Operating System.

Dot Net framework (available as additional component in Windows installation disk)

Instructions to install In Silico-iTTV

1. Extract the KIT.rar
2. Paste iTTV and Blast folders in C: drive
3. Install the modules in Setup folder that are not present in your system

Instructions for using In Silico-iTTV

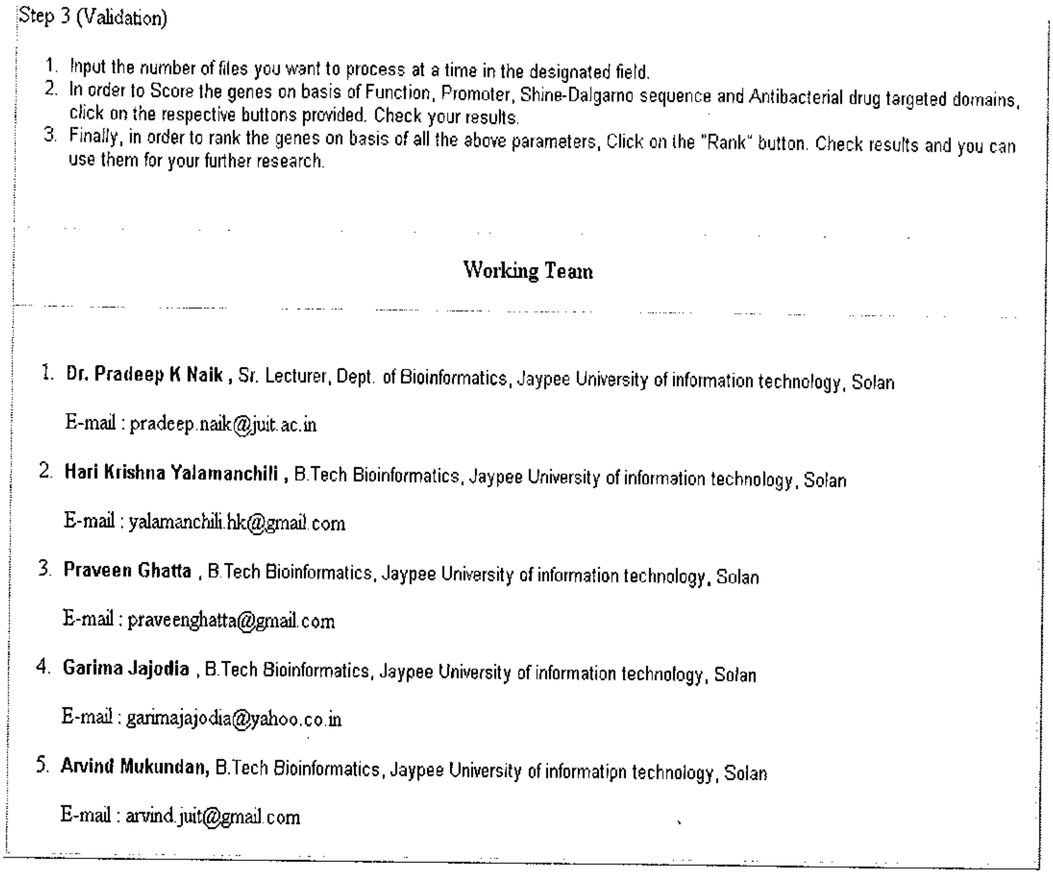
The use of this tool kit for Target identification and validation process can be understood with the following protocol. This tool executes in 3 basic steps viz. Gene prediction, extracting Unique Essential Genes and Validation.

Step 1

1. Click on the "Gene Prediction" button and select your option as "1" (Default).Input a genome or a clipping into the window prompted in FASTA format.
2. The output will be generated in a file "GeneAnnotation 1.txt" in the iTTV folder. Please remove the space in this file name and you can rename it to "anything.txt" without blank spaces.
3. Give the path of your genome file, Gene prediction file (output of the previous step) and output file location.
4. Check the output folder location for gene files generated. These are the number of gene sequences present in the input genome.

Step 2

1. The BLAST executable path has already been provided in the designated space. You can change it if it differs in your system.
2. Provide input of the gene files generated as a reult of Step 1, output path, E-value cutoff (default is e-10), File count (number of files you want to be executed at a time), The database Path has been provided as default in the designated space.
3. On clicking the "BLAST" button, the output files will be generated in the output folder path for all the N genes you have input.
4. In order to screen these genes w.r.t bitscore, provide another output path in the field "Screened Output Path" and click the "Screen Results" button. The input to this step i.e. the BLAST output files have automatically been taken. The output as a result of this step is the number of genes that are essential and unique to the bacteria.



Step 3 (Validation)

1. Input the number of files you want to process at a time in the designated field.
2. In order to Score the genes on basis of Function, Promoter, Shine-Dalgarno sequence and Antibacterial drug targeted domains, click on the respective buttons provided. Check your results.
3. Finally, in order to rank the genes on basis of all the above parameters, Click on the "Rank" button. Check results and you can use them for your further research.

