12-4-2006

CAD Tools for DNA Micro-Array Design, Manufacture and Application

Nisar Hundewale

CAD TOOLS FOR DNA MICRO-ARRAY DESIGN, MANUFACTURE AND APPLICATION

by

Nisar Hundewale

Under the Direction of Professor Alexander Zelikovsky

ABSTRACT

Motivation: As the human genome project progresses and some microbial and eukaryotic genomes are recognized, numerous biotechnological processes have attracted increasing number of biologists, bioengineers and computer scientists recently. Biotechnological processes profoundly involve production and analysis of high-throughput experimental data. Numerous sequence libraries of DNA and protein structures of a large number of micro-organisms and a variety of other databases related to biology and chemistry are available. For example, microarray technology, a novel biotechnology, promises to monitor the whole genome at once, so that researchers can study the whole genome on the global level and have a better picture of the expressions among millions of genes simultaneously. Today, it is widely used in many fields- disease diagnosis, gene classification, gene regulatory network, and drug discovery. For example, designing organism specific microarray and analysis of experimental data require combining heterogeneous computational tools that usually differ in the data format; such as, GeneMark for ORF extraction, Promide for DNA probe selection, Chip for probe placement on microarray chip, BLAST to compare sequences, MEGA for phylogenetic analysis, and ClustalX for multiple alignments.

Solution: Surprisingly enough, despite huge research efforts invested in DNA array applications, very few works are devoted to computer-aided optimization of DNA array design and manufacturing. Current design practices are dominated by ad-hoc heuristics incorporated in proprietary tools with unknown suboptimality. This will soon become a bottleneck for the new generation of high-density arrays, such as the ones currently being designed at Perlegen [109].

The goal of the already accomplished research was to develop highly scalable tools, with predictable runtime and quality, for cost-effective, computer-aided design and manufacturing of DNA probe arrays. We illustrate the utility of our approach by taking a concrete example of combining the design tools of microarray technology for Harpes B virus DNA data.

INDEX WORDS: CAD tools, design, DNA microarray, RNA Array, manufacture, optimization, ORF, probe placement, Universal Tag Array, workflow.

CAD TOOLS FOR DNA MICRO-ARRAY DESIGN, MANUFACTURE AND APPLICATION

by

Nisar Hundewale

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia State University

2006

CAD TOOLS FOR DNA MICRO-ARRAY DESIGN, MANUFACTURE AND APPLICATION

by

Nisar Hundewale

Major Professor:	Prof. Alexander Zelikovsky
Committee:	Prof. Yi Pan
	Prof. Robert Harrison
	Prof. Ludmila Perelygina

Electronic Version Approved:

Office of Graduate Studies
College of Arts and Sciences
Georgia State University
December 2006

to The Creator of universes

ACKNOWLEDGMENTS

Thanks to my advisor Prof. Alexander Zelikovsky for giving me advise and guide me in the research. I am grateful to the Chair Prof. Yi Pan for his constant encouragement through out my stay at GSU. I would like to give special thanks to Prof. Harrison for sharing his valuable knowledge. I thank Prof. Ludmila Perelygina for her time to explain the biological basis of the work. I thank all the colleagues and faculty members who have spend their time discussing with me various research problems.

I thank all the student peers who have also spent great deal of their time working together on different research topics. My special thanks to Mourad Atlas who did spend lot of time working together on interesting bioinformatics and biostatistics problems.

Special thanks to my family members who have patiently been my moral support thorough out my education and career.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

DNA probe arrays – DNA arrays or DNA chips for short – are the means of choice for performing a wide range of genomic analyses, including gene expression monitoring, mutation detection, and single nucleotide polymorphism analysis (see, e.g., [89] for a survey). The reasons for this wide acceptance are a unique combination of robust manufacturing, massive parallel measurement capabilities, and highly accurate and reproducible results. Current commercial DNA arrays integrate hundreds of thousands of different probes on a surface only slightly larger than 1cm^2, enabling researchers to perform fast and reliable genome-wide analyses using small sample volumes. Next-generation designs are expected to integrate up to hundreds of millions of different probes [109]. Finally, a future "killer application" is to build arrays with billions of probes that would allow one to analyze all SNPs in a humane genome in a single experiment.

Today, most DNA arrays are manufactured through a highly scalable process, referred to as *Very Large-Scale Immobilized Polymer Synthesis (VLSIPS)*, that combines photolithographic technologies adapted from the semiconductor industry with combinatorial chemistry [3, 109, 56]. Similar to Very Large Scale Integration (VLSI) circuit manufacturing, multiple copies of a DNA array design are simultaneously synthesized on a *wafer*, typically made out of quartz. When synthesis is complete, the wafers are diced and arrays are packaged individually. Depending on the number of distinct probes per array, a single 5" square wafer can yield between 49 and 400 arrays. To initiate synthesis, linker molecules including a photolabile protective group are attached to the wafer, forming a regular 2-dimensional pattern of synthesis sites. Probe synthesis then proceeds in successive steps, with one nucleotide (A, C, T, or G) being synthesized at a selected set of sites in each step. To select which sites will receive nucleotides, photolithographic masks, or *reticles*, are placed over the wafer. Exposure to light de-protects linker molecules at the non-masked sites. Once the desired sites have been activated in this way, a solution containing a single type of nucleotide (which bears its own photolabile protection group to prevent the probe from growing by more than one nucleotide) is flushed over the wafer's surface. Protected nucleotides attach to the unprotected linkers, initiating the probe synthesis process. In each subsequent step, another mask is used to enable selective de-protection and single-nucleotide synthesis. This cycle is repeated until all probes have been fully synthesized. After the chip is synthesized, it is hybridized and image scanning then indicates where successful and unsuccessful hybridization of control probes occur as given by the image intensities [89]. Bright control spots indicate successful hybridization resulting in sound control probes, while dim control spots indicate unsuccessful hybridization resulting in defective control probes.

As the number of DNA array designs is expected to ramp up in coming years with the ever-growing number of applications [63, 131], there is an urgent need for high-quality software tools to assist in the design and manufacturing process. The biggest challenges to rapid growth of DNA array technology are:

- drastic increases in design sizes with simultaneous decrease of array cell sizes – next-generation designs are envisioned to have hundreds of millions of cells of sub-micron size;
- increased *non-recurring* design and manufacturing costs, which quickly become prohibitive for designs that have low production volumes; and

- increased complexity of the design process, which leads to unpredictability of design quality and design turnaround time.

Surprisingly enough, despite huge research efforts invested in DNA array applications, very few works are devoted to computer-aided optimization of DNA array design and manufacturing. Current design practices are dominated by ad-hoc heuristics incorporated in proprietary tools with unknown suboptimality. This will soon become a bottleneck for the new generation of high-density arrays, such as the ones currently being designed at Perlegen [109].

The goal of the already accomplished research was to develop highly scalable tools, with predictable runtime and quality, for cost-effective, computer-aided design and manufacturing of DNA probe arrays. The challenges identified above were addressed on several levels:

- accurate formalization and modelling of the entire DNA array design and manufacturing process;
- identification of intermediate optimization objectives highly correlated with final design quality and cost;
- optimization of individual design steps, and integration of an entire design flow by introducing flow-aware optimizations and feedback loops;
- development of novel algorithm techniques leading to highly scalable, predictable, and near-optimal tools for all design phases;
- extensive algorithm engineering and empirical study that thoroughly tests the efficiency of proposed algorithms on both synthetic and industrial benchmarks;
- exploration and simulation of *new manufacturing technology solutions and methodologies* that extend beyond the current state-of-the-art; and

- creation of electronic media for distribution of high-quality, open-source software implementations of CAD tools, synthetic and industry benchmarks, and results of empirical studies.

The next chapter summarizes previous work on DNA array design. The third chapter addresses DNA microarray design and the detailed basic design flow as laid out by [12]. The fourth chapter deals with the design flow of universal tag microarrays (UTA), as it enhances the design and manufacturing steps while reducing the cost and time of manufacture of microarray as laid out by [72]. This area is the least researched and the newest of interest. With the advent of microarray technology high-throughput data poses a greater challenge in terms of processing it. The time and hardware resources required in order to analyze the data is enormous therefore requires to choose a suitable algorithm. In chapter five, we present parallel statistical validation of clustering algorithms for the analysis of microarray data.

Affymetrix [3] in July 2006 announced that it plans to introduce a one million-SNP product by the first quarter of 2007. With the dramatic reduction of the price of its current two-chip 500K SNP genotyping set to $250, and also increasing throughput and enabling scientists to run more samples, and thereby increase the power of their genetic association studies. For example, thousands of samples are needed to find genes that are associated with the complex diseases like diabetes. We perform disease association study and present the results in chapter six.

The chapter seven is the conclusion and summary of the work and remaining problems mention in the dissertation. In the eighth chapter we discuss the prospective microarray design technologies. We describes RNA microarray design and propose the software tools for creating a RNA design workflow. Also, we illustrate the workflow of synthesis methodology for digital microfluidics-based biochips.

We list the relevant paper publications, and author's other publications in chapters nine and ten.

At last, the bibliography of referred publications marks the end of this dissertation.

CHAPTER 2
PREVIOUS WORK

2.1 DNA Array Design

Bearing to the similarity between the manufacturing of DNA arrays and that of VLSI chips, the design of DNA arrays shares many common principles with VLSI circuit design. In this section we describe the physical design of DNA microarray.

Under ideal manufacturing conditions, the functionality of a DNA array is not affected by the placement of the probes on the chip, or the particular order in which nucleotides of each probe are synthesized. In practice, since manufacturing process is prone to errors, probe placement and synthesis schedules affect to a great degree the hybridization sensitivity and ultimately the functionality of the array. There are several types of synthesis errors that take place during array manufacturing. First, a probe may not loose its protective group when exposed to light, or the protective group may be lost but the nucleotide to be synthesized may not attach to the probe. Second, due to diffraction, internal reflection, and scattering, unintended illumination may occur at sites that are geometrically close to intentionally exposed regions. The first type of manufacturing errors can be effectively controlled by careful choice of manufacturing process parameters, e.g., by proper control of exposure times and by insertion of correction steps that irrevocably end synthesis of all probes that are unprotected at the end of a synthesis step [3, 109]. Errors of second type of errors result in synthesis of unforeseen sequences in masked sites and compromises interpretation of experimental data. To reduce such uncertainty, one can exploit freedom in how probes are assigned to array sites by optimizing the placement and embedding of the probes such that the sum of border lengths in all masks is minimum.[1]

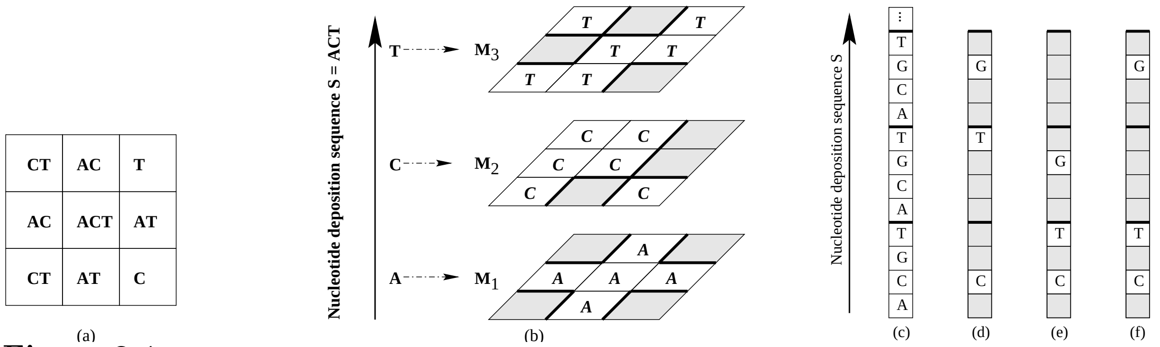
Kahng et al [81] view array design as a *three-dimensional placement* problem (Figure 2.1(a-b)): two dimensions represent the site array, and the third dimension represents the nucleotide deposition sequence S. Each layer in the third dimension corresponds to a mask that induces deposition of a particular nucleotide (A, C, G, or T); a probe is *embedded* within a "column" of this three-dimensional placement representation. Border length of a given mask is computed as the number of *conflicts*, i.e., pairs of adjacent exposed and masked sites in the mask. Given two adjacent embedded probes p and p', the *conflict distance* $d(p, p')$ is the number of conflicts between the corresponding columns. The border length of the embedding is the sum of conflict distances between adjacent probes, and the *border minimization problem (BMP)* seeks to minimize this quantity.

The border minimization problem was considered for uniform arrays (i.e., arrays containing all possible probes of a given length) by Feldman and Pevzner [54], who proposed an optimal solution based on 2-dimensional Gray codes. For non-uniform arrays, the border minimization problem was formulated by Hannenhalli et al. [65], who gave methods for placement (at array sites) and embedding (in the mask sequence) of probes in a *synchronous* synthesis regime, in which the nucleotide deposition sequence S is periodic, and the i^{th} fperiod ($ACGT$) of S synthesizes a single (the i^{th}) nucleotide in each probe. This implies a unique and trivially computed embedding of each probe p in the sequence S; see Figure 2.1(d).

In the synchronous synthesis context, the conflict distance between two probes is $d(p, p') = 2h(p, p')$, with $h(p, p')$ denoting the Hamming distance between p and p', i.e., the number of positions in which p and p' differ. As recounted in [65], the first array

[1]Reducing unwanted illumination improves the signal to noise ratio in image analysis after hybridization, and thus permits smaller array sites or more probes per array [71].

design at Affymetrix used a travelling salesman problem (TSP) heuristic to arrange all probes in a tour that heuristically minimized Hamming distance between neighboring probes in the tour. The tour was then *threaded* into the two-dimensional array of sites, using a technique similar to one previously used in VLSI design [86]. [65] enhanced this threading approach to achieve up to 20% border length reduction for large chips. Recently, in [79], Kahng et al suggested an *epitaxial* placement heuristic which places a random probe in the center of the array and then continues to insert probes in sites adjacent to already-placed probes, so as to greedily minimize the number of induced conflicts. Epitaxial, or "seeded crystal growth", placement is a technique that has been well-explored in the VLSI circuit placement literature [107, 115]. Our epitaxial placement method improves over the previous TSP-based approach by up to 10%.



CT AC T
AC ACT AT
CT AT C
(a)
Nucleotide deposition sequence S = ACT
T → M3
C → M2
A → M1
(b)
Nucleotide deposition sequence S
(c)
(d)
(e)
(f)

Figure 2.1. (a) 2-dimensional probe placement. (b) 3-dimensional probe embedding. The nucleotide deposition sequence $S = (ACT)$ corresponds to the sequence of three masks M_1, M_2 and M_3. In each mask the masked sites are shaded and the borders between exposed and masked sites are thickened. (c) Periodic nucleotide deposition sequence S. (d) Synchronous embedding of probe CTG into S; shaded sites denote the masked sites in the corresponding masks. (e-f) Two different asynchronous embeddings of the same probe.

Recently, in [79], Kahng et al introduced the border minimization problem for the asynchronous synthesis regime, which allows arbitrary probe embeddings, as illustrated in Figure 2.1(e-f). Asynchronous synthesis has identical technological requirements with synchronous synthesis. Asynchronous synthesis is already mandated by minimization of the number of synthesis steps. At the same time, asynchronous embedding offers more flexibility for reducing total border length.

Note that in the asynchronous context, the conflict distance between two adjacent probes depends on their embedding. In [79] Kahng et al proposed dynamic programming algorithms that embed a given probe optimally with respect to fixed embeddings of the probe's neighbors. We also suggest two ways to improve the embedding of probes that have already been placed (e.g., by threading as in [65], or by epitaxial methods) onto array sites. We describe the *chessboard* and *batched greedy* methods for optimizing probe embeddings *after* assignment of probes to their positions in the array. The *chessboard* method alternates between optimal embedding adjustment of "black" and "white" sites with respect to their neighbors (which always have different color). The *batched greedy* method implements non-conflicting optimal embeddings in a greedy fashion. Our experiments show that the chessboard method (winning slightly over the batched greedy method) decreases by 15.5-21.8% the number of conflicts in the original synchronous placement.

Furthermore, in [79] Kahng et al give the first known non-trivial *a priori* lower bounds on the total border length of the optimum synchronous solution based on Hamming distance, and of the optimum asynchronous solution based on the length of the Longest Common Subsequence (LCS). These afford an estimate of potential future progress due to improved probe placement algorithms for synchronous embedding. We also give an LCS-distance based lower bound method that yields bounds on possible improvement from exploiting the freedom to change probe embeddings but not their placement.

In ongoing work, Kahng et al are developing high-quality, scalable methods for designing large DNA arrays, seeking methods that allow more accurate modelling of the array design problem, and hence form the basis of more practically relevant heuristics. Two of our main design goals are: (i) to enable scaling to 100 million or more placeable objects envisioned for the next-generation of DNA arrays [89] (say, within the current or next generation of workstations), and (ii) to enable easy parallelism (implying near-linear speedup on workstation clusters) wherever possible. We next note that the placer design can be made according to two basic functions [115]: *initial* *placement* and (iterative) *placement improvement.* Each of these functions can apply to (i) the assignment of probes to array sites, and/or (ii) the embedding of probes in the mask sequence S. Within an overall placement metaheuristic, the requirements for one function are strongly dependent on the implementation of the other function.

The following extensions to the border length minimization problem are important in practice, but have not been addressed by previous works on the problem [65, 79].

1. **Distance-dependent border conflict weights.** Back-reflection of light affects not only adjacent array cells, but also cells that are as far as 3 cells apart [71]. This implies that we should weight conflicts according to the distance between cells.

2. **Position-dependent border conflict weights.** The weight of border conflicts depends on the position in the probe since contamination errors are more harmful in the middle of the probe [71]. Suggested weights are given by the square root of the distance to the closer endpoint (so, conflict weight varies from 1 to $\sqrt{12}$ in a 25-mer).

3. **Polymorphic probes.** Some of the synthesized DNA probes occur both unmodified and mutated in the middle position (e.g., for detection of single nucleotide polymorphism in the target DNA or for reliability of the hybridization test). To minimize border length the SNPs are placed together, so the general BMP requires placing and aligning a mixture of single probes, 2- and 4-ominoes.

In a later paper [80], Kahng et al reported improved algorithms for the border minimization problem that, unlike the previous methods in [65, 79], account for the above practical extensions. More importantly, unlike the methods in [65, 79], which have at least quadratic time complexity, our methods have nearly-linear runtime and are hence practically scalable to hundreds of millions of probes. In the remaining of this section present the description and experimental validation of the "engineering" of a scalable, high-quality asynchronous placement heuristic for DNA array design [80]. Kahng et al demonstrated the value of simple ordering-based methods for initial placement. Kahng et al also proposed the use of scalable sliding-window and row-epitaxial techniques having antecedents in large-scale integrated circuit placement [47, 70, 107, 115], as well as a local improvement operator based on reassignment of an “independent” set of probes. A recurring motif is the analogy between silicon chip design and DNA chip design, pointing to the value of technology transfer between the 40-year old VLSI CAD field and the newer realm of probe array design. Experimental results showed to confirm the linear scaling of runtime complexity and better solution quality compared to best previous methods.

CHAPTER 3

DNA MICROARRAY SOFTWARE TOOLS INTEGRATION

3.1 DNA Array Flow

In [12], we presented a concatenated software solution for the entire DNA array flow exploring all steps of a consolidated software tool. The proposed software tool has been tested on Herpes B virus as well as simulated data. Our experiments showed that the genomic data follow the pattern predicted by simulated data although the number of border conflicts (quality of the DNA array design) is several times smaller than for simulated data. We also reported a trade-off between the number of border conflicts and the running time for several proposed algorithmic techniques employed in the physical design of DNA arrays.

We describe the main flow of the producing a DNA chip, methods employed to read a genomic data from database and producing ORF's in FASTA format. We illustrate the models used for probe selection, Promide and Tm checker, with specific details regarding the production of pool of probes that must optimize hybridization affinity, and satisfy the constraints on the melting temperature. We describe VLSI CAD models of chip-physical design, and how to engage CAD algorithms in probe placement and embedding. We discuss mask and array manufacturing process that is a combination of photolithography and combinatorial chemistry, as well as a hybridization experiment and the analysis of gene intensities. We discuss a specific application of the DNA chip production to analyze the genome of the Herpes B virus, and we give the experimental results. We present the DNA Array flow (see 3.1).

3.2 Herpes B Virus

Herpes B virus generally causes only mild localized or asymptomatic infections in its natural hosts, Asian monkeys of the genus *Macaca* [83, 84]. In contrast, B virus infections in foreign hosts, humans or monkey species other than macaques, often result in encephalitis encephalomyelitis, and death [105, 132]. Recently, complete sequencing of the herpes B virus genome has been achieved [106]. The genome of B virus is 156,789 bp in length. 72 genes exist as a single copy and within unique genomic regions whereas two genes appear twice due to the duplication of the large and small repeat regions where they reside. The genomic sequence is proposed to facilitate identification of the genetic basis and possible molecular mechanisms of enhanced B virus neurovirulence in humans, which results in an 80% mortality rate following zoonotic infection.

The best known means for achieving this challenging goal are DNA probe arrays. The standard approach is to design a chip consisting of approximately 20 probes per gene and include also control probes which will constitute to around 1500 probes which is a small chip. Unfortunately, all gene sequences are very similar since 74.5% of all bases are G+C. This makes finding sufficient amount of unique probes for each gene a challenging if not impossible task. We propose to use universal DNA tag arrays for gene expression assay for Herpes B virus.

3.3 DNA Microarray Workflow

The basic flow for DNA microarray design begins with "ORF extraction" [12] (where ORF is the open reading frame). Probe selection, physical design, manufacturing and finally hybridization and analysis are the remaining steps of the design flow. [12] Each of these steps is discussed below with accompanied research paper references. The majority of this chapter is devoted to probe selection and design, since probe selection is common among DNA microarray, RNA, and UTA. The methods discussed in the probe selection portion are applicable to both the RNA design and UTA design chapters.

In the following sections we discuss each stage of DNA Array Flow.

3.4 Reading Genomic Data and ORF Extraction:

In this step, we use ORF extraction programs to extract the desired ORF.

ORF Extraction: ORF extraction is the process of extracting the usable portions of the DNA from the genome sequence. Portions that have looped around and hybridized with itself are unusable, and must not be included. Thus the usable portions or ORFs are extracted. [12] The importance of the ORF is that it is the portion that can be "transcribed into mRNA." [12]

The extraction process can be done using the ORF Finder software. [12] ORF Finder is a product of NCBI. The definition of ORF used by ORF Finder is the prokaryotic definition. [12] The significance of the definition used is that it only allows ORF Finder to be "useful in detecting the initial exons in a eukaryotic gene sequence." [12]

Another option for ORF extraction is GenMark. GenMark's Orflist function produces the list of ORFs. Both the position of the ORFs, the DNA strands, and coding frame are output. [12]

ORF Extraction Using ORF Finder: Extracting ORF sequence from the genome sequence is straightforward using ORF Finder. ORF can be extracted by means of the genome's sequence or id. ORF Finder is an extremely simple program at the NCBI that can be used to visualize open reading frames in a DNA sequence. An open reading frame is simply a subsequence of DNA that could potentially be transcribed into mRNA.

ORF Finder uses the prokaryotic definition of an open reading frame, and therefore is only useful in detecting the initial exons in a eukaryotic gene sequence. Each of these

ORFs represents a potential initial exon in a gene. The interface to ORF Finder is extremely functional, allowing the user to select and BLAST individual ORFs, which can be a somewhat effective way of finding unknown genes with initial exons that are similar to known genes. Also, the documentation and support available from the NCBI is good, especially considering the simplicity of ORF Finder. However, ORF Finder is a very limited tool for predicting genes within DNA sequences, and there are many more other programs available.

ORF Extraction Using GenMark: GenMark server provides e-mail based identification of protein coding regions in DNA sequences from prokaryotic and eukaryotic species. GeneMark server accepts a formatted message containing a DNA sequence in text format (numbers and spaces allowed, see below). Orflist is a function in GenMark, which is responsible for producing a list of evaluated open reading frames. The option 'Orflist' instructs to return a list of area's of the DNA sequence where GeneMark has identified a coding region. In the ORF list output, the "left end" and "right end" columns denote the physical position of the ORF in the sample sequence in absolute nucleotides relative to the beginning of the sample sequence. "DNA strands" indicates the strand in which the ORF lies. "Coding frame" indicates the reading frame in which the ORF lays relative beginning to the sequence. "Avg prob" denotes the mean coding potential over the indicated range and reading frame. "Start Prob" denotes the start probability, which is a measure of the likelihood that indicated start marks the true beginning of the ORF. There are several outputs; however, our concern will focus on the physical position and the name of the ORFs.

ORF Parser: In this step, we used the ORFs produced by GenMark because of the accuracy of GenMark over ORF Finder. In this step we parse the ORFs produced by GenMark to suit the input format acceptable by Promide. The output of GenMark is lists of start and end positions of each ORF along with their names. The parser locates each ORF using their positions within the entire sequence and creates a file with all ORF's in FASTA format. The content of pool of ORF's using provided information from GenMark is parsed to extract the target of the sequence in question. Although, Perl has strong native string matching capability; however, working at the character level is not as efficient compared to C++ or Java that is why we choose to implement the first parser using C++. The convenience here is that each time the genome sequence changes, we do not have to rewrite our parser.

3.5 Probe Selection

This step is analogous to the logic synthesis in VLSI design; the probe selection step is responsible for implementing the desired functionality of the DNA array. Although probe selection is application dependent, many underlying criteria are common all DNA micro-array technologies. The challenge for probe selection is how to identify the optimum probes for each gene. The basic criteria for the probes are the following: The probe should be a unique sequence in the all-coding sequences of the target genome, even with allowable number of mismatches, should not be self complementary, be close to the 3' end of that gene, does not contain single nucleotide multiple repeat regions, such as AAAAAA, does not a representative of low complexity region, and should have almost the same GC percentage as that of the target genome. Empirically, the optimum probe for a gene would be the one with minimum hybridization free energy for that gene (under the appropriate hybridization conditions) and, maximum hybridization free energy for all other genes in the hybridization pool. Probe selection presents the first algorithm that can select suitable custom oligonucleotide probes (e.g. 25-mers) for microarray experiments on a truly large scale. For example, oligos for human genes can be found within 50 hours. This becomes possible by using the longest common substring as a specificity measure for candidate oligos. Promide [110] presents an algorithm based on a suffix array with additional information that is efficient both in terms of memory usage and running time to rank all candidate oligos according to their specificity. Promide also introduce the concept of master sequences to describe the sequences from which oligos are to be selected. Constraints such as oligo length, melting temperature, and self-complementarily are incorporated in the master sequence at a preprocessing stage and thus kept separate from the main selection problem. We want to select 25-mers for every known ORF of budding sequence using Promide. We assume that the oligonucleotide probe sequence is the coding sequence of the gene; this choice depends on the type of assay being used. We already obtain a FASTA file with all genome ORF sequences using GenMark and Parser1.

Checking Temperature of Hybridization: In this step, we check the temperature of our probe that we produce from Promide. We calculate the temperature using the formula, 2 * number of A and number of T plus 4 times the number of G and C. Using this formula, we find that the temperature chosen before in Ocand is closed to the one calculated.

The selection of probes is what defines the functionality of the array [12]. The main issue facing probe selection is the problem of finding the best probes for the gene in question. [12], according to the criteria listed earlier in this section. Atlas et al [12] describe the "optimal probe for a gene" to be the probe with "the minimum hybridization free energy for that gene and, maximum hybridization free energy for all other genes in the hybridization pool". [12]

Probe selection methods can be broken down into statistical approaches, thermodynamic approaches, and other miscellaneous approaches. The following subsections deal with each of the above mentioned methodologies and present research that utilizes those methods.

3.5.1 Statistical Approaches

A statistical approach to probe selection may also utilize some thermodynamic properties, but the overwhelming methodology behind the selection is statistical in nature. The next few subsections feature some statistical approaches to probe selection from [113], and [111].

