# Exact and Approximation Algorithms for DNA Tag Set Design

by

Dragoş N. Trincă

B.S., A.I.Cuza University (Romania), 2003

A Thesis

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

at the

University of Connecticut

2005

APPROVAL PAGE

Master of Science Thesis

# Exact and Approximation Algorithms for DNA Tag Set Design

Presented by

Dragoş N. Trincă

Major Advisor: _____
Ion I. Mandoiu

Associate Advisor: _____
Sanguthevar Rajasekaran

Associate Advisor: _____
Alexander Russell

University of Connecticut

2005

**Abstract**

In this thesis we propose new solution methods for designing tag sets for use in universal DNA arrays. First, we establish upper bounds for an extended version of a previous formalization. Second, we give integer linear programming formulations for two previous formalizations of the tag set design problem, and show that these formulations can be solved to optimality for instance sizes of practical interest by using general purpose optimization packages. Third, we note the benefits of periodic tags, and establish an interesting connection between the tag design problem and the problem of packing the maximum number of vertex-disjoint directed cycles in a given graph. We show that combining a simple greedy cycle packing algorithm with a previously proposed alphabetic tree search strategy yields an increase of over 40% in the number of tags compared to previous methods. Most of the results presented in this thesis have already appeared in the following publications [31, 30]:

1. I.I. Mandoiu and D. Trinca. Exact and Approximation Algorithms for DNA Tag Set Design. In *Proceedings of the 16th Annual Symposium on Combinatorial Pattern Matching (CPM 2005)*, June 19–22, 2005, Korea. Lecture Notes in Computer Science, volume 3537, pages 383–393, Springer-Verlag. Extended version accepted to *Journal of Computational Biology*, and also available as ACM Computing Research Repository report cs.DS/0503057.

2. I.I. Mandoiu, C. Prajescu, and D. Trinca. Improved Tag Set Design and Multiplexing Algorithms for Universal Arrays. In *Proceedings of the 5th International Conference on Computational Science (ICCS 2005)/ 2005 International Workshop on Bioinformatics Research and Applications (IWBRA)*, May 22–25, 2005, Atlanta, GA, USA. Lecture Notes in Computer Science, volume 3515, pages 994–1002, Springer-Verlag. Extended version appeared in *LNCS Transactions on Computational Systems Biology*, volume 3680, pages 124–137, 2005, Springer-Verlag; also available as ACM Computing Research Repository report cs.DS/0502054.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Oligonucleotides are short single-stranded pieces of DNA (typically 15-50 nucleotides) made by chemical synthesis. In solution, oligonucleotides tend to specifically hybridize with their Watson-Crick complements ([41]), and form a stable DNA duplex. This specificity is exploited in molecular hybridization assays, in which oligonucleotides are used as probes to identify any complementary (or near complementary) DNA from a complex mixture of target DNA.

Array-based hybridization assays, introduced in the late 1980s [14, 23, 26, 38, 8, 12], offer the possibility of simultaneously monitoring a multitude (currently up to tens of thousands) of hybridization reactions. In such an assay, a target-specific set of oligonucleotides is synthesized on a solid support surface (e.g., silicon or glass). A fluorescently labeled target sample mixture of DNA or RNA fragments is then brought in contact with the treated surface, and allowed to hybridize with the synthesized oligonucleotides. Scanning the fluorescent labels of the fragments attached to the array reveals information about the content of the sample mixture. Theoretically, the assay conditions are such that hybridization only occurs in sites on the surface that are Watson-Crick complements to some substring in the target. In practice, cross-hybridization is a main source of cross-signal contamination in any array-based hybridization assay.

Array-based hybridization assays show great potential for many different applications such as SNP genotyping [19], gene expression profiling [3], and resequencing DNA [24, 19]. Recently, S. Brenner and others [10, 28] suggested an alternative approach based on universal arrays containing oligonucleotides called *antitags*. The Watson-Crick complement of each antitag is called a *tag*. The tag-antitag pairs are designed so that

each tag hybridizes strongly to its complementary antitag, but not to any other antitag. In this approach, the analysis of a DNA sample consists of two steps: solution-phase hybridization followed by solid-phase hybridization. In the first step, hybridization takes place between the target DNA in solution and a set of oligonucleotide precursors called reporter molecules. Each reporter molecule consists of a target-specific part ligated to a unique tag. Reporter/target hybridization events are registered (e.g., by an enzymatic reaction). In the second step the modified precursors are introduced to the array. Tags form duplexes with the corresponding antitags. Thus, the reporter molecules are sorted into different locations on the array and hybridization events can be determined. This approach has several advantages:

- Complicated array manufacturing processes are required only for the fixed, universal component of the assay. These universal components can therefore be mass-produced, significantly reducing manufacturing costs.

- The assay components that need to be designed for a specific target are involved in solution phase processes. The underlying nucleic acid chemistry and thermodynamics are better understood than the same aspects of surface-based processes. Therefore a more efficient and effective design process is facilitated.

As an example, we describe a multiplexed SNP genotyping assay. SNPs (single nucleotide polymorphisms) are differences, across the population, in a single base, within an otherwise conserved genomic sequence [15]. Genotyping is a process that determines the variants present in a given sample, over a set of SNPs. This assay uses off-the-shelf universal components: a universal set of oligonucleotide tags and a universal array of antitags. The antitags, immobilized on the array, are Watson- Crick complements of the tags in the mixture. The whole system will be called a DNA Tag/AntiTag system and in short a DNA TAT system. Consider a set of SNPs to be genotyped. The assay is performed as follows:

1. A set of reporter molecules (one for each SNP) is synthesized in solution. Each reporter molecule consists of two parts that are ligated (in string language: concatenated) together. The first part is the Watson-Crick complement of the upstream

sequence that immediately precedes the polymorphic site of the SNP. The second part of each reporter molecule is a unique tag from the universal set of tags.

2. When an individual is to be genotyped, a sample is prepared that contains the sequences flanking each of the SNP loci. The sample is mixed with the reporter molecules. Solution-phase hybridization then takes place. Assuming that specificity is perfect, this results in the flanking sequences of the SNPs paired only with the appropriate reporter molecule.

3. Single nucleotides, `A`,`C`,`T`,`G`, fluorescently labeled with four distinct colors, are added to the mixture. These labeled nucleotides hybridize to the polymorphic site of each SNP and are ligated to the corresponding reporter molecule. That is, each reporter molecule is extended by exactly one labeled nucleotide.

4. The extended reporter molecules are separated from the sample fragments, and brought into contact with the universal array. Assuming that specificity is perfect, the tag part of each reporter molecule will only hybridize to its complementary antitag on the array. Thus the extended reporter molecules sort into the array sites where the corresponding antitag is present.

5. For each site of the array, the fluorescent colors present at that site are detected. The colors indicate which bases were used for the extension at the corresponding SNP site, and thus reveal the SNP variations present in the individual.

## 1.1 DNA Microarrays

A DNA microarray consists of a solid surface, usually a microscope slide, onto which DNA molecules have been chemically bonded. The purpose of a microarray is to detect the presence and abundance of labelled nucleic acids in a biological sample, which will hybridise to the DNA on the array via Watson-Crick duplex formation, and which can be detected via the label. In the majority of microarray experiments, the labelled nucleic acids are derived from the mRNA of a sample or tissue, and so the microarray measures gene expression. The power of a microarray is that there may be many thousands

of different DNA molecules bonded to an array, and so it is possible to measure the expression of many thousands of genes simultaneously.