Working Team

1. **Dr. Pradeep K Naik** , Sr. Lecturer, Dept. of Bioinformatics, Jaypee University of information technology, Solan

   E-mail : pradeep.naik@juit.ac.in

2. **Hari Krishna Yalamanchili** , B.Tech Bioinformatics, Jaypee University of information technology, Solan

   E-mail : yalamanchili.hk@gmail.com

3. **Praveen Ghatta** , B.Tech Bioinformatics, Jaypee University of information technology, Solan

   E-mail : praveenghatta@gmail.com

4. **Garima Jajodia** , B.Tech Bioinformatics, Jaypee University of information technology, Solan

   E-mail : garimajajodia@yahoo.co.in

5. **Arvind Mukundan,** B.Tech Bioinformatics, Jaypee University of informatipn technology, Solan

   E-mail : arvind.juit@gmail.com

Figure 3.4: Snapshot of our Webpage uploaded on university web server

REFERENCES

J. Drews. Genomic sciences and the medicine of tomorrow. *Nature Biotechnology*, 14: 1516-
1518, 1996.

B.P. Zambrowicz, and A.T. Sands. Knockouts model the 100 best-selling drugs – will they model the next 100? *Nat. Rev. Drug Discov.*, 2: 38-51, 2003.

Wheeler, D. L., Church, D. M., Edgar, R., Federhen, S., Helmberg, W., Madden, T. L., Pontius, J. U., Schuler, D. G., Schriml, L. M., Sequeira, E., Suzek, T. O., Tatusova, T. A. and Wagner, L. (2004). Database resources of the National Center for Biotechnology Information: update. Nucleic Acids Res. 32, D35-D40.

Miesel, L., Greene, J. and Black, T. A. (2003). Genetic strategies for antibacterial drug discovery. Nature Rev. Genet. 4, 442-456.

International Human Genome Sequencing Consortium. Finishing the euchromatic sequence of the human genome. *Nature*, 431:931-945, 2004.

J. Drews. Drug discovery: a historical perspective. *Science*, 287: 1960-1964, 2000.

G.C. Terstappen, A. Reggiani, G.C. Terstappen, and A. Reggiani. In *silico* research in drug discovery. *Trends in Pharmacological Sciences*, 22: 23-26, 2001.

Howbrook DN, van der Valk AM, O'Shaughnessy MC, Sarker DK, Baker SC, Lloyd AW: Developments in microarray technologies. Drug Discov Today 2003, 8:642-651.

Sem DS, Yu L, Coutts SM, Jack R: Object-oriented approach to drug design enabled by NMR SOLVE: first real-time structural tool for characterizing protein-ligand interactions. J Cell Biochem Suppl 2001, 37:99-105

Lander, E. S., Linton, L. M., Birren, B., Nusbaum, C., Zody, M. C., Baldwin, J., Devon, K., Dewar, K., Doyle, M., FitzHugh, W., *et al.* (2001). Initial sequencing and analysis of the human genome. Nature 409, 860-921.

Venter, J. C., Adams, M. D., Myers, E. W., Li, P. W., Mural, R. J., Sutton, G. G., Smith, H. O., Yandell, M., Evans, C. A., Holt, R. A., *et al.* (2001). The sequence of the human genome. Science 291, 1304-1351.

Galperin, M. Y. and Koonin, E. V. (1999). Searching for drug targets in microbial genomes. Curr. Opin. Biotechnol. 10, 571-578.

Mushegian, A. R. and Koonin, E. V. (1996). A minimal gene set for cellular life derived by comparison of complete bacterial genomes. Proc. Natl. Acad. Sci. USA 93, 10268-10273.

Bruccoleri, R. E., Dougherty, T. J. and Davison, D. B. (1998). Concordance analysis of microbial genomes. Nucleic Acids Res. 26, 4482-4486.

Itaya, M. (1995). An estimation of the minimum genome size required for life. FEBS Lett. 362, 257-260.

Tatusov, R. L., Koonin, E. V. and Lipman, D. J. (1997). A genomic perspective of protein families. Science 278, 631-637.

Kobayashi, K., *et al.* (2003). Essential *Bacillus subtilis* genes. Proc. Natl. Acad. Sci. USA 100, 4678-4683.

Koonin, E. V., Tatusov, R. L. and Galperin, M. Y. (1998). Beyond complete genomes - from sequence to structure and function. Curr. Opin. Struc. Biol. 8, 355-363.

Sakharkar, K. R., Sakharkar, M. K. and Chow, V. T. K., (2004). A novel genomics approach for the identification of drug targets in pathogens, with special reference to *Pseudomonas aeruginosa*. In Silico Biol. 4, 0028.

Zhang, R., Ou, H. Y. and Zhang, C. T (2004). DEG: A database of essential genes. Nucleic Acids Res. 32, D271-D272

Singh, N.K., Selvam, S.M., and Chakravarthy, P. 2006. T-iDT: Tool for identification of drug target in bacteria and validation by *Mycobacterium tuberculosis*. In Silico Biology. 6, 45-53.

D.B. Allison. Statistical methods for microarray research for drug target identification 2002. *Proceedings of the American Statistical Association*, Biopharmaceutical Section [CD-ROM].
Alexandria, (VA): American Statistical Association, 2002.

J. Augen. The evolving role of information technology in the drug discovery process. *Drug Discovery Today*, 7: 315-323, 2002.

T. Head-Gordon, and J.C. Wooley. Computational challenges in structural and functional genomics. *IBM Systems J.*, 40: 265, 2001.

Jacobs, M. A., Alwood, A., Thaipisuttikul, I., Spencer, D., Haugen, E., Ernst, S., Will, O., Kaul, R., Raymond, C., Levy, R., Chun-Rong, L., Guenthner, D., Bovee, D., Olson, M. V.

and Manoil, C. (2003). Comprehensive transposon mutant library of *Pseudomonas aeruginosa*. Proc. Natl. Acad. Sci. USA 100, 14339-14344.

PUBLICATION

Dr. P.K. Naik, Hari Krishna.Y, Arvind. M , Praveen.G , Garima. J. **In Silico Identification of Therapeutic Targets and Validation.** Communicate to Protein Structure, function and Bioinformatics.