Group Testing with DNA Chips: Generating Designs and Decoding Experiments [113] Schliep et al [113] propose an approach to the design of microarrays "based on group testing." [113] The approach is purported by Schliep et al [113] to be the first "that explicitly takes cross-hybridization and experimental errors into account while accommodating several targets." [113] The approach used is statistical and non-adaptive. [113]

The "goal is to devise a group testing design which covers each target with a certain number of probes and allows identification of several targets simultaneously while using a reasonably small total number of probes." [113] Due to the difference in microarrays from traditional subjects of group testing some of the standards of group testing had to be adjusted and some added to cope with the specifics of microarrays. First the individuals cannot be randomly assigned to groups. [113] Other areas that differ from conventional group testing situations is that cross-hybridization still presents a problem, the possibility of high error rates exist, and the targets are capable of constantly changing. These departures from traditional group testing create further areas of concern in designing the testing scheme. [113]

The following notation is used to help describe the specifics of [113]ś work. "The m target sequences are denoted by $t_i(1 \leq i \leq m)$ the initial n_0 probe candidates by $p\prime_k(1 \leq k \leq n_0)$ and the final $n \leq n_0$ probes selected for the design by $p_j(1 \leq j \leq n)$. Thus every p_j is equal to some $p\prime_k$." [113] The "target set incidence matrix H" is defined as "$H_{ik} := 1$ if target t_i hybridizes to probe candidate $p\prime_k$, and $H_{ik} := 0$ otherwise." [113] D is the design matrix, and is a "sub matrix of H." [113] "$D_{ij} = 1$

if target t_i hybridizes to the selected probe p_j." [113] $P(i)$ is the "set of probes hybridizing to target t_i." [113]

Groups are designed such that "a potential target group T only exists if there is a probe that binds to all." [113] The probes "should not be self-complementary and not cross-hybridize to targets outside their intended target set." [113] The selection of candidates uses the "longest common factor." [113] "$lcf(s, t)$ of two strings s and t is the maximum length of a substring that occurs in both s and t." [113] The longest common factor statistic is defined as "$lcfs(o, l)$... as the number of genes $g\prime \neq g$ containing an oligo $o\prime$ with $lcf(o, o\prime) = l$." [113]

In order to find a good group testing design it is imperative that sets that are not too large be distinguishable. [113] The following is a definition of the separability of target sets. [113]

Definition 1 *Let S be set of target sequences. We say that a probe p hybridizes to the set S when p hybridizes to at least one target in S. By P(S) we denote the set of all probes hybridizing to S, i.e.,*$P(S) := \cup_{t_i \in S} P(t_i)$.

[113] This definition can be more clearly stated that if a particular probe can hybridize with one set of target probes but not with another, then the probe separates the two sets of target probes.

Probes are added "until every target is covered by at least d oligos." [113] All "pairs of targets are separated by at least d oligos." [113] When the hybridization is complete the process of deciphering which "targets were present in the sample," and "using the results for the different probes," to do so. [113] In other words the results must be decoded.

The probability that "a set T of targets constitutes all targets present in a sample given the result vector r" [113] can be expressed as follows.

$$P[T|r] = \frac{P[r|T] \cdot P[T]}{P[r]} \tag{3.1}$$

[113] The likelihood that the view is of a specific result is as follows.

$$P[r|T] = \prod_{p_j} f(r_j, |T(j) \cap T|) \tag{3.2}$$

[113]

Schliep et al [113] "formulate a Bayesian prior for the presence of a group of targets in the sample." [113] "Impendence between targets" [113] is assumed and there are two factors. [113] The first is the number of appearances of each target "t_i in samples containing at least one target is denoted f_i," [113] and the second is "the distribution describing the likelihood of finding a particular number of different targets in one sample." [113] c_k is the "prior probability of observing k different targets in one sample." [113] The "quantity proportional to the prior" [113] can be defined as the following assuming T to be the "set of targets." [113]

$$P[T] \propto C_{|T|} \cdot \prod_{t_i \in T} f_i \prod_{t_i \ni T} (1 - f_i) \tag{3.3}$$

[113]

The marginals are not easily computable due to the fact that the "exact computation requires work exponential in the number of targets." [113] $P[r]$ not being available "precludes...computing the posterior in closed form." [113] Marginals are the actual interest. [113] The following is the marginals "expressed in "terms of the posterior." [113]

$$P[tpresentinsample|r] \propto \sum_{T\, t \in T} P[T|r] \tag{3.4}$$

[113]

To cope with the above difficulties, the Markov Chain Monte Carlo (MCMC) approach can be used. [113] The marginal $P[tpresentinsample|r]$ can be estimated using the Monte Carlo approach. [113] The estimation is "the relative frequency with which t is continued in the sets T_k." [113] Sampling must be done "from $P[T|r]$," [113] which leads to "construction of a Markov chain over the space of all sets T." [113] "$P[T|r]$ is the stationary distribution." [113] Using Bayes' theorem, the "fraction $P[T_k|r]/P[T_k\delta\{t_i\}|r]$ equals" [113]

$$\frac{c_{|T_k|}(1-f_i)}{c_{|T_k|+1}f_i} \prod_{p_j \in P(i)} \frac{f(r_j, |T(j)\hat{T}_k|)}{f(r_j, |T(j)\hat{T}_k|+1)} \tag{3.5}$$

[113] "if $t_i \ni T_k$ and , for $t_i \in T_k$

$$\frac{c_{|T_k|+1}f_i}{c_{|T_k|}(1-f_i)} \prod_{p_j \in P(i)} \frac{f(r_j, |T(j)\hat{T}_k|)}{f(r_j, |T(j)\hat{T}_k|-1)} \tag{3.6}$$

[113]

The main focus of [113] was to present a "DNA microarray design methodology based on non-unique oligos and group testing." [113] The method is predominantly geared toward dealing with the problem that arises due to the "closely related target sequences." [113] This might be seen with related viruses. This high level of relation poses the interesting problem of finding probes for every target. The work as stated before is the "first that explicitly considers cross-hybridization issues and experimental errors within this framework." [113] The method has high "robustness", and is able to identify "92% of the targets present." [113] Schliep et al [113] remark that more work could be done to acquire more information about "the dynamics of hybridization reactions." [113] Other areas needed for work are those examining the usage of "several unique probes for each target." It is currently unknown if this can be done with other methods. Another area needing further examination is that of "detecting recombination events." [113]

Fast and Sensitive Probe Selection for DNA Chips Using Jumps in Matching Statistics Rahmann et al [111] presents a design approach to DNA microarrays that attempts to deal with the design problems associated with large scale

DNA arrays while preserving the accuracy. The approach is "based on jumps in matching statistics." [111]

The standard notation used in the description of Rahmann et al's work [111] follows. $\sum$ is used to denote a finite alphabet, and $\sum^*$ is the set of strings "consisting of characters from $\sum and \sum^+$ for all non-empty substrings." [111] Prefix of a string is defined to be the "a factor that starts at the beginning of the string." [111] Suffix of a string is defined to be " a factor that ends a string." [111] $|s|$ denotes the length of the string s, and $s_{i,\cdots,j}$ denote the substring from indexes of s from i to j. $s_{(i)} = s_{i\cdots|s|-1}$ denotes the "suffix string starting at position i." [111]

Matching statistics need to be explored in order for the proposed approach to be understood. The following definitions and lemmas are necessary in order to explain the jumping.

Definition 2 (Matching Statistics) *For strings s and t, the matching statistics of s against t are $ms^{s|t} = ms = (ms_0, \cdots, ms_{|s|-1})$, where ms_i is the length of the longest prefix of $s_{(i)}$ that occurs somewhere in t.*

[111]

Lemma 1 (Suffix property of matching statistics) *Consecutive matching statistics decrease by at most one: $ms_i^{s|t} \geq ms_{i-1}^{s|t} - 1 for all i = 1, \cdots, |s| - 1$*

[111]

Definition 3 (Jumps in matching statistics) *We say that a jump occurs at position $i > 0$ in $ms^{s|t}$ if and only if $ms_i^{s|t} \neq 0$ and $ms_i^{s|t} > ms_{i-1}^{s|t} - 1$.*

[111] This definition leads to the following notation that will be used to describe the [111] 's work. The jump level is represented as $J_i = ms_i^{s|t}$ if there is a "jump at position i." [111] A jump is defined to be a pair. The pair is composed of the jump

level and position of the jump. As an example the following is the jump at position i, (i, J_i)

Taking "u as a substring of s," [111] the following Lemma defines the matching statistics of substrings.

Lemma 2 (Matching statistics of substrings) *Let* $u = s_{i \cdots i+L-1}$*, so* $|u| = L$*. Then* $ms)k^{u|t} = min(ms^{s|t}_{i+k}, L-k)$ *(*$k = 0, \cdots, L-1$*)*

[111]

Lemma 3 (Jumps in matching statics of substrings) *Jumps in* $ms^{u|t}$ *can occur at position* $k = 0$*, and otherwise at most at those positions* $0 < k < L$ *where a jump occurs in* $ms^{s|t}$ *at position* $i + k$*.*

[111]

Definition 4 *We define the* $[i, L]$*-jump-set of* $ms^{s|t}$ *as the set consisting of jumps between positions* $i+1$ *and* $i+L-1$*(inclusive), and the closest jump to the left of* $i+1$ *at position* $i-d$ *for the appropriate distance* $d \geq 0$

[111]

Corollary 1 *Let* u *be the substring of length* L *of* s *that starts at position* i*. To obtain the jumps in* $ms^{u|t}$*, it suffices to look at the* $[i, L]$*-jump-set of* $ms^{s|t}$

[111]

The definition matching statistics with mismatches follows.

Definition 5 (Matching statistics with f mismatches) *For strings* s *and* t*, the matching statistics allowing* f *mismatches of* s *against* t *are* $ms^{s|t;f} = (ms^f_0, \cdots, ms^f_{|s|-1})$*, where* ms^f_i *is the length of the longest prefix of* $s_{(i)}$ *that occurs somewhere in* t *with at most* f *mismatches.*

[111] Jumps can subsequently be defined as before through the following Lemma.

Lemma 4 *Let u be a substring of s starting at position i. the longest common substring of u and t with at most f mismatches is the maximal level of all jumps in $ms^{u|t;f}$. It can be computed from the $[i, |u|]$-jump set of $ms^{s|t;f}$*

[111]

In order to determine the frequency of jumps in matching statistics, [111] considers "the number of occurrences K_L of a random string of length L in t." [111] $n - L + 1$ is the number of "possible starting positions." [111] p^L is the "probability that the length L substring a t a given position agrees with a random string of length L." [111]

Lemma 5 $E[K_L] = (n - L + 1) \cdot p^L$

[111] It is theoretically possible that exact distribution of K_L can be determined by taking into account all autocorrelations of all words. [111] It has also been proven that the estimation $P(K_L = 0) \approx e^{-E[K_L]}$ is accurate. [111] "K_L has a Poisson distribution," [111] The occurrences of K_L in t are independent if a random word has "no or low self-overlap," which will occur if π is not "close to a Dirac distribution." [111] The K_L occurrences in t that are independent and non overlapping, are rare due to the constraint that $E[K_L] << n$. [111] From the previous lemma is can be approximated that:

$$P(K_L = k) = e^{-\lambda} \cdot \frac{\lambda^k}{k} \tag{3.7}$$

where $\lambda = E[K_L] = (n - L + 1) \cdot p^L$ [111] $E)L$ is the expected number of jumps in "$ms^{s|t}$ to jump level L." [111] The probability that "a jump to level L occurs at position i in s," follows. [111]

Lemma 6 $p_L = e^{-\lambda \cdot (1-q^2)} - e^{-\lambda}$, *where* $\lambda = (n - L + 1) \cdot p^L$

[111]

Theorem 1 *An estimate of the expected number E_L of jumps in $ms^{s|t}$ with jump level L is given by $E_L = (m-L-1)e^{-\lambda\cdot(1-q^2)} + 2e^{-\lambda p} - (m-L+1)e^{-\lambda}$, where lambda has he same value as in the previous lemma.*

[111] The discussion will be restricted to only one mismatch because with greater number of mismatches the Poisson distribution of K_L breaks down.

Theorem 2 *The expected number E_L of jumps to level L in the matching statistics with one mismatch $ms^{m|s;1}$ is approximately given by $E_L = (m-L+1)(e^{-\lambda\cdot(1-q^2)} - e^{-\lambda})$, where $\lambda = (n-L+1)\cdot L \cdot p^{L-1}\cdot q$*

[111]

Considering "L^* where $E^+_{L^*} \geq 1$ but $E^+_{L^*+1} < 1$, " jumps to level L^* are observed but not jumps "to or above L^*+1." [111] L^* is an accurate "estimate of $lcf(s,t)$." [111] For no mismatches $\lambda \approx np^L$, $e^{-\lambda x} \approx 1 - \lambda x$, and thus $E_L \approx mq^2\lambda \approx mnq^2p^L$. [111] The following apply to the case with only one mismatch. $\lambda \approx nLqp^{L-1}$, and $E_L \approx Lmnq^3p^{L-1}$. [111] For exact matches the following thus applies.

$$E^+_L \approx mnqp^L \tag{3.8}$$

For one mismatch the following therefore applies.

$$E^+_L \approx Lmnq^2p^{L-1} \tag{3.9}$$

Since E^+_L is defined for both integers and non-integer values the following definition defines the center of the distribution of lcf.

Theorem 3 *The center $L^o$of the lcf of two random strings with lengths m and $n = \theta(m)$ generated with close to uniform character distribution π on $\sum$ is*

$$L^o \approx \frac{\ln(mn) + \ln q}{\ln(\frac{1}{p})} \tag{3.10}$$

This simplifies to $L^o \approx \log_{|\sum|}(mn)$ when π is uniform and $|\sum| >> 1$. For the length of the longest common factor with one mismatch (lcf^1), the center L^o is the solution of the equation

$$mnq^2 \cdot L^o \cdot p^{L^o-1} = 1 \tag{3.11}$$

[111]

Probe selection done via jumps in matching statistics measure oligo specificity using free energy. [111] $\delta G(o,T)$ denotes the "Gibbs free energy of an oligo o at temperature T." [111] The free energy is the measure of the oligos "tendency not to hybridize to its perfect match." [111] A negative value of the free energy thus says the oligo tends to hybridize to its match at the given temperature T. [111] The stability can be evaluated quickly. [111] The stability is denoted $-\Delta G(o,T)$. Of main interest here is the "stability of oligo-gene hybridizations $(o, g\prime)$, where $g\prime$, does not contain an exact match of o." [111] The "most stable perfect and near-perfect partial match of o and $g\prime$" are treated as if they were exact. The following is the estimation, where G_0 is $\Delta G(o^{-0}, T)$, and G_1 is $\Delta G(o^{-1}, T)$.

$$\Delta G(o, g\prime, T) \approx \min\{G_0, G_1 + \gamma\} \tag{3.12}$$

[111] This is synopsized by the measure of "the stability of the hybridization of $g\prime$ to o (is measured) by the lowest free energy of either a perfect partial match (G_0) or of a penalized near-perfect partial match ($G_1 + \gamma$)." [111] The quantification of the specificity is obtained through the difference between the estimate and the value of the free energy. [111]

The value of the estimation that is desired is one that leaves a large difference between the estimated and the known value of the free energy. Thus the oligos that are being sought are those for which the above is true. A preprocessing step is introduced to process the jumps. $ms^{g|g\prime;0}$ and $ms^{g|g\prime;1}$ are the list of jumps in both "for all other genes $g\prime \neq g$." [111] A threshold can be implemented to restrict the jumps. The threshold should be maximized. The computation need for the jump list is expensive, but must only be done once. [111] Finding the "gene-specific oligos in gene g, the following (is done to) each other gene $g\prime$ for exact matching statistics and one mis-matching statistics, and for each candidate oligo o from left to right." [111]

- Let i be the stating position of the current oligo o. for each gene $g\prime \neq g$, we do the following.

  1. Find $l = ms_o^{o|g\prime}$ from the first element of the $[i, |o|]$-jump set of $ms^{g|g\prime}$. This value is updated in constant time from the previous oligo. Evaluate the stability $-\Delta G(\bar{o}, T)$ of the partial match $\bar{o} = o_{o \cdots l-1}$.
  2. For each remaining jump $(i + k, J_{i+k})$ in the $[i, |o|]$-jump set , determine whether there is also a jump at position k in the substring matching statistics $ms^{o|g\prime}$. If yes, let $l = \min\{J_{i+k}, |o| - k\}$ denote the jump level and evaluate the stability $-\Delta G(\bar{o}, T)$ of $\bar{o} = o_{k \cdots k_l - 1}$.
  3. The most stable match found in this way is compared tot eh perfect match stability, and difference is recorded as the specificity of o with respect to $g\prime$.

- Oligo o is rejected when the specificity with respect to too many genes $g\prime$ is low.

[111]

It is suggested by [111] that the increase in time during the selection phase is worth while since the chip design is not time limited, when considering an increase by a factor of 2 or 3. It is unclear however, whether the energy based relation due to its dependence on sequence composition, would be an improvement and worth while. It is suggested that it be tested on a large scale.

3.5.2 Thermodynamic Approaches

A thermodynamic approach to probe selection is one in which the thermodynamic characteristics are considered in the probe selection process. Melting temperature is frequently one of the main considerations for selecting probes. The melting temperature is used to help reduce the number of cross hybridizations and tends to be the dominant methodology used for probe selection algorithms. The following subsections feature the work of [121], and [78].

Fast and Accurate Probe Selection Algorithm for Large Genomes: [121] Sung et al [121], proposes an algorithm for probe selection based on the "homogeneity, sensitivity, and specificity"; however, no heuristics are used which allows for greater accuracy. Homogeneity filter, sensitivity filter and specificity filter are used to eliminate unsuitable probes based the corresponding criteria. [121]

Homogeneity filter addresses the melting temperature, and "eliminates probes that hybridize at a temperature that is out of the experiment temperature range." [121] Next the sensitivity filter is applied and "eliminates probes with secondary structures." [121] Cross-hybridization is eliminated using the specificity filter. [121] To further understand the approach used by Sung et al [121] it is necessary to formally define the probe selection problem. "*Given a set of genes $G = \{g_1, g_2, \cdots, g_n\}$ and a parameter m which specifies the length of the probes, the probe design problem finds, for every gene g_i, a length m probe (that is substring of g_i) which satisfies (1) Homogeneity, (2) Sensitivity, and (3) Specificity.*" [121]

The first filtering applied is the "homogeneity filter" [121]. This filter considers the melting temperature in relation to the experiment temperature and the number of GC complexes. The number of the "GC rich complexes" should not be too high or too low due the increased stability of these bonds. [121] The melting temperature of the probes must fall inside some predefined range, and this melting temperature for the probe of length m is calculated using the "nearest-neighbor method." [121] The following is the equation for melting temperature used.

$$T_m(p) = \frac{\Delta H(p)}{\Delta S(p) + Rx\log(C_T)} - 273.15 + 16.6\log(Na) \tag{3.13}$$

In the previous equation R =molar gas constant, C_T =total molar concentration of the annealing oligonucleotides. [121] "Na is the salt concentration of the solution in which the oligomers are dissolved." [121] $\Delta H(p)$ and $\Delta S(p)$ are the enthalpy and entropy for helix formation of p respectively." [121] The "optimal hybridization temperature T_h of p is determined from $T_m(p)$ by" [121]

$$T_h(p) = T_m(p) - 25 - 0.62(C_f) \tag{3.14}$$

The calculation of the melting temperature and GC content requires only "one pass" through the probes "the homogeneity of a probe can be determined in linear time." [121]

The next filtering that takes place is the sensitivity filtering. This step in the filtering process determines if the probes contain secondary structures. If a probe does contain a secondary structure it is eliminated. The basic technique used is to take "a segment of length x from the $3\prime$ end of each probe, if it can form a consecutive length y complementary segment with itself, it is said to have a secondary structure." [121] The sensitivity filter runs in linear time.

The specificity filter is the next filter to be used to eliminate unsuitable probes. This is the most complicated filter. The specificity filter attempts to minimize "cross-hybridization of probes with other DNA sequences." [121] In order to determine cross-hybridization it must find "for each length -m probe p in gene g, whether there exists a length -m substring q in $G - \{g\}$ such that $H(p, q) <$ some threshold w. If such a q is found, then the probe p is said to be able to cross-hybridize with other genes." [121] Thus any probes meeting this criteria must be eliminated. To do the specificity filtering the "Pigeon Hole Principle" is used in an effort to "speed up the searching process." [121]

The following is the "key lemma" of the "Pigeon Hole Principle to determine whether a probe will cross-hybridize." [121]

Lemma 7 *If there exist length-m probes p and q such that $H(p,q) < w$, then we can find length-k substrings $p\prime = p[i \cdots i+k-1]$ and $q\prime = [i \cdots i+k-1]$ such that $H(p\prime, q\prime) \leq v$ where $v = [(w-1)\frac{k}{m}]$*

[121] The probes deemed to be "bad" can be filtered out using the following algorithm, which is based on the previous lemma. Also, "for any two length-k substrings s_1 and s_2 in the genome, ...they form a hit if $H(s_1, s_2) \leq v$ where $v = [(w-1)\frac{k}{m}]$...the set of hits for s can be found by enumerating all substrings in the genome that has Hamming distance $\leq v$ with s." [121]

Algorithm 1 *For every $g \in G$,*

- *For every hit between some length-k substring s_1 of gene g and some length-k substring s_2 of gene $g\prime \in G - \{g\}$,*
  - *For every length-m substring p and q which contain s_1 and s_2, respectively.*
    - *if $H(p,q) < w, p$ can cross-hybridize and we remove it from the candidate probe list.*

[121] In the worst possible case the "complexity of the basic algorithm is as bad as the brute force approach." [121] However, in practice the algorithm performs much better than the brute force algorithm. [121]

A further improvement can be made to the algorithm. The improvement is based on gapped hashing. [121] The previous lemma can be extended to the following.

Lemma 8 *If there exist length-m probes p and q such that $H(p,q) < w$, then we can find length-k substrings $p\prime = p[i]p[i+\frac{m}{k}]p[i+2\frac{m}{k}] \cdots p[i+(k-1)\frac{m}{k}]$, $q\prime = q[i]q[i+$*

$\frac{m}{k}]q[i + 2\frac{m}{k}] \cdots q[i + (k-1)\frac{m}{k}]$ *for* $i = 0, 1, 2, \cdots, \frac{m}{k} - 1$ *such that* $H(p\prime, q\prime) \leq v$ *where* $v = [(w-1)\frac{k}{m}]$

[121] This can be further improved by using preprocessing. [121] The hamming distance can be precomputed and stored. The extension to the lemma and preprocessing further reduce the complexity of the algorithm. [121]

Results show that the filtering makes it possible for this algorithm to be used for "finding oligo probes for long or short genomes." [121] "Divide and conquer" was used to overcome the difficulty of "genomes whose size exceeds the memory limit of a computer." [121] This leads to less restriction based on hardware constraints. [121] Further examination of methods that avoid heuristics should be considered, as the use of heuristics increase the efficiency and reduce the accuracy.

Selecting signature oligonucleotides to identify organisms using DNA arrays: Probes must be chosen for each sequence to be attached to an array. The selection of probes to effectively perform this task is of concern. Probes must be selected so that they do not hybridize with any other sequences. The probes are then attached to the surface and then hybridized to the sequence.

The formal problem is expressed by [78] as follows: *Given n target sequences* $t_1, t_2, \ldots t_n$*, find a temperature T and n probe sequences* $p_1, p_2, \ldots, p_n$ *such that:*

$$T_M(p_i, t_i) - \epsilon > T > T_M(p_i, t_i) + \epsilon \tag{3.15}$$

for all $k \neq z$*,* $i = 1, \ldots, n$*, where* $T_M(x, y)$ *is the temperature below which the two strands x and y are bound and above which they denature.* [78]

The proposed algorithm only considers probes that are perfect compliments to the sequence under consideration, probes that are unique, and then the remaining probes are evaluated for melting temperature using the Nearest Neighbor Thermodynamic Model. The algorithm requires that the following be known:

- Melting Temperature between all compliments.
- Which basepairs are going to form the duplex.
- The alignment resulting in the largest T_M.

[78] To minimize the amount of time to find the desired probes, all probes that can be eliminated due to reasons other than melting temperature are eliminated before the temperature is computed. This is because calculation of the temperature is costly.

The first step in the algorithm is the construction of a Suffix Tree (ST). [78] The ST contains the "inverse compliments of all target sequences." [78] The suffix tree is then used to find those prospective probes that do not meet the uniqueness requirement. The redundancy of the probes is exploited through the use of the ST. If a calculation of melting temperature has already been done on a subsequence of a probe then it does not have to be recomputed since the information is already stored in the ST. The probe information is stored in the suffix tree and then used to make probe selection suggestions. [78]

The T_M requires knowledge of the alignment. The alignment of sequences with a weight function $w(.,.)$ will be dealt with using a dynamic programming approach. Considering two sequences, "the general idea is to consecutively extend the alignment, starting with an alignment of prefixes of the two sequences x and y." [78] This algorithm is from Needleman-Wunsch. The modification of the algorithm for Kaderali's purposes is to "calculate ΔH and ΔS for all prefix-alignments, choosing the one resulting in the highest local melting temperature." [78] "The dynamic programming recursion is as follows:

$$\Delta H_{i,j} = \begin{cases} \Delta H_{i-1,j-1} + \Delta\Delta H(x_i, y_j) & \text{if t=0,} \\ \Delta H_{i-1,j} + \Delta\Delta H(x_i, -) & \text{if t=1,} \\ \Delta H_{i,j-1} + \Delta\Delta H(-, y_j) & \text{if t=2} \end{cases} \tag{3.16}$$

$$\Delta S_{i,j} = \begin{cases} \Delta S_{i-1,j-1} + \Delta\Delta S(x_i, y_j) & \text{if t=0,} \\ \Delta S_{i-1,j} + \Delta\Delta S(x_i, -) & \text{if t=1,} \\ \Delta S_{i,j-1} + \Delta\Delta S(-, y_j) & \text{if t=2} \end{cases} \tag{3.17}$$

and $t\epsilon\{0,1,2\}$ is to be chosen such that

$$T_M(i,j) = \frac{\Delta H_{i,j}}{\Delta S_{i,j} + R \ln \frac{C_T}{4}} \tag{3.18}$$

[78] Here "$\Delta\Delta H(x_i, y_j)$ and $\Delta\Delta S(x_i, y_j)$ denote the nearest neighbor parameters," when x and y are paired with '-' it represents the gap in the alignment. [78]

To further reduce the running time of the algorithm, computation is minimized further by only computing information once. Given the dynamic programming table "for two sequences," and that they need to be aligned. [78] If the two share a subsequence then they can share a sub-table. The shared sub-table part need not be recomputed or the other sequence. The generalized suffix tree is used to "identify common prefixes of substring pairs of all the different sequences." [78] The "defining property" of the suffix tree "is that each path from the root node to a leaf corresponds to a suffix of the string represented by the tree, and vice versa." [78] The construction of the suffix tree is linear. [78]

Probe pre-selection is done to minimize the cost of temperature calculations. The criteria used for the pre-selection follows:

- Probe Length - The assumption is made that variables minlen and maxlen and all probes must satisfy $minlen \leq probe \leq maxlen$
- Unique Probes - Only probes that are complementary to exactly one substring of all target sequences.
- Probe Melting Temperature - Minimum temperature that a probe-target duplex should be able to withstand.