### Making Microarrays

There are two main technologies for making microarrays: *robotic spotting* and *in-situ synthesis*. Spotting is the technology by which the first microarrays were manufactured. The array is made using a spotting robot via three main steps:

1. Making the DNA probes to put on the array.
2. Spotting the DNA onto the glass surface of the array with the spotting robot.
3. Postspotting processing of the glass slide.

There are three main types of spotted array, which can be subdivided in two ways: by the type of DNA probe, or by the attachment chemistry of the probe to the glass. The DNA probes used on a spotted array can either be polymerase chain reaction (PCR) products or oligonucleotides. In the first case, highly parallel PCR is used to amplify DNA from a clone library, and the amplified DNA is purified. In the second case, DNA oligonucleotides are presynthesised for use on the array.

The attachment chemistry can either be covalent or non-covalent. With covalent attachment, a primary aliphatic amine ($NH_2$) group is added to the DNA probe and the probe is attached to the glass by making a covalent bond between this group and chemical linkers on the glass. With oligonucleotide probes, the amine group can be added to either end of the oligonucleotide during synthesis, although it is more usual to add it to the $5'$ end of the oligonucleotide. With cDNA probes, the amine group is added to the $5'$ end of the PCR primer from which the probes are made. Thus the cDNA probes are always attached from the $5'$ end. With non-covalent attachment, the bonding of the probe to the array is via electrostatic attraction between the phosphate backbone of the DNA probe and $NH_2$ groups attached to the surface of the glass. The interaction takes place at several locations along the DNA backbone, so that the probe is tethered to the glass at many points. Because most oligonucleotide probes are shorter than cDNAs, these interactions are not strong enough to anchor oligonucleotide probes to glass. Therefore, non-covalent attachment is usually only used for cDNA microarrays. The DNA probes

are organised in microtiter well plates, typically 384 well plates. Most modern spotting robots will use a number of plates to print arrays, so the plates are arranged in a "hotel," whereby the robot is able to gain successive access to each of the plates. The spotting robot itself consists of a series of pins arranged as a grid and held in a cassette. The pins are used to transfer liquid from the microtiter plates to the glass array.

There are a number of different designs of pins. The first spotting robots used solid pins; these can only hold enough liquid for one spot on the array, thus requiring the pin cassette to return to the plate containing probe before printing the next spot. Most array-making robots today have pins with a reservoir that holds liquid. This enables higher throughput production of arrays because each probe can be spotted on several arrays without the need to return the pins to the sample plates. The typical printing process follows five steps:

1. The pins are dipped into the wells to collect the first batch of DNA.
2. This DNA is spotted onto a number of different arrays, depending on the number of arrays being made and the amount of liquid the pins can hold.
3. The pins are washed to remove any residual solution and ensure no contamination of the next sample.
4. The pins are dipped into the next set of wells.
5. Return to step 2 and repeat until the array is complete.

In the final phase of array production, the surface of the array can be fixed so that no further DNA can attach to it. There are many fixing processes that depend on the precise chemistry on the surface of the glass. The desired outcome is always the same: we do not want DNA target from the sample to stick to the glass of the array during hybridisation, so the surface must be modified so this does not happen. It is also common to modify the surface so that the glass becomes more hydrophilic because this aids mixing of the target solution during the hybridisation stage. Some microarray production facilities do not fix their arrays.

These arrays are fundamentally different from spotted arrays: instead of presynthesising oligonucleotides, oligos are built up base-by-base on the surface of the array. This

takes place by covalent reaction between the $5'$ hydroxyl group of the sugar of the last nucleotide to be attached and the phosphate group of the next nucleotide. Each nucleotide added to the oligonucleotide on the glass has a protective group on its $5'$ position to prevent the addition of more than one base during each round of synthesis. The protective group is then converted to a hydroxyl group either with acid or with light before the next roundof synthesis. The different methods for deprotection lead to the three main technologies for making in-situ synthesised arrays:

1. Photodeprotection using masks: this is the basis of the Affymetrix [1] technology.
2. Photodeprotection without masks: this is the method used by Nimblegen and Febit.
3. Chemical deprotection with synthesis via inkjet technology: this is the method used by Rosetta [34], Agilent [2] and Oxford Gene Technology [32].

### Affymetrix Technology

Affymetrix arrays use light to convert the protective group on the terminal nucleotide into a hydroxyl group to which further bases can be added. The light is directed to appropriate features using masks that allow light to pass to some areas of the array but not to others. This technique is known as photolithography and was first applied to the manufacture of silicon chips. Each step of synthesis requires a different mask, and each mask is expensive to produce. However, once a mask set has been designed andmade, it is straightforward to produce a large number of identical arrays. Thus Affymetrix technology is well suited for making large numbers of "standard" arrays that can be widely used throughout the community.

### Maskless Photodeprotection Technology

This technology is similar to Affymetrix technology in that light is used to convert the protective group at each step of synthesis. However, instead of using masks, the light is directed via micromirror arrays, such as those made by Texas Instruments [39]. These are solid-state silicon devices that are at the core of some data projectors: an array of mirrors is computer controlled and can be used to direct light to appropriate parts of the glass slide at each step of oligonucelotide synthesis. This is the technology used by Nimblegen and Febit.

### Inkjet Array Synthesis

Instead of using light to convert the protective group, deprotection takes place chemically, using the same chemistry as a standard DNA synthesiser. At each step of synthesis, droplets of the appropriate base are fired onto the desired spot on the glass slide via the same nozzles that are used for inkjet printers; but instead of firing cyan, magenta, yellow and black ink, the nozzles fire `A`, `C`, `G`, and `T` nucleotides. One of the main advantages of micromirror and inkjet technologies over both Affymetrix technology and spotted arrays is that the oligonucleotide being synthesised on each feature is entirely controlled by the computer input given to the array-maker at the time of array production. Therefore, these technologies are highly flexible, with each array able to contain any oligonucleotide the operator wishes. However, these technologies are also less efficient for making large numbers of identical arrays.

### Synthesis Yields

The different methods of oligonucleotide synthesis have different coupling efficiencies: this is the proportion of nucleotides that are successfully added at each step of synthesis. Photodeprotection has a coupling efficiency of approximately 95%, whereas acid-mediated deprotection of dimethoxytrityl protecting groups has a coupling efficiency of approximately 98%. The effect on the yield of full-length oligos is dependent on the length of the oligonucleotide being synthesised: the longer the oligonucleotide, the worse the yield. This dependence is multiplicative, so that even a small difference in coupling efficiency can make a large difference in the yield of long oligonucleotides. The composition of the final population of oligonucleotides produced depends on whether or not a capping reaction is included during synthesis. Capping is used by Affymetrix and prevents further synthesis on a failed oligonucleotide. As a result, all oligonucleotides on a feature will have the same start, but will be of different lengths (e.g., with a coupling efficiency of 95%, each feature will be 4.8% monomers, 4.5% dimers, 4.3% trimers, etc.). In contrast, uncapped oligonucleotides allow further synthesis to take place. Therefore, all the oligonucleotides on a feature will be of similar length but may contain random deletions (e.g., with a coupling efficiency of 95% and synthesis of 20 mers, the average probe length would be 19 bases, with such probes containing one deletion).