APPENDIX

2.1 Perl Program used to download essential bacterial genes in Html files from DEG:

```
use LWP::Simple;
$id='DEG';
$counter=10010001;
foreach(1..243)
{
push(@array,$counter);
$counter++;
}
$count=0;
foreach(@array)
{
$_=$id.$_;
open(GG,">>Seq.html");
$count++;
print $count."). Fetching ".$_."....\n";
my $esearch_result = get("http://tubic.tju.edu.cn/deg/information.php?ac=".$_);
print GG $esearch_result."\n";
print "Fetched....\n";
close GG;
}
print "Finished...";
<stdin>;
```

2.2 Perl program for extracting genes from the input genome using index file.

```
@conn;
$n=0;
foreach(@ARGV)
{
$conn[$n]=$_;
}
$f=$conn[0];
$ff=$conn[1];
$l=@conn;

@spl = split('',$f);
$l=@spl;
$i = 1;

$search="C:";
$bo=rindex $f,$search;
#print $bo;
$cmd3=substr($f,$bo,($l-$bo));

$temp=substr($f,0,($bo));
$bo=rindex $temp,$search;

@sp= split('',$temp);
```

```
$l= @sp;
$cmd2=substr($temp,$bo,($l-$bo));

$cmd1=substr($f,0,$bo);
open(HH,$cmd1);
@content=<HH>;
close HH;
$seq='';
chomp(@content);
foreach(@content)
{
if($_=~ /^>.*./)
{
next;
}
else
{
$seq=$seq.$_;
}
}
open(GOOD,$cmd2);
chomp(@con=<GOOD>);
close GOOD;
@forwardgenes=();
@backgenes=();
$genecount=0;
foreach(@con)
{
@array=();
@array=split(' ',$_);
if($array[0]=~ /^[0-9]/)
{
    if($array[1] eq '+')
    {
    my $frag=substr($seq,($array[2]-1),$array[4]);
    $genecount++;
    open(HAND,'>'.$cmd3.$genecount.'.txt');
    print HAND ">gene".$genecount."\n$frag";
    close HAND;

    }
    if($array[1] eq '-')
    {
    my $frag=substr($seq,($array[2]-1),$array[4]);
    $frag=~ tr/ATGC/TACG/;
    $genecount++;
    open(HAND,'>'.$cmd3.$genecount.'.txt');
    print HAND ">gene".$genecount."\n$frag";
    close HAND;
    } } }
```

2.3 Perl code for screening BLAST outputs:

```
@conn;
$n=0;
foreach(@ARGV)
{
$conn[$n]=$_;
}
$f=$conn[0];
$ff=$conn[1];
$l=@conn;

@spl = split(",$f);
$l=@spl;
$i = 1;

$search="C:";
$search2 = "-";
$bi=rindex $f,$search2;
$cmd5 = substr ($f,$bi+1,($l-$bi));
print $cmd5;

$temp2 = substr($f,0,$bi);
print "\n";
@str2 = split(",$temp2);
$l2 = @str2;

$bi = rindex $temp2,$search2;
$cmd4 = substr($temp2, $bi+1, ($l2-$bi));
print $cmd4;
$f2 = substr($temp2,0,$bi);
@str3 = split(",$f2);
$l=@str3;
$bo = rindex $f2,$search;
$cmd3=substr($f2,$bo,($l-$bo));
print "\n";
print $cmd3;

$temp=substr($f2,0,($bo));
$bo=rindex $temp,$search;

@sp= split(",$temp);
$l= @sp;
$cmd2=substr($temp,$bo,($l-$bo));
print "\n";
print $cmd2;

$cmd1=substr($f2,0,$bo);
print "\n";
print $cmd1;

$m=1;
$lcount=0;
$status=1;
$hp=$cmd4-1;
$tempc=0;
for($m=$cmd4;$m<=$cmd5;$m++)
```

```
{
open(GOOD,$cmd1.$m.".out");
chomp(@con=<GOOD>);
close GOOD;
foreach(@con)
{
$lcount++;
@array=();
@array=split(' ',$_);
$al=scalar(@array);

if(($lcount>=22)&& ($array[0]=~/^ref.*./))
{
if($array[$al-2]>=100)
{
$status=0;
}
}
}
if($status==1)
{
@gene=();
open(HANDLE,$cmd2.$m.".txt");
@gene=<HANDLE>;
close HANDLE;
$hp++;
open(HH,'>'.$cmd3.$hp.'.txt');
print HH "@gene";
close HH;
}
$status=1;
}
```

2.4 The program for scoring the genes on basis of functions

```
$fcount=$ARGV[0];
$status=0;
open(HARII,'>C:\DTITK\Protein_Function.txt');
for($i=1;$i<=$fcount;$i++)
{
open(GOOD,'C:\DTITK\BO\BO'.$i.'.out');
chomp(@con=<GOOD>);
close GOOD;
foreach(@con)
{
$lcount++;
@array=();
@array=split(' ',$_);
$len=@array;
if($array[0]=~ /^Pro.*./)
{
if($array[$len-2]>=100)
{
@qu=split(' ',$con[8]);
print HARII ">".$qu[1]." ";
$searchitem=$array[0];
$status=1;
}
}
#if($array[0]=~ /Query=/)
#{
#$temp=$array[1];
#print HARII $temp;
#print HARII " ";
#}
}
if($status==1)
{
open(HANDLE,'C:\DTITK\917_ScoredProteins_Functions.txt');
chomp(@stor=<HANDLE>);
foreach(@stor)
{
@arr=();
@arr=split(' ',$_);
if($arr[0] eq ">".$searchitem)
{
print HARII $arr[1]." ";
print  HARII "score: ";
print HARII $arr[2];
print HARII "\n";
break;
}
}
}
$status=0;
}
```