[78] After constructing a suffix tree of the "compliments of all target sequences ... applying the above criteria and removing all infeasible probes from the tree is straightforward." [78]

The last step of the algorithm is to determine the melting temperatures of the "duplexes formed between all substrings left in the subtree and all target sequences." [78] The calculations of temperatures for some substrings can be reused to decrease running time in cases where the substring occurs in more than one sequence. By design the suffix tree causes grouping of the prefixes in such a way as to cause common prefixes to be grouped together. [78]

Once the melting temperatures have been calculated, the "objective now is to select a temperature T and one probe from the list for each of the target sequences, such that the array experiment can be carried out at temperature T, and the probes selected will hybridize only to their intended target sequence." [78] The selection of the temperature T, must adhere to the following condition from Kaderali. *"Given n DNA or RNA target sequences $t_1, t_2, \ldots, t_n$, given furthermore for each target sequence t_i a finite set of probe sequences P_i where $P_i \bigcap P_j = 0$ for all $i, j; i \neq j$. Furthermore given for all target sequences t_i and all probe candidates $p_j \epsilon \bigcup_{k=1}^{n} P_k$ the melting temperatures $T_M(t_i, p_j)$ at which target t_i and probe p_j dissociate. For each probe $p_k \epsilon P_i$, and for each target sequence t_i find a temperature T.*

$$T_M(t_i, p_k) \geq T > T_M(t_j, p_k) for all i \neq j \tag{3.19}$$

The idea here is to sort all probes according to descending melting temperature, and consecutively move down the scale removing all those "probes that will cross-hybridize at the new temperature." [78] This is an iterative process and continues "until...a feasible, unambiguous probe is found for every target or until all probes have been removed." [78]

The algorithm has been tested on "randomly generated sequences of different lengths." [78] However, effective analysis of the running times could not be obtained because the complexity and running time are heavily dependent on the selection of "filtering criteria." [78] This selection of criteria is a problem facing algorithms in this area.

3.5.3 Other Approaches To Probe Selection

This section features areas related to probe selection, but not necessarily algorithms for probe selection. [117] offers a parallel architecture to help with probe selection demands, while [67] offers a linear algebra solution to the probe selection problem. Finally, [87] gives an entropy estimator for consideration. While all these are related to probe selection they are not actual algorithms for solving the probe selection problem via a statistical method or thermodynamic method.

Real-Time Primer Design for DNA Chips [117] Simmler et al [117], offers a parallel architecture to assist in overcoming the hurdles associated with the complexity of the probe selection algorithms and the hardware constraints faced. The main concentration is that of primer design.

The hybridization conditions or "parameters taken into account for selecting each optimal primer set have major influence on the quality of the hybridization process." [117] Primer length, melting temperature (calculated the same as in [121]), GC content, secondary structures (specifically self annealing, self end annealing, pair annealing, and pair end annealing) are all parameters to be considered.

The primer length is the "amount of bases that build the biological primer." [117] Therefore the length is a definition of the "selectivity of the primer." [117] The melting temperature is considered to make sure the prospective primer has a melting temperature that falls within the predefined temperature range of the experiment. Due to the stability of the GC pairs, it is also critical to limit the number of these bonds present. Finally the consideration of secondary structures is the most complex. There are several "criteria used for the detection of secondary structure effects", these will be explored next. [117]

The following "score function" [117] is used to compare the "primer sequences and examines the possibility of a hybridization to itself or another primer." [117]

$$s(p_i, q_j) = \begin{cases} 2, & \text{if } \{p_i, q_j\} = \{A, T\}; \\ 4, & \text{if } \{p_i, q_j\} = \{C, G\}; \\ 0, & \text{else.} \end{cases} \tag{3.20}$$

[117] uses p and q to represent to primers, that are being "analyzed for secondary structures." [117] The secondary structure possibilities can be further broken down into sub areas with different calculations for each.

The first to consider is when the primer hybridizes with itself. This is referred to as self annealing (SA). [117] To make the SA calculation the primer and the "opposite version of the same primer" is used. [117] The "calculation of the SA score starts with the left shifted opposite primer, where only one overlapping position with the original primer exists. It compares each single overlapping position using the score function and accumulates each single score values to an alignment score...the opposite primer is shifted one position at the right end of the original primer." [117] The SA function follows:

$$SA(p, q) = \max_{k=-(n-1),\cdots,m-1} \sum_{i=1}^{n} s(p_i, q_{i+k}) \tag{3.21}$$

The next to consider is the self end annealing(SEA). The difference between the SA and the SEA calculation is that the SEA calculation "considers only those alignments where the 3′ end of the original primer belongs to the overlapping region." [117] The other difference is the "SEA score is accumulated only for these overlaps which are continuous." [117]

Pair annealing(PA), "takes the interaction of different primers into account and calculates all possible primer pairs." [117] The last to consider is the pair end annealing(PEA). "PEA is similar to SEA and evaluates all possible binding starting from the 3′ end of the primer." [117]

The previously discussed methods are "combined into a scoring vector." [117] The following is the scoring vector.

$$sc_{PCR}(p,q) = (length(p), GC(p), T_m(p), sa(p), sea(p),$$
$$length(q), GC(q), T_m(q), sa(q), sea(q), pa(p,q), pea(p,q)).$$

[117] The ideal score vector follows.

$$sc_{PCR,ideal} = (length_f, GC_f, T_{M,f}, 0, 0, length_r, GC_r,$$
$$T_{M,r}, 0, 0, 0, 0)$$

[117] The "melting temperatures, and GC content are set to the forward and reverse primer." [117] "The deviation between the calculated value and the given value is computed for each parameter of the scoring...each is then weighted." [117]

l_{PCR} is the quality score and is calculated as follows.

$$l_{PCR} = \sum_{i=1}^{12} k_i |sc(p,q)_i - (sc_{ideal})_i| \tag{3.22}$$

[117] The scoring vector and the ideal vector for the chip are as follows.

$$sc_{Chip}(p) = (lenght(p), GC(p), T_m(p), sa(p), sea(p)) \tag{3.23}$$

[117]

$$sc_{Chip,ideal}(p) = (length_f, GC_f, T_{m,f}, 0, 0) \tag{3.24}$$

[117] The "distance l_{Chip} is used for the selection the optimal primer is calculated" as follows. [117]

$$l_{Chip} = \sum_{i=1}^{5} k_i |sc(p,q)_i - (sc_{ideal})_i| \tag{3.25}$$

[117]

The calculations for the quality score are extremely expensive in relation to computation time. Therefore Simmler et al [117] attempts to lessen the complexity through a "primer design architecture." [117] The architecture suggested by [117] is a parallel architecture.

The first part of the quality score calculation is that of computing "several parameters for the given sequence pair," [117] while the second part is the calculation for the distance values. [117] The parameters that comprise the first part of the calculation include the "length, GC content, the melting temperature, and SA SEA, PA and PEA value." [117]

The first and second calculation portions of the quality score are independent of one another and can thus be computed via a parallel architecture. [117] The first architecture used to perform the first part of the quality score calculations will compute the "length, GC content, melting temperature, SA score, and SEA score." [117] "These values are calculated for each single primer defined by the length range and the window length." [117] "A sub score sc_1 is calculated for each primer." [117] The primers are then filtered and selected, and stored "using the window position, the sub score, and the length of the primer." [117]

The optimal primer pair is then selected, after primers are computed for both windows. [117] The second architecture will calculate the PA and PEA values. [117] The calculations of the PA and PEA values is done in parallel. [117] The parallel approach decreases processing time. [117] "The results are compared to the given ranges, multiplied with the given weight and combined to the sub score sc_2." [117] "The quality score is calculated using this sub score sc_2 and adding the sub scores of the forward primer and the reverse primer." [117] The best primer is then selected via a filter. [117]

The goal of [117] is to propose a method to "accelerate the execution time of a primer design application." [117] This is accomplished, and the complexity of the calculations are reduced to a level of acceptability. Since the main computation is that of the annealing value, the complexity could be further reduced if the cost of the annealing value could be further reduced. [117] It is also stated by [117] that "config assembly or a precise primer based database search (might) also gain (from) using an annealing matrix." [117]

Linear Reduction Methods for Tag SNP Selection: He et al, propose a linear algebra solution to the tag SNP selection problem. The "tag SNP selection problem" is stated below. [67]

Problem 1 *Given the full pattern of all SNP's for a small sample, select minimum number of tag SNP's that will allow to reconstruct the full data set, i.e., reconstruct any haplotype from tag SNP's.*

[67]

P is defined to be a population of haplotype vectors, n as the number of sets of haplotype vectors $H = h_i | i = 1, \cdots, n$. [67] Each H has m coordinates.[67] "Each of n haplotype vectors corresponds to a haplotype drawn from the population P and each of m positions corresponds to a SNP site in a haplotype." [67]

"Tag SNP's are k position-sites" $t_1, t_2, \cdots, t_k, t_i \in i, \cdots, m$. [67] According to He et al, it is possible to reconstruct a haplotype from "its tag SNP vector-like value" $h_k = (h_{t_1}, \cdots, h_{t_k})$. [67] The "statistical tag SNP selection and haplotype reconstruction problem (STTS)" is as follows:

Problem 2 *Given a set of n haplotype vectors H on m sites and $k < m$, find k tag sites $t_1, \cdots, t_k$ and a reconstruction function $f = (f_1, \cdots, f_m)$, such that for any haplotype $h \in P$ expected Hamming distance between h and its prediction $\bar{h}$ is minimized.*

[67]

The formulation given in this chapter does not "specify any restrictions on what tag SNP's can be used when the value of a particular SNP is decided." [67] The assumption in He et al's paper is that the population P is "a set of already sequenced haplotypes." [67] Below is the "new optimization problem formulation." [67]

Problem 3 *Given population as a set P of p haplotypes on m sites, a population sample $H \in P$ of n haplotypes and an integer $k < m$, find k tag sites $t_1, \cdots, t_k$ and a reconstruction function $f = (f_1, \cdots, f_m)$, such that average Hamming distance $|h, \bar{h}|$ nryerrnsny haplotype $\frac{h \in P}{H}$ and predicted haplotype $\bar{h} = f(h_k)$ is minimized.*

[67]

It is common if sites are synonymous (which occurs frequently) to drop all but one that are the same since no additional information is included. For He et al's [67] needs, a more general statement can be made. If there are k columns that are "dependent", then k^{th} site can be dropped. "Two sites are synonymous if and only if they are collinear." [67]

Theorem 4 *Let H be a set of haplotypes obtained from two haplotypes by recombination events at g sites. Then the linear rank of H, rank(H)$\leq g + 2$.*

[67]

The steps for linear reduction proposed in He et al's paper are the following.

- *Form the population sample H extract $r = rank(H)$ of sites $T(H) = (t_1, \cdots, t_r)$ forming a basis of columns-sites.*
- For each column-site in f_j,$j = 1, \cdots, m$ in H find a unique representation $f_j = \sum_{i=1}^{r} a_{i,j} h_(t_i)$
- *Output the set of tag SNP's $T(H)$ and the reconstruction function $f = (f_1, \cdots, f_m)$.*

[67]

The implementation of the reduction can be done using Gaussian elimination. This is efficient. Using Gaussian elimination the echelon form of the matrix can be found and it will be called $H\prime$. "Let F be the matrix $H\prime$ in which zero rows are dropped, so F is an rxm matrix...for any haplotype h with the tag SNP values h_r, the predicted reconstruction $\bar{h} = f(h_r)$ equals:

$$\bar{h} = h_r F \tag{3.26}$$

[67]

There are two linear reduction implementation compared. The first is "where the SNPs are taken in the order in which they are given in H" and the second is the Randomized LR (RLR) where it is randomized. [67] Experimental results of comparing the 3RLRP (Randomized LR with Post Processing with 3 times more SNP's), RLRP, RLR, and LR, show that for a population of size 1000, the error is dependent on the size of the population, and "the number of tag SNP's is always close to the size of the sample." [67]

Further work is planned in order to "apply tag selection to genotype data." [67] There is also some interest by [67] to "explore different possibilities of combining (the method) with block methods." [67]

Estimating DNA Sequence Entropy: The goal of [87] is to present an entropy estimator. The previous entropy estimators used the probability of "n-tuples for large n" [87], however, these estimates converged too slowly to be a good estimator for the entropy. [87]

Other proposed methods include "Biocompress", which is inaccurate due to the additional expense of compression. Thus the estimates are always too large. Another previously suggested was the "match length entropy estimator". [87] The problem with this estimator is that is assumes the sequence is generated by Markov. CDNA is another program to estimate the entropy, but convergence has not been proven. However, in the paper presented by [87], an entropy estimator is developed called GTAC(Grammar Transform Analysis and Compression).

GTAC is universal, and based on the grammar based codes. The comparisons done with GTAC and other estimators are based on the following guidelines:

- *Is the code universal with respect to any stationary source? That is, will the entropy estimate converge to the actual entropy if the sequence is long enough? A limited one, such as the Match Length entropy estimator must make the addition assumption that the source is a Markov process.*
- Is the run time linear?
- *How good are the algorithm's entropy estimates?*

[87]

A brief background in context free grammars, and grammar based code is presented to lay the ground work for an entropy estimator based on the context free grammars, and grammar based code (proposed by Kieffer and Yang). The definition of a context free grammar (CFG) is $G = (V, T, P, S)$. V is " a finite non-empty set of variables, T is a finite non-empty set of terminal symbols that is disjoint from V, S is a distinguished element of V called the start symbol, and P is a set of production rules which map elements of V onto $V \cup T$)." [87]

Kieffer and Yang's proposition is that a sequence x is "transformed into a CFG G_x from which x can be fully recovered." [87] G_x is then compressed, using "arithmetic coder". [87] In order to retrieve x, G_x must adhere to the following rules:

- *the language generated by G_x consist of only x.*
- G_x is deterministic, that is, any variable in V appears only once on the left had side of the production rules, P

- *P does not contain the empty string on the right hand side of any rule.*
- G_x has no useless symbols. That is, during the process of deriving x from the production rule corresponding to the start symbol S, each production rule in G_x is used at least once.

[87] If G_x adheres to the rules then it is classified as an "admissible grammar". [87]

$w(G_x)$ is called the sequence that is "obtained by concatenating the right hand side of all production rules of G_x in some order and then deleting the first appearance of each variable." [87]

$$H(G_x) = \sum_s n(s) \log \frac{|w(G_x)|}{n(s)} \tag{3.27}$$

[87] In the above equation $n(s)$ is the number of times s appears in $w(G_x)$. s is either a variable or terminal symbol.

Theorem 5 *According to arithmetic coding or enumerative coding, one can assign a uniquely decodable binary codeword $B(G_x)$ to each admissible CFG G_x(or its equivalent form) such that:*

$$|B(G_x)| = f(G_x) + H(G_x) \tag{3.28}$$

where $|B(G_x)|$ denotes the length of the binary codeword $B(G_x)$, and $f(G_x)$ represents the overhead paid to the universality of grammar based codes. In (2), $f(G_x)$ is negligible as compared to $H(G_x)$ and is upper bounded, in the worst case scenario by :

$$f(G_x) \leq 5|G_x| + \alpha \tag{3.29}$$

where $|G_x|$ denotes the total entities in the right hand side o all production rules of G_x, and α is the cardinality of the source alphabet and is 4 in the case of DNA sequence.

[87] From the above theorem it can be seen that the grammar transform should provide $H(G_x)$ of G_x and $f(G_x)$ are small.

Irreducible grammars produce "efficient universal compression algorithms." [87] Therefore, the following theorem:

Theorem 6 *For any sequence x, let $\zeta(x)$ be the set consisting of all irreducible grammars G representing x. Then the following hold:*

- *There is a constant c, which depends only the cardinality of the source alphabet, such that for any sequence x*

$$max_{G\in\zeta(x)}|G| \le \frac{c|x|}{\log|x|} \tag{3.30}$$

  where $|x|$ denotes the length of x.

- *For any stationary, ergodic source $X_{i}{}_{i=1}^{\infty}$ with entropy H, the quantity:*

$$max\{|\frac{|B(G)|}{n} - H| : G \in \zeta(X_1 \cdots X_n)\} \tag{3.31}$$

  goes to 0 with probability one as $n \to \infty$.

[87] This leads to the following remark.

Remark 1 *Part b of Theorem 2 represents the worst case scenario. The actual convergence rate at which*

$$\frac{|B(G_{x^n})|}{n} - H \tag{3.32}$$

where G_{x^n} is an irreducible grammar representing $X^n = X_1 \cdots X_n$, goes to 0 depends on the source $X_{i}{}_{i=1}^{\infty}$ and the irreducible grammar transform $X^n \to G_{x^n}$.

[87]

The suggested estimator uses a normalized grammar entropy $\frac{H(G_x)}{|x|}$ of G_x.

Theorem 7 *Let $X_{i}{}_{i=1}^{\infty}$ be any data source. Then for a constant $d > 0$, the following holds with probability at least $1 - n^{-d}$:*

$$\frac{|H(G_{x^n})|}{n} \ge \frac{-1}{n}\log P(X^n) - \frac{f(G_{x^n})}{n} - \frac{d\log n}{n} \tag{3.33}$$

for any grammar transform $x \to G_x$, where $X^n = X_1 \cdots X_n$ and $P(X^n)$ denotes the probability of X^n.

[87] This theorem leads to the following remark.

Remark 2 *From information theory one can interpret $\frac{(-\log P(X^n))}{n}$ as the entropy in bits per letter of X^n. From theorems 1, and 2 it follows that for irreducible grammar transforms, $\frac{f(G_{x^n})}{n}$ is quite small and upper bounded, in the worst case scenario by* $\mathrm{O}(\frac{1}{\log n})$. Therefore, theorem 3, says that with a high probability, the entropy estimators associated with grammar based codes with irreducible grammar transforms will never severely underestimate the actual entropy.

[87]

The algorithm for GTAC is based on solving the "longest non-overlapping patter (LNP) problem." [87] The problem can be formalized as follows:

Problem 4 (LNP Problem) *Given a set of strings, P, find the longest substring β such that β occurs in at least two non-overlapping positions somewhere in P.*

[87] Adding the following constraints we can define GTAC.

- *$G = (V, T, P, S)$, when we let P be the set of all right hand sides of the production rules P.*
- The length of β is at least two.

[87]

The basic description of the steps of GTAC is that the algorithm finds and reduces the LNP repeatedly, and in the process it creates a new rule. The following more explicitly describes the algorithm based on if "an LNP β appears in the following form:" [87]

- $A \rightarrow \alpha_1 * \beta * \alpha_2 * \beta * \alpha_3$

  Then rewrite the previous rule as two rules.

- $A \to \alpha_1 * B * \alpha_2 * B * \alpha_3$

  $B \to \beta$

  In an LNP β appears in different rules,

- $A \to \alpha_1 * \beta * \alpha_2$

  $B \to \alpha_3 * C * \alpha_4$

  then rewrite the previous rules and introduce a new one as follows.

- $A \to \alpha_1 * C * \alpha_2$

  $B \to \alpha_3 * C * \alpha_4$

  $C \to \beta$

[87] GTAC also have the ability to recognize reverse complements which of integral necessity for an estimator. Reverse complements are handled by having two sets of non-terminals. One set are just non-terminals, the others are the reverse complement. These sets are denoted $A_1, \cdots$ and $R_1, \cdots$ respectively. The algorithms deals with reverse complements as follows. When given an input, the algorithm creates a grammar. GTAC then finds the LNP. If "there is two or more occurrences of β, create a rule $A_i \to \beta$ ignoring any occurrences of $\bar{\beta}^r$(reverse compliment of β). If there is only one occurrence of β and one of $\bar{\beta}^r$, then create a rule using one of the reverse complement non-terminals, $R_i \to \beta$, which means interpret the second occurrence of the non-terminal in the right hand side of a rule as the reverse complement." [87] After the preceding has taken place the entropy can be calculated. [87] The following is the algorithm to implement GTAC.

```
• GTAC(x) begin
  create rule $S \to x$ ;
  $T$=new suffix tree(x) ;
  md = max depth(T) ;
  Q[] = new array of queries(size=md);
```

```
• for each interior node n of T do
  if(n is an LNP)
  add n to Q[depth(n)];
  end while;

• for l=md down To 2 do
  while (Q[l] is non-empty) do
  n=pop(Q[l])
  B=new non_terminal

• for each $\beta$[] = path to node n do
  p[] = 2l chars to left of $\beta$[] in y;
  for i=1 to 2l do

• if(suffix(p[i]) contains $\beta$[1] in T)
  remove suffix in T after p[2l];
  end for;

• for i=1 to l do
  if(suffix($\beta$[i]) goes beyond $\beta$[l] in T)
  remove suffix in T after $\beta$[l]
  end for;

• replace $\beta$ with B in rules;
  end for;

• create rule $B \rightarrow \beta$;
  end while;
  end for;
  estimate entropy based on grammar;
  end alg;
```

[87]

The running time of the algorithm runs in linear time. Comparing GTAC to Bicompress-2 shows that GTAC always gives better performance. In eight of the ten results GTAC beat CDNA. Work is projected to attempt to combine GTAC with other methods, since other approaches to estimating entropy currently converge too slowly. [87]

3.6 Physical Design

This is the main step of the flow. Its inputs are pools of probes from the probe selection step. In this step also, even though we face the conflict problem of the nucleotides, the Chip program introduces new algorithms to reduce the number of conflicts when we place and embed the probes. The Chip program also gives running time for each algorithm. Physical design for DNA arrays is equivalent to the physical design phase in VLSI design. It consists of four steps: deposition sequence design, which is a basic optimization in DNA array design, is minimizing the number of synthesis steps, or, equivalently, minimizing the number of photo-lithographic masks used in the manufacturing process; test control, which is equivalent of built-in self-test (BIST) structures in VLSI design, and aim at detecting catastrophic manufacturing defects, i.e., defects that irrevocably compromise the functionality of the DNA chip. Additionally, DNA chip designs incorporate control structures for ensuring reliable interpretation of results; probe placement, which is responsible for mapping selected probes onto locations on the chip, and probe embedding, which embeds each probe into the deposition sequence (i.e., determines synthesis steps for all nucleotides in the probe). The result of physical design is the complete description of the reticles (photomasks) used to manufacture the microarray.

Deposition Sequence Design: Fundamental optimization in DNA array design is minimizing the number of synthesis steps, or, equivalently, minimizing the number of photolithographic masks used in the manufacturing process. Current methodologies use a predefined deposition sequence, typically periodic. Chip authors propose to optimize the deposition sequence with respect to a given set of selected probe pools, and add a feedback loop to provide updated design rules and parameters to the probe selection step. So, the number of synthesis steps affects manufacturing time and the cost of the mask set, and also directly affects the quantity of defective probes synthesized on the chip. Therefore, a basic optimization in DNA array design is to minimize the number of synthesis steps. In the simplest model, this optimization has been reformulated as the classical shortest common super sequence (SCS) problem [[82], [123]]

2-D Probe Placement: Under ideal manufacturing conditions, the functionality of a DNA array is not affected by the placement of the probes on the chip, or the particular order in which nucleotides of each probe are synthesized. In practice, since manufacturing process is prone to errors, probe placement and synthesis schedules affect to a great degree the hybridization sensitivity and ultimately the functionality of the array. There are several types of synthesis errors that take place during array manufacturing.

3-D Probe embedding: Recently, in [79], Kahng et al introduced the border minimization problem for the asynchronous synthesis regime, which allows arbitrary probe embeddings. Asynchronous synthesis has identical technological requirements with synchronous synthesis. Asynchronous synthesis is already mandated by minimization of the number of synthesis steps. At the same time, asynchronous embedding offers more flexibility for reducing total border length.

The physical design problem of DNA chips is similar to that faced by VLSI design. [12] Attempting to maximize the number of probes present on a chip while minimizing the chip size. According to [12] there are four main steps in the physical design portion of the design flow.

- Deposition Sequence Design
- Test Control
- Probe Placement
- Probe Embedding

[12] The four steps can be further explained as the minimization of ”synthesis steps,” [12] for deposition sequence design. Test control is said by [12] to be analogous to ”built-in self-test structures in VLSI design.” [12] The probe placement and probe embedding are placement of probes on the chip, and how the probes adhere to the chip. The result of the physical design is the information needed for manufacturing. [12]

3.6.1 Manufacturing:

In this step, DNA array goes through a combination of photolithography and combinatorial chemistry process, resulting in many of the arrays' powerful capabilities. With a calculated minimum number of synthesis steps, DNA arrays with hundreds of thousands of different probes are packed at an extremely high density. This feature enables researchers to obtain high quality, genome-wide data using small sample volumes. Manufacture is scalable because the length of the probes, determines the number of synthesis steps required. This automated production process yields arrays with highly reproducible properties, which reduces user set-up time by eliminating the need for individual labs to produce and test their own arrays.

3.6.2 Evaluation of Placement Techniques for DNA Probe Array Layout

[81] The main focus of [81] work is to apply the concepts of VLSI design to DNA probe placement. The primary concern is minimizing ”total border cost.”[81]

"Portioning based algorithms" are proposed, as well as a "simple in-place probe re-embedding algorithm," an experiment is also conducted to evaluate the probe placement currently available, and the ones given by [81].

The main problem addressed is the border cost problem. According to [81] the problem is formally defined as "finding a three-dimensional placement of the probes: two dimensions represent the site array and the third dimension represents the nucleotide deposition sequence S." [81] Border length is computed as the "number of conflicts , i.e. pairs of adjacent exposed and masked sites in the mask." [81] The conflict distance between two probes is the number of "conflict between the corresponding columns." [81] The total border length is the sum of the conflict distances between pairs of probes. [81]

The partitioning algorithm proposed by Kahng et al [81]is based on the portioning used for VLSI design. The algorithm is a new "centroid-based quadrisection method that applies the recursive partitioning paradigm to DNA probe placement." [81] A probe set R is quadrisecting into the following partitions "R_1, R_2, R_3, R_4," [81] and each probe p that is in the probe set R is assigned to a partition of R. The partition of R chosen is based on minimizing the conflicts. The conflict is calculated between p and a "centroid." [81] There is one centroid for each of the four partitions of R. The centroids are chosen from R based on their total distance to each other. The four probes of R with the greatest distance are assigned as centroids of R. The following is the centroid selection process formally defined.

- Input: Partition(set of probes) R
- Output: Probes C_0, C_1, C_2, C_3 to be used as centroids for the 4 subpartitions
  - Randomly select probe C_0 in R
  - Choose $C_1 \epsilon R$ maximizing $d(C_1, C_0)$
  - Choose $C_2 \epsilon R$ maximizing $d(C_2, C_0) + d(C_2, C_1)$

```
    – Choose $C_3 \epsilon R$ maximizing $d(C_3, C_0) + d(C_3, C_1) + d(C_3, C_2)$
    – Return $\{C_0, C_1, C_2, C_3\}$
```

[81]

After the centroid selection is completed, and the max depth is reached, a placement step is done. The placement step places the partitions probes into the section of the chip corresponding to the partition. The following is the complete "partitioning based placement algorithm for DNA arrays." [81]

```
• Input: Chip size $SxS$; set $P$ of DNA probes
• Output: Probe placement which heuristically minimizes total conflicts
    – Let $l = 0$ and Let $L = maximumrecursiondepth$
    – Let $R^l_{1,1} = P$
    – For $l = 0$ to $L - 1$
        ∗ For $i = 1$ to $2^l$
            · For $j = 1$ to $2^l$
            · Let the set of (next-level) subpartitions be $R_{next} = \{R^{l+1}_{2i-1,2j-1} = 0, R^{l+1}_{2i-1,2j} = 0, R^{l+1}_{2i,2j} = 0\}$
            · SelectCentroid ($R_{next}$)
            · For all probes $p \epsilon R^l_{i,j}$
            · Insert $p$ into the yet-unfilled partition of $R_{next}$ whose centroid has minimum distance to $p$
    – For $i = 1$ to $2^L$
        ∗ For $j = 1$ to $2^L$
            · Rept$x(R^L_{i,j}, R^L_{i,j+1})$
```

When comparing the new re-embedding algorithm "given a two-dimensional probe placement, improves the embedding of the probes without re-placing," to the chessboard and the batched greedy algorithm the following was noted. [81] The sequential new algorithm takes a different outlook from the chessboard and greedy. Instead of re-embedding "an independent set of sites on the DNA chip," it is proposed by Kahng et al [81] that "dropping this requirement permits faster propagation of the effects of any new embedding, and hence convergent to a better local optimum." [81] It is shown that the "re-embedding of the probes in a sequential row-by-row order leads to a reduction in the border cost by 0.8% compared to the chessboard algorithm." [81] A current open problem given by [81] is that of "developing a tighter lower bound." [81]

3.6.3 A Design Method of DNA chips for SNP Analysis Using Self Organizing Maps

[48] To solve the problem of chip size Douzono et al [48] uses self organizing maps (SOM). The goal is to "obtain common features of DNA sequences with small number of probes which efficiently cover the target sequence with sufficient resolution for finding the correct position of SNPs." [48]

To attempt to minimize the size of the DNA chip, SOM was employed to select probes. SOM will work for long sequences, which is untrue for the clustering approach. [48] SOM has been explored before in reference to DNA chips, but this work uses SOM for finding probes "which were sufficient to represent the target sequences, detecting feature of DNA sequences." [48] The reason for choosing SOM is based on the fact that "SOM can organize the generic feature of the DNA sequences by sufficient learning of known DNA sequences." [48] The following is the algorithm used to "train the self organizing map of fixed length probes." [48]

1. Initialize the map of probes using random sequences of specified length.

2. Select a position of reference sequence (RS) randomly and find the closest probes on the map to the sub-sequence which starts from that position.