### Spot Quality

The quality of the features depends on the method of array production. Spotted array images can be of variable quality. Affymetrix arrays have the problem that the masks refract light, so light leaks into overlapping features; Affymetrix compensates for this with their image-processing software, so the user need not worry about this problem. Inkjet arrays tend to produce the highest quality features.

## 1.2 Universal DNA Arrays

Specific hybridization of oligonucleotides and their analogs is a fundamental process that is employed in a wide variety of research, medical, and industrial applications, including the identification of disease-related polynucleotides in diagnostic assays, screening for clones of novel target polynucleotides, identification of specific polynucleotides in blots of mixtures of polynucleotides, amplification of specific target polynucleotides, therapeutic blocking of inappropriately expressed genes, DNA sequencing, and the like.

Specific hybridization has also been proposed as a method of tracking, retrieving, and identifying compounds labeled with oligonucleotide tags. For example, in multiplex DNA sequencing oligonucleotide tags are used to identify electrophoretically separated bands on a gel that consist of DNA fragments generated in the same sequencing reaction. In this way, DNA fragments from many sequencing reactions are separated on the same lane of a gel which is then blotted with separate solid phase materials on which the fragment bands from the separate sequencing reactions are visualized with oligonucleotide probes that specifically hybridize to complementary tags. Similar uses of oligonucleotide tags have also been proposed for identifying explosives, potential pollutants, such as crude oil, and currency for prevention and detection of counterfeiting. More recently, systems employing oligonucleotide tags have also been proposed as a means of manipulating and identifying individual molecules in complex combinatorial chemical libraries.

The successful implementation of such tagging schemes depends in large part on the success in achieving specific hybridization between a tag and its complementary probe. That is, for an oligonucleotide tag to successfully identify a substance, the number of false positive and false negative signals must be minimized. Unfortunately, such spurious sig-

nals are not uncommon because base pairing and base stacking free energies vary widely among nucleotides in a duplex or triplex structure. For example, a duplex consisting of a repeated sequence of deoxyadenine (A) and thymidine (T) bound to its complement may have less stability than an equal-length duplex consisting of a repeated sequence of deoxyguanidine (G) and deoxycytidine (C) bound to a partially complementary target containing a mismatch. Thus, if a desired compound from a large combinatorial chemical library were tagged with the former oligonucleotide, a significant possibility would exist that, under hybridization conditions designed to detect perfectly matched AT-rich duplexes, undesired compounds labeled with the GC-rich oligonucleotide–even in a mismatched duplex–would be detected along with the perfectly matched duplexes consisting of the AT-rich tag. In the molecular tagging system proposed by Brenner et al., the related problem of mis-hybridizations of closely related tags was addressed by employing a so-called "commaless" code, which ensures that a probe out of register (or frame shifted) with respect to its complementary tag would result in a duplex with one or more mismatches for each of its five or more three-base words, or "codons."

Even though reagents, such as tetramethylammonium chloride, are available to negate base-specific stability differences of oligonucleotide duplexes, the effect of such reagents is often limited and their presence can be incompatible with, or render more difficult, further manipulations of the selected compounds, e.g. amplification by polymerase chain reaction (PCR), or the like.

Such problems have made the simultaneous use of multiple hybridization probes in the analysis of multiple or complex genetic loci, e.g. via multiplex PCR, reverse dot blotting, or the like, very difficult. As a result, direct sequencing of certain loci, e.g. HLA genes, has been promoted as a reliable alternative to indirected methods employing specific hybridization for the identification of genotypes.

The ability to sort cloned and identically tagged DNA fragments onto distinct solid phase supports would facilitate such sequencing, particularly when coupled with a non gel-based sequencing methodology simultaneously applicable to many samples in parallel.

## Constructing Oligonucleotide Tags from Minimally Cross-Hybridizing Sets of Subunits

The nucleotide sequences of the subunits for any minimally cross-hybridizing set are conveniently enumerated by simple computer programs following a general algorithm. Such an algorithm computes all minimally cross-hybridizing sets having subunits composed of three kinds of nucleotides and having length of four.

Preferably, minimally cross-hybridizing sets comprise subunits that make approximately equivalent contributions to duplex stability as every other subunit in the set. In this way, the stability of perfectly matched duplexes between every subunit and its complement is appoximately equal.

A preferred embodiment of minimally cross-hybridizing sets are those whose subunits are made up of three of the four natural nucleotides. As will be discussed more fully below, the absence of one type of nucleotide in the oligonucleotide tags permits target polynucleotides to be loaded onto solid phase supports by use of the $5' \rightarrow 3'$ exonuclease activity of a DNA polymerase.

Oligonucleotide tags and their complements are conveniently synthesized on an automated DNA synthesizer, using standard chemistries, such as phosphoramidite chemistry and the like. Alternative chemistries, e.g. resulting in non-natural backbone groups, such as phosphorothioate, phosphoramidate, and the like, may also be employed provided that the resulting oligonucleotides are capable of specific hybridization. In some embodiments, tags may comprise naturally occuring nucleotides that permit processing or manipulation by enzymes, while the corresponding tag complements may comprise non-natural nucleotide analogs, such as peptide nucleic acids, or like compounds, that promote the formation of more stable duplexes during sorting.

When microparticles are used as supports, repertoires of oligonucleotide tags and tag complements are preferably generated by subunit-wise synthesis via "split and mix" techniques. Briefly, the basic unit of the synthesis is a subunit of the oligonucleotide tag. Preferably, phosphoramidite chemistry is used and $3'$ phosphoramidite oligonucleotides are prepared for each subunit in a minimally cross-hybridizing set. Synthesis proceeds in direct analogy with the techniques employed to generate diverse oligonucleotide libraries using nucleosidic monomers and the like. Generally, these techniques simply call for the application of mixtures of the activated monomers to the growing oligonucleotide during

the coupling steps. Double stranded forms of tags are made by separately synthesizing the complementary strands followed by mixing under conditions that permit duplex formation. Such duplex tags may then be inserted into cloning vectors along with target polynucleotides for sorting and manipulation of the target polynucleotide.

In embodiments where specific hybridization occurs via triplex formation, coding of tag sequences follows the same principles as for duplex-forming tags; however, there are further constraints on the selection of subunit sequences. Generally, third strand association via Hoogsteen type of binding is most stable along homopyrimidine-homopurine tracks in a double stranded target. Usually, base triplets form in T-A*T or C-G*C motifs (where "-" indicates Watson-Crick pairing and "*" indicates Hoogsteen type of binding); however, other motifs are also possible. For example, Hoogsteen base pairing permits parallel and antiparallel orientations between the third strand (the Hoogsteen strand) and the purine-rich strand of the duplex to which the third strand binds, depending on conditions and the composition of the strands. Conditions for annealing single-stranded or duplex tags to their single-stranded or duplex complements are well known.