2.5 (a) Program in JAVA to score sample promoters

```
import java.io.*;

public class Demo1
{
public static void main(String arg[])throws IOException
{
String s= new String();
FileInputStream f = new FileInputStream("1.txt");
int i=0,j=0;
float c[][] = new float[100][6];
float pb[][] = new float[100][6];
char promoter_10[][]= new char [100][100];
while(true)
{
        int a = f.read();
        if(a == -1)
                break;
                char temp = (char)a;
                if((temp == 'A')||(temp == 'T')||(temp == 'G')||(temp == 'C'))
                {
                  promoter_10[i][j] = temp;
```

```
                        j++;
                        if(j==6)
                        {
                         i++;
                         j=0;
                        }

                      }
 }
 /****/
 for(int l=0;l<i;l++)
 {
 for(int k=0;k<6;k++)
 {
 if(promoter_10[l][k] == 'A')
 {
 c[0][k]++;
 }
 if(promoter_10[l][k] == 'T')
 {
 c[1][k]++;
 }
 if(promoter_10[l][k] == 'G')
 {
 c[2][k]++;
 }
 if(promoter_10[l][k] == 'C')
 {
 c[3][k]++;
 }
 }
 }
 /****/
 for(int m=0;m<4;m++)
 {
 for(int n=0;n<6;n++)
 {
 pb[m][n] = (float)(c[m][n])/i;
 }
 }

 /****/
 float tc[][][]= new float [6][4][4];
 for(int q=0;q<i;q++)
 {
 for(int r=0;r<6;r++)
 {
 if(promoter_10[q][r] == 'A')
 {
 if(promoter_10[q][(r+1)] == 'A')
 {
 tc[r][0][0]++;
 }
 if(promoter_10[q][(r+1)] == 'T')
 {
 tc[r][0][1]++;
 }
```

```
if(promoter_10[q][(r+1)] == 'G')
{
tc[r][0][2]++;
}
if(promoter_10[q][(r+1)] == 'C')
{
tc[r][0][3]++;
}
}
if(promoter_10[q][r] == 'T')
{
if(promoter_10[q][(r+1)] == 'A')
{
tc[r][1][0]++;
}
if(promoter_10[q][(r+1)] == 'T')
{
tc[r][1][1]++;
}
if(promoter_10[q][(r+1)] == 'G')
{
tc[r][1][2]++;
}
if(promoter_10[q][(r+1)] == 'C')
{
tc[r][1][3]++;
}
}
if(promoter_10[q][r] == 'G')
{
if(promoter_10[q][(r+1)] == 'A')
{
tc[r][2][0]++;
}
if(promoter_10[q][(r+1)] == 'T')
{
tc[r][2][1]++;
}
if(promoter_10[q][(r+1)] == 'G')
{
tc[r][2][2]++;
}
if(promoter_10[q][(r+1)] == 'C')
{
tc[r][2][3]++;
}
}
if(promoter_10[q][r] == 'C')
{
if(promoter_10[q][(r+1)] == 'A')
{
tc[r][3][0]++;
}
if(promoter_10[q][(r+1)] == 'T')
{
tc[r][3][1]++;
}
```

```
if(promoter_10[q][(r+1)] == 'G')
{
tc[r][3][2]++;
}
if(promoter_10[q][(r+1)] == 'C')
{
tc[r][3][3]++;
}
}
}
}
}
/****/

System.out.println("\n");

for(int o=0;o<4;o++)
{
if(o==0)
System.out.print("A at Pos 1 to 6  ");
if(o==1)
System.out.print("T at Pos 1 to 6  ");
if(o==2)
System.out.print("G at Pos 1 to 6  ");
if(o==3)
System.out.print("C at Pos 1 to 6  ");
for(int p=0;p<6;p++)
{
System.out.print(pb[o][p] + " ");
}
System.out.println("\n");
}
/****/
System.out.println("\n");
for(int x=0;x<5;x++)
{
System.out.println("TRANSITION MATRIX FOR COLUMN " + (x+1));
System.out.println("\n"+"   A   T   G   C "+"\n");
for(int y=0;y<4;y++)
{
if(y==0)
System.out.print("A");
if(y==1)
System.out.print("T");
if(y==2)
System.out.print("G");
if(y==3)
System.out.print("C");
for(int z=0;z<4;z++)
{
System.out.print(" "+ (tc[x][y][z]/i) +" ");
}
System.out.println("\n");
}
System.out.println("\n\n");
}
/****/
char r[]= {'T','A','T','A','A','T',' '};
```

```
double d[]= new double[100];
double score=0;
for (int g=0;g<i;g++)
{
for(int h=0;h<6;h++)
{
if(promoter_10[g][h]=='A')
{
d[g]=d[g]+pb[0][h];
if(promoter_10[g][(h+1)]=='A')
{
d[g]=d[g]+(tc[h][0][0]/i);
}
if(promoter_10[g][(h+1)]=='T')
{
d[g]=d[g]+(tc[h][0][1]/i);
}
if(promoter_10[g][(h+1)]=='G')
{
d[g]=d[g]+(tc[h][0][2]/i);
}
if(promoter_10[g][(h+1)]=='C')
{
d[g]=d[g]+(tc[h][0][3]/i);
}
}
if(promoter_10[g][h]=='T')
{
d[g]=d[g]+pb[1][h];
if(promoter_10[g][(h+1)]=='A')
{
d[g]=d[g]+(tc[h][1][0]/i);
}
if(promoter_10[g][(h+1)]=='T')
{
d[g]=d[g]+(tc[h][1][1]/i);
}
if(promoter_10[g][(h+1)]=='G')
{
d[g]=d[g]+(tc[h][1][2]/i);
}
if(promoter_10[g][(h+1)]=='C')
{
d[g]=d[g]+(tc[h][1][3]/i);
}
}
if(promoter_10[g][h]=='G')
{
d[g]=d[g]+pb[2][h];
if(promoter_10[g][(h+1)]=='A')
{
d[g]=d[g]+(tc[h][2][0]/i);
}
if(promoter_10[g][(h+1)]=='T')
{
d[g]=d[g]+(tc[h][2][1]/i);
}
```

```
if(promoter_10[g][(h+1)]=='G')
{
d[g]=d[g]+(tc[h][2][2]/i);
}
if(promoter_10[g][(h+1)]=='C')
{
d[g]=d[g]+(tc[h][2][3]/i);
}
}
if(promoter_10[g][h]=='C')
{
d[g]=d[g] + pb[3][h];
if(promoter_10[g][(h+1)]=='A')
{
d[g]=d[g]+(tc[h][3][0]/i);
}
if(promoter_10[g][(h+1)]=='T')
{
d[g]=d[g]+(tc[h][3][1]/i);
}
if(promoter_10[g][(h+1)]=='G')
{
d[g]=d[g]+(tc[h][3][2]/i);
}
if(promoter_10[g][(h+1)]=='C')
{
d[g]=d[g]+(tc[h][3][3]/i);
}
}
}
}
for(int fe=0;fe<i;fe++)
{
System.out.print("Score for ");
for(int e=0;e<6;e++)
{
System.out.print(promoter_10[fe][e]);
}
System.out.print(" is "+d[fe]);
System.out.println();
}

double max;
double min;
double avg=0;
max=d[0];
min=d[0];
for(int aa=0;aa<i;aa++)
{
        if(d[aa]< d[(aa+1)])
        {
        max=d[(aa+1)];
        }
        if((d[aa] < d[(aa+1)]))
        {
        if(d[(aa+1)]==0)
        {
            min=d[aa];
            }
            min=d[aa];
    }

avg=avg+d[aa];
}

System.out.println();
System.out.println("********");
System.out.println("MAXIMUM SCORE IS : "+ max);
System.out.println("MINIMUM SCORE IS : "+ min);
System.out.println("AVERAGE SCORE IS : "+ (avg/i));
/****/
}
}
```