3. Update the closest probe Pr found in step 2 as follows.

    - For each symbol in the probe
    - If the symbol is A (G,T,C)
    - Then modify the value Pr. A(G,T,C) = Pr.A(G,T,C)+1
    - If Pr.A(G,T,C)¿Th-U
    - Then update the symbol to A(G,T,C) and set all Pr.A(G,T,C) to 0.

4. Update the probe Pr' whose distance is closer than M-Dist from Pr geometrically using the same procedure in step 3.

[48] "Repeat steps 2-4, changing the value Th-U and M-Dist." [48] Pr.A(T,C,G) denote the "intermediary value which are introduced to gradually update the discrete values." [48] If the threshold(Th-U) is reached the updating occurs. [48]

In order for a map to be trained, " a reference sequence whose length are sufficiently long to detect the common feature of DNA sequences," [48] were used. [48] "made SNP analysis using each set of the probes obtained form each map," after training. [48] The "rates detecting SNPs of all single nucleotide changes," [48] where calculated. It was determined that longer probes give worse coving rates. This is due to the fact that longer probes require more probes. It was also determined that the algorithm presented by [48] is high in calculation time and is not suitable to "train larger maps." [48]

The algorithm using SOM by [48] can in fact select small numbers of probes that are in fact "representative of the feature DNA sequence for SNP analysis." [48] Thus it is determined to be acceptable for SNP analysis, but unacceptable for "sequencing by hybridization." [48] This is true because an insufficient number of probes are organized by the algorithm. This is one area where the algorithm needs improvement. "For sequencing by hybridization not enough probes will hybridize to target sequences." [48] In order for the algorithm to be used on larger maps, improvement needs to be made in the algorithm. [48]

3.6.4 Soft Lithography for Oligonucleotide Arrays Fabrication

[68] design a chip fabrication method for DNA chips. The method is based on a chemistry protocol. The overall goal is to reduce the expense, and increase the accuracy and reliability of the chips. [68]

There are currently two approaches to probe fabrication. [68] The "sequential individual probe fixation," is useful for "low-density arrays." [68] However, the "on-chip synthesis" is useful for high-density arrays. [68] The method used by [68] is of the "on-chip synthesis" type. The method uses molecular stamping or soft lithography to "fabricate the oligonucleotide arrays." [68]

The steps needed to perform soft lithography begin when the surface of the "substrate is treated so that it could bind single nucleotides." [68] The following are the steps necessary to conduct the "stamping coupling." [68] "The mixed acetonitrile solution with nucleoside monomer and tetrazole as reactants is spread on features of the modification stamp, then transferred onto the modified substrate surface by machine alignment stamping until acetonitrile is vaporized to nearly dryness,...the nucleoside monomer on features of the stamp is coupled with the predefined regions on the substrate." [68] Therefore, the all spots on the chip except those which will have an A are covered by the first stamp. The second stamp allows sites to couple with T. The third is for C, and the fourth is for G. After four stampings, "oxidation, capping and detritylation are conducted." [68] The next layer then precedes in the same fashion.

[68] tested the soft lithography method by adhering the "same oligonucleotide sequence... to different feature of the ...slide." [68] The results of the test showed that the method was acceptable, and that it was "steady, and could be reused." [68]

3.7 Hybridization and Analysis

In this step, hybridization experiments are performed. During the hybridization experiment label of each probe is quantified, and probes are diluted so that all are at an equal concentration. Usually, a duplicate filter or micro-array is prepared for each probe to be assayed. Probes are hybridized separately with each array. Filter arrays are incubated with probe and washed in much the same way as is done for Southern or Northern blotting. For glass microarrays, hybridization is done under a cover slip. Dipping into wash solutions washes slides along with cover slips. Commercially produced arrays come in cassettes, in which hybridization, washing, and detection is done.

3.8 Experimental Study

Herpes B virus is a member of the subfamily Alphaherpesvirinae from the genus Simplex virus. Herpes B (HB) virus is mild localized or asymptotic infection in its internal hosts [106]. In contrast HB virus infection in foreign host, humans or monkeys species other than macaques often result in encephalitis, encephalomyelitis, and death. Herpes B virus genome is 156,798 bps long and includes 74 genes [106]. In this study, we design a chip to carry on experiments to study HB virus. The chip should contain 20 to 50 probes for each of 74 genes. To do that we follow the following step: First, we extract the genome sequence from GenBank using BioPerl [22] tools by giving the genome id to Genbank [61]. After getting the genomic data, we feed it to GenMark [61], which gives a set of ORF for Herpes B Virus. The ORF generated from GenMark is going to be the input for Probe Selection step, Promide [110]. However, the format provided by GenMark is not suitable for Promide input; therefore we use our parser Parser1 as tool to change the format of the input. Parser1 get as input the ORF's name, their left and right physical position. After parsing the data, Parser1 produces a set of ORF in FASTA format. The ORF generated form Parser1 will be provided to Promide for probe selection [110] that is to generate a set of pools of probes. Since melting temperature play very important role in DNA arrays, Promide gives a freedom of choosing the melting temperature for probes. Checking melting temperature [116] for probes will be offered by Sigma-Genosys. We use the temperature 60 and 65 as best melting temperature to carry on our experiments. [[110], [78]], the generated pool of probes from both temperatures has the same targets but the average pool size for each temperature is different. The pool of probes spawned will be nosh to the Chip program [79]. However, the pool of probes given is not in the proper format of Chip's input. For this reason we create the second parser for DNA array flow, "Parser2". Parser 2 will read probes from Promide- which in the form of ACTG- and output them in the format-0123- where 0 stand for A, 1 for C, 2 for T and 3for G. Beside converting the format of Promide, our parser can choose from the pool of probes the number of candidate for each probe, which make a powerful tools when we want to change the chip size. The pool of probes produce by Parser will be given to Chip program to produce a chip and compare the number of conflict [80] for each algorithm, CPU time usage of Herpes B virus and simulated data. We will repeat the experiment for different chip size, temperature and number of candidate chosen from each probe. The Chip Program does not allow us to read a real data [80]; it uses random generated pool of probes. We added a module to the Chip program that can read data from a file "Readpool". This module read real data file chooses a number of candidate wanted for each probe. Readpool model measures a number of conflicts between probes and gives CPU time for each algorithm. The goal of this study is to design a chip for HB virus. To measure the quality of our design we have to minimize

Algorithm	Herpes B Virus		Simulated Data	
(K=2)	# Conflicts	CPU Time(sec)	# Conflicts	CPU Time(sec)
Initial	107577		265992	
Tsort	98830	0.17	231526	0.08
Tsp	95640	0.22	227960	0.09
Lalign	79254	0.25	189272	0.1
Reptx 2	64830	4.45	154766	1.58
Chessboard	63594	15.58	150812	7.1

Table 3.1. DNA Flow Results, K=1.

Algorithm	Herpes B Virus		Simulated Data	
(K=2)	# Conflicts	CPU Time(sec)	# Conflicts	CPU Time(sec)
Initial	54205		265328	
Tsort	49746	0.3	232954	0.14
Tsp	48541	0.34	227762	0.15
Lalign	42858	0.42	182972	0.16
Reptx 2	32098	7.84	149332	3.16
Chessboard	31498	20.93	146708	10.89

Table 3.2. DNA Flow Results, K=2.

the number of conflict between the probes. For this reason we perform the following experiments in HB virus using the chip program. The results of the experiment, using following parameters, are presented in Tables 1 and 2. Melting Temperature: In our experiment, we choose 65 C as the melting temperatures for our DNA probe array. Number of Candidates (K): We experimented with different values of K (number of candidates) for each pools of probes: 1 and 2. Chip Size: We ran our Experiments with chip size 60x60. Pool Size: In order to design a chip of 60x60, we selected 47 probes from each set of probes. The algorithms in Chip Program as well as Promide were implemented in C. The parser1 was implemented in C++; however, parser 2 was implemented in Perl. We run all the code on Linux server. The experiments were run on randomly generated data and herpes B Virus genome data.

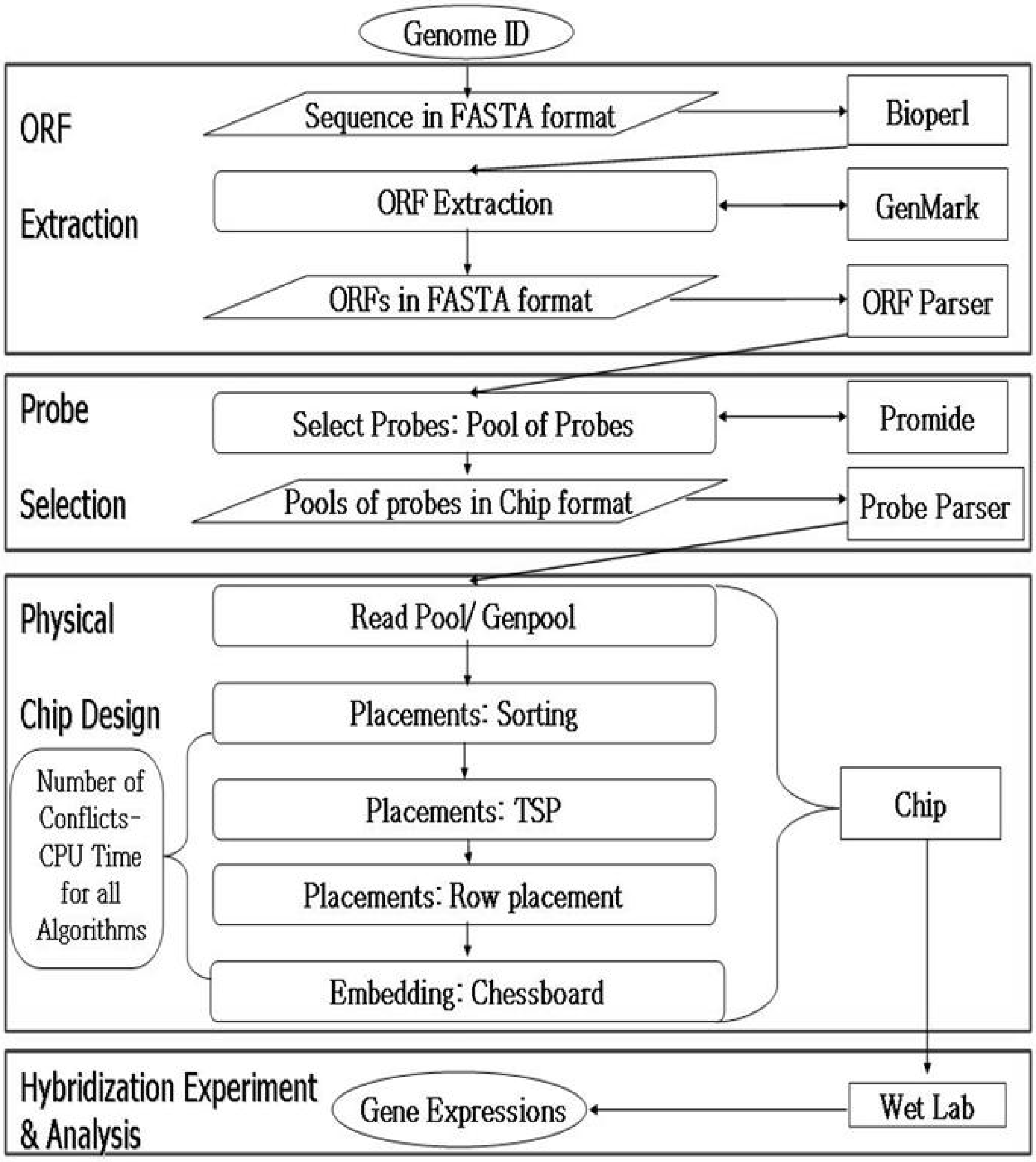


Figure 3.1. Detailed DNA Array Design Flow

CHAPTER 4

UNIVERSAL TAG ARRAY (UTA) DESIGN

Universal Tag Arrays (UTA) consist "of a set of DNA strings called tags, designed such that each tag hybridizes strongly to its antitag (Watson-Crick Complement)" and will not hybridize strongly to any other antitag. [72] Hundewale et al [72] report that "sample analysis is typically performed by a sequence of hybridization and single-base extension reactions involving reporter probes consisting of application specific primers ligated to antitags." [72] UTAs have a defined design flow. Hundewale et al [72] built on the work of Atlas et al [12] and defined this design flow as consisting of five steps. The first step is the "reading of genomic data," then the "open reading frame (ORF) extraction." [72] The next step is probe selection, then tag assignment, and finally "hybridization experiment and analysis." [72] Below is the flow provided by [72].

The two design areas for UTAs that are reviewed in this chapter are probe selection, and tag assignment.

4.1 Probe Selection

The Probe selection step is the step that defines the purpose of the array. [12] According to Atlas et al [12] the "challenge for probe selection is how to identify the optimum probes for each gene." Probe selection for UTAs is the same as for DNA arrays. Therefore, it is clear that the same problems are faced in probe selection for UTAs as was seen for DNA arrays. The methods discussed in the previous probe selection section are applicable here as well.

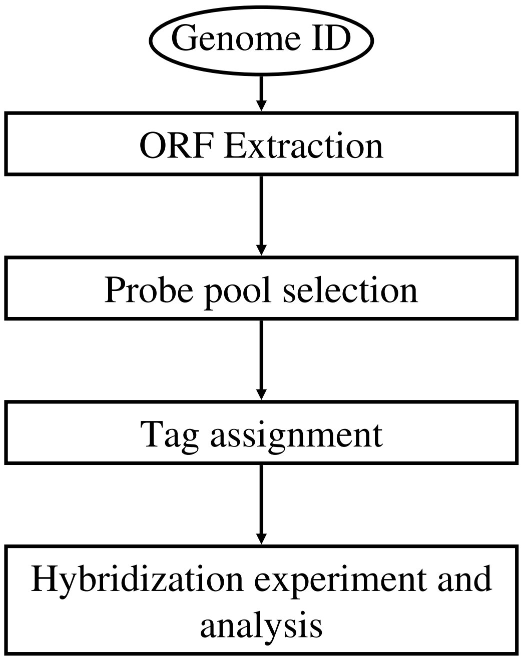


Figure 4.1. DNA Universal Tag Array Design Flow

Atlas et al [12] gives the following criteria for probe selection: ”probe should be a unique sequence in the all-coding sequences of the target genome, should not be self-complementary, be close to the 3énd of that gene, does not contain single nucleotide multiple repeat regions, does not a representative of low complexity region and should have almost the same GC percentage as that of the target genome.” [12]

4.2 Tag Assignment

Tag assignment is the assignment of tags to primers. This is a complicated issue due the fact that undesired hybridizations between primers and tags can occur. [72] The tag assignment problem is stated by Hundewale et al [72] as the assignment of anti-tags/tags to primers.

Ben-Dor et al [19] presents a combinatorial tag design scheme, and later addresses assay specific sources of error in [20]. Mandoiu et al [93], suggests an improved tag set design and multiplexing algorithm for universal arrays, and later in [94] suggests an exact and approximation algorithm for DNA tag set design.

4.2.1 Universal DNA Tag System: A Combinatorial Design Scheme

. Ben-Dor et al [19] in their paper address the problem of containing the maximum number of probes in an array while minimizing cross-hybridization. Ben-Dor et al [19], formalize the problem using a thermodynamic model of hybridization. [19] It is also proven in their paper that the combinatorial approach is "near-optimal" for construction. [19]

The melting temperature of a duplex is when half of the duplexes are hybridized and half are melted (single strand form).[19] This melting temperature can be used to define the hybridization affinity of two oligonucleotides. [19] Using temperature parameters C and H, then $(C < t < H)$. [19] "if a duplex has a melting temperature of at most C, at most a fraction of the duplexes will form. Similarly, if a duplex has a melting temperature of least H, then at temperature t, at least a fraction of the duplexes will form." [19] This leads to the formalization of the design goal. [19] The goal is to create a tag/antitag system such that:

- *for each tag U,* $t(U, \bar{U}) \geq H$ *and*
- *for any two distinct tags U and V,* $t(U, \bar{V}) < C$

[19] 't' is the melting temperature, and C and H are temperatures. This would prevent cross-hybridization [19] The 2-4 rule is used for estimating the melting temperature. [19] The 2-4 rule gives the melting temperature to be 2 times the number of A-T bases and 4 times the number of C-G bases. [19]

Formally the design problem according to [19] is to create a tag/antitag "system with a maximum number of tag-antitag pairs such that the following are satisfied:" [19]

- *for each tag-antitag pair* $(U, \bar{U})$ *the melting temperature (using the 2-4 rule) satisfies* $t(U, \bar{U}) \geq H$
- *for any two distinct tags U and V and for each oligonucleotide x that occurs as a substring in both U and V,* $t(x, \bar{x}) < C$

[19] All strings are assigned numbers equal to half their melting temperature. [19] The following definition from [19] formalizes the weights given to each string.

Definition 6 *The weight w(s) of a string* $s = a_1 a_2 ... a_k$ *is* $\sum_{i=1}^{k} w(a_i)$ *where w(A)=w(T)=1 and w(C)=w(G)=2. Given two parameters c and h, we call a set T of strings or "tags" a valid c-h code if the following two conditions are satisfied.*

[19]

- Condition 1 *Each tag has a weight of h or more*
- Condition 2 *Any substring of weight c or more occurs at most once*

[19] The "problem of finding a maximum valid c-h code" [19] is called the "Combinatorial Tag Design Problem". [19] The upper bound of the number of tags in a valid c-h code is defined as the global upper bound divided by the minimum amount of resource used by each tag. [19] The resource considered here is made up of those substrings that are of weight c or more. [19] These substrings can only occur once in a valid code. [19] These substrings are called c-tokens and are defined by [19] as follows:

Definition 7 *We call a string t a c-token if* $w(t) \geq c$*, but does not properly contain a suffix of weight* $\geq c$

[19] The tail weight of a T is the sum of all the tail weights of all the c-tokens contained inside the tag. [19] The c-token tail weight is the weight of the last character of the c-token. [19]

Lemma 9 *Any tag in a valid c-h code has a till weight of at least h-c+1*

[19] The upper bound on the tail weight of valid c-h code is based on conditions 1 and 2, and "we use $< n >$ to denote the set of strings with weight $n \in N$, and G_n, to denote the number of such strings. [19] It is straight forward to derive the recurrence $G_1 = 1$, $G_2 = 2$, and $G_n = 2 * G_{n-2} + 2 * G_{n-1} for n \geq 3$." [19] The maximum total weight of the tokens is derived from the tail weights of the different classes of tokens. [19] Using 'S' to denote strong characters (those characters with a weight of 2, either C or G), and using 'W' to denote weak characters (those characters with a weight of 1, either A or T). [19] The following table are the classes of tokens.

	Max. occurrences	**Max.**
Token class	**in valid code**	**tail weight**
$< c-2 > S$	$2 * G - c - 2$	$4 * G_{c-2}$
$S < c-3 > S$	$4 * G - c - 3$	$8 * G_{c-3}$
$< c-1 > W$	$2 * G - c - 1$	$2 * G_{c-1}$
$S < c-2 > W$	$2 * G - c - 2$	$4 * G_{c-2}$

[19]

This table leads to the following theorem that provides the upper bound.

Theorem 8 *any valid c-h code contains at most* $\frac{2*G_{c-1}+6*G_{c-2}+8*G_{c-3}}{h-c+1}$ *tags.*

[19] There are two parts to the construction, the first stage is to create a circular string of tokens. [19] These tokens must only occur once. [19] The second portion of the construction is to extract the tags from the circular string. [19] The extracted tags must meet condition 1 and must have a weight of h or more. [19] In order to satisfy condition 2 the weight of the overlap between tags must have a weight of at most c-1. [19]

The second portion of the construction can be done using a greedy algorithm. [19] Iterating through the circular string and finding the first tag of weight h or more, then moving backward in the circular string until the weight of c or more is reached and them moving forward one to begin the next tag. [19] Each tag will have a weight of h+1 at most, and the overlap between tags is at least c-2. [19] If we call the circular string C then we are in search of the at least $\frac{w(C)}{h-c+3} - 1$ tags. [19] This is the lower bound for the number of tags. [19]

Definition 8 (Circular String Problem) *Given the parameters $c > 0$ and $h > c$, construct a set C of circular strings that contain any substring of weight $\geq c$ at most once, and maximize*

[19]

$$\sum_{C \in c} \left(\frac{w(C)}{h-c+3} - 1\right) \tag{4.1}$$

[19] Encoding is given to each character in order to assign each character a meta character and a bit. [19] A=(W,0), T=(W,1), C=(S,0), and G=(S,1) are the assignments for the meta characters and bits respectively. [19] Each string is assigned a meta-string and a bit string, based on the encoding given to each character. [19] Every circular string is a meta-string that is composed of a repeated meta-string. [19] It is guaranteed to not contain two of the same tokens since the meta-string will be paired with a different sequence of the bit-string. [19] DeBruijn sequence is a binary sequence in which each substring only occurs once. [19] These sequences are constructed in linear time and it is assumed that a "DeBruijn sequence of order k is given for each $k \in N$." [19] Starting from some fixed position with in the sequence and labelling it the origin gives a string denoted by D_k^2. [19]

If the meta-string contains the following properties:

- gcd($|\mu|$ +1, $2^{|\mu|}$)=1

- μW cannot be represented as a concatenation of two or more identical sub-strings.

[19] the cycle will be:

$$C_o(u) = ((\mu * W)^{2^{|\mu|}}, (D^o_{|\mu|})^{|\mu|+1}) \tag{4.2}$$

[19] A more general case of the cycle is if α is the shortest period of μW, and set k = k(μ) = gcd($|\alpha|$, $2^{|\mu|}$). "For any meta-strings μ with w(μ)=c, our code contains the k cycles:" [19]

$$C_i(\mu) = ((\alpha)^{\frac{2^{|\mu|}}{k}}, (D^2_{|\mu|})^{\frac{|\alpha|}{k}}), 0 \leq i \leq k \tag{4.3}$$

[19] The cycles constructed are:

$$C := \bigcup_{w(\mu)=c} \bigcup_{i=0}^{k(\mu)-1} C_i(\mu) \tag{4.4}$$

[19]

All cycles are combined into one cycle before tags are extracted. Two cycles can be pasted together if they share a "common substring s with a weight of c-1". [19] The following two Lemmas explain the pasting method.

Lemma 10 *For any two cycles A and B that share a common substring of weight c-1, past (A,B) contains exactly the union of the tokens contained in B.*

[19]

Lemma 11 *There exists a sequence of past operations that merges all cycles of C into a single cycle.*

[19]

Proof 1 *The central observation is that each cycle $C_i(\mu)$, where μ is a meta-string containing $k \geq 1$ strong characters, can be pasted with a cycle of the form $C_j(v)$, where v contains only k-1 strong characters. To see how this works, observe that the* *circular meta-string of $C_i(\mu)$ can be expressed as a repetition of μ'S for some meta-string μ' of weight c-1. Consider an instance s of μ' in $C_i(\mu)$. The string sT is a token of weight c and , according to Lemma 6, does occur in some cycle $C_j(v)$ with v=μ'W. Observe that the meta-string v contains only k-1 strong characters. Since both cycles $C_i(\mu)$ and $C_j(v)$ contain s as a substring, they can be pasted together.*

[19] This proof and the two previous Lemmas give the following theorem.

Theorem 9 *The above construction yields at least*

$$\frac{2 * G_{c-1} + 6 * G_{c-2} + 4 * G_{c-3}}{h - c + 3} - 1 \tag{4.5}$$

tags.

[19]

[19] claim that the Circular String Problem is solved optimally. Using the following Lemmas the optimum performance of the solution to the Circular String Problem is proven. [19] Allowing r to equal c/2, and assuming c to be even and C' to be the set of cycles. [19] A bound is placed on the number of weak characters and the number of strong characters. [19]

Lemma 12 *for $k \in 0, \cdots, r-1$, the number of instances of the meta-string s^kW in C' is at most $2^k G_{2(r-k)}$.*

[19]

Lemma 13 *C' contains at most 2^r instances of the meta-string S^r*

Lemma 14 *The number of strong characters in C' equals the number of instances of the meta-string S^kW (over $k = 1, \cdots, r-1$), plus the number of instances of S^r.*

[19] The next Lemma shows that the upperbound is in fact G_{2r-1}.

Lemma 15

$$\sum_{k=1}^{r-1} 2^k G_{2(r-k)} = G_{2r-1} \tag{4.6}$$

[19] This leads to the following theorem.

Theorem 10 *The total weight of any valid set of cycles is at most* $G_c + 2 * G_{c-1}$

[19]

This theorem proves that it is in fact optimal. However, it must be validated that the assumption by [19] that "violations are infrequent" is in fact true in order for the method to be useful. [19] Choosing of antitags for screening is also a persistent problem. [19]

4.2.2 Optimally Multiplexed Applications of Universal DNA Tag Systems

. The major focus of [20] is to show ways of avoiding cross-hybridization caused by assay specific components. Primer to anti-tag cross-hybridization is the focus of the paper by Ben-Dor et al [20]. The goal is more clearly stated as an attempt to "maximize the multiplexing rate for a given set of SNPs, under given primer to antitag cross-hybridization constraints."[20] Ben-Dor et al [20] "choose how to partition the set of SNPs into assignable subsets and to assign tags to SNP sites" in order to "control the multiplexing rate." [20]

The multiplexing problem can be modelled using a bipartite graph.[20] One side of the graph is the primers and the other side is the tags. [20] Each edge is a possible cross-hybridization. [20] The cross-hybridization is between a primer and an antitag. This means the problem is really "the problem of covering the primer vertices, using a minimum number of subgraphs of max degree one." [20]

In order to formally define the problem it is necessary to label some variables. The tag sequence will be denoted as T, the antitags that correspond to the tags will be denoted as $\bar{T}$. P is the set of primers, $m = |P|$, $n = |T|$.[20] The graph will be called G, the subset of the vertices will be R. The subgraph of G "induced by R" is

G_R.[20] $V(G)$ is the set of vertices, $E(G)$ is the set of edges of G.[20] An assumption is made that all cross-hybridization is represented in binary matrix form.[20] Letting A be a binary $mbyn$ matrix, "such that:"

$$A_{p,t} = \begin{cases} 1 & \text{if } p \in P \text{ potentially hybridizes with } \bar{t} \in \bar{T}, \\ 0 & \text{otherwise.} \end{cases} \tag{4.7}$$

[20]

Definition 9 *A set of reporter molecules $(p_1, t_1), \cdots, (p_k, t_k)$ (with distinct $p_i \in P$ and distinct $t_j \in T$) is said t be non-cross-hybridizing if $A_{p_i,t_i} = 0$ for all $i \neq j$.*

[20]

Definition 10 *A set of k distinct primers $p_1, \cdots, p_k \subseteq P$ is called assignable if there exists a non-cross-hybridizing set of reporter molecules $\{(p_1, t_1), \cdots, (p_k, t_k)\}$ for k distinct tags $t_1, \cdots, t_k \in T$.*

[20]

Definition 11 *A sub-permutation matrix is a square 0-1-matrix whose rows and columns can be permuted such that all entries outside the main diagonal are* 0

[20]

Observation 1 *A set of primers $\acute{P} \subseteq P$ is assignable if and only if $\acute{P}$ corresponds to the row set of a sub-permutation sub-matrix of A.*

[20] From the above observation, another observation can be made from viewing A as a bipartite graph, and $G = (P, T, A)$.[20] If A is a bipartite graph then P are the vertices of primers, T are the vertices that are tags, and the edges represent the cross-hybridization.[20] Furthermore, " a subgraph $H = (\prime P, \prime T, \prime E)$ of G is called balanced if $|\prime P| = |\prime T|$. H is an assignable subgraph if H is a balanced induced subgraph of maximum degree 1." [20]

Observation 2 *A matrix A with a set of rows P and a set of columns T is a sub-permutation matrix if and only if the bipartite graph $G = (P, T, A)$ is an assignable graph.*

[20] These observations lead to the formal statement of the problem.[20]

Problem 5 *Minimum Primer cover (MPC). Given a bipartite graph $G = (P, T, A)$, find a minimum primer cover of P.*

[20]

Ben-Dor et al [20], prove that the MPC problem is NP-complete, "even if the number of tags is required to be greater or equal to the number or primers." [20] Every primer can have "edges with at most d tags." [20] This is due to the fact that every primer in the primer set P must be bound by some constant d.[20] This bounding is due to the fact that primer substrings of any excessive length would lead to cross hybridization.[20] d is some constant that bounds the degree of every $p \in P$. [20] the "optimum solution of cardinality" is "at least 2", for the input MPC instance. For d=1, "polynomial for $m \leq n$." [20]

Lemma 16 *Let $G = (P, T, A)$ ba 1-bounded instance of MPC, with $m \leq n$. Then a minimum primer cover for G can be found in polynomial time.*

[20] The result here is a "polynomial algorithm", that will produce a solution "of cardinality at most $\frac{\frac{m}{n}}{d+1}$ for a d-bounded input instance." [20]

Theorem 11 *Let $G = (P, T, A)$ be a d-bounded input instance of MPC. Then we can find, in polynomial time, a solution to MPPC on G of cardinality at most $\frac{\frac{m}{n}}{(d+1)}$*

It can be noted here that for $m \leq n$ the approximation ratio is $\frac{\frac{\frac{m}{n}}{d+1}}{2}$, and for $m > n$ "at least $\frac{m}{n}$ subgraphs are needed in order to cover the primer set." [20]

Ben-Dor et al [20], next addresses the greedy approach to MPC. The greedy algorithm works by finding the largest assignable subset and removing it recursively.[20] The downfall to this is the fact that it is NP-hard.[20]

Problem 6 *Maximum Assignable Primer set (MAP). Given a bipartite graph G, find a maximum assignable subgraph of G.*

[20] As stated earlier the MAP problem is NP-hard and is proven to be so by Ben-Dor et al [20].