Oligonucleotide tags may range in length from 12 to 60 nucleotides or basepairs. Preferably, oligonucleotide tags range in length from 18 to 40 nucleotides or basepairs. More preferably, oligonucleotide tags range in length from 25 to 40 nucleotides or basepairs. Most preferably, oligonucleotide tags are single stranded and specific hybridization occurs via Watson-Crick pairing with a tag complement.

### Attaching Tags to Molecules

Oligonucleotide tags may be attached to many different classes of molecules by a variety of reactive functionalities well known in the art. When the functionalities and counterpart reactants are reacted together, after activation in some cases, a linking group is formed. Moreover, tags may be synthesized simultaneously with the molecules undergoing selection to form combinatorial chemical libraries.

A class of molecules particularly convenient for the generation of combinatorial chemical libraries includes linear polymeric molecules of the form

$$-(M-L).sub.n-$$

wherein $L$ is a linker moiety and $M$ is a monomer that may selected from a wide range of chemical structures to provide a range of functions from serving as an inert non-sterically hindering spacer moiety to providing a reactive functionality which can serve as a branching point to attach other components, a site for attaching labels; a site for attaching oligonucleotides or other binding polymers for hybridizing or binding to a therapeutic target; or as a site for attaching other groups for affecting solubility, promotion of duplex and/or triplex formation, such as intercalators, alkylating agents, and the like. The sequence, and therefore composition, of such linear polymeric molecules may be encoded within a polynucleotide attached to the tag. However, after a selection event, instead of amplifying then sequencing the tag of the selected molecule, the tag itself or an additional coding segment can be sequenced directly–using a so-called "single base" approach described below–after releasing the molecule of interest, e.g. by restriction digestion of a site engineered into the tag. Clearly, any molecule produced by a sequence of chemical reaction steps compatible with the simultaneous synthesis of the tag moieties can be used in the generation of combinatorial libraries.

### Solid Phase Supports

Solid phase supports may have a wide variety of forms, including microparticles, beads, and membranes, slides, plates, micromachined chips, and the like. Likewise, solid phase supports may comprise a wide variety of compositions, including glass, plastic, silicon, alkanethiolate-derivatized gold, cellulose, low cross-linked and high cross-linked polystyrene, silica gel, polyamide, and the like. Preferably, either a population of discrete particles are employed such that each has a uniform coating, or population, of complementary sequences of the same tag (and no other), or a single or a few supports are employed with spacially discrete regions each containing a uniform coating, or population, of complementary sequences to the same tag (and no other). In the latter embodiment, the area of the regions may vary according to particular applications. Preferably, such regions are spacially discrete so that signals generated by events, e.g. fluorescent emissions, at adjacent regions can be resolved by the detection system being employed. In some applications, it may be desirable to have regions with uniform coatings of more than one tag complement, e.g. for simultaneous sequence analysis, or for

bringing separately tagged molecules into close proximity.

Tag complements may be used with the solid phase support that they are synthesized on, or they may be separately synthesized and attached to a solid phase support for use. Preferably, tag complements are synthesized on and used with the same solid phase support, which may comprise a variety of forms and include a variety of linking moieties. Such supports may comprise microparticles or arrays, or matrices, of regions where uniform populations of tag complements are synthesized. A wide variety of microparticle supports may be used, including microparticles made of controlled pore glass (CPG), highly cross-linked polystyrene, acrylic copolymers, cellulose, nylon, dextran, latex, polyacrolein, and the like. Microparticle supports further include commercially available nucleoside-derivatized CPG and polystyrene beads; derivatized magnetic beads; polystyrene grafted with polyethylene glycol; and the like. Selection of the support characteristics, such as material, porosity, size, shape, and the like, and the type of linking moiety employed depends on the conditions under which the tags are used. For example, in applications involving successive processing with enzymes, supports and linkers that minimize steric hinderance of the enzymes and that facilitate access to substrate are preferred.

As mentioned above, tag complements may also be synthesized on a single (or a few) solid phase support to form an array of regions uniformly coated with tag complements. That is, within each region in such an array the same tag complement is synthesized.

### Attaching Target Polynucleotides to Microparticles

An important aspect is the sorting of populations of identical polynucleotides, e.g. from a cDNA library, and their attachment to microparticles or separate regions of a solid phase support such that each microparticle or region has only a single kind of polynucleotide. This latter condition can be essentially met by ligating a repertoire of tags to a population of polynucleotides followed by cloning and sampling of the ligated sequences. A repertoire of oligonucleotide tags can be ligated to a population of polynucleotides in a number of ways, such as through direct enzymatic ligation, amplification, e.g. via PCR, using primers containing the tag sequences, and the like. The initial ligating step produces a very large population of tag-polynucleotide conjugates such that a single tag

is generally attached to many different polynucleotides. However, by taking a sufficiently small sample of the conjugates, the probability of obtaining "doubles," i.e. the same tag on two different polynucleotides, can be made negligible. (Note that it is also possible to obtain different tags with the same polynucleotide in a sample. This case simply leads to a polynucleotide being processed, e.g. sequenced, twice). As explained more fully below, the probability of obtaining a double in a sample can be estimated by a Poisson distribution since the number of conjugates in a sample will be large, e.g. on the order of thousands or more, and the probability of selecting a particular tag will be small because the tag repertoire is large, e.g. on the order of tens of thousand or more. Generally, the larger the sample the greater the probability of obtaining a double. Thus, a design trade-off exists between selecting a large sample of tag-polynucleotide conjugates–which, for example, ensures adequate coverage of a target polynucleotide in a shotgun sequencing operation, and selecting a small sample which ensures that a minimal number of doubles will be present. In most embodiments, the presence of doubles merely adds an additional source of noise or, in the case of sequencing, a minor complication in scanning and signal processing, as microparticles giving multiple fluorescent signals can simply be ignored. As used herein, the term "substantially all" in reference to attaching tags to molecules, especially polynucleotides, is meant to reflect the statistical nature of the sampling procedure employed to obtain a population of tag-molecule conjugates essentially free of doubles. The meaning of substantially all in terms of actual percentages of tag-molecule conjugates depends on how the tags are being employed. Preferably, for nucleic acid sequencing, substantially all means that at least eighty percent of the tags have unique polynucleotides attached. More preferably, it means that at least ninety percent of the tags have unique polynucleotides attached.

### Single Base DNA Sequencing

The mechanism presented can be employed with conventional methods of DNA sequencing, but for parallel, or simultaneous, sequencing of multiple polynucleotides, a DNA sequencing methodology is preferred that requires neither electrophoretic separation of closely sized DNA fragments nor analysis of cleaved nucleotides by a separate analytical procedure, as in peptide sequencing. Preferably, the methodology permits the stepwise

identification of nucleotides, usually one at a time, in a sequence through successive cycles of treatment and detection.

A "single base" method of DNA sequencing which is suitable for use with the method described and which requires no electrophoretic separation of DNA fragments comprises the following steps: (a) ligating a probe to an end of the polynucleotide having a protruding strand to form a ligated complex, the probe having a complementary protruding strand to that of the polynucleotide and the probe having a nuclease recognition site; (b) removing unligated probe from the ligated complex; (c) identifying one or more nucleotides in the protruding strand of the polynucleotide by the identity of the ligated probe; (d) cleaving the ligated complex with a nuclease; and (e) repeating steps (a) through (d) until the nucleotide sequence of the polynucleotide is determined. As is described more fully below, identifying the one or more nucleotides can be carried out either before or after cleavage of the ligated complex from the target polynucleotide. Preferably, whenever natural protein endonucleases are employed, the method further includes a step of methylating the target polynucleotide at the start of a sequencing operation.