2.5 (b) The program for scoring the genes on basis of Promoter sequence

```
$fcount=$ARGV[0];
$cmd = $ARGV[1];
open(HARI,'>c:\DTITK\Promoter_Scores.txt');
%pb=
(
'1A'=>0.0,
'2A'=>0.84615386,
'3A'=>0.07692308,
'4A'=>0.46153846,
'5A'=>0.53846157,
'6A'=>0.07692308,
'1T'=>0.9230769,
'2T'=>0.07692308,
'3T'=>0.6923077,
'4T'=>0.15384616,
'5T'=>0.15384616,
'6T'=>0.9230769,
'1C'=> 0.07692308,
'2C'=>0.0,
'3C'=>0.07692308,
'4C'=>0.07692308,
'5C'=>0.23076923,
'6C'=>0.0,
'1G'=>0.0,
'2G'=>0.07692308,
'3G'=>0.15384616,
'4G'=>0.30769232,
'5G'=>0.07692300,
'6G'=>0.0,
);
%tp=
(
'1AA'=>0.0,
'2AA'=>0.0,
'3AA'=>0.07692308,
'4AA'=>0.23076923,
'5AA'=>0.0,
'1AT'=>0.0,
'2AT'=>0.61538464,
'3AT'=>0.0,
'4AT'=>0.07692308,
'5AT'=>0.53846157,
'1AG'=>0.0,
'2AG'=>0.15384616,
'3AG'=>0.0,
'4AG'=>0.0,
'5AG'=>0.0,
'1AC'=>0.0,
'2AC'=>0.15384616,
'3AC'=>0.0,
'4AC'=>0.15384616,
'5AC'=>0.0,
#########
'1TA'=>0.7692308,
'2TA'=>0.7692308,
'3TA'=>0.30769232,
'4TA'=>0.7692308,
'5TA'=>0.7692308,
'1TT'=>0.7692308,
'2TT'=>0.0,
'3TT'=>0.7692308,
'4TT'=>0.0,
'5TT'=>0.7692308,
'1TG'=>0.7692308,
'2TG'=>0.0,
'3TG'=>0.23076923,
'4TG'=>0.7692308,
'5TG'=>0.0,
'1TC'=>0.0,
'2TC'=>0.0,
'3TC'=>0.23076923,
'4TC'=>0.0,
'5TC'=>0.0,
#########
'1GA'=>0.0,
'2GA'=>0.0,
'3GA'=>0.7692308,
'4GA'=>0.15384616,
'5GA'=>0.0,
'1GT'=>0.0,
'2GT'=>0.7692308,
'3GT'=>0.0,
'4GT'=>0.7692308,
'5GT'=>0.7692308,
'1GG'=>0.0,
'2GG'=>0.0,
'3GG'=>0.7692308,
'4GG'=>0.0,
'5GG'=>0.0,
'1GC'=>0.0,
'2GC'=>0.0,
'3GC'=>0.0,
'4GC'=>0.7692308,
'5GC'=>0.0,
#########
```

```
'1CA'=>0.7692308,
'2CA'=>0.0,
'3CA'=>0.0,
'4CA'=>0.7692308,
'5CA'=>0.0,
'1CT'=>0.0,
'2CT'=>0.0,
'3CT'=>0.7692308,
'4CT'=>0.0,
'5CT'=>0.23076923,
'1CG'=>0.0,
'2CG'=>0.0,
'3CG'=>0.0,
'4CG'=>0.0,
'5CG'=>0.0,
'1CC'=>0.0,
'2CC'=>0.0,
'3CC'=>0.0,
'4CC'=>0.0,
'5CC'=>0.0,
);
for($i=1;$i<=$fcount;$i++)
{
open(HANDLE,'C:\DTITK\SG\SG'.$i.'.txt');
@arr=<HANDLE>;
close HANDLE;
$tag=$arr[0];
$tag=~ s/\n//;
$arr[0]='';
$gene=join('',@arr);
$gene=~ s/ //;
open(HH,"$cmd");
@con=<HH>;
$con[0]='';
$m=join('',@con);
$m=~ s/\n//gi;
#print "Gene:".$gene;
#print "\nGenome:".$m;
#<>;
$index = index $m,$gene;
if($index == -1)
{
$gene=~ tr/ATGCatgc/TACGtacg/;
$index = index $m,$gene;
}
print $index."\n";