"Portioning a bipartite graph into vertex-disjoint assignable subgraphs that cover the set of primers" can be useful only when "the number of primers is at most the number of tags." [20]

Problem 7 *(Minimum Partition into disjoint Assignable Subgraphs (MPDAS)). Given a bipartite graph $G = (P, T, A)$, find a minimum set of vertex-disjoint assignable subgraphs that cover P.*

[20] The problem of MPDAS is also a NP-complete problem.[20]

Theorem 12 *Let $G = (P, T, A)$ be an input bipartite graph in which the degree of each $p \in P$ is bounded by d, and $m \leq n$. Then we can find, in polynomial time a solution to MPDAS on G of cardinality at most $2D$.*

[20] If the number of tags is greater than the number of primers, smaller covers can be produced. [20]The suggestion of Ben-Dor et al [20] is that MPDAS is useful for "multiplexing the solution-phase experiments, it is possible to perform the genotyping using a single array, at the cost of performing slightly more solution-phase experiments." [20] This is in reference to the method of "introducing blocking oligonucleotides", in between the extension and hybridization steps. [20]

The first algorithm to address MPC is based on the theoretical analysis, and the second algorithm is based on the set cover approximation.[20] Built in the first algorithm is a cover " with size at most $\frac{\frac{m}{n}}{d+1}$, for any set of primers with degree bounded by d.[20]

Algorithm 2 *1.* $\varepsilon \leftarrow 0$

```
2. Unmark all vertices of $T$
3. Sort the tags in $T$ in non-decreasing order based on their degrees in $G_{P \cup T}$
4. $\acute{T} \leftarrow 0$
5. While there are unmarked tags do:
    (a) Find an unmarked tag $t \in T$ or $t \in \acute{T}$ with lowest degree
    (b) Mark $t$
    (c) If $\acute{T} \cup t$ is assignable then $\acute{T} \leftarrow \acute{T} \cup t$
6. Find a set $\acute{P}$ of —$\acute{T}$— primers that form a non-cross-hybridizing set with $\acute{T}$
7. $\varepsilon \leftarrow \varepsilon \cup \acute{P}$ (add $\acute{P}$ to the cover)
8. Update $P \leftarrow P$ or $P \leftarrow \acute{P}$
9. If $P = 0$ then halt else go to step 2
```

[20] The second algorithm needs some clarification for step number 3. In step number 3 a primer is removed the method by which the primer is selected is based on the primer's potential. [20] The potential is the sum of the potentials of the neighbor tags. [20] The primer that is removed is the primer with the maximum potential.[20] The potential of a tag of degree w is 2^{-w}.[20] The additional rule of subtracting $\frac{1}{2}$ is used when a primer is next to a tag of degree 1.[20] The following is the second algorithm based on the set cover.

Algorithm 3

```
1. $\varepsilon \leftarrow 0$
2. $\acute{P} \leftarrow P$
3. While $\acute{P}$ is not assignable remove a primer of maximum potential from $\acute{P}$.
4. $\varepsilon \leftarrow \varepsilon \cup \acute{P}$ (add $\acute{P}$ to the cover).
```

5. *Update* $P \leftarrow P$ *or* $P \leftarrow \acute{P}$.

6. *If* $P = 0$ *then halt else go to 2.*

[20]

The second algorithm based on set cover outperformed the first algorithm.[20] This was true for all simulations. [20] There are many assay specific sources of error including primer to antitag cross-hybridization, sandwich cross-hybridization, primer to primer mis-extension, and primer to tag mis-extension. [20] only address the problem of primer to antitag cross-hybridization. Thus the other are still possible sources of error and should be researched more completely for possible solutions.

4.2.3 Improved Tag Set Design and Multiplexing Algorithms for Universal Arrays

. Mandoiu et al, address two problems with tag set design. The first problem addressed is the tag set design problem that was defined by [19]; however here [19] work is extended to include the problem of anti-tag to anti-tag hybridization. The second problem addressed is the tag assignment problem. Mandoiu et al [93] attempts to improve the multiplexing rate. The goal of improving the multiplexing rate comes from the fact that it is not "possible to assign all tags to primers in an array experiment due to unwanted cross-hybridizations." [93]

More generally the universal array tag set design problem is maximizing the tags.[93] If the number of tags is maximized then more "reactions ... can be multiplexed using a single array." [93] In order for the assay to function correctly the following constraints are placed on the tags and antitags.

- *(H1) Every antitag hybridizes strongly to its tag;*

- *(H2) No antitag hybridizes to a tag other than its complement and*

- *(H3) There is no antitag to antitag hybridization.*

[93]

As was used in [19], "hybridization affinity" is approximated using the 2-4 rule.[93] However, the constraints on tags has been modified from that defined by [19] . The departure from [19] is that [19] allowed tags of unequal length.[93] However, [93] requires that C1, C2, and C3 be met. The following describes the requirements for valid tags.

Definition 12 *For given constants l, h, and c with $l \leq h \leq 2l$, a set of tags $T \subseteq A, C, T, G$ is called valid if the three following conditions are satisfied:*

- (C1) Every tag in T has a weight h or more
- (C2) Every DNA string of weight c or more appears as substring at most once in the tags of T
- (C3) If a DNA string x of weight c or more appears as a substring of a tag, then $\bar{x}$ does not appear as a substring of a tag unless $x = \bar{x}$.

[93] Formalization of the tag set design problem is given in the following:

Problem 8 (Universal Array Tag Set Design Problem:) *Given constants l, h, and c with $l \leq h \leq 2l$, find a tag set of maximum cardinality.*

[93] A new upperbound is calculated based on the new constraints of the tags, and a greedy algorithm for creating tag sets is proposed.[93]

Using c-tokens and tail weights described by [19] , the following Lemma defines the number of c-tokens that are allowed in a valid tag.

Lemma 17 *Let $c \geq 4$. Then the total number of c-tokens that appear as substrings in a valid tag set is at most $3G_{c-2} + 6G_{c-3} + G_{\frac{c-3}{2}}$ if c is odd, and at most $3G_{c-2} + 6G_{c-3} + \frac{1}{2}G_{\frac{c-3}{2}}$ if c is even. Furthermore, the total tail weight of c-tokens that appear as substrings in a valid tag set is at most $2G_{c-1} + 4G_{c-3} + 2G_{\frac{c-3}{2}}$ if c is odd and at most $2G_{c-1} + 4G_{c-3} + G_{\frac{c-3}{2}} + G_{\frac{c-4}{2}}$*

[93]

Theorem 13 *For every l, h, c with $l \leq h \leq 2l$ and $c \geq 4$, the number of tags in a feasible tag set is at most. min*$\{\frac{3G_{c-2}+6G_{c-3}+G_{\frac{c-3}{2}}}{l-c+1}, \frac{G_{c-2}+6G_{c-3}+\frac{1}{2}G_{\frac{c-3}{2}}}{h-c+1}\}$

for c odd and at most
min$\{\frac{G_{c-2}+6G_{c-3}+\frac{1}{2}G_{\frac{c-3}{2}}}{l-c+1}, \frac{2G_{c-1}+4G_{c-3}+G_{\frac{c-3}{2}}+G_{\frac{c-4}{2}}}{h-c+1}\}$

for c even.

[93]

The greedy algorithm to construct the tags, begins with an empty set of tags.[93] During each iteration the next letter in the DNA alphabet is added.[93] If the added letter completes a c-token that has already been used in the current tag or in a previous tag the letter is removed and the next letter in the alphabet is tried.[93] When there are no further choices of letters backtracking is done to begin the sequence of trying to add a letter again.[93] This continues until a complete tag is formed.[93] Once a tag is generated the tag is saved, and "backtrack to the last letter of its first c-token." [93]

To avoid the formation of nucleation complexes that may produce undesired hybridization, primers are assayed using large number of arrays and the assignment of antitags to primers adhere to the following constraints.[93]

- *(A1) If primer p forms the configuration where a primer and a tag other than the compliment of the ligated antitag, then antitag $\bar{t}$' is not assigned to any primer in array experiments in which p is assayed, unless it is assigned to p itself.*

- *(A2) If primer p forms the configuration where a primer and an antitag are hybridized, then antitag $\bar{t}'$ is not assigned to any primer in array experiments in which p is assayed (this time assigning $\bar{t}'$ to p is not allowable).*

- *(A3) If primers p and p' form the configuration where two primers hybridize, then they are assayed on different array experiments.*
- *(A4) Antitag $\bar{t}$ is never assigned to primer p if they form the configuration in which two reported probe substrings hybridize and at least one of which straddles a ligation point, with t'=t.*
- *(A5) If antitag $\bar{t}$ is assigned to primer p in an array experiment, and the resulting reporter probe forms the configuration in which two reporter probe substrings hybridize and at least one of which straddles a ligation point with t'$\neq$ t, then antitag $\bar{t}$' is not assigned to any primer in that experiment.*

[93] The following is the formalized multiplexing problem.

Problem 9 (Universal Array Multiplexing Problem) *Given primers $P = p_1, \cdots, p_m$ and tag set $T = t_1, \cdots, t_m$, find a partition of P into the minimum number of assignable sets.*

[93] Since in most cases there are multiple primers available, multiplexing rates can be increased by combining the primer selection and tag assignment.[93] This maximizes the hybridization patterns available.[93] Proposed recently in other research is a modification of primer selection tools.[93] The idea is that the tools will return "pools".[93] These "pools" will contain all primer candidates.[93] A primer pool is "assignable if we can select a primer for each pool to form an assignable set of primers."[93]

Problem 10 (Pooled Universal Array Multiplexing Problem) *Given primer pools $P = P_1, \cdots, P_m$, and tag set $T = t_1, \cdots, t_m$, find a partition of P into the minimum number of assignable sets.*

[93] [93], only addresses solving the pooled multiplexing problem when (Al) is enforced. The authors purport that for the constraint (A2) the solution for (A1) can be extended, and for constraints (A3-A5) the best solution is to re-assign primers

violating the constraints.[93] The algorithm for solving the multiplexing problem for only (A1) constraint being enforced follows.

Algorithm 4 (Primer pool assignment algorithm) *Input:* Primer pools $P = \{P_1, \cdots, P_m\}$ and tag set T

*Output*Triples (p_i, t_i, k_i), $1 \leq i \leq m$, where $p_i \in P_i$ is the selected primer for pool i t_i is the tag assigned to p_i and k_i, is the index of the array on which p_i is assayed

```
$k \leftarrow 0$
*While* $P \neq 0$ *do*
$k \leftarrow k + 1$
$P\prime \leftarrow P$
*While* $|X(P\prime)| + |Y(P\prime)| < |P\prime|$ *do*
*Remove the primer p of maximum potential from the pools in* $P\prime$
*If p's pool becomes empty then remove it from* $P\prime$
*End While*
*Assign pools in* $P\prime$ *to tags on array k using Lemma 2*
$P \leftarrow \frac{P}{P\prime}$
*End While*
```

[93]

Lemma 18 *A set P of primer pools is assignable iff* $|X(P)| + |Y(P)| \geq |P|$

[93]

For each iteration of the algorithm Lemma is checked, to see if it is satisfied.[93] If Lemma is not satisfied then a primer of maximum "potential" is deleted.[93] If the last primer is deleted then the pool is deleted from the set of pools.[93]

The result of using the greedy algorithm for tag selection, using constraint (C3) halved the number of tags selected compared to only using (C1) and (C2).[93] The results for the integrated primer selection and tag assignment, showed marked improvement in multiplexing rates, this was achieved with the pooling aware algorithm.[93] None of the pool aware algorithms seems to consistently out perform the others, so the best suggestion is to run all and choose the one that gives the best solution.[93] The multiplexing rate could be examined and evaluated to see if improving the rate is possible.

4.2.4 Exact and Approximation Algorithms for DNA Tag Set Design

. Mandoiu et al [94] focus on solutions for the design of tag sets for universal tag arrays. [94] give an "integer linear programming formulations for two previous formalizations of the tag set design problem."[94] Also presented is a correlation between the tag set design problem and the "problem of packing the maximum number of vertex-disjoint directed cycles in a given graph." [94]

The design of tag sets contains two types of constraints.[94] The first type of constraint is the stability constraint, and the second type is the non-interaction constraint.[94] The hybridization model used by Mandoiu et al [94] is the model used by Ben-Dor et al [19] . The model used by Ben-Dor et al [19] is the c-token hybridization model. The most basic description of the c-token model is that hybridization only occurs if "one oligo contains as substring the complement of a substring of weight c or more of the other, where c i8s a given constant." [94]

The stability constraint used by Mandoiu et al [94] is both the predetermined length for tags and matching lengths (l, where $l = 20$) for tags as used by Affymetrix's GenFlex arrays, and the variable length tag with minimum length h (h here is a predefined constant), presented by Ben-Dor et al [19] .[94]

The non-interaction constraints used by Mandoiu et al [94] are both of the following, denoted as C and $\bar{C}$ respectively.

(C) *"For every feasible tag set τ, let $N_\tau(x) \leq 1$ for every DNA string x of weight c or more."* [94]

$(\bar{C})$ *"For every feasible tag set τ, $N_\tau(x) + N_\tau(\bar{x}) \leq 1$ for every DNA string x of weight c or more."* [94]

The notation used by Mandoiu et al [94] is for the previously defined constraints is $\gamma\epsilon 1, multiple$ specifies "whether or no the c-token uniqueness constrain is enforced." [94] $\alpha\epsilon l, h$ specifies which tag length constraint is used, and $\beta\epsilon C, \bar{C}$ specifies if the anti-tag/tag constraint is used or the anti-tag/tag, and anti-tag/anti-tag constraint is used.[94] The "maximum tag set design problem with constrains α, β, γ, is denoted MTSDP($\alpha|\beta|\gamma$)." [94]

Other notations used by Mandoiu et al [94] are the following:

- "Let N denote the number os c-tokens". [94]
- "$C = c_1, \cdots, c_N$ denote the set of all c-tokens". [94]
- "Let $C_0 \subseteq C$ denote the set of c-tokens of weight $c + 1$ that end with a weak base (of the form $S < c - 2 > W$)" [94]
- "Let $C_2 \subseteq C$ denote the set of c-tokens of weight c that end with a strong base (of the form $< c - 2 > S$)" [94]

[94]

The first integer linear programming formulation is that of MTSDP($l|C|1$).[94] The "ILP (integer linear programming) formulation uses an auxiliary directed graph

$G = (V, E)$ with $V = s, t \cup \bigcup_{1 \leq i \leq N} V_i$, where $V_i = v_i^k || c_i| \leq k \leq l$." [94] "$G$ has a directed arc from v_i^k to v_j^{k+1} for every triple i, j, k such that $|c_i| \leq k \leq l - 1$ and c_j is obtained from c_i by appending ta single nucleotide and removing the maximal prefix that still leaves a valid c-token." [94] "G also has an arc from s to every $v \epsilon V_{first}$, where $V_{first} = v_i^{|c_i|} | c_i \epsilon \frac{C}{C_0} \cup v_i^{|c_i|+1} | c_i \epsilon C_2$, and an arc from v_i^l to t for every $l \leq i \leq N$." [94]

The following is the integer program:

$$maximize \sum_{v \epsilon V_{first}} (x_v) \tag{4.8}$$

$$subjecttox_v = \sum_{e \epsilon in(v)} (y_e) = \sum_{e \epsilon out(v)} (y_e), (\frac{v \epsilon V}{s, t} \tag{4.9}$$

$$\sum_{v \epsilon V_i} (x_v \leq 1), 1 \leq i \leq N \tag{4.10}$$

$$x_v, y_e \epsilon 0, 1, \frac{v \epsilon V}{s, t, e \epsilon E} \tag{4.11}$$

$$\sum_{v \epsilon V_i \cup V_j} x_v \leq 1, c_i \epsilon C_0 \; c_j = \hat{c}_i \; i < j \tag{4.12}$$

[94]

Mandoiu et al [94] then describe two algorithms for MTSDP(l—C—multiple), and note that the other variants of MTSDP(*—*—multiple) can be handled with same algorithms containing minimal changes.[94]

The first algorithm uses an alphabetic tree search algorithm, that "marks a c-token as unavailable only when a complete tag is found." [94] "The algorithm performs an alphabetical traversal of a 4-ary tree representing all 4^l possible tags, skipping over subtrees rotted at internal vertices that correspond to tag prefixes including unavailable c-tokens." [94] Mandoiu et al [94] notes that the "alphabetic tree search algorithm produces a maximal feasible set of tags τ."

By defining a tag to be periodic if "t is the length l prefix of an infinite string x^∞, where x is a DNA string with $|x| < |t|$," [94] the following Lemma can be formulated.[94]

Lemma 19 *For every c and l, there exists an optimal tag set τ in which every tag has the uniqueness property or is periodic.*

[94] Periodic tags "make better use of the limited number of available c-tokens," than the tags that fit the uniqueness property.[94] This is due to the fact that "a periodic tag whose shortest period has length p contains as substrings exactly p c-tokens," and those tags that meet the uniqueness property "contain between $l-c+1$ and $\frac{l-c}{2+1}$ c-tokens." [94] Therefore a "feasible solution for MTSDP(l—C—multiple) consisting of n tags." is the "vertex-disjoint packing of n cycles in H_c." [94] Here H_c is a graph with C "as its vertex set, and in which a token c_i is connected by an arc to token c_j iff c_i and c_j can appear consecutively in a tag, i.e., iff c_j is obtained from c_i by appending a single nucleotide and removing the maximal prefix that still leaves a valid c-token." [94] The problem can then be expressed as the "Maximum vertex-disjoint directed cycle packing problem". [94] The problem is formally formulated as follows:

Given a directed graph G, find a maximum number of vertex-disjoint directed cycles in G.[94]

It can be proven that the solution to the maximum vertex-disjoint directed cycle packing problem is "APX-hard even for regular directed graphs with in-degree and out-degree of 2." [94] Mandoiu et al [94] used a greed algorithm to solve the problem. They enumerated "possible tag periods in pseudo-lexicographic order and check for each period if all c-tokens are available for the resulting tag," and [94] refer to "this algorithm as the greed cycle packing algorithm." [94] Further extensions are mentioned by [94]. One open problem listed by [94] is that of finding a ”tight upper bound and exact method for MTSDP formulations.” [94]

4.2.5 Multiplexed Genotyping with Sequence-Tagged Molecular Inversion Probes

[17] Baner et al [17] present a ”strategy that combines DNA detection specificity and sensitivity with the potential to analyze large numbers of target sequences in parallel.” [17] The steps used are:

1. ”padlock probes with universal tag sequences were reacted with target DNA”
2. ”molecularly inverted”
3. ”amplified together”
4. ”identified in a multiplex analysis ”

/citebaner03 The method proposed by Baner et al [17] is purported to result in the ”lowering of the scale, cost and sample requirements of high-throughput genotyping.” [17]

Padlock probes were used to create ”linear dimeric molecules,” [17] that are ”easily distinguished from circularized probes by exonucleolyic degradation.” [17] This reduces the rise of ”nonspecific amplification products,” as seen when PC amplification probes are used to evaluate target sequences. The ”exonuclease treatment protocol” removes a significant number of the dimeric molecules and little effects are seen in the circularized probes. [17]

The molecular inversion probe genotyping was accomplished by ”redesigning the padlock probes to be locus specific to avoid the need for balancing allele-specific probes at every locus.” [17] This was done ”before increasing the multiplexing level.” [17] This is accomplished by leaving out the ”polymorphic nucleotide a the 3énd.” [17] The gap that was created in this process was then filled during ”four separate allele-specific polymerization and ligation reactions." [17] The melting temperature was considered during tag selection since all were intended to have similar melting temperatures. The base composition was also considered in order to ensure the similarity in base composition. When these are considered the tags can all hybridize under the same conditions.

After the amplification is complete, "the products are hybridized on four DNA microarrays and the components are deco0ded by measuring the fluorescence signals at the corresponding complementary tag site on the DNA array." [17]

The results show that "16% of the probes were inactive during a single synthesis." [17] The projection of the problem as assumed to be due to "errors in the database, probe design or failures of oligonucleotide synthesis, probe synthesis, ro the assay itself." [17] Out of 23,450 assays, 21,336 where called "full genotypes," [17] and 1,746 "half genotypes." [17]

CHAPTER 5

PARALLEL STATISTICAL-VALIDATION OF CLUSTERING ALGORITHMS FOR THE ANALYSIS OF MICROARRAY DATA

Currently, clustering applications use classical methods to partition a set of data (or objects) in a set of meaningful sub-classes, called clusters. A cluster is therefore a collection of objects which are "similar" among them, thus can be treated collectively as one group, and are "dissimilar" to the objects belonging to other clusters. However, there are a number of problems with clustering. Among them, as mentioned in [46], dealing with large number of dimensions and large number of data items can be problematic because of computational time. In this chapter [13], we investigate all clustering algorithms used in [46] and we present a parallel solution to minimize the computational time. We apply parallel programming techniques to the statistical algorithms as a natural extension to sequential programming technique using R. The proposed parallel model has been tested on a high throughput dataset. It is microarray data on the transcriptional profile during sporulation in budding yeast. It contains more than 6,000 genes. Our evaluation includes clustering algorithm scalability pertaining to datasets with varying dimensions, the speedup factor, and the efficiency of the parallel model over the sequential implementation. Our experiments show that the gene expression data follow the pattern predicted in [46] that is Diana appears to be solid performer also the group means for each cluster coincides with that in [46]. We show that our parallel model is applicable to the clustering algorithms and more useful in applications that deal with high throughput data, such as gene expression data.

5.1 Introduction

In fact, there has been a great deal of work on gene expression analysis, each using distinct data sets of gene expression, cluster-ing techniques. However, the majority of these works has given emphasis on the biological results, with no critical evaluation of the computational time of the clustering algorithm used in high dimension data. In the few works in which clustering algorithms was applied with gene expression data, the focus was only on the evaluation of the proposed validation methodology. As a consequence, so far, with the exception of [46], there is no study on which clustering methods are used in parallel computing for the analysis of data from high throughput gene expression time series. Based on this, a data driven comparative study of parallel computing, clustering and validation methods used in the literature of high throughput gene expression analysis is accomplished in this chapter. More specifically, parallel computing in R is used which allows a programmer to employ several machines, or processors, during program execution. This means that set of algorithms must be partitioned in such a manner as to be made optimal for the statistical problem at hand. In terms of clustering algorithms, six algorithms are analyzed in the experiments. In terms of validation methods, versions of three validation methods are used to compare the clustering algorithms. All the experiments are performed with data sets of time series gene expression of the yeast. This data was chosen because there is a wide availability of public data, as well as the comparison of the time taken by [46] and the solution proposed in this chapter.

5.2 Algorithms And Implementation

First we implement each of these clustering techniques using S+ for the sporulation data. Chu et al. [37] advocated grouping the expressed genes into seven temporal classes on biological grounds. Following Chu et al. [37], the number of clusters was set to seven in each case. As expected, there are some differences in the results of the various algorithms. Overall, K-means and Diana seem to be most effective in achieving good separation and almost distinct class boundaries. One potential problem with Fanny is that it typically produces only few distinct hard clusters. For this data set, only three clusters were produced, even though seven were desired. Further details, including pictures, can be obtained from the supplementary website.

5.3 Results and Discussion

From table 1, it appears that the parallel solution follow the pattern predicted by [46]. More specifically, the mean for each cluster did not change for all the algorithms using all the validation methods, and Fanny had the worst performance. On the other hand, we notice that the time taken by each process is slightly increased by couple of seconds due to the communication between the slave and the master program.

Table 1.? the weight of each cluster and parallel cpu time for each algorithm

CV	Cl4	Cl5	Cl6	Cl7	Cl8	Cl9	Cl10	Cl11	Cl12	Time
HC1	0.225	0.22	0.164	0.185	0.182	0.227	0.244	0.26	0.252	12
KM1	0.28	0.3	0.158	0.319	0.402	0.37	0.384	0.388	0.38	12
DI1	0.144	0.198	0.266	0.314	0.327	0.365	0.376	0.361	0.36	16
HC2	0.478	0.477	0.497	0.511	0.525	0.551	0.526	0.515	0.519	17
MO1	0.451	0.507	0.501	0.53	0.591	0.582	0.587	0.592	0.597	17
KM2	1.132	0.867	0.453	0.701	0.906	0.88	0.917	0.905	0.881	18
DI2	0.437	0.534	0.807	0.869	0.901	0.948	0.895	0.822	0.826	19
MO2	3.256	3.156	3.132	3.001	3.102	3.005	2.985	2.899	2.851	21
PLS1	0.13	0.126	0.288	0.337	0.213	0.22	0.255	0.253	0.258	240
PLS2	2.224	2.215	2.315	2.173	2.171	2.403	2.436	2.756	2.768	245
FA1	0.157	0.202	0.272	0.277	0.314	0.282	0.256	0.439	0.363	988
FA2	0.625	0.542	0.671	0.655	0.71	0.645	0.59	0.963	0.872	1004
KM3	2.414	2.358	2.07	2.065	1.996	1.965	1.912	1.869	1.837	1732
DI3	2.331	2.28	2.251	2.219	2.197	2.18	2.12	2.03	2.002	1830
HC3	3.434	3.431	3.324	3.321	3.306	3.265	3.242	3.226	3.214	2143
MO3	2.985	2.6	2.59	2.351	2.229	2.213	2.15	2.185	2.179	2197
PLS3	3.436	3.412	3.431	3.412	3.281	3.278	3.233	3.224	3.19	2428
FA3	2.658	2.6	2.626	2.648	2.625	2.619	2.606	2.741	2.7	2819

5.3.1 Speed up and efficiency

Hypothetically, speedup, or Sp, should be perfectly linear when using an embarrassingly parallel algorithm. This is a somewhat unrealistic assumption, given the fact that large amounts of data must be transferred to the various nodes for the use in the computations. In addition to this, the parallel solution, or their contributions to the statistics of interest, must be collected and output by the master node at the end of the program which can reduce speedup. Again, each row of Table 1 is for a run of the parallel solution program for a corresponding number of nodes p. All execution times are compared to that of the serial, or p = 1, run of the program. The last column of the table contains the CPU time take by the parallel solution which is T(p) = 2869 second equivalent to 1 hour and 8 minutes. We computed the absolute speedup as S(p)=T(s)/T(p), where T(s) is the execution time of the sequential program and T(p) the time needed for performing the parallel version on p processors. Therefore, the speedup is S(p)=T(s)/T(p)= 39914/2869=14.19. The absolute efficiency is then computed as speedup/p, where p = 16. Thus, the performance of our parallel solution is: E(p)=S(p)/p=14.16/16=88.7%. The performance loss of the program with only one level running in parallel in comparison to the ideal efficiency (= 1) is due to communication and management cost that sequential program is spared.