An important feature of the method is the probe ligated to the target polynucleotide. Generally, the probes are double stranded DNA with a protruding strand at one end. The probes contain at least one nuclease recognition site and a spacer region between the recognition site and the protruding end. Preferably, probes also include a label, which in this particular embodiment is illustrated at the end opposite of the protruding strand. The probes may be labeled by a variety of means and at a variety of locations, the only restriction being that the labeling means selected does not interfer with the ligation step or with the recognition of the probe by the nuclease.

It is not critical whether protruding strand of the probe is a $5'$ or $3'$ end. However, it is important that the protruding strands of the target polynucleotide and probes be capable of forming perfectly matched duplexes to allow for specific ligation. If the protruding strands of the target polynucleotide and probe are different lengths the resulting gap can be filled in by a polymerase prior to ligation. Preferably, the number of nucleotides in the respective protruding strands are the same so that both strands of the probe and

target polynucleotide are capable of being ligated without a filling step. Preferably, the protruding strand of the probe is from 2 to 6 nucleotides long.

The complementary strands of the probes are conveniently synthesized on an automated DNA synthesizer, using standard chemistries. After synthesis, the complementary strands are combined to form a double stranded probe. Generally, the protruding strand of a probe is synthesized as a mixture, so that every possible sequence is represented in the protruding portion.

### Parallel Sequencing

The tagging system can be used with single base sequencing methods to sequence polynucleotides up to several kilobases in length. The tagging system permits many thousands of fragments of a target polynucleotide to be sorted onto one or more solid phase supports and sequenced simultaneously. In accordance with a preferred implementation of the method, a portion of each sorted fragment is sequenced in a stepwise fashion on each of the many thousands of loaded microparticles which are fixed to a common substrate–such as a microscope slide–associated with a scanning system. The size of the portion of the fragments sequenced depends of several factors, such as the number of fragments generated and sorted, the length of the target polynucleotide, the speed and accuracy of the single base method employed, the number of microparticles and/or discrete regions that may be monitored simultaneously; and the like. Preferably, from 12-50 bases are identified at each microparticle or region; and more preferably, 18-30 bases are identified at each microparticle or region. With this information, the sequence of the target polynucleotide is determined by collating the 12-50 base fragments via their overlapping regions.

Fragments may be generated from a target polynucleotide in a variety of ways, including so-called "directed" approaches where one attempts to generate sets of fragments covering the target polynucleotide with minimal overlap, and so-called "shotgun" approaches where randomly overlapping fragments are generated. Preferably, "shotgun" approaches to fragment generation are employed because of their simplicity and inherent redundancy. For example, randomly overlapping fragments that cover a target polynucleotide are generated in the following conventional "shotgun" sequencing protocol. As

used herein, "cover" in this context means that every portion of the target polynucleotide sequence is represented in each size range, e.g. all fragments between 100 and 200 basepairs in length, of the generated fragments. Briefly, starting with a target polynucleotide as an insert in an appropriate cloning vector, the vector is expanded, purified and digested with the appropriate restriction enzymes. Typically, the protocol results in about $500 - 1000$ subclones per microgram of starting DNA. The insert is separated from the vector fragments by preparative gel electrophoresis, removed from the gel by conventional methods, and resuspended in a standard buffer, such as TE (Tris-EDTA). The restriction enzymes selected to excise the insert from the vector preferably leave compatible sticky ends on the insert, so that the insert can be self-ligated in preparation for generating randomly overlapping fragments. The circularized DNA yields a better random distribution of fragments than linear DNA in the fragmentation methods. After self-ligating the insert, e.g. with T4 ligase using conventional protocols, the purifed ligated insert is fragmented by a standard protocol. After fragmentation the ends of the fragments are repair, and the repaired fragments are separated by size using gel electrophoresis. Fragments in the $300 - 500$ basepair range are selected and eluted from the gel by conventional means, and ligated into a tag-carrying vector as described above to form a library of tag-fragment conjugates.

## 1.3 Summary of results

In this thesis we propose new solution methods for designing tag sets for use in universal DNA arrays. First, we establish upper bounds for an extended version of a previous formalization. Second, we give integer linear programming formulations for two previous formalizations of the tag set design problem, and show that these formulations can be solved to optimality for instance sizes of practical interest by using general purpose optimization packages. Third, we note the benefits of periodic tags, and establish an interesting connection between the tag design problem and the problem of packing the maximum number of vertex-disjoint directed cycles in a given graph. We show that combining a simple greedy cycle packing algorithm with a previously proposed alphabetic tree search strategy yields an increase of over 40% in the number of tags compared to

previous methods. Most of the results presented in this thesis have already appeared in the following publications [31, 30]:

1. I.I. Mandoiu and D. Trinca. Exact and Approximation Algorithms for DNA Tag Set Design. In *Proceedings of the 16th Annual Symposium on Combinatorial Pattern Matching (CPM 2005)*, June 19–22, 2005, Korea. Lecture Notes in Computer Science, volume 3537, pages 383–393, Springer-Verlag. Extended version accepted to *Journal of Computational Biology*, and also available as ACM Computing Research Repository report cs.DS/0503057.

2. I.I. Mandoiu, C. Prajescu, and D. Trinca. Improved Tag Set Design and Multiplexing Algorithms for Universal Arrays. In *Proceedings of the 5th International Conference on Computational Science (ICCS 2005)/ 2005 International Workshop on Bioinformatics Research and Applications (IWBRA)*, May 22–25, 2005, Atlanta, GA, USA. Lecture Notes in Computer Science, volume 3515, pages 994–1002, Springer-Verlag. Extended version appeared in *LNCS Transactions on Computational Systems Biology*, volume 3680, pages 124–137, 2005, Springer-Verlag; also available as ACM Computing Research Repository report cs.DS/0502054.

# Chapter 2

# Problem Formulations and Previous Work

Universal DNA tag arrays [10, 29, 18], described in detail in the previous chapter, offer a flexible and cost-effective alternative to custom-designed DNA arrays for performing a wide range of genomic analyses. As already pointed out, a universal tag array consists of a set of DNA strings called *tags*, designed such that each tag hybridizes strongly to its own *antitag* (Watson-Crick complement), but not to any other antitag. A typical assay based on universal tag arrays performs Single Nucleotide Polymorphism (SNP) genotyping using the following steps [5, 21]: (1) A set of *reporter oligonucleotide probes* is synthesized by ligating antitags to the $5'$ end of primers complementing the genomic sequence immediately preceding the SNP. (2) Reporter probes are hybridized in solution with the genomic DNA under study. (3) Hybridization of the primer part ($3'$ end) of a reporter probe is detected by a single-base extension reaction using the polymerase enzyme and dideoxynucleotides fluorescently labeled with 4 different dyes. (4) Reporter probes are separated from the template DNA and hybridized to the universal array. (5) Finally, fluorescence levels are used to determine which primers have been extended and learn the identity of the extending dideoxynucleotides.