$tata=substr($m,($index-14),9);
@prom;
$prom[0]=substr($tata,0,6);
$prom[1]=substr($tata,1,6);
$prom[2]=substr($tata,2,6);
$prom[3]=substr($tata,3,6);
$prom[4]=substr($tata,4,6);
$prom[5]=substr($tata,5,6);
@score=();
for($k=1;$k<=6;$k++)
{
```

```
@temp=split('',$prom[$k-1]);
{
for($j=0;$j<=6;$j++)
{
if($temp[$j] eq 'A')
{
$score[$k]=$score[$k]+$pb{$j.'A'};
if($temp[($j+1)] eq 'A')
{
$score[$k]=$score[$k]+$tp{$j.'AA'};
}
if($temp[($j+1)] eq 'T')
{
$score[$k]=$score[$k]+$tp{$j.'AT'};
}
if($temp[($j+1)] eq 'G')
{
$score[$k]=$score[$k]+$tp{$j.'AG'};
}
if($temp[($j+1)] eq 'C')
{
$score[$k]=$score[$k]+$tp{$j.'AC'};
}
}
if($temp[$j] eq 'T')
{
$score[$k]=$score[$k]+$pb{$j.'T'};
if($temp[($j+1)] eq 'A')
{
$score[$k]=$score[$k]+$tp{$j.'TA'};
}
if($temp[($j+1)] eq 'T')
{
$score[$k]=$score[$k]+$tp{$j.'TT'};
}
if($temp[($j+1)] eq 'G')
{
$score[$k]=$score[$k]+$tp{$j.'TG'};
}
if($temp[($j+1)] eq 'C')
{
$score[$k]=$score[$k]+$tp{$j.'TC'};
}
}
if($temp[$j] eq 'G')
{
$score[$k]=$score[$k]+$pb{$j.'G'};
if($temp[($j+1)] eq 'A')
{
$score[$k]=$score[$k]+$tp{$j.'GA'};
}
if($temp[($j+1)] eq 'T')
{
$score[$k]=$score[$k]+$tp{$j.'GT'};
}
if($temp[($j+1)] eq 'G')
{
```

```
$score[$k]=$score[$k]+$tp{$j.'GG'};
}
if($temp[($j+1)] eq 'C')
{
$score[$k]=$score[$k]+$tp{$j.'GC'};
}
}
if($temp[$j] eq 'C')
{
$score[$k]=$score[$k]+$pb{$j.'C'};
if($temp[($j+1)] eq 'A')
{
$score[$k]=$score[$k]+$tp{$j.'CA'};
}
if($temp[($j+1)] eq 'T')
{
$score[$k]=$score[$k]+$tp{$j.'CT'};
}
if($temp[($j+1)] eq 'G')
{
$score[$k]=$score[$k]+$tp{$j.'CG'};
}
if($temp[($j+1)] eq 'C')
{
$score[$k]=$score[$k]+$tp{$j.'CC'};
}
}
}
}
}
####################################################
$max=$score[1];
if($score[2]>$max)
{
$max=$score[2];
}
if($score[3]>$max)
{
$max=$score[3];
}
if($score[4]>$max)
{
$max=$score[4];
}
if($score[5]>$max)
{
$max=$score[5];
}
if($score[6]>$max)
{
$max=$score[6];
}
if($max>4.4)
{
print HARI $tag."\t"."Promoter Score: 4  ".$max."\n";
}
if(($max>=3) && ($max<=4.44))
```

```
{
print HARI $tag."\t"."Promoter Score: 2  ".$max."\n";
}
if($max<3)
{
print HARI $tag."\t"."Promoter Score: 1  ".$max."\n";
}
$score[1]=0;
$score[2]=0;
$score[3]=0;
$score[4]=0;
$score[5]=0;
$score[6]=0;
$max=0;
}
```