The main contribution of this chapter was to present a comparative analysis of sequential and parallel clustering and validation algorithm applied to the analysis of gene expression data. In order to do so, a parallel model implementing the clustering algorithms introduced in [46] was proposed. The study carried out in this chapter is more complete than previous ones, as it used the same data sets that used in [37], and included methods not evaluated before. Furthermore, no comparative analysis of parallel clustering performed before on high throughput time series gene expression data.

In summary, we have presented a simple system for parallelization which many statisticians will be able to use immediately to de-crease computational processing time for many computationally intensive problems. We consciously do provide a solution flexible enough to express all classes of parallel algorithms. Instead, we focus on a system that is very easy to learn to use, yet is powerful enough to express a large class of important parallel computations. Parallelizing a computation reduces computing time, but the effort required to develop and deploy a parallel solution must be considered as well. The simplicity of our solution helps to reduce this effort. It is therefore an effective solution for statistical research projects which create and evaluate computationally intensive statistical procedures, and for distributing the resulting tools to other users.

With the success from parallelizing these techniques, the next step is to wonder whether other clustering methods of high throughput matrices would see similar results. Another area is, we suggest that the entire parallel model be made available through web services, so that users can upload high throughput data, whose matrix is already available, and with an option of choosing to set the required parameters the suite will produce the means measure for each cluster calculated by each clustering algorithms and statistical validation chosen . Webservices have increasingly been of interest because of the ease and simplicity of usage. With Webservices the usage complexity of several tools can be abstracted. Users do not have to manually execute the tools and utilities. Considering the users may not be computer wizards, web interface would provide added simplicity for experimenters to effectively use the suite.

CHAPTER 6

GENOTYPE SUSCEPTIBILITY AND INTEGRATED RISK FACTORS FOR COMPLEX DISEASES

Recent improvements in the accessibility of high-throughput genotyping have brought a great deal of attention to disease association and susceptibility studies. This chapter [96] explores possibility of applying discrete optimization methods to predict the genotype susceptibility for complex disease. The proposed combinatorial methods have been applied to publicly available genotype data on Crohn's disease and autoimmune disorders for predicting susceptibility to these diseases. The result of predicted status can be also viewed as an integrated risk factor. The quality of susceptibility prediction algorithm has been assessed using leave-one-out and leave-many-out tests and shown to be statistically significant based on randomization tests.The best prediction rate achieved by the prediction algorithms is 69.5% for Crohn's disease and 63.9% for autoimmune disorder. The risk rate of the corresponding integrated risk factor is 2.23 for Crohn's disease and 1.73 for autoimmune disorder.

6.1 Introduction

Recent improvement in accessibility of high-throughput genotyping brought a great deal of attention to disease association and susceptibility studies[133]. High density maps of single nucleotide polymorphism (SNPs)[66] as well as massive genotype data with large number of individuals and number of SNPs become publicly available[43, 59, 76]. A catalogue of all human SNPs is hoped to allow genome-wide search of SNPs associated with genetic diseases.

Success stories when dealing with diseases caused by a single SNP or gene were reported. But some complex diseases, such as psychiatric disorders, are characterized by a non mendelian, multifactorial genetic contribution with a number of susceptible genes interacting with each other[99, 25]. In general, a single SNP or gene may be impossible to associate because a disease may be caused by completely different modifications of alternative pathways. Furthermore, there are no reliable tools applicable to large genome ranges that could rule out or confirm association with a disease. It is even difficult to decide if a particular disease is genetic, e.g., the nature of Crohn's disease has been disputed [10]. Although answers to above questions may not explicitly help to find specific disease-associated SNPs, they may be critical for disease prevention. Indeed, knowing that an individual is (or is not) susceptible to (or belong to a risk group for) a certain disease will allow greatly reduce the cost of screening and preventive measures or even help to completely avoid disease development, e.g., by changing a diet.

Disease association analysis is usually searching for a SNP which can be a statistically significant risk factor. In epidemiology, the importance of the integrated risk factor is commonly measured by risk rates. The risk rate is the ratio of disease incidence proportion in the population exposed to the risk factor to that in the non-exposed population. Unfortunately, the traditional direct statistical association so far is unsatisfactory and arguably is not applicable to complex diseases [39]. Even if an individual SNP has a significant relative risk rate it may give only negligible absolute increase of probability, e.g., from 10 in a million to 20 in a million. Note that the cumulative power of several SNPs is difficult to assess because of SNP linkage. So it would be desirable to have a tool that would integrate different genetic risk factors resulted in high disease prediction rate and high risk rate.

This study is devoted to the problem of assessing accumulated information targeting to predict genotype susceptibility to complex diseases with significantly high accuracy and statistical power. In this chapter, we first give several discrete optimization based algorithms for prediction disease susceptibility. We then compare leave-one- and leave-many-out tests demonstrating that prediction accuracy of suggested methods is sufficiently resilient to discarding case/ control data implying that leave-one-out test is a trustworthy accuracy measure. The randomization techniques have been used for computing the statistical significance level of proposed methods and resulted prediction weights. We show that prediction rate and statistical significance are well correlated.

The proposed methods are applied to two publicly available data: Crohn's disease [43] and autoimmune disorder [126]. In the leave-one-out cross-validation tests the proposed linear programming (LP) based method achieves prediction rate of 69.5%(p-value below 2%) and 61.3%(p-value below 62%) and the risk rates of 2.23 and 0.98, respectively. We also show that SVM methods used in [91, 129] are not much worse than our proposed LP-based method.

The next section formally defines the problem and describes several universal and adhoc methods for predicting genotype susceptibility to complex diseases. Section 6.3 describes real case/control data sets, discusses prediction and risk rate measures and compares results for several susceptibility prediction methods. We draw the conclusion in the last section.

6.2 Prediction Methods for Genotype Susceptibility

In this section we first describe the input and the output of prediction algorithms and how to predict genotype susceptibility. We the describe several universal and adhoc prediction methods.

Specifications of prediction algorithms. Data sets have n genotypes and each has m SNPs. The input for a prediction algorithm includes:

(G1) Training genotype set $g_i = (g_{i,j}), i = 0, \dots, n-1, j = 1, \dots m$, $g_{i,j} \in \{0, 1, 2\}$

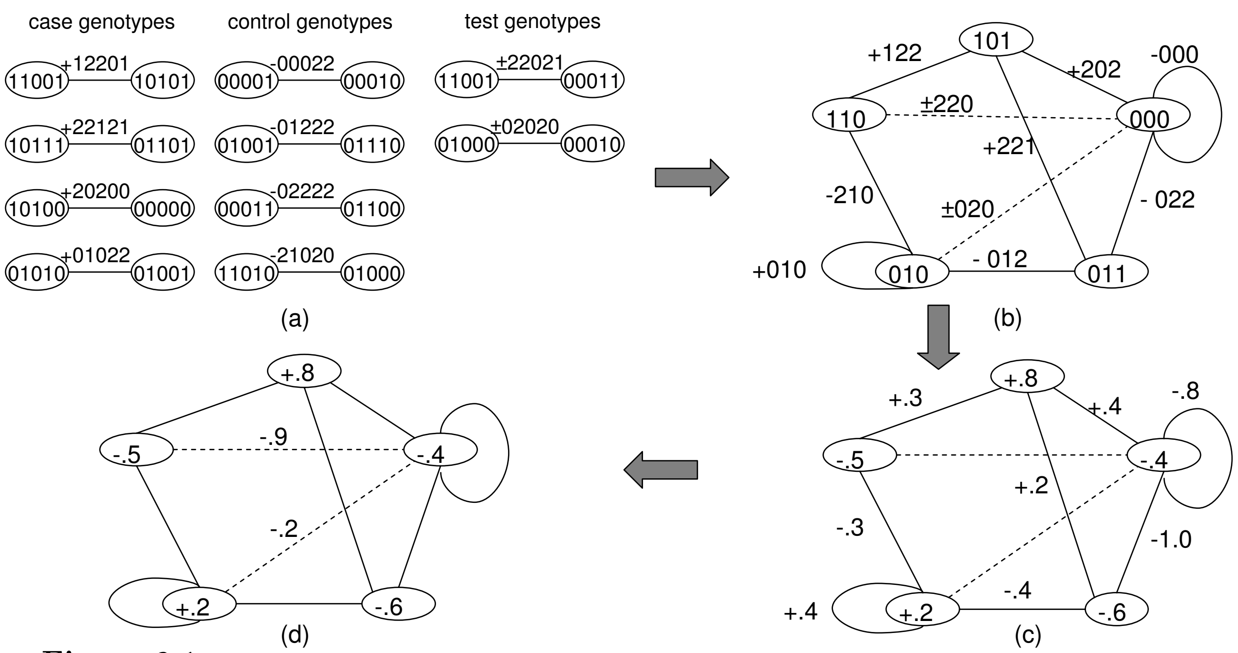


Figure 6.1. LP-based Prediction Method. (a) The set of case, control and test genotypes are phased resulting in the sparse graph with vertices-haplotypes and edges-genotypes. (b) The last two SNPs are dropped without collapsing case and control edges resulting in a denser graph. (c) The LP finds optimal weights for vertices-haplotypes. (d) The status of test genotypes is predicted from the sign of the sum of weights of their endpoints.

(G2) Disease status $s(g_i) \in \{-1, 1\}$, indicating if $g_i, i = 0, \ldots, n-1$, is in case (1) or in control (-1) , and

(G3) Testing genotype g_n without any disease status.

The input data can also be phased, then each genotype is represented by a pair of haplotypes. We will refer to the parts (G1-G2)of the input as *training set* and to the part (G3) as the test case. The output of prediction algorithms is the disease status of the genotype g_n, i.e., $s(g_n)$.

Below we describe several universal prediction methods. These methods are adaptations of general computer-intelligence classifying techniques.

Closest Genotype Neighbor (CN). For the test genotype g_t, find the closest (with respect to Hamming distance) genotype g_i in the training set, and set the status $s(g_t)$ equal to $s(g_i)$.

Support Vector Machine Algorithm (SVM). Support Vector Machine (SVM) is a generation learning system based on recent advances in statistical learning theory. SVMs deliver state-of-the-art performance in real-world applications and have been used in case/control studies [91, 129]. We use SVM-light [88] with the radial basis function with $\gamma = 0.5$.

Random Forest (RF). A random forest is a collection of CART-like trees following specific rules for tree growing, tree combination, self-testing, and post-processing. We use Leo Breiman and Adele Cutler's original implementation of RF version 5.1 [28]. This version of RF handles unbalanced data to predict accurately. RF tries to perform regression on the specified variables to produce the suitable model. RF uses bootstrapping to produce random trees and it has its own cross-validation technique to validate the model for prediction/classification.

CDPG: Tomita [124] introduced the Criterion of Detecting Personal Group (CDPG) for extracting risk factor candidates(RFCs). RFCs are extracted using binomial test and random permutation tests. CDPG performs exhaustive combination analysis using case/control data and assumes the appearance of case and control subjects belonging to a certain rule as a series of Bernoulli trials, where two possible outcomes are case and control subjects with some probabilities.

We now describe two ad hoc prediction methods (i.e., classifying techniques taking in account the nature of the classification problem). The first method is 2-SNP method [85] and the second method is a variation of the LP-based method [95].

Most Reliable 2 SNP Prediction [85] (MR). This method chooses a pair of adjacent SNPs (site of s_i and s_{i+1}) to predict the disease status of the test genotype g_t by voting among genotypes from training set which have the same SNP values as g_t at the chosen sites s_i and s_{i+1}. They chose the 2 adjacent SNPs with the highest prediction rate in the training set.

LP-based Prediction Algorithm (LP). This method are based on the following *genotype* graph $X = \{H, G\}$, where the vertices H are distinct haplotypes and the edges G are genotypes each connecting its two haplotypes (vertices) (see Figure 6.1(a)).

When applying graph heuristics to X, we found that it is necessary to increase the density of X. This can be achieved by dropping certain SNPs (or, equivalently, keeping only certain tag SNPs). Indeed, dropping a SNP may result in collapsing of certain vertices in X, i.e., different vertices become identical. Collapsing vertices may also result in collapsing certain edges (genotypes). Discarding a SNP is not allowed if it results in collapsing edges from case and control populations, but collapsing of edges from the same population is allowed (see Figure 6.1(b)).

A simple greedy strategy consists of traversing all the SNPs and dropping a SNP if it is allowed. The resulted set of SNPs is a minimal subset of SNPs which do not collapse genotypes from opposite disease status. Unfortunately, in the original graph X we may already have collapsed edges from opposite populations - in fact, Daly *et al* data contain such pair of genotypes. Only such original collapsing is allowed – the status of such edges is assumed to be the one of majority of genotypes. Our experiments show that on average, we are left with 21 tag SNP's out of 103 for Daly *et al* [43] data and 29 tag SNP's out of 108 for Ueda *et al* [126] data (see description of the next section). The selected set tag SNPs are better candidates for being disease associated, in fact only such tag SNPs were used in the prediction methods with the highest accuracy.

After collapsing the graph X we add the edge corresponding to the test-case genotype g_n. If the edge g_n collapses with another edge g_i, then we set the predicted disease status $s(g_n) = s(g_i)$. Otherwise, we apply one of the following two methods for computing the disease status $s(g_n)$. The LP-based method assumes that certain haplotypes are susceptible to the disease while others are resistant to the disease.

The genotype susceptibility is then assumed to be a sum of susceptibilities of its two haplotypes.

We want to assign a positive weight to susceptible haplotypes and a negative weight to resistant haplotypes such that for any control genotype the sum of weights of its haploptypes is negative and for any case genotype it is positive (see Figure 6.1(c)). We would also like to maximize the confidence of our weight assignment which can be measured by the absolute values of the genotype weights. In other words, we would like to maximize the sum of absolute values of weights over all genotypes.

Formally, for each vertex h_i (corresponding to haplotype) of the graph X we wish to assign the weight p_i, $-1 \leq p_i \leq 1$ such that for any genotype-edge $e_{ij} = (h_i, h_j)$, $s(e_{ij})(p_i + p_j) \geq 0$ where $s(e_{ij}) \in \{-1, 1\}$ is the disease status of genotype represented by edge e_{ij}.

The total sum of absolute values of genotype weights is maximized

$$\sum_{e_{ij}=(h_i,h_j)} s(e_{ij})(p_i + p_j) \tag{6.1}$$

The above formulation with the objective (6.1) is the linear program which can be efficiently solved by a standard linear program solver such as GNU Linear Programming Kit (GLPK) [64].

For the left-out testing genotype g_n, we compute the sum of weights of its haplotypes. If the sum is strictly positive, the genotype is attributed to the case, if the sum is strictly negative, it is attributed to the control (see Figure 6.1(d)), otherwise $s(g_n)$ is assigned according to 2-SNP prediction algorithm [85].

6.3 Quality of Susceptibility Prediction Methods

In this section we describe the two real case/control population samples and the results of leave-one-out and leave-many-out cross-validation tests estimating susceptibility prediction methods on these sets.

Data Sets. The data set Daly *et al* [43] is derived from the 616 kilobase region of human Chromosome $5q31$ that may contain a genetic variant responsible for Crohn's disease by genotyping 103 SNPs for 129 trios. All offspring belong to the case population, while almost all parents belong to the control population. In entire data, there are 144 case and 243 control individuals. The missing genotype data and haplotypes have been inferred using 2SNP phasing method [30]. The highest risk rate for single SNP is 2.7.

The data set of Ueda *et al* [126] are sequenced from 330kb of human DNA containing gene CD28, CTLA4 and ICONS which are proved related to autoimmune disorder. A total of 108 SNPs were genotyped in 384 cases of autoimmune disorder and 652 controls. Similarly, the missing genotype data and haplotypes have been inferred. The highest risk rate for single SNP is 1.9.

Cross-validation Tests. In the leave-one-out cross-validation, the disease status of each genotype in the data set is predicted while the rest of the data is regarded as the training set. In the leave-many-out cross-validation, n individuals are uniformly at random picked up from the data set, marked and put back, where n is the size of the data set. This way, approximately $2/3$ of the individuals are picked at least once and marked while the rest will not be marked. The training set consists of marked data and the testing set consists of unmarked data.

Quality Measures. In cross-validation tests, the predicted and the actual disease statuses are compared and the standard confusion matrix is filled (see Table 6.1). Predicted cases and predicted controls are notated by pCS and pCO respectively. We report sensitivity, specificity, and accuracy of the prediction methods. We also report

Table 6.1. Confusion Table.

	True Data (Golden Standard)		
	Case	Control	
pCS	True Positive TP	False Positive FP	Positive Prediction Rate PPR= TP/(TP+FP)
pCO	False Negative FN	True Negative TN	Negative Prediction Rate NPR= TN/(FN+TN)
	Sensitivity TP/(TP+FN)	Specificity TN/(FP+ TN)	Accuracy (TP+TN)/(TP+FP+FN+TN)

the the risk rate of the corresponding integrated risk factor associated with each prediction method. It is computed as the the ratio of the probability of developing disease among those predicted susceptible to the probability of developing disease among those predicted non-susceptible [41]:

$$Risk\ Rate = \frac{TP}{TP+FP} / \frac{FN}{TN+FN}$$

We report the 95% confidence intervals (CI) for accuracy and risk rate, for leave-one-out test 95% CI is computed using bootstrapping. We also compute significance level, *p*-value, for the accuracy of prediction algorithms computed using 5000 randomized instances. On the randomized instances, the average prediction rate for SVM and RF has been 60% and for all other methods except has been 50%.

Results and Discussion. Table 6.2 compares 6 different prediction methods for both data sets. Column C denotes performed cross-validation tests, LOO stays for leave-one-out test and LMO stays for leave-many-out test. For leave-one-out test, the best accuracy is achieved by LP – 69.5% on Daly *et al.* [43] data and by MR – 63.9% on Ueda *et al.* [76] data. For leave-many-out test, the accuracy only slightly degrades showing resiliency to the size of the data. The risk rates for the integrated risk factor associated with prediction methods are comparable with risk rates for individual SNPs – for the first data set, 2.23 (LP method) vs 2.7 and for the second data set, 1.73 (RF method) and 1.64 (MR method) vs 3.2. The good performance of SVM

Table 6.2. The comparison of sensitivity, specificity, accuracy and risk rate with 95% confidence intervals (CI) and p-value for 6 prediction methods for two real data sets.

Daly *et al.* [43]							
C	Quality measure	Prediction Methods					
		CN	SVM	RF	CDPG	MR	LP
	sensitivity	45.5	20.8	34.0	68.8	30.6	37.5
	specificity	63.3	88.8	85.2	58.0	85.2	88.5
L	accuracy	54.6	63.6	66.1	62.2	65.5	69.5
O	95%-CI	±.9	±.5	±.6	±.8	±.9	±.5
O	p-value	0.03	0.04	0.30	0.04	0.03	0.01
	risk rate	1.25	1.52	1.83	1.49	2.00	2.23
	95%-CI	±.09	±.04	±.03	±.02	±.02	±.05
	sensitivity	45.9	18.0	30.0	59.7	28.0	36.0
L	specificity	54.0	89.3	82.2	55.6	76.5	82.3
M	accuracy	52.2	62.9	64.2	57.1	58.5	68.4
O	95%-CI	±.9	±.5	±.5	±.9	±.9	±0.5
	risk rate	0.99	1.45	1.67	1.47	1.15	2.01
	95%-CI	±.06	±.26	±.12	±.01	±.01	±.01

Ueda *et al.* [76]							
C	Quality measure	Prediction Methods					
		CN	SVM	RF	CDPG	MR	LP
	sensitivity	37.7	14.3	18.0	58.6	6.9	7.1
	specificity	64.5	88.2	92.8	61.7	97.2	91.2
L	accuracy	54.8	60.9	65.1	60.5	63.9	61.3
O	95%-CI	±.9	±.3	±.4	±.8	±.9	±.3
O	p-value	0.04	0.70	0.73	0.05	0.04	0.62
	risk rate	1.05	1.15	1.73	1.67	1.64	0.86
	95%-CI	±.01	±.03	±.03	±.01	±.01	±.03
	sensitivity	34.8	12.7	13.4	56.0	7.2	8.0
L	specificity	64.8	90.5	83.5	56.9	89.4	82.7
M	accuracy	53.4	61.8	62.4	56.6	58.4	59.3
O	95%-CI	±.9	±.3	±.3	±.9	±.9	±.6
	risk rate	0.98	1.22	1.25	1.38	0.76	0.98
	95%-CI	±.06	±.03	±.03	±.01	±.01	±.01

and certain other universal methods indicate that they can possibly be adjusted to improve specific ad hoc methods for prediction of susceptibility to complex diseases.

CHAPTER 7
CONCLUSION

DNA Array Flow

Our experiments show that the genomic data follow the pattern predicted by simulated data. In case of Herpes B virus, like simulated data, increasing number of candidates per probe (k) decreases number of border conflicts during the probe placement algorithms. However, the number of border conflicts is several times smaller than for simulated data. We give the trade-off between number of border conflicts and the CPU time taken for the various algorithms that are defined in the physical design. In [12], we give a concatenate software solution for the entire DNA array flow, and we explore all steps in a single automated software suite of tools. We suggest that the entire software suite be made available through web services, so that users can enter name of organism, whose DNA sequence is already available at GenBank, and with an option of choosing to set the required parameters the suite will produce the DNA probe micro-array chip layout. Web Services have increasingly been of interest because of the ease and simplicity of usage. With Web Services the usage complexity of several tools can be abstracted. Users do not have to manually execute the tools and utilities. Considering the users could be biologists or those who may not be computer wizards, web interface would provide added simplicity for experimenters to effectively use the suite.

The design flow for DNA array is comprised of five steps. The first is ORF extraction, followed by probe selection, physical design, manufacturing and finally hybridization and analysis. [12] The probe selection and physical design are the main areas where research is prevalent. Probe selection applies to DNA, RNA and UTA. The physical design presents a problem similar to that of VLSI design.

Probe selection is divided into statistical methods and thermodynamic methods. [113] and [111] approached probe selection from a statistical point of view while thermodynamic approach remains the method most commonly seen.

[113] approached the design issues using a group testing method. This method had to be adapted to cope with the untraditional nature of probe selection compared to that of a more common subject for group testing. The main focus of this approach is that of coping with "closely related target sequences." [113] Remaining areas of interest include the usage of "several unique probes for each target," [113] using other methods, and finding more information about "dynamics of hybridization reactions." [113] The approach used by Rahmann et al [111] attempted to deal with the design problems associated with large scale DNA arrays while preserving the accuracy. The approach is "based on jumps in matching statistics." [111] There is an increase in the time cost of the selection phase, but it is argued that it is acceptable since it is only an increase by a factor of 2 or 3. [111] Testing on a large scale is still needed to solidify the results. [111]

[121] presents a thermodynamic approach to probe selection. The approach does not use heuristics, and employs a series of filters to perform the selection. [121] The algorithm is practical for long or short genomes, and a divide and conquer approach is used to deal with hardware constraints for large genomes. Additional examination of eliminating heuristics from some approaches might is an interesting idea to increase accuracy at the cost of efficiency.

[78] gives an thermodynamic approach as well. The algorithm proposed by [78] uses the NN thermodynamic model, and probes must be a perfect compliment to the sequence. All probes that can be eliminated prior to calculating the melting temperature are eliminated to reduce the time cost of the calculations, and redundancy is

attempted to be avoided by use of a suffix tree. [78] The algorithm has been tested on "randomly generated sequences of different lengths." [78] However, effective analysis of the running times could not be obtained because the complexity and running time are heavily dependent on the selection of "filtering criteria." [78] This selection of criteria is a problem facing algorithms in this area.

Other attempts to assist with probe selection include the work by [117], who proposed a parallel architecture to assist overcoming hardware constraints that occur due to the intensive resources necessary during probe selection. [67] offers a linear algebra approach to probe selection, and finally [87] develops an entropy estimator.

The physical design of the DNA array is closely linked to the design of VLSI chips. Maximizing the number of probes, while minimizing the size of the chip is the goal.

[81] uses the logical approach and utilizes principles of VLSI design to improve the physical design. The method centers around minimizing border cost. [81] When comparing the new re-embedding algorithm "given a two-dimensional probe placement, improves the embedding of the probes without re-placing," to the chessboard and the batched greedy algorithm the following was noted. [81] It is shown that the "re-embedding of the probes in a sequential row-by-row order leads to a reduction in the border cost by 0.8% compared to the chessboard algorithm." [81] A current open problem given by [81] is that of "developing a tighter lower bound." [81]

[48] attempts to use self organizing maps to solve the physical design problem. The method is determined to be acceptable for SNP analysis, but unacceptable for "sequencing by hybridization." [48] This is true because an insufficient number of probes are organized by the algorithm. This is one area where the algorithm needs improvement.

Probe selection and physical design are the most researched and also have some the larger problems. Probe selection is important because it directly effect the outcome of the experiment and is a source of much energy and time.

Universal Tag Array (UTA) Design [19] addresses the problem of maximizing the number of probes, while minimizing cross-hybridization. The approach used in [19] is that of the thermodynamic model. The combinatorial approach used to construct the array has proven to be near-optimal. [19] The assumptions made by [19] are the following:

1. "If a sequence has a weight greater than or equal to h (corresponding to melting temperature greater than or equal to $2h$ in the 2-4 model) then the sequence will hybridize to its complement." [19]

2. "Sequence x will fail to hybridize to sequence y provided that there is no string z of weight greater than or equal to c such that z is a substring of x and z^C us a substring of y." [19]

However, concerns such as "secondary structure in a tag may cause it to fail to hybridize to its antitag" [19]. Ben-Dor et al [19] do note that the scheme provided in their paper will only be useful if can be verified that "violations are infrequent" and those violations that do exist are eradicated "by deleting some tags." [19]

In [20] the main issue is addressing assay specific cross hybridization." While [19] addresses the cross-hybridization "between tags and foreign anti-tags," [20] points out other areas of assay specific cross hybridization not discussed in [19]. Examples of possible problems are, "primer to antitag cross-hybridization, ...sandwich cross-hybridization,...primer to primer mis-extension, and primer to tag mis-extensions." [20] However, the only source of cross-hybridization specifically addressed by [20] is the primer to antitag cross hybridization. This problem is address by associating the problem with the "problem of covering the vertices of one side of a bipartite graph by a minimum number of balanced sub-graphs of maximum degree 1." [20]

Mandoui et al [93] tackle two problems the first problem addressed is the tag set design problem extended to include anti-tag to anti-tag hybridization. The second problem addressed is the tag assignment problem. An attempt is made to improve the multiplexing rate. The desired result is to address the problem of assigning "all tags to primers in an array experiment." This is addressed since this is not possible due to cross-hybridization. [93] Constraints are placed on the tags as follows.

- *(H1) Every antitag hybridizes strongly to its tag;*
- *(H2) No antitag hybridizes to a tag other than its complement and*
- *(H3) There is no antitag to antitag hybridization.*

[93] A greedy algorithm is used and the result did experience improvement in the multiplexing rates. Although addressing the lack of a truly good pooling aware algorithm, is an example of an area that could be researched for completely as well as improving the multiplexing rate further.