A main objective of universal array designers is to maximize the number of tags, which directly determines the number of reactions that can be multiplexed using a single array. At the same time, tag sets must satisfy a number of *stability* and *non-interaction* constraints [9]. The full set of constraints depends on factors such as the array manufacturing technology and the intended application. In this chapter we formalize the most

important stability and non-interaction constraints using the hybridization model in [6].

**Hybridization model.** Hybridization affinity between two oligonucleotides is commonly characterized using the *melting temperature*, defined as the temperature at which half of the duplexes are in hybridized state and the other half are in melted state. However, accurate melting temperature estimation is computationally expensive, e.g., estimating the melting temperature between two non-complementary oligonucleotides using the near-neighbor model of SantaLucia [36] is an NP-hard problem [22]. A conservative hybridization model based on the observation that stable hybridization requires the formation of an initial *nucleation complex* between two perfectly complementary substrings of the two oligonucleotides was formalized by [6, 5]. For nucleation complexes, hybridization affinity is modeled using the classical *2-4 rule* [40], according to which the melting temperature of the duplex formed by an oligonucleotide with its complement is proportional to the sum between the number of *weak* bases (i.e., `A` and `T`) and twice the number of *strong* bases (i.e., `G` and `C`).

Following [6], we define the *weight* $w(x)$ of a DNA string $x = a_1a_2 \dots a_k$ by $w(x) = \sum_{i=1}^{k} w(a_i)$, where $w(\texttt{A}) = w(\texttt{T}) = 1$ and $w(\texttt{C}) = w(\texttt{G}) = 2$. Throughout the thesis we assume the following *c-token hybridization model*: hybridization between two oligonucleotides takes place only if one contains as substring the complement of a substring of weight $c$ or more of the other, where $c$ is a given constant. The *complement* of a string $x = a_1a_2 \dots a_k$ over the DNA alphabet $\{\texttt{A}, \texttt{C}, \texttt{T}, \texttt{G}\}$ is defined as $\bar{x} = b_1b_2 \dots b_k$, where $b_i$ is the Watson-Crick complement of $a_{k-i+1}$.

**Hybridization stability.** Current industry designs require a predetermined tag length $l$, e.g., GenFlex universal tag arrays manufactured by Affymetrix use $l = 20$ [1]. The model proposed in [6] allows tags of unequal length and instead require a minimum tag weight of $h$, for a given constant $h$. In this thesis we consider both types of stability constraints, and use the parameter $\alpha \in \{l, h\}$ to denote the specific model used for hybridization stability.

**Pairwise non-interaction constraints.** A basic constraint in this category is that every antitag must not hybridize to non-complementary tags [6]. For a DNA string $x$ and a set of tags $\mathcal{T}$, let $N_{\mathcal{T}}(x)$ denote the number of tags in $\mathcal{T}$ that contain $x$ as a substring. Using the $c$-token hybridization model, this antitag-to-tag hybridization constraint is formalized as follows:

**($C$)** For every feasible tag set $\mathcal{T}$, $N_{\mathcal{T}}(x) \leq 1$ for every DNA string $x$ of weight $c$ or more.

In many assays based on universal tag arrays it is also required to prevent antitag-to-antitag hybridization, since the formation of antitag-to-antitag duplexes or antitag hairpin structures prevents reporter probes from performing their function in the solution-based hybridization steps [9]. The combined constraints on antitag hybridization are formalized as follows:

**($\bar{C}$)** For every feasible tag set $\mathcal{T}$, $N_{\mathcal{T}}(x) + N_{\mathcal{T}}(\bar{x}) \leq 1$ for every DNA string $x$ of weight $c$ or more.

In the following we use the parameter $\beta \in \{C, \bar{C}\}$ to specify the type of pairwise non-interaction constraints.

**Substring occurrences within a tag.** Previous works on DNA tag set design [6] have imposed the following *c-token uniqueness constraint* in addition to constraints ($C$) and ($\bar{C}$): a DNA string of weight $c$ or more can appear as a substring of a feasible tag at most once. This uniqueness constraint simplifies analysis – e.g., it is the key property enabling the DeBruijn sequence based heuristics in [6]) – but is *not* required for ensuring correct assay functionality. In the following we will use the parameter $\gamma \in \{1, \mathit{multiple}\}$ to specify whether or not the $c$-token uniqueness constraint is enforced.

**Problem formulation.** For every $\alpha \in \{l, h\}$, $\beta \in \{C, \bar{C}\}$, and $\gamma \in \{1, \mathit{multiple}\}$, the *maximum tag set design problem with constraints* $\alpha, \beta, \gamma$, denoted MTSDP($\alpha|\beta|\gamma$), is the following: given constants $c$ and $l/h$, find a tag set of maximum cardinality satisfying constraints $\alpha$, $\beta$, and $\gamma$.

**Previous work on tag set design.** The tag set design problem was identified in previous work and several formulations and solutions were proposed [16, 10, 28, 37, 18]. These papers differ both in the way hybridization is modeled, and in the algorithmic approach employed to find a good DNA TAT system. In [16] a TAT system is described as a part of a strategy for surface-based DNA computing. The authors take a coding theory approach and choose to model cross-hybridization constraints as general Hamming distance conditions. A set of 108 8-mers, with a 50% GC content, which differ in at least 4 bases from each other, is constructed, and experimentally tested for cross-hybridization. In [10] the method of using a DNA TAT system to sort target DNA is presented, together with several examples of applications. The model assumption is that two oligonucleotides of length $n$ need to have perfectly complementary substrings of length more than $\lambda$ in order to form a reasonably stable duplex. A set of $n$-mers is said to be a $\lambda$-*free code* if no two elements of the set have a common substring of length more than $\lambda$. Given $n$, the design problem implied in [10] is to construct the largest possible $\lambda$-free code. The $c$-token model for oligonucleotide hybridization and the MTSDP($h|C|1$) problem are formalized in [6]. Ben-Dor et al. also established a constructive upper bound on the optimal number of tags for this formulation, and gave a nearly optimal tag selection algorithm based on DeBruijn sequences. Similar upper bounds are established for the MTSDP($l|C|1$) and MTSDP($*|\bar{C}|1$) problems in the next chapter. For a comprehensive survey of hybridization models, results on associated formulations for the tag set design problem, and further motivating applications in the area of DNA computing, we direct the reader to [9].

# Chapter 3

# Upper Bounds for MTSDP$(*|\bar{C}|1)$

The constructive upperbound is based on counting the minimal strings, called *c-tokens*, that can occur as substrings only once in the tags and antitags of a feasible set. Formally, a DNA string $x$ is called a $c$-token if the weight of $x$ is $c$ or more, and every proper suffix of $x$ has weight strictly less than $c$. The *tail weight* of a $c$-token is defined as the weight of its last letter. Note that the weight of a $c$-token can be either $c$ or $c+1$, the latter case being possible only if the $c$-token starts with a `G` or a `C`. As in [6], we use $G_n$ to denote the number of DNA strings of weight $n$. It is easy to see that $G_1 = 2$, $G_2 = 6$, and $G_n = 2G_{n-1} + 2G_{n-2}$; for convenience, we also define $G_0 = 1$. Using these notations, the upper bound established by Ben-Dor et al. for MTSDP$(h|C|1)$ can be stated as follows:

**Theorem 1** *A feasible solution to MTSDP(h|C|1) has at most*

$$\frac{2G_{c-1} + 6G_{c-2} + 8G_{c-3}}{h - c + 1} \text{ tags.}$$

Regarding our upper bounds, we first establish two lemmas on self-complementary DNA strings, i.e., strings $x \in \{\mathtt{A}, \mathtt{C}, \mathtt{T}, \mathtt{G}\}^+$ with $x = \overline{x}$.