2.6 The program for scoring the genes on basis of Shine Dalgarno sequence

```
$fcount=$ARGV[0];
$cmd=$ARGV[1];
%pb=
(
'1A'=>0.0,
'2A'=>0,
'3A'=>0,
'4A'=>0,
'5A'=>0,
'6A'=>0,
'1T'=>0.25,
'2T'=>0,
'3T'=>0.25,
'4T'=>0.5,
'5T'=>0,
'6T'=>0,
'1C'=> 0.75,
'2C'=>1,
'3C'=>0.75,
'4C'=>0.5,
'5C'=>1,
'6C'=>1,
'1G'=>0.0,
'2G'=>0,
'3G'=>0,
'4G'=>0,
'5G'=>0,
'6G'=>0.0,
);
%tp=
(
'1AA'=>0.0,
'2AA'=>0.0,
'3AA'=>0,
'4AA'=>0,
'5AA'=>0.0,
'1AT'=>0.0,
'2AT'=>0,
'3AT'=>0.0,
'4AT'=>0,
'5AT'=>0,
'1AG'=>0.0,
'2AG'=>0,
'3AG'=>0.0,
'4AG'=>0.0,
'5AG'=>0.0,
'1AC'=>0.0,
'2AC'=>0,
'3AC'=>0.0,
'4AC'=>0,
'5AC'=>0.0,
#########
'1TA'=>0,
'2TA'=>0,
'3TA'=>0,
'4TA'=>0,
'5TA'=>0,
'1TT'=>0,
'2TT'=>0.0,
'3TT'=>0,
'4TT'=>0.0,
'5TT'=>0,
'1TG'=>0,
'2TG'=>0.0,
'3TG'=>0,
'4TG'=>0,
'5TG'=>0.0,
'1TC'=>0.25,
'2TC'=>0.0,
'3TC'=>0.25,
'4TC'=>0.5,
'5TC'=>0.0,
#########
'1GA'=>0.0,
'2GA'=>0.0,
'3GA'=>0,
'4GA'=>0,
'5GA'=>0.0,
'1GT'=>0.0,
'2GT'=>0,
'3GT'=>0.0,
'4GT'=>0,
'5GT'=>0,
'1GG'=>0.0,
'2GG'=>0.0,
'3GG'=>0,
'4GG'=>0.0,
'5GG'=>0.0,
'1GC'=>0.0,
'2GC'=>0.0,
'3GC'=>0.0,
'4GC'=>0,
'5GC'=>0.0,
#########
'1CA'=>0,
'2CA'=>0.0,
'3CA'=>0.0,
'4CA'=>0,
'5CA'=>0.0,
'1CT'=>0.0,
'2CT'=>0.25,
'3CT'=>0.5,
'4CT'=>0.0,
'5CT'=>0.23076923,
'1CG'=>0.0,
'2CG'=>0.0,
'3CG'=>0.0,
'4CG'=>0.0,
'5CG'=>0.0,
'1CC'=>0.75,
'2CC'=>0.75,
'3CC'=>0.25,
'4CC'=>0.5,
'5CC'=>1,
);
open(HARI,'>c:\DTITK\SHINESCORE.txt');
for($i=1;$i<=$fcount;$i++)
{
open(HANDLE,'C:\DTITK\SG\SG'.$i.'.txt');
@arr=<HANDLE>;
close HANDLE;
$tag=$arr[0];
$tag=~ s/\n//;
$arr[0]=";
$gene=join(",@arr);
$gene=~ s/ //;
open(HH,"$cmd");
@con=<HH>;
$con[0]=";
$m=join(",@con);
$m=~ s/\n//gi;
$index = index $m,$gene;
if($index == -1)
{
$gene=~ tr/ATGCatgc/TACGtacg/;
$index = index $m,$gene;
}
$shine=substr($m,($index+4),11);
@shine1;
$shine1[0]=substr($shine,0,6);
$shine1[1]=substr($shine,1,6);
$shine1[2]=substr($shine,2,6);
$shine1[3]=substr($shine,3,6);
$shine1[4]=substr($shine,4,6);
$shine1[5]=substr($shine,5,6);
for($k=1;$k<=6;$k++)
{
@temp=split(",$shine1[$k-1]);
{
for($j=0;$j<=6;$j++)
{
if($temp[$j] eq 'A')
{
```

```
$score[$k]=$score[$k]+$pb{$j.'A'};
if($temp[($j+1)] eq 'A')
{
$score[$k]=$score[$k]+$tp{$j.'AA'};
}
if($temp[($j+1)] eq 'T')
{
$score[$k]=$score[$k]+$tp{$j.'AT'};
}
if($temp[($j+1)] eq 'G')
{
$score[$k]=$score[$k]+$tp{$j.'AG'};
}
if($temp[($j+1)] eq 'C')
{
$score[$k]=$score[$k]+$tp{$j.'AC'};
}
}
if($temp[$j] eq 'T')
{
$score[$k]=$score[$k]+$pb{$j.'T'};
if($temp[($j+1)] eq 'A')
{
$score[$k]=$score[$k]+$tp{$j.'TA'};
}
if($temp[($j+1)] eq 'T')
{
$score[$k]=$score[$k]+$tp{$j.'TT'};
}
if($temp[($j+1)] eq 'G')
{
$score[$k]=$score[$k]+$tp{$j.'TG'};
}
if($temp[($j+1)] eq 'C')
{
$score[$k]=$score[$k]+$tp{$j.'TC'};
}
}
if($temp[$j] eq 'G')
{
$score[$k]=$score[$k]+$pb{$j.'G'};
if($temp[($j+1)] eq 'A')
{
$score[$k]=$score[$k]+$tp{$j.'GA'};
}
if($temp[($j+1)] eq 'T')
{
$score[$k]=$score[$k]+$tp{$j.'GT'};
}
if($temp[($j+1)] eq 'G')
{
$score[$k]=$score[$k]+$tp{$j.'GG'};
}
if($temp[($j+1)] eq 'C')
{
$score[$k]=$score[$k]+$tp{$j.'GC'};
}
```

```
}
if($temp[$j] eq 'C')
{
$score[$k]=$score[$k]+$pb{$j.'C'};
if($temp[($j+1)] eq 'A')
{
$score[$k]=$score[$k]+$tp{$j.'CA'};
}
if($temp[($j+1)] eq 'T')
{
$score[$k]=$score[$k]+$tp{$j.'CT'};
}
if($temp[($j+1)] eq 'G')
{
$score[$k]=$score[$k]+$tp{$j.'CG'};
}
if($temp[($j+1)] eq 'C')
{
$score[$k]=$score[$k]+$tp{$j.'CC'};
}
}
}
}
}
####################################################
$max=$score[1];
if($score[2]>$max)
{
$max=$score[2];
}
if($score[3]>$max)
{
$max=$score[3];
}
if($score[4]>$max)
{
$max=$score[4];
}
if($score[5]>$max)
{
$max=$score[5];
}
if($score[6]>$max)
{
$max=$score[6];
}
if($max>4.4)
{
print HARI $tag."\t"."Shine dalagarno  Score: 4 ".$max."\n";
}
if(($max>=3) && ($max<=4.44))
{
print HARI $tag."\t"."Shine dalagarno  Score: 2 ".$max."\n";
}
if($max<3)
{
print HARI $tag."\t"."Shine dalagarno score: 1 ".$max."\n";
```

```
}
$score[1]=0;
$score[2]=0;
$score[3]=0;
$score[4]=0;
$score[5]=0;
$score[6]=0;
$max=0;
}
```