[94] focused on solutions for the design of tag sets for universal tag arrays. The tag set design must adhere to two types of constraints.[94] The stability constraint, and the non-interaction constraint are applied.[94] The same model used by [19] for hybridization is used by [94].

[94] used a greedy algorithm to solve the problem, called the "greed cycle packing algorithm." A remaining problem is that of finding a "tight upper bound and exact method for MTSDP formulations." [94]

Genotype Susceptibility In this chapter, we discuss motivation behind the genotype susceptibility studies. The developed ad hoc susceptibility prediction method based on linear programming is shown to have high prediction rates and high relevant risk rate for associated integrated risk factors for two completely differentcase/control studies for Crohn's disease [43] and autoimmune disorders [76]. The extensive computational results show great potential of the proposed prediction methods. In our future work we are going to continue validation of the proposed method.

CHAPTER 8
PROSPECTIVE DESIGN APPLICATION

8.1 RNA Microarray Design

RNA Arrays are the least researched area discussed in this chapter. DNA microarrays have been researched extensively, while RNA Arrays have experienced little research. DNA microarrays use reverse transcription which is an indirect method of analyzing mRNA expressions. [6] The following is a presentation of the current research on RNA-DNA and RNA-RNA interactions. RNA-DNA is where RNA bonds with DNA, while RNA-RNA is RNA to RNA interactions. The design flow for RNA-DNA or RNA-RNA is comprised of the same steps as DNA microarray, except that the ORF extraction is no longer necessary as a step in the process. It can also be seen via the flow chart below that probe selection is present as a step in the design flow.

8.1.1 RNA-DNA Microarray Design

Alsaidi et al, focus on the "development of a direct, simple, rapid, accurate and sensitive system for RNA detection and quantification." [6] This differs from current methods using DNA microarray and realtime PC, since they use an indirect approach to RNA detection. [6]

The system is "based on the RNA 3́-labelling by using labelled-dNTPs and DNA polymerase on a DNA template." [6] Solid phase RNA detection and quantification "by immobilization of the template," [6] is currently being explored (see Figure 8.2). To "perform detection and quantification for a specific mRNA in the solid phase,its

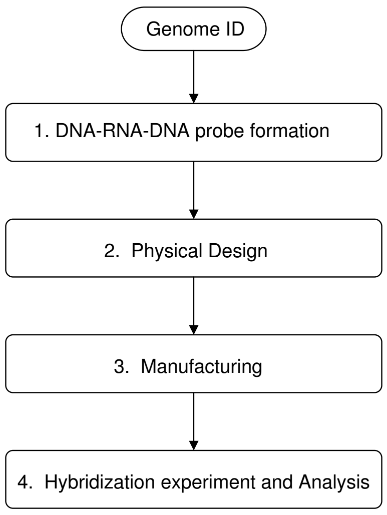

Genome ID
1. DNA-RNA-DNA probe formation
2. Physical Design
3. Manufacturing
4. Hybridization experiment and Analysis

Figure 8.1. RNA Design Flow

3′-region needs to be removed so as to expose its unique internal sequence for selective labelling and detection." [6] RNase H is used as an "RNA endonuclease." [6] RNase H is only used in this way when a "DNA guiding sequence" is present. [6] Since RNase H does not "digest RNA-RNA duplex" and can split RNA in a RNA-DNA duplex, the proposed design by Alsaidi et al is a "5′-DNA- 3′hybrid template in which the DNA and RNA sequences serve as a guiding sequence and a protecting sequence, respectively." [6]

"Immobilization of the 3′-terminus of the DNA-RNA hybrid template on a microplate allows immediate Klenow labelling following RNase H digestion of the mRNA 3′-region." [6] This double digestion leaves a product of "short RNA fragment hybridized to its template in the solid phase for detection." [6] The length of the template affects the amount of interference. Thus the template needs to be long.

The immobilization of the "hybrid template," is done through a "3́-NH_2 group on a microplate by N-hydroxylsuccinimide displacement." [6] Specific RNA are hybridized to the template and others are washed away after "incubation of a mixed RNA population on the functionalized microplate." [6] "Enzyme-binder conjugates" are used to bond to the "immobilized RNA target through the hapten label." [6] The "detection of the specific RNA" is achieved when the "immobilized enzyme catalyzes a chemiluminescence reaction in the presence of substrates." [6]

When long periods of exposure where used the background was visible, but shortening the time removed the background. The appearance of the background was due to the "nonspecific binding of the conjugate." [6]

The result of the paper by Alsaidi et al is the design of a "novel system for RNA-specific detection on a microplate by immobilizing the hybrid templates and using an enzyme label." [6] The implication of this is that it is the first such system, and can be used for "rapid detection of pathogens and diseases" as well as for "analysis of environmental samples, in which mRNAs are partially degraded." [6] The method developed by [6] is "direct, simple, cost-effective, and rapid without reverse transcription, PC transcription, etc..." unlike DNA array technology. [6] also suggest that sensitivity can be increased through use of smaller chips, as well as background reduction. This method can be used for "environmental samples" where some degradation has occurred. It is also suggested that the work has the potential to later be used for "developing methodologies for rapid on site detection of bacteria and viruses." [6] This area is so new that many areas are as yet unexplored and have left many open areas for future research.

8.1.2 RNA-RNA Microarray Design

There have been many studies involving "antisense" RNAs. Antisense RNAs are the string of nucleotides that are the complements to the message sense. mRNA and antisense RNA can form duplexes just like DNA does, providing that they are complements of one another. According to [5] "small RNAs have ... been artificially constructed to knock-out genes of interest in humans and other organisms for the purpose of finding out more about their function." One of the problems facing RNA duplexes is that while algorithms exist to predict the "secondary structure of a single RNA molecule," [5] no algorithms have been developed to predict the "joint secondary structure of two interacting RNA molecules." [5] Thus one of the main focuses of [5] is the presentation of the "RNA-RNA interaction prediction (RIP) problem." [5] Algorithms are also proposed to solve the RIP problem. Free energy minimization is the focus of the algorithms for predicting the "secondary structure of two interactive RNA molecules." [5]

The basics of the RIP problem are that given two RNA S and R, how can the joint secondary structure be predicted. [5] The "joint secondary structure between S and R" is defined by [5] as being the "set of pairings where each nucleotide of S and R is paired with at most one other nucleotide either from S or R." [5] In order to explain the three energy models used in [5], it is necessary to define some notation used in the definitions provided by [5].

- $S[i]$ = the i^{th} nucleotide of an RNA sequence S
- $S[i, j]$ = the substring of S extending from $S[i] to S[j]$
- $S[k, k] = S[k]$
- $S[i, i-1]$ = empty sequence
- $S[i, i-1]^r$ = reverse of $S[i-1, i]$
- $S[1]$ = 5énd of S
- $R[1]$ = 3énd of R

[5]

The three models used to compute the "free energy of interaction RNA sequences" [5] follow:

1. "Sum of free energies of individual Watson-Crick base pairs as a crude approximation to the total joint free energy. The basepair energy model is known to be inaccurate." [5]

2. The "second free energy model is based mostly on stacked pair energies...which provide the main contribution to the energy model employed by the Mfold program." [5] To cope with the lack if "thermodynamic information on pseudoknots or kissing loops," [5] incorporate another approach, "to differentiate the thermodynamic parameters of external bonds from the internal bonds by multiplying the external parameters with a weight slightly smaller than 1." [5]

3. The last free energy model is one that uses both the stacked pair and basepair models. This model sums "up the free energies of various types of internal loops and stacked pairs." [5] "This model...will be referred to as the loop energy model." [5]

The RIP problem for both the stacked pair and basepair models is proven by [5] to be an NP-Complete problems. The proofs follow from the proof of the "longest common subsequence of multiple binary strings" being a NP-Complete problem.

For the basepair model, the free energy between two interacting RNA is approximated "as the sum of the free energies of bonded nucleotide pairs." [5] If x and y are nucleotides, then the "Watson-Crick free energy of a bond" [5] between x and y is denoted ase(x, y), for internal bonds, and $\prime e(x, y)$ for external bonds. [5] However, for [5] it is accepted that $\prime e = e$. $E(S[i, j], R[\prime i, \prime j])$ is the "free energy between interacting RNA strands $S[i, j]$ and $R[\prime i, \prime j]$ for all $i < j$ and $\prime i < \prime k$." [5] E will be computed to minimize the free energy. The assignment of $E(S[i, i], R[\prime i, \prime i])$ to

$e(S[i], R[\prime i])$, allows for the computation of $E(S[i, i], R[\prime i, \prime i])$. [5] This computation will be done "inductively as the minimum of the following."

1. $min_{i-1 \leq j; \prime i-1 \leq \prime\prime j : (k \neq i-1 or \prime k \neq \prime i-1),(k \neq or \prime k \neq \prime j)}$ $E(S[i, k], R[\prime i, \prime k]) + E(S[k+1, j], R[\prime k+1, \prime j])$

2. $E(S[i+1, j-1], R[\prime i, \prime j]) + e(S[i], R[\prime j])$

3. $E(S[i, j], R[\prime i+1, \prime j-1]) + e(R[\prime i], R[\prime j])$

[5] [5] proves that the "above dynamic programming is correct", and that the "table E can be computed in time $\theta(|S|^3 \cdot |R|^3)$ and in space $\theta(|S|^2 \cdot |R|^2)$." [5]

In the stacked pairs model "the bonds between nucleotide pairs form uninterrupted sequences." [5] Internal stacked pair $X[i]-X[j], X[i+1]-X[j-1])$ will be represented by $ee(X[i, i+1], X[j-1, j])$, and "the energy of the external stacked pair $X[i] - Y[j], X[i+1] - Y[j-1])$" [5], will be represented as $\prime ee(X[i, i+1], Y[j, j+1])$. [5] For [5] $\prime ee = \sigma ee$. It is noted that there is no symmetry between ee and $\prime ee$. [5] The following "four energy functions" [5] are necessary in order to "compute the joint structure between S and R." [5]

1. $E_S(S[i, j], R[\prime i, \prime j])$ denotes the free energy between S and R such that $S[i]$ bonds with $S[j]$. [5]

2. $E_R(S[i, j], R[\prime i, \prime j])$ denotes the free energy between S and R such that $R[\prime i]$ bonds with $R[\prime j]$. [5]

3. $E_l(S[i, j], R[\prime i, \prime j])$ denotes the free energy between S and R such that $S[i]$ bonds with $R[\prime i]$. [5]

4. $E_r(S[i, j], R[\prime i, \prime j])$ denotes the free energy between S and R such that $S[j]$ bonds with $R[\prime j]$. [5]

"The overall energy is defined to be: " [5]

$$E(S[i,j],R['i,'j]) = \begin{cases} E_S(S[i,j],R[i',j'], \\ E_R(S[i,j],R['i,'j]), \\ E_r(S[i,j],R['i,'j]), \\ E_l(S[i,j],R['i,'j]), \\ min_{i\leq k\leq j-1;'i\leq 'k\leq 'j-1}\{E(S[i,k], \\ R['i,'j])+E(S[k+1,j],R['k+1,'j])\}, \\ min_{i\leq k\leq j-1}\{E(S[i,k],-)+ \\ E(S[k+1,j]),R['i,'j])\}, \\ min_{'i\leq 'k\leq 'j-1}\{E(S[i,j],R['i,'k])+ \\ E(-,R['k+1,'[j])\}, \\ min_{'i\leq 'k\leq 'j-1}\{E(-,R['i,'k])+ \\ E(S[i,j],R['k+1,'[j])\}. \end{cases} \tag{8.1}$$

[5] The following are the initial values of the energy functions listed above.

$$E_l(S[i,j],-)=\infty, E_r(S[i,j],-)=\infty \tag{8.2}$$

[5]

$$E_l(-,R['i,'j])=\infty, E_r(-,R['i,'j])=\infty \tag{8.3}$$

[5]

$$E_l(S[i,i],R['i,'i])=0, E_r(S[i,i],R['i,'i])=0 \tag{8.4}$$

[5]

$$E_S(S[i,i],-)=\infty, E_R(-,R['i,'i])=\infty \tag{8.5}$$

[5] The following is the "complete description of the dynamic programming formulation." [5]

$$E_l(S[i,j], R['i,'j]) = min\{E_l(S[i+1,j], R['i+1,'j]) +$$
$$'ee(S[i,i+1], R['i,'i+1]), E(S[i+1,j], R['i+1,'j])\}$$

[5]

$$E_r(S[i,j], R['i,'j]) = min\{E_r(S[i,j-1], R['i,'j-1]) +$$
$$'ee(S[j-1,j], R['j-1,'j]), E(S[i,j-1], R['i,'j-1])\}$$

[5]

$$E_S(S[i,j], R['i,'j]) = min\{E_S(S[i+1,j-1], R['i,'j]) +$$
$$ee(S[i,i+1], S[j-1,j]), E(S[i+1,j-1], R['i,'j])\}$$

[5]

$$E_R(S[i,j], R['i,'j]) = min\{E_R(S[i,j], R['i+1,'j-1]) +$$
$$ee(R['j-1,'j]^r, R['i,'i+1]^r), E(S[i,j], R['i+1,'j-1])\}$$

[5] It is also proven by [5] that the "above dynamic programming formulation is correct." [5] E_S, E_R, E_l, E_r can also be shown to have running time $\theta(|S|^3 \cdot |R|^3)$, and can be "computed ... in space," $\theta(|S|^2 \cdot |R|^2)$. [5]

Due to the expense (in relation to time), the stacked pair energy model is excessive. For short strands of RNA the model can take 15 minutes to complete. For longer strands this is excessive and prohibitive. [5] An observation is made by [5], that the "(predicted)self structures are mostly preserved in the joint secondary structures."

[5] It is then proposed that it might be "sufficient to compute the joint structure of two RNA sequences by simply computing the set of loop pairs that form bonds to minimize the total joint free energy." [5] This approach can be extended to one that "maintains that each RNA sequence will tend to preserve much of its original secondary structure after interacting with the other RNA sequence." [5] The sequence that is preserved is referred to as "independent subsequence." [5]

Definition 1 *Independent Subsequences:*
Given an RNA sequence R and its secondary structure, the substring $R(i,j)$ is an independent subsequence of R if it satisfies the following conditions.

- *$R[i]$ is bonded with $R[j]$*
- *$j - 1 \leq \kappa$ for some user specified length κ*
- *There exists no $'i < i$ and $'j < j$ such that $R['i]$ is bonded with $R['j]$ and $'j - 'i \leq \kappa$*

[5] It is possible to use the following greedy algorithm "to compute the independent sequences of a given RNA molecule." [5]

1. Let IS be the set of independent subsequences in R; initially we set $IS = 0$.
2. Starting from the first nucleotide of R find the first nucleotide $R[i]$ which bonds with another nucleotide $R[j], (j > i)$.
3. If $j - i \leq \kappa$ then update $IS = IS \cup R[i,j]$ and move to $R[j+1]$. Else move to $R[i+1]$
4. Repeat step 2.

[5] It can be proven by [5] that the above greedy algorithm finds the "correct independent subsequences." [5] The following is used to "calculate the joint structure between

R and S by minimizing the total free energy of their independent subsequences (ISs) via means of establishing bonds between their nucleotides." [5] If "the minimum free energy of the joint secondary structure of the two ISs, $S_{IS}[i]$ and $R_{IS}[j]$ be $e_{IS}(i,j)$. The value of $e_{IS}(i,j)$ can be computed via" the greedy algorithm described above. [5] "The minimum joint free energy between the consecutive sets of ISs of R and S is calculated once $e_{IS}(i,j)$ is computed for all i,j. Let n and m denote the number of ISs in S and R respectively." [5] "Let $E(S_{IS}[i], R_{IS}[j]) = E[i,j]$ bet the smallest free energy of the interacting IS lists $S_{IS}[i]$ and $R_{IS}[j]$ interact with each other." [5]

Definition 1 *Let $S_{IS}[i]$ and $S_{IS}[j]$ be two ISs in a given RNA sequence S. The distance between $S_{IS}[i]$ and $S_{IS}[j]$, denoted $d(S_{IS}[i], S_{IS}[j])$ is defined as the number of nucleotides $S[k]$ that do not lie between a bonded pair of nucleotides $S[h]$ and $S['h]$ that are both located between $S_{IS}[i]$ and $S_{IS}[i]$.*

[5] To compute the value of $E[i,j]$, dynamic programming can be used as in the following: [5]

$$E[i,j] = min_{'i<i,'j<j|d(S_{IS}[i],S_{IS}[j])\leq(1+\epsilon)\cdot d(R_{IS}['i],R_{IS}[j]+\sigma}$$
$$(E['i,'j] + e_{IS}(i,j) + \Sigma_{'i<''i<i} e_{IS}(''i,0) + \Sigma_{'j<''j<j} e_{IS}(0,''j))$$

$$min_{\forall i,j} E[i,j] + \Sigma_{i<'i} e_{IS}('i,0) + \Sigma_{j<'j} e_{IS}(0,'j) \tag{8.6}$$

[5] It is also proven by [5] that "the above dynamic programming formulation correctly computes $E[i,j]$.

Searching "for target sequences for specific antisense RNA molecules in whole genomic and plasmid sequences," is a necessity for the algorithms presented. [5] "The Stacked Pair Model is not efficient when searching through large sequences." [5] The Loop Energy Model is used for the target prediction and contains two steps. [5]

1. "Using the cDNA annotation available for genomic sequences," [5] the "candidate target sequences...that is known to include the target," [5] is found. The "cDNA is extended towards the 5ánd 3ÚTR ends," [5] in order to "compute the potential mRNA." [5] The cDNA is extended as follows: [5]

   (a) "Each cDNA is extended up to l_1 nucleotides at its 5ÚTR, and by l_2 nucleotides a its 3ÚTR, where l_1 and l_2 are user defined parameters." [5]

   (b) The "extended cDNA sequence is trimmed from both ends via a dynamic programming routine in order to compute its subsequence which has the lowest energy density." [5] The prediction of the "resulting mRNA of each such cDNA" [5] is its "trimmed extension." [5]

2. The "joint secondary structure predication algorithms based on Loop Energy Model" is then run to "determine if there are any external bonds formed between each candidate target sequence and the antisense RNA sequence." [5] There are three constraints used.

   (a) "At least one IS in the candidate target sequence which lies before the start codon should interact with an independent subsequence in the query sequence." [5]

   (b) "All predicted interactions between pairs of ISs should include at least ξ uninterrupted bonds for some user specified constraint ξ." [5]

   (c) "At least two pairs of independent sequences must be interacting with each other." [5]

[5]

The program used "40 hrs on a PC equipped with 2GHz Pentium IV processor and i GB of main memory to detect all targets of the CopA sequence on the complete R1 plasmid." [5] The only potential target returned was the correct target CopT. [5]

Other tests returned similar results. [5] No other papers were located that discussed this area thus much research is need and many areas are open for further exploration.

RNA Microarray Design Conclusion

Since little research is available on this area, only two works were summarized and presented. The first is that of [6], who presented work on DNA-RNA and the second is that of [5], who presented work on RNA-RNA.

[6] attempted to develop a "direct, simple, rapid, accurate, and sensitive system for RNA detection." [6] This is different than that of the DNA since the DNA is an indirect approach while RNA is direct. [6] The system is "based on the RNA 3'-labelling by using labelled-dNTPs and DNA polymerase on a DNA template." [6] The result of the paper by Alsaidi et al is the design of a "novel system for RNA-specific detection on a microplate by immobilizing the hybrid templates and using an enzyme label." [6] It is suggested by [6] that sensitivity can be increased through use of smaller chips, as well as background reduction.

One of the problems facing RNA duplexes is that while algorithms exist to predict the "secondary structure of a single RNA molecule," [5] no algorithms have been developed to predict the "joint secondary structure of two interacting RNA molecules." [5] Thus one of the main focuses of [5] is the presentation of the "RNA-RNA interaction prediction (RIP) problem." [5] Free energy minimization is the focus of the algorithms for predicting the "secondary structure of two interactive RNA molecules." [5] No other papers were located that discussed this area thus much research is need and many areas are open for further exploration.

8.2 Digital Microfluidics-Based Biochips Design

Chakrabarty [35] presents design, testing, and applications of digital microfluidics-based biochips.

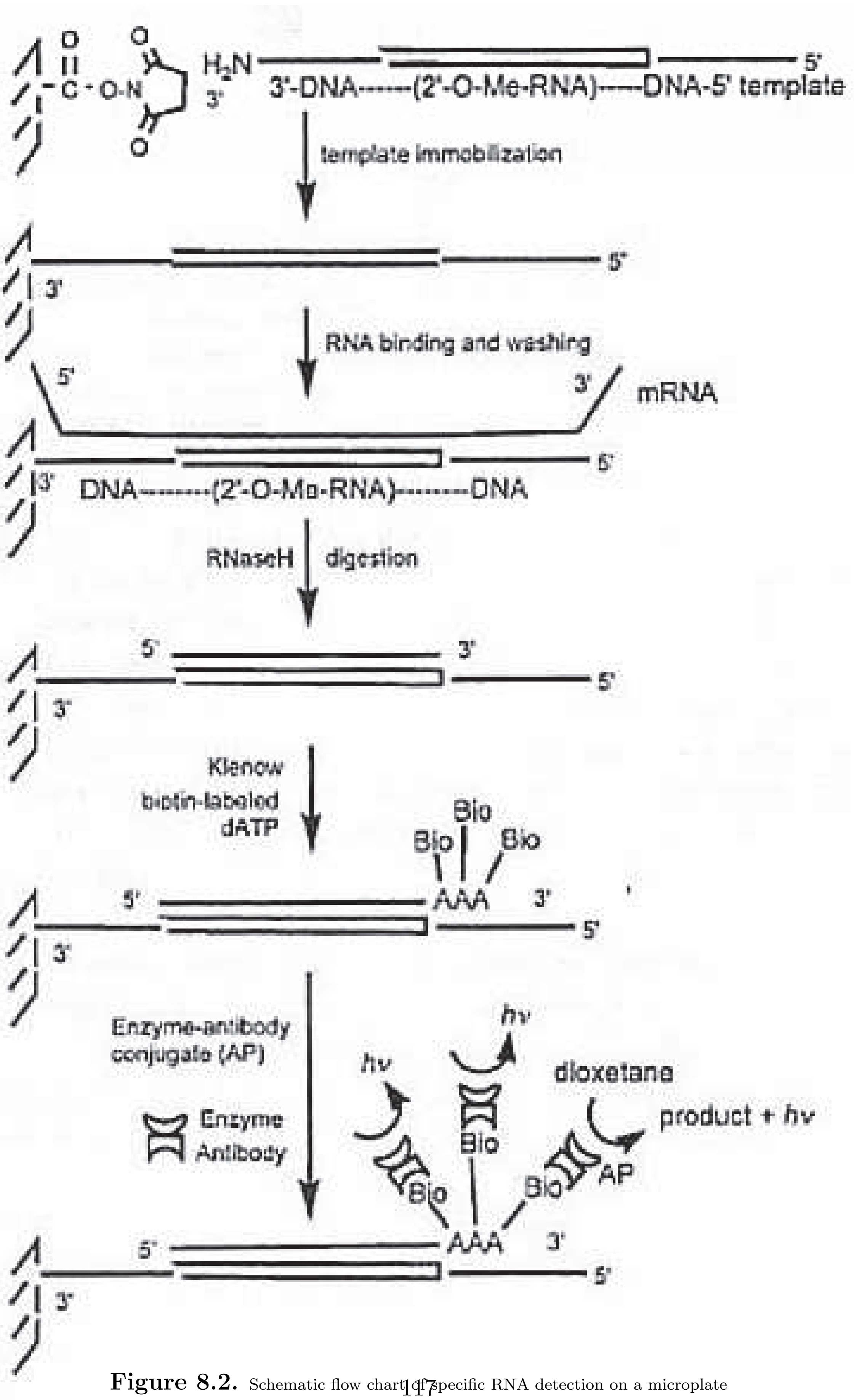


Figure 8.2. Schematic flow chart of specific RNA detection on a microplate

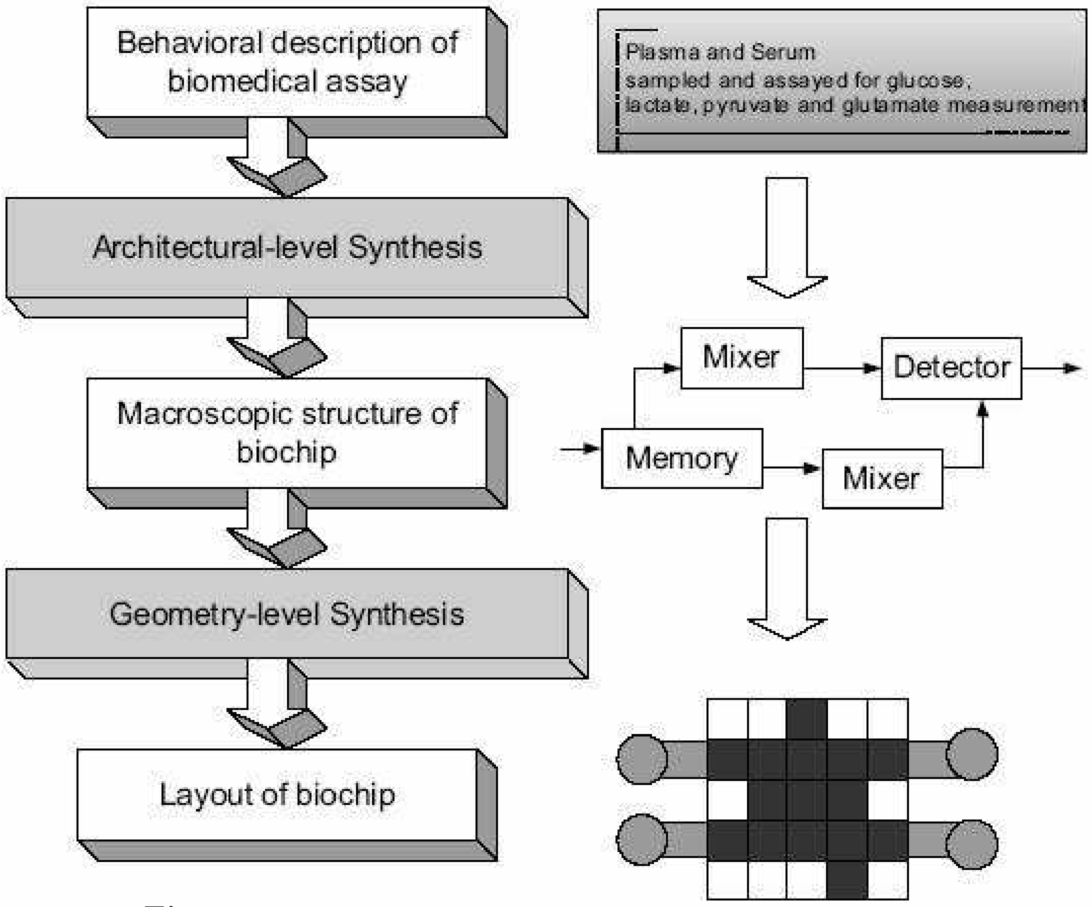

Behavioral description of biomedical assay
Architectural-level Synthesis
Macroscopic structure of biochip
Geometry-level Synthesis
Layout of biochip
Plasma and Serum sampled and assayed for glucose, lactate, pyruvate and glutamate measurement
Mixer
Detector
Memory
Mixer

Figure 8.3. Synthesis methodology for digital microfluidics-based biochips.

CHAPTER 9

RELATED PUBLICATIONS

1. M. Atlas, **N. Hundewale**, S. Datta, ”Parallel Statistical-Validation of Clustering Algorithm for the Analysis of High Throughput Data”, Bioinformatics Oxford University Press (under review).

2. M. Atlas, **N. Hundewale**, S. Datta, ”Parallel Statistical-Validation of Clustering Algorithm for the Analysis of High Throughput Data”, Research Louisville 2005 (Best third poster Award).