**Lemma 1** *If $x$ is self-complementary then $|x|$ and $w(x)$ are both even.*

*Proof.* Let $x = x_1 x_2 \ldots x_p$ be a self-complementary DNA string. If $p = 2q + 1$, by the definition of the complement we should have $x_{q+1} = \overline{x}_{q+1}$, which is impossible. Thus, $p = 2q$. Since $x_1 = \overline{x}_{2q}$,$x_2 = \overline{x}_{2q-1}$,..., $x_q = \overline{x}_{q+1}$, and the weight of complementary bases is the same, it follows that $w(x) = 2\sum_{i=1}^{q} w(x_i)$. □

**Lemma 2** *Let $H_n$ be the number of self-complementary DNA strings of weight $n$. $H_n = 0$ if $n$ is odd, and $H_n = G_{n/2}$ if $n$ is even.*

*Proof.* By Lemma 1, self-complementary strings must have even length and weight. For even $n$, the mapping $x_1 \dots x_q x_{q+1} \dots x_{2q} \mapsto x_1 \dots x_q$ gives a one-to-one correspondence between self-complementary strings of weight $n$ and strings of weight $n/2$. □

Now, we prove the following:

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

*Proof.* Let `W` and `S` denote weak and strong DNA bases (`A` or `T`, respectively `G` or `C`), and let $<w>$ denote the set of DNA strings with weight $w$. The $c$-tokens can be partitioned into the seven classes given in Table 3.1, depending on total token weight ($c$ or $c+1$) and the type of starting and ending bases. This partitioning is defined so that, for every $c$-token $x$, the class of the unique $c$-token suffix of $\overline{x}$ can be determined from the class of $x$. Note that $\bar{x}$ is itself a $c$-token, except when $x \in \mathtt{S}<c-3>\mathtt{WW} \cup \mathtt{S}<c-4>\mathtt{SW}$.

Let $N_{cls}$ denote the number of $c$-tokens of class *cls* occurring in a feasible tag set.

### $c$ odd

Since $\mathtt{W}<c-3>\mathtt{S} \cup \mathtt{S}<c-3>\mathtt{W}$ can be partitioned into $4G_{c-3}$ pairs $\{x, \bar{x}\}$ of complementary $c$-tokens, and at most one token from each pair can appear in a feasible tag set,

$$N_{\mathtt{W}<c-3>\mathtt{S}} + N_{\mathtt{S}<c-3>\mathtt{W}} \leq 4G_{c-3}. \tag{3.1}$$

Table 3.1: Classes of $c$-tokens.

| Class of $x$ | $c$-token suffix of $\overline{x}$ |
|---|---|
| $\mathtt{W}<c-3>\mathtt{S}$ | $\mathtt{S}<c-3>\mathtt{W}$ |
| $\mathtt{S}<c-4>\mathtt{S}$ | $\mathtt{S}<c-4>\mathtt{S}$ |
| $\mathtt{S}<c-3>\mathtt{S}$ | $\mathtt{S}<c-3>\mathtt{S}$ |
| $\mathtt{W}<c-2>\mathtt{W}$ | $\mathtt{W}<c-2>\mathtt{W}$ |
| $\mathtt{S}<c-3>\mathtt{W}$ | $\mathtt{W}<c-3>\mathtt{S}$ |
| $\mathtt{S}<c-3>\mathtt{WW}$ | $\mathtt{W}<c-3>\mathtt{S}$ |
| $\mathtt{S}<c-4>\mathtt{SW}$ | $\mathtt{S}<c-4>\mathtt{S}$ |

Similarly, class $\mathtt{W}\!<\!c-2\!>\!\mathtt{W}$ can be partitioned into $2G_{c-2}$ pairs $\{x, \bar{x}\}$ of complementary $c$-tokens, $\mathtt{W}\!<\!c-3\!>\!\mathtt{S} \cup \mathtt{S}\!<\!c-3\!>\!\mathtt{WW}$ can be partitioned into $4G_{c-3}$ triples $\{x, \bar{x}A, \bar{x}T\}$ with $x \in \mathtt{W}\!<\!c-3\!>\!\mathtt{S}$, $\mathtt{S}\!<\!c-3\!>\!\mathtt{W} \cup \mathtt{S}\!<\!c-3\!>\!\mathtt{WW}$ can be partitioned into $4G_{c-3}$ triples $\{x, xA, xT\}$ with $x \in \mathtt{S}\!<\!c-3\!>\!\mathtt{W}$, and $\mathtt{S}\!<\!c-4\!>\!\mathtt{S} \cup \mathtt{S}\!<\!c-4\!>\!\mathtt{SW}$ can be partitioned into $2G_{c-4}$ 6-tuples $\{x, \bar{x}, xA, xT, \bar{x}A, \bar{x}T\}$ with $x \in \mathtt{S}\!<\!c-4\!>\!\mathtt{S}$. Since at most one $c$-token can appear in a feasible tag set from each such pair, triple, respectively 6-tuple,

$$N_{\mathtt{W}<c-2>\mathtt{W}} \leq 2G_{c-2}, \tag{3.2}$$

$$N_{\mathtt{W}<c-3>\mathtt{S}} + N_{\mathtt{S}<c-3>\mathtt{WW}} \leq 4G_{c-3}, \tag{3.3}$$

$$N_{\mathtt{S}<c-3>\mathtt{W}} + N_{\mathtt{S}<c-3>\mathtt{WW}} \leq 4G_{c-3}, \tag{3.4}$$

$$N_{\mathtt{S}<c-4>\mathtt{S}} + N_{\mathtt{S}<c-4>\mathtt{SW}} \leq 2G_{c-4}. \tag{3.5}$$

Using Lemma 2, it follows that $\mathtt{S}\!<\!c-3\!>\!\mathtt{S}$ contains $2G_{\frac{c-3}{2}}$ self-complementary $c$-tokens. Since the remaining $4G_{c-3} - 2G_{\frac{c-3}{2}}$ $c$-tokens can be partitioned into complementary pairs each contributing at most one $c$-token to a feasible tag set,

$$N_{\mathtt{S}<c-3>\mathtt{S}} \leq \frac{1}{2}\left(4G_{c-3} - 2G_{\frac{c-3}{2}}\right) + 2G_{\frac{c-3}{2}} = 2G_{c-3} + G_{\frac{c-3}{2}}. \tag{3.6}$$

Adding inequalities (3.1), (3.3), and (3.4) multiplied by 1/2 with (3.2), (3.5), and (3.6) implies that the total number of $c$-tokens in a feasible tag set is at most

$$2G_{c-2} + 8G_{c-3} + 2G_{c-4} + G_{\frac{c-3}{2}} = 3G_{c-2} + 6G_{c-3} + G_{\frac{c-3}{2}}. \tag{3.7}$$