2.7 Program in Python to retrieve the target sequences, PROSITE motif signatures, for the calculation involving the total number of occurrences of these motifs present in each of the input sequences and finally the RANK assignment to each of these protein sequences.

```
import re
import string
import Bio
from Bio import Fasta
#from pros_pattall123 import assign
parser = Fasta.RecordParser()

#Retrieval of target sequences......
seqs = []
for i in range(135):
    k = []
    fh1 = "%d.txt" %i
    fh = open(fh1)
    iterator = Fasta.Iterator( fh, parser )
    while 1:
        record = iterator.next()
        if not record:
            break
        k.append(record.sequence)
    seqs.append(k)
    fh.close()
#End of retrieval ......

#Opening of prositedom.txt file .....
fh2 = open("prositedom.txt")
list1 = []
for i in fh2.readlines():
    if( i == "\n"):
        break
    else:
        list1.append(i.strip())
fh2.close()
#End of domain Retrieval....

#Training data from the target dataset .. ..
listvalues = []
for a in list1:
    flag = 0
    counter = 0
    pattern = re.compile(a)
    for i in range(len(seqs)):
        for j in range(len(seqs[i])):
            result = re.search( pattern,seqs[i][j] )
            if(result):
                flag += 1
    listvalues.append(flag)

m = 0
#For getting the sequences from the input file 'X.txt' and storing in a list.....
prott = []
prots = []
parser = Fasta.RecordParser()
#inputseq = raw_input("Enter the File Name: ")
infile = open("C:\DTITK\prot.txt")
iterator = Fasta.Iterator(infile,parser)
while 1:
    record = iterator.next()
    if not record:
        break
    prots.append(record.sequence)
    prott.append(record.title)
#End of retrieval of the protein sequences.....

#Opening of prositedom.txt containing all the the motif definitions.....
fh = open("prositedom.txt")
list1 = []
for i in fh.readlines():
    if( i == "\n"):
        break
    else:
        list1.append(i.strip())
fh.close()
#End of domain Retrieval....

#Forming a Dictionary...containing 'motifs as keys' n 'its corresponding values'
ab = {}
for i in range(len(list1)):
    ab[list1[i]] = listvalues[i]
#Dictionary formed.....

#Calculation of Score.....
score_list = []
for i in range(len(prots)):
    score = 0
    for a in ab.keys():
        pattern = re.compile(a)
        result = re.search(pattern,prots[i])
        if(result):
            score = score + ab[a]
    score_list.append(score)
#Scores calculated.....

#Sorting the values in the list
pos = []
for i in range(len(score_list)):
    pos.append(i)
for i in range(len(score_list)):
    for j in range(len(score_list)-1):
        if( score_list[j] < score_list[j+1] ):
            temp1 = score_list[j]
            score_list[j] = score_list[j+1]
            score_list[j+1] = temp1
            temp2 = pos[j]
            pos[j] = pos[j+1]
            pos[j+1] = temp2
#Scores sorted.....

def comparer( a,b ):
    if( a == b ):
        return 0
    else:
        return 1

#Writing the titles,scores and corresponding ranks to a file....
fh = open("rank.txt","w+t")
k = 1
n = -1
j = 0   #for scorelist....
R1 = 0   #for rank....
R2 = 1
for i in pos:
    fh.write(">")
    fh.write(prott[i])

    fh.write(" Score: ")
    a = score_list[j]
    b = "%d" %a
    fh.write( b )

    if( j < len(score_list)-1 ):
        R1 = comparer(score_list[j], score_list[j+1])
        k4 = "%d\n" %R2
        fh.write(" Rank: ")
        fh.write(k4)
        R2  += R1

    j = j+1
    #n = n+1
z = "%d" %R2
fh.write( " Rank: " )
fh.write(z)
fh.close()
#File written....

print "\nCheck your results at rank.txt\n"
```