3. W. Mao, **N. Hundewale**, S. Gremalschi, D. Brinza, A. Zelikovsky, Genotype Susceptibility using Decision Fusion Methods 2006 IEEE International Conference on Granular Computing (IEEE-GrC2006) pp. 754-757.

4. **N. Hundewale**, A. Zelikovsky, I. Mandoiu, P. Prajecue, ”Integrated Design of Tag System” Proc. The Ninth Annual International Conference on Research in Computational Molecular Biology (RECOMB’05), Cambridge, MA, May 2005, refereed poster.

5. M. Atlas, **N. Hundewale**, L. Perelygina, A. Zelikovsky, ”Consolidating Software Tools for DNA Microarray Design and Manufacturing,” Proc. International Conf. of the IEEE Engineering in Medicine and Biology (EMBC’04), September 2004, 172-175.

CHAPTER 10
OTHER PUBLICATIONS

1. **N. Hundewale**, S. Jung, A. Zelikovsky, "Energy Efficient Node Caching and Load Balancing Enhancement of Reactive Ad Hoc Routing Protocols", Journal of Universal Computer Science (JUCS), Springer, vol. 13 (2007), to appear in the special issue.

2. Q. Cheng, Y. Zhang, X. Hu, **N. Hundewale**, A. Zelikovsky, "Routing Using Messengers in Sparse and Disconnected Mobile Sensor Networks", Atlantic Web Intelligence Conference 2006, Studies in Computational Intelligence 26, pp. 31-40.

3. **N. Hundewale**, Q. Cheng, X. Hu, A. Bourgeois, A. Zelikovsky, Autonomous Messenger Based Routing Technique in Disjoint Clusters of Mobile Sensor Networks Agent-Directed Simulation 2006 Huntsville, AL, April 2006, pp. 57-64.

4. S. Jung, **N. Hundewale**, A. Zelikovsky, "Energy Efficiency of Load Balancing in MANET Routing Protocols," Proc. International Workshop on Self-assembling Wireless Networks, (SAWN 2005). pp 476-483.

5. S. Jung, **N. Hundewale**, A. Zelikovsky, "Node Caching Enhancement of Reactive Ad Hoc Routing Protocols," Proc. IEEE Wireless Communication and Networking Conference (WCNC'05), March 2005, Volume 4, pp 1970- 1975.

BIBLIOGRAPHY

[1] Miron Abramovici and Melvin A. Breuer and Arthur D. Friedman, (1993) ‘Digital Systems Testing and Testable Design’, *Wiley-IEEE Press.*

[2] Affymetrix Inc. ‘GenFlex Tag Array Technical Note No. 1’, *http://www.affymetrix.com/support/technical/technotes/genflex_technote.pdf.*

[3] http://www.affymetrix.com

[4] S.B. Akers, (1981)‘On the use of the linear assignment algorithm in module placement’, *Proc. 18th Design Automation Conference (DAC 1981)*, pp. 137–144.

[5] Can Alkan, Emre Karakoc, Joe Nadeau, S Cenk Sahinalp and Kaizhong Zhang (2005) ‘RNA-RNA Interaction Prediction and Antisense RNA Target Search’, *Proc. of the Ninth Annual International Conference on Research in Computational Molecular Biology (RECOMB 2005 - LNBI 3500)*, pp. 152–171.

[6] Mohammed Alsaidi and Elena Lum and Zhen Huang (2004) ‘Direct Detection of a Specific Cellular mRNA on Functionalized Microplate’, *ChemBioChem*, pp. 1136-1139.

[7] C. J. Alpert and A. B. Kahng, (1994) ‘Multi-Way Partitioning Via Spacefilling Curves and Dynamic Programming’, *Proc. ACM/IEEE Design Automation Conf.*, pp. 652-657.

[8] C. J. Alpert and A. B. Kahng, (1995) ‘Multi-Way Partitioning Via Geometric Embeddings, Orderings, and Dynamic Programming’, in *IEEE Trans. on CAD* 14(11), pp. 1342-1358.

[9] C. J. Alpert, A. B. Kahng and D. S. Yao, (1998) ‘Spectral Partitioning: the More Eigenvectors, the Better’, *Discrete Applied Mathematics.*

[10] Anderson, M. (2001) ‘Crohn’s: An Autoimmune or Bacteria-Related Disease?’, *The Scientist*, 22:15-16.

[11] A.A. Antipova, P. Tamayo and T.R. Golub, (2002)‘ A strategy for oligonucleotide microarray probe reduction’, *Genome Biology* 3(12):research0073.1-0073.4.

[12] M. Atlas, **N. Hundewale**, L. Perelygina and A. Zelikovsky (2004) ‘Consolidating Software Tools for DNA Microarray Design and Manufacturing’, *Proc. IEEE International Conference of Engineering in Medicine and Biology (EMBC)*, pp. 172–175.

[13] M. Atlas, **N. Hundewale** and S. Datta (2005) ‘Parallel Statistical-Validation of Clustering Algorithm for the Analysis of High Throughput Data’, *Research Louisville 2005.*

[14] Marco Autili, Paola Inverardi and Massimo Tivoli,(2004) ‘Automatic Adaptor Synthesis for Protocol Transformation’, *First International Workshop on Coordination and Adaptation Techniques for Software Entities (WCAT04) in conjunction with ECOOP.*

[15] K. Nandan Babu and S. Saxena, (1997) ‘Parallel algorithms for the longest common subsequence problem’, *Proc. 4th Intl. Conf. on High-Performance Computing*, pp. 120-125.

[16] Dusan Balek and Franktisek Plasil, (2001) 'Software and Connectors and Their Role in Component Deployment', *In Proceedings of DAIS'01, Krakow, Kluwer.*

[17] Johan Baner, Paul Hardenbol, Mats Nilsson, Eugeni A. Namsaraev, George A Karlin-Neumann, Hossein Fakhrai-Rad, Mostafa Ronaghi, Thomas D. Willis, Ulf Landegren and Ronlad W. Davis (2003) 'Multiplexed genotyping with sequence-tagged molecular inversion probes', *Nature Biotechnology 21*, pp. 673–678.

[18] J. J. Bartholdi and L. K. Platzman, (1982) 'An O(N log N) Planar Travelling Salesman Heuristic Based On Spacefilling Curves', *Operations Research Letters* 1(4), pp. 121-125.

[19] A. BenDor, R. Karp, B. Schwikowski and Z. Yakhini (2000) 'Universal DNA tag systems: a combinatorial design scheme', *Journal of computational Biology*, vol. 7(3-4), pp. 503–519.

[20] A. BenDor, T. Hartman, B. Schwikowski, R. Sharan and Z. Yakhini (2003) 'Towards optimally mulitplexed applications of universal DNA tag systems', *Proc. 7th Annual International Conference on Research in Computational Molecular Biology*, pp. 48–56.

[21] Casey Best, Margaret-Anne Storey, and Jeff Michaud (2002) 'Designing a component-based framework for visualization in software engineering and knowledge engineering', *SEKE 2002* pp. 323–322.

[22] http://www.bioperl.org

[23] Bjerre LM, Verheij TJM, Kochen MM (2003) 'Antibiotics for community acquired pneumonia in adult outpatients', *Clinical Evidence* 10:1724-37.

[24] M. Borodovsky, 'GeneMark', *http://opal.biology.gatech.edu/GeneMark.*

[25] Botstein, D., Risch, N. (2003) ‘Discovering Genotypes Underlying Human Phenotypes: Past Successes for Mendelian Disease, Future Approaches for Complex Disease’, *Nature Genetics*, 33:228-237.

[26] J. Branke and M. Middendorf, (1996) ‘Searching for shortest common supersequences by means of a heuristic-based genetic algorithm’, *report*, July 1996. See also: ‘Searching for shortest common supersequences’, *Proc. Second Nordic Workshop on Genetic Algorithms and Their Applications*, pp. 105-113.

[27] J. Branke, M. Middendorf and F. Schneider, (1998) ‘Improved heuristics and a genetic algorithm for finding short supersequences’, *OR Spektrum* 20(1), pp. 39-46.

[28] Breiman, L. and Cutler, A. http://www.stat.berkeley.edu/users/breiman/RandomForests/

[29] S. Brenner (1997) ‘Methods for sorting polynucleotides using oligonucleotide tags’, *US Patent 5,604,097.*

[30] Brinza, D. and Zelikovsky, A. (2006) 2SNP: Scalable Phasing Based on 2-SNP Haplotypes, *Bioinformatics.*

[31] Kyle Brown, Phillip Eskelin, and Nat Pryce (1999) ‘A Mini-pattern language for Distributed Component Design’, *Pattern Languages of Programs (PLoP) Conference.*

[32] Lubomir Bulej and Tomas Bures (2003) ‘A Connector Model Suitable for Automatic Generation of Connectors’, *Tech. Report No. 2003/1, Dep. of SW Engineering, Charles University, Prague.*

[33] A. E. Caldwell, A. B. Kahng and I. L. Markov, (1999)‘Optimal Partitioners and End-Case Placers for Standard-Cell Layout’, *Proc. ACM Intl. Symp. on Physical Design*, pp. 90-96.

[34] Carlos Canal, (2004) ‘On The Dynamic Adaptation Of Component Behaviour’, *First International Workshop on Coordination and Adaptation Techniques for Software Entities (WCAT04) in conjunction with ECOOP.*

[35] Krishnendu Chakrabarty (2005) ‘Design, Testing, and Applications of Digital Microfluidics-Based Biochips’, *Proceedings of the 18th International Conference on VLSI Design held jointly with 4th International Conference on Embedded Systems Design (VLSID05) IEEE* pp.

[36] T. F. Chan, J. Cong, T. Kong and J. R. Shinnerl, (2000)‘Multilevel Optimization for Large-Scale Circuit Placement’, *Proc. IEEE/ACM Intl. Conf. on Computer Aided Design*, pp. 171-176.

[37] Chu S., DeRisi J., et al., (1998) ‘The transcriptional program of sporulation inv budding yeast’, *Science*, 282, pp. 699705.

[38] Ciprian-Bogdan Chirila, Pierre Crescenzo, and Philippe Lahire (2004) ‘A Reverse Inheritance Relationship Dedicated to Reengineering: The Point of View of Feature Factorization’, *MASPEGHI Workshop at ECOOP 2004, MechAnisms for SPEcialization, Generalization and inHerItance Oslo.*

[39] Clark AG. (2003) ‘Finding Genes Underlying Risk of Complex Disease by Linkage Disequilibrium Mapping’, *Curr Opin Genet Dev.*, 13(3):296-302.

[40] Clifton, T., and Teahan, W. J. (2004) ‘Question Answering with Knowledgeable Agents’, *Artificial Intelligence and Intelligent Agents* vol. 04.4.

[41] Clinical Epidemiology Glossary, http://www.med.ualberta.ca/ebm/define.htm.

[42] D. R. Cutting, D. R. Karger, J. O. Pederson and J. W. Tukey, (1992) 'Scatter/Gather: A Cluster-Based Approach to Browsing Large Document Collections', (15th Intl. ACM/SIGIR Conference on Research and Development in Information Retrieval) *SIGIR Forum*, pp. 318-329.

[43] Daly, M., Rioux, J., Schaffner, S., Hudson, T. and Lander, E. (2001) 'High Resolution Haplotype Structure in the Human Genome', *Nature Genetics*, 29:229-232.

[44] V. Dancik, (1998) 'Common subsequences and supersequences and their expected length', *Combinatorics, Probability and Computing* 7(4), pp. 365-373.

[45] B. DasGupta, K.M. Konwar, I.I. Mandoiu, and A.A. Shvartsman (2005) 'Highly scalable algorithms for robust string barcoding', *Proc. 2005 International Workshop on Bioinformatics Research and Applications (IWBRA'05).*

[46] Datta, S and Datta, S. (2003) 'Comparisons and validation of statistical clustering techniques for microarray gene expression data', *Bioinformatics*, 19, 459-466.

[47] K. Doll, F. M. Johannes, K. J. Antreich, (1994) 'Iterative Placement Improvement by Network Flow Methods', *IEEE Transactions on Computer-Aided Design* 13(10), pp. 1189-1200.

[48] Hiroshi Douzono, Shigeomi Hara and Yoshio Noguchi (2001) 'A Design Method of DNA chips for SNP Analysis Using Self Organizing Maps', *Proc. IEEE*, pp.

[49] Scott F. Dowell, M.D., M.P.H. (2004) 'Surviving Pneumonia-Just a Short-Term Lease on Life?', *American Journal of Respiratory and Critical Care Medicine* Vol 169. pp. 895-896.

[50] http://www.photomasks.com/

[51] Eggimann P, Revelly JP. (2006) 'Should antibiotic combinations be used to treat ventilator-associated pneumonia?', *Semin Respir Crit Care Med* 27(1):68-81.

[52] Scott J. Emrich1, Mary Lowe and Arthur L. Delcher (2003) ‘PROBEmer: a web-based software tool for selecting optimal DNA oligos’, *Nucleic Acids Research* Vol. 31, No. 13, pp. 3746-3750.

[53] Philip Eskelin (1999) ‘Layering Frameworks in Component-Based Development’, *Proceedings of Pattern Languages of Programs.*

[54] W. Feldman and P.A. Pevzner, (1994) ‘Gray code masks for sequencing by hybridization’, *Genomics*, 23, pp. 233–235.

[55] Thomas M. File Jr, MD, James S. Tan, MD, Joseph F. Plouffe, MD.(1998). Lower respiratory tract infections the role of atypical pathogens: mycoplasma pneumoniae, chlamydia pneumoniae, and legionella pneumophila in respiratory infection’, *Infectious Disease Clinics of North America* Volume 12, Number 3.

[56] S. Fodor, J. L. Read, M. C. Pirrung, L. Stryer, L. A. Tsai and D. Solas (1991) ‘Light-Directed, Spatially Addressable Parallel Chemical Synthesis’, *Science* 251, pp. 767-773.

[57] D. E. Foulser, M. Li and Q. Yang, (1992) ‘Theory and algorithms for plan merging’, *Artificial Intelligence* 57(2-3), pp. 143-181.

[58] C. B. Fraser and R. W. Irving, (1995) ‘Approximation algorithms for the shortest common supersequence’, *Nordic J. Computing* 2, pp. 303-325.

[59] Gabriel, G., Schaffner, S., Nguyen, H., Moore, J., Roy, J., Blumenstiel, B., *et al.* (2002) ‘The Structure of Haplotype Blocks in the Human Genome’, *Science*, 296:2225-2229.

[60] Martin Gaedke, Joern Rehse, and Guntram Graef (1999) ‘A Repository to Facilitate Reuse in Component-Based Web Engineering’, *International Work-*

shop on Web Engineering at the 8th International World-Wide Web Conference (WWW8), Toronto, Canada.

[61] http://www.ncbi.nlm.nih.gov

[62] N.P. Gerry, Nancy Witowski, Joseph Dat, Robert P. Hammer, George Barany and Francis Branay (1999) 'Universal DNA Mutations', *Journal of Molecular Biology*, vol. 292, pp. 251–262.

[63] D.H. Geschwind and J.P. Gregg (Eds.), (2002)*Microarrays for the neurosciences: an essential guide*, MIT Press, Cambridge, MA.

[64] GLPK (2000). GNU Linear Programming Kit. http://www.gnu.org.

[65] S. Hannenhalli, E. Hubbell, R. Lipshutz and P.A. Pevzner, (2002) 'Combinatorial Algorithms for Design of DNA Arrays', *Chip Technology*, (ed. J. Hoheisel), Springer-Verlag.

[66] The International HapMap Project, http://www.hap map.org

[67] J. He and Z. Yakhini (2004) 'Linear Reduction for Haplotype Inference', *Proc. Workshop on Algorithms in Bioinformatics (WABI)*, pp. 352–371.

[68] N.Y He, P.F. Xiao, Z.C. Ziu and Z.H. Lu (2001) 'Soft Lithography for Oligonucleotide Arrays Fabrication', *Proc. of the 23rd Annual EMBS International Conf.*, pp.

[69] Earl A. Hubbel and David P. Smith, (2000)'Techniques for Synthesis Integrity Evaluation Utilizing Cycle Fidelity Probes', *United States Patent*, patent number 6,130,046.

[70] D. J. Huang and A. B. Kahng, (1997)'Partitioning-Based Standard Cell Global Placement With an Exact Objective', *Proc. ACM Intl. Symp. on Physical Design*, pp. 18-25.

[71] E. Hubbell and M. Mittman , (2002)‘Personal communication’, *Affymetrix, Santa Clara, CA*, July 2002.

[72] **Nisar Hundewale**, Ion Mandoiu, Claudia Prajescu and Alexander Zelikovsky (2005) ‘Integrated Design Flow for Universal DNA Tag Arrays’, *Proc. of the Ninth Annual International Conference on Research in Computational Molecular Biology (RECOMB 2005 - LNBI 3500)* pp.

[73] Hussain S, McCurry K, Dauber J, Singh N, Kusne S.(2002) ‘Nocardia infection in lung transplant recipients’, *Journal of Heart Lung Transplant* 21(3):354-9.

[74] Ayaz Isazaheh (2004) ‘Software Engineering: Integration’, *Applied And Computational Mathematics: An International Journal* vol. 3:1, pp. 56–66.

[75] T. Jiang and M. Li, (1995) ‘On the approximation of shortest common supersequences and longest common subsequences’, *SIAM J. Computing* 24(5), pp. 1122-1139.

[76] Johnson, G.C.L., Esposito, L., Barratt, B.J., Smith, A.N., Heward, *et al.* (2001) ‘Haplotype Tagging for the Identification of Common Disease Genes’, *Nature Genetics*, 29:233-237.

[77] Leon Jololian and Murat M. Tanik (2001) ‘A Framework for a Meta-Semantic Language for Smart Component-Adapters’, *Journal of Sytems Integration* vol. 10(3), pp. 269–297.

[78] Lars Kaderali and Alexander Schiep (2002) ‘Selecting signature oligonucleotides to identify organisms using DNA arrays’, *Bioinformatics 18* , pp. 1340–1349.

[79] A.B. Kahng, I.I. Măndoiu, P.A. Pevzner, S. Reda, and A. Zelikovsky, (2002)‘Border Length Minimization in DNA Array Design’, *Proc. 2nd International Workshop on Algorithms in Bioinformatics (WABI 2002),* R. Guigó and D. Gusfield

(Eds.), Springer-Verlag Lecture Notes in Computer Science Series 2452, pp. 435-448.

[80] A.B. Kahng, I.I. Măndoiu, P.A. Pevzner, S. Reda, and A. Zelikovsky, (2003)'Engineering a Scalable Placement Heuristic for DNA Probe Arrays', *Proc. 7th Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, pp.

[81] Andrew B. Kahng, Ion Măndoiu, Sherief Reda, Xu Xu and Alex Z. Zelikovsky (2003) 'Evaluation of Placement Techniques for DNA Probe Array Layout', *Proc. IEEE International Conference on Computer Design(ICCD)*, pp. 116–123.

[82] S. Kasif, Z. Weng, A. Derti, R. Beigel, and C. DeLisi, (2002) 'A computational framework for optimal masking in the synthesis of oligonucleotide microarrays', *Nucleic Acids Research* vol. 30, e106.

[83] Keeble S. A. (1960) 'B virus infection in monkeys', *Ann. N. Y. Acad. Sci.* 85:960969.

[84] Keeble S. A., Christofinis G. J. and Wood W. (1958) 'Natural B virus infection in rhesus monkeys', *J. Pathol. Bacteriol*, 76:189-199.

[85] Kimmel, G. and Shamir R.. (2005) A Block-Free Hidden Markov Model for Genotypes and Its Application to Disease Association. J. of Computational Biology, Vol. 12, No. 10: 1243-1260.

[86] T. Kozawa et al., (1983)'Automatic Placement Algorithms for High Packging Density VLSI', *Proc. 20th Design Automation Conference (DAC 1983)*, pp. 175–181.

[87] J. Kevin Lanctot, Ming Li and En-hui Yang ‘Estimating DNA sequence entropy’, *SODA ’00: Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms* (2000), pp. 409–418.

[88] Joachims, T. http://svmlight.joachims.org/

[89] R.J. Lipshutz, S.P. Fodor, T.R. Gingeras, D.J. Lockhart, (1999) ‘High density synthetic oligonucleotide arrays’, *Nat. Genet.* 21, pp. 20–24.

[90] F. Li and G.D. Stormo, (2001)‘Selection of optimal DNA oligos for gene expression arrays,’ *Bioinformatics* 17(11), pp. 1067-1076.

[91] Listgarten, J., Damaraju, S., Poulin B., Cook, L., Dufour, J., Driga, A., Mackey, J., Wishart, D., Greiner,R., and Zanke, B.. (2004) Predictive Models for Breast Cancer Susceptibility from Multiple Single Nucleotide Polymorphisms. *m*phClinical Cancer Research, Vol. 10, 2725C2737, 2004.

[92] Francisca Losavio, Dinarle Ortega, and Mara A. Prez (2002) ‘Modeling EAI’, *SCCC 2002* pp. 195–203.

[93] I. Mandoiu, C. Prajescu, and D. Trinca (2005) ‘Improved Tag Set Design and Multiplexing Algorithms for Universal Arrays’, *Proc. International Workshop on Bioinformatics Research and Applications (IWBRA)* pp. .

[94] Ion Mandoiu and D. Trinca (2005) ‘Exact and Approximation Algorithms for DNA Tag Set Design’, *Proc. 16th Annual Symposium on Combinatorial Pattern Matching (CPM 2005)*, pp.

[95] Mao, W., He, J., Brinza D. and Zelikovsky, A. (2005) ‘A Combinatorial Method for Predicting Genetic Susceptibility to Complex Diseases’, Proc. International Conf. of the IEEE Engineering in Medicine and Biology (EMBC’05).

[96] W. Mao, **N. Hundewale**, S. Gremalschi, D. Brinza, A. Zelikovsky, (2006) 'Genotype Susceptibility using Decision Fusion Methods', *IEEE International Conference on Granular Computing (IEEE-GrC2006)*, pp. 754-757.

[97] McCracken G. H. Jr. (2000) 'Diagnosis and Management of pneumonia in children', *Pediatr. Infect. Dis J.* 19(9):924-8.

[98] B. Medjahed, B. Benatallah, A. Bouguettaya, A. H. H. Ngu, and A. K. Elmagarmid (2003) 'Business-to-business interactions: issues and enabling technologies', *The VLDB Journal - The International Journal on Very Large Data Bases* vol. 12:1.

[99] Merikangas, KR., Risch, N. (2003) 'Will the Genomics Revolution Revolutionize Psychiatry', *The American Journal of Psychiatry*, 160:625-635.

[100] Michelow Ian C., Olsen Kurt, BS, Lozano Juanita, Rollins Nancy K., Ziegler Thedi, Kauppila Jaana, Leinonen Maija and McCracken George H. (2004) 'Epidemiology and Clinical Characteristics of Community Acquired Pneumonia in Hospitalized Children', *Pediatrics* Vol. 113 No. 4, pp. 701-707.

[101] http://www.mosis.org

[102] Mikhail V. Naganov (2005) 'Towards Automatic Generation of Q Adaptors', *IASTED International Conference on Software Engineering in Innsbruck, Austria.*

[103] NIH, 'Orf finder', *http://www.ncbi.nih.gov/gorf/gorf.html.*

[104] Michael Ostapchuk, M.D., Donna M. Roberts, M.D., and Richard Haddy, M.D. (2004) 'Community-Acquired Pneumonia In Infants and Children', *American Family Physician*70:899-908.

[105] Palmer A. E., (1987) 'B virus, herpes virus simile: historical perspective', *J. Med. Primatol.*, 16:99-130.

[106] Perelygina L., Zhu L., Zurkuhlen H., Mills R., Borodovsky M. and Hilliard J.K. (2003) 'Complete Sequence and Comparative Analysis of the Genome of Herpes B Virus (Cercopithecine herpesvirus 1) from a Rhesus Monkey', *J. Virology*, 77(11):61676177.

[107] B. T. Preas and M. J. Lorenzetti, eds., (1988)'Physical Design Automation of VLSI Systems', *Benjamin-Cummings.*

[108] Paul Pop, 'A Survey of Three Approaches to the Automation of Design Patterns', *Lecture Notes, Computer and Information Science Dept. Linkpings universitet.*

[109] http://www.perlegen.com

[110] Sven Rahmann, (2002) 'Rapid large-scale oligonucleotide selection for microarrays', *Proc. IEEE Computer Society Bioinformatics Conference (CSB'02).*

[111] Sven Rahmann (2003) 'Fast and Sensitive Probe Selection for DNA Chips Using Jumps in matching Statistics', *Proc. of the computational Systems Bioinformatics*, pp.

[112] Resor, III , et al. (1992)'Apparatus and method for making large area electronic devices, such as flat panel displays and the like, using correlated, aligned dual optical systems', *US Patent RE33836.*

[113] Alexander Schliep and Sven Rahmann (2003) 'Group Testing with DNA Chips: Generating Designs and Decoding Experiments', *Proc. of the Computational Systems Bioinformatics*, pp.

[114] R. Sengupta and M. Tompa, (2002) 'Quality Control in Manufacturing Oligo Arrays: a Combinatorial Design Approach', *Journal of Computational Biology* 9, pp. 1–22.

[115] K. Shahookar and P. Mazumder, (1991) 'VLSI Cell Placement Techniques', *Computing Surveys* 23(2), pp. 143-220.

[116] http://www.sigma-genosys.com

[117] H. Simmler, H. Singpiel and R. Manner (2003) 'Real-Time Primer Design for DNA Chips', *IEEE Computer Society*, pp.

[118] Gerald Benoit (2003) 'Bioinformatics', , pp.

[119] Kaleigh Smith (2002) 'Universal Microarrays: An Algorithmic Approach', , pp.

[120] L. Steinberg, (1961)'The backboard wiring problem: a placement algorithm", *SIAM Review* 3, pp. 37–50.

[121] Wing-Kin Sung and Wah-Heng Lee (2003) 'Fast and Accurate Probe Selection Algorithm for Large Genomes', *Proc. of the computational Systems Bioinformatics*, pp.

[122] V. G. Timkovsky, (1993) 'Ten etudes on shortest common nonsubsequence and supersequence approximations', *XXX*, pp.

[123] A.C. Tolonen, D.F. Albeanu, J.F. Corbett, H. Handley, C. Henson, and P. Malik, (2002) 'Optimized in situ construction of oligomers on an array surface', *Nucleic Acids Research* vol. 30, e107.

[124] Tomita, Y., Yokota, M. and Honda, H. (2005) Classification method for prediction of multifactorial disease development using interaction between genetic and environmental factors, *IEEE computational systems bioinformatics conference*, abstract.

[125] http://www.eetimes.com/semi/news/OEG20020517S0074

[126] Ueda, H., Howson, J.M.M., Esposito, L. et al. (2003) 'Association of the T Cell Regulatory Gene CTLA4 with Susceptibility to Autoimmune Disease', *Nature*, 423:506-511.

[127] C. Pierrat, (2002)'Multiple image reticle for forming layers', *US Patent*, July 2002.

[128] C. Holmes, (2002)'Cyclic and substituted immobilized molecular synthesis', *US Patent 6468740*, October 2002.

[129] Waddell, M., Page,D., Zhan, F., Barlogie, B., and Shaughnessy, J..(2004) Predicting Cancer Susceptibility from SingleNucleotide Polymorphism Data: A Case Study in Multiple Myeloma. *Proceddings of BIOKDD 2005*, 05, 2005.

[130] Wang X, Seed B. (2003) 'Selection of oligonucleotide probes for protein coding sequences', *Bioinformatics* 19(7):796-802.

[131] J.A. Warrington, R. Todd, and D. Wong (Eds.) (2002) 'Microarrays and cancer research', *BioTechniques Press/Eaton Pub.*, Westboro, MA.

[132] Weigler B. J., (1992) 'Biology of B virus in macaque and human hosts: a review', *Clin. Infect. Dis.*, 14:555-567.

[133] Zhang, K., Calabrese, P., Nordborg, M., Sun, F. (2002) 'Haplotype Block Structure and Its Applications in Association Studies: Power and Study Design', *The American Journal of Human Genetics*, 71:1836-1894.