Furthermore, adding (3.1), (3.2), and (3.3) with inequalities (3.5) and (3.6) multiplied by 2 implies that the total tail weight of the $c$-tokens in a feasible tag set is at most

$$2G_{c-2} + 12G_{c-3} + 4G_{c-4} + 2G_{\frac{c-3}{2}} = 2G_{c-1} + 4G_{c-3} + 2G_{\frac{c-3}{2}}. \tag{3.8}$$

**$c$ even**

Inequalities (3.1), (3.3), and (3.4) continue to hold for even values of $c$. Since $c-3$ is odd, $\mathtt{S}\!<\!c-3\!>\!\mathtt{S}$ contains no self-complementary tokens and can be partitioned into $2G_{c-3}$ pairs $\{x, \bar{x}\}$,

$$N_{\mathtt{S}<c-3>\mathtt{S}} \leq 2G_{c-3}. \tag{3.9}$$

By Lemma 2, there are $2G_{\frac{c-4}{2}}$ self-complementary tokens in $\mathtt{S}{<}c-4{>}\mathtt{S}$. Therefore $\mathtt{S}{<}c-4{>}\mathtt{S} \cup \mathtt{S}{<}c-4{>}\mathtt{SW}$ can be partitioned into $2G_{\frac{c-4}{2}}$ triples $\{x, xA, xT\}$ with $x \in \mathtt{S}{<}c-4{>}\mathtt{S}$, $x = \bar{x}$ and $2G_{c-4} - G_{\frac{c-4}{2}}$ 6-tuples $\{x, \bar{x}, xA, xT, \bar{x}A, \bar{x}T\}$ with $x \in \mathtt{S}{<}c-4{>}\mathtt{S}$, $x \neq \bar{x}$. Since a feasible tag set can use at most one $c$-token from each triple and 6-tuple,

$$N_{\mathtt{S}<c-4>\mathtt{S}} + N_{\mathtt{S}<c-4>\mathtt{SW}} \leq 2G_{c-4} + G_{\frac{c-4}{2}}. \tag{3.10}$$

Using again Lemma 2, we get

$$N_{\mathtt{W}<c-2>\mathtt{W}} \leq 2G_{c-2} + G_{\frac{c-2}{2}}. \tag{3.11}$$

Adding inequalities (3.1), (3.3), and (3.4) multiplied by 1/2 with (3.9), (3.10), and (3.11) implies that the total number of $c$-tokens in a feasible tag set is at most

$$2G_{c-2} + 8G_{c-3} + 2G_{c-4} + G_{\frac{c-2}{2}} + G_{\frac{c-4}{2}} = 3G_{c-2} + 6G_{c-3} + \frac{1}{2}G_{\frac{c}{2}}. \tag{3.12}$$

Finally, adding (3.1), (3.3), and (3.11) with inequalities (3.9) and (3.10) multiplied by 2 implies that the total tail weight of the $c$-tokens in a feasible tag set is at most

$$2G_{c-2} + 12G_{c-3} + 4G_{c-4} + G_{\frac{c-2}{2}} + 2G_{\frac{c-4}{2}} = 2G_{c-1} + 4G_{c-3} + G_{\frac{c-2}{2}} + 2G_{\frac{c-4}{2}}. \tag{3.13}$$

□

**Theorem 2** *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*

$$\min\left\{\frac{3G_{c-2} + 6G_{c-3} + G_{\frac{c-3}{2}}}{l-c+1}, \frac{2G_{c-1} + 4G_{c-3} + 2G_{\frac{c-3}{2}}}{h-c+1}\right\}$$

*for $c$ odd, and at most*

$$\min\left\{\frac{3G_{c-2} + 6G_{c-3} + \frac{1}{2}G_{\frac{c}{2}}}{l-c+1}, \frac{2G_{c-1} + 4G_{c-3} + G_{\frac{c-2}{2}} + 2G_{\frac{c-4}{2}}}{h-c+1}\right\}$$

*for $c$ even.*

*Proof.* The proof follows from Lemma 3 by observing that every tag contains at least $l-c+1$ $c$-tokens, with a total tail weight of at least $h-c+1$. □

# Chapter 4

# Integer Linear Programming Formulations for MTSDP$(*|C|1)$

Before stating our integer linear program formulations, we introduce some additional notations.

Following [6], a DNA string $x$ of weight $c$ or more is called a *c-token* if all its proper suffixes have weight strictly less than $c$. Clearly, it suffices to enforce constraints $(C)$ or $(\bar{C})$ for all $c$-tokens $x$. Let $N$ denote the number of $c$-tokens, and $\mathcal{C} = \{c_1, \ldots, c_N\}$ denote the set of all $c$-tokens. The results in [6] imply that $N = \Theta((1+\sqrt{3})^c)$. Note that the weight of a $c$-token can be either $c$ or $c+1$, the latter case being possible only if the $c$-token starts with a strong base (`G` or `C`). We let $\mathcal{C}_0 \subseteq \mathcal{C}$ denote the set of $c$-tokens of weight $c+1$ that end with a weak base, i.e., $c$-tokens of the form `S`$<c-2>$`W`, where `W` (`S`) denotes a weak (strong) base, and $<c-2>$ denotes an arbitrary string of weight $c-2$. We also let $\mathcal{C}_2 \subseteq \mathcal{C}$ denote the set of $c$-tokens of weight $c$ that end with a strong base, i.e., $c$-tokens of the form $<c-2>$`S`.

Clearly, there is at most one $c$-token ending at every letter of a tag. It is easy to see that each $c$-token $x \in \mathcal{C}_0$ contains a proper prefix which is itself a $c$-token, and therefore $x$ cannot be the first $c$-token of a tag, i.e., cannot be the $c$-token with the leftmost ending. All other $c$-tokens can appear as first $c$-tokens. When a $c$-token in $C \setminus (\mathcal{C}_0 \cup \mathcal{C}_2)$ is the first in a tag, then it must be a prefix of the tag. On the other hand, tokens in $\mathcal{C}_2$ can be first both in tags that they prefix and in tags in which they are preceded by a weak base not covered by any $c$-token.

The ILP formulation for MTSDP$(l|C|1)$ uses an auxiliary directed graph $G = (V, E)$

with $V = \{s,t\} \cup \bigcup_{1 \leq i \leq N} V_i$, where $V_i = \{v_i^k \mid |c_i| \leq k \leq l\}$. $G$ has a directed arc from $v_i^k$ to $v_j^{k+1}$ for every triple $i,j,k$ with $|c_i| \leq k \leq l-1$ for which $c_j$ can be obtained from $c_i$ by appending a single nucleotide and removing the maximal prefix that still leaves a valid $c$-token. Finally, $G$ has an arc from $s$ to every $v \in V_{first}$, where $V_{first} = \{v_i^{|c_i|} \mid c_i \in \mathcal{C} \setminus \mathcal{C}_0\} \cup \{v_i^{|c_i|+1} \mid c_i \in \mathcal{C}_2\}$, and an arc from $v_i^l$ to $t$ for every $1 \leq i \leq N$. Notice that $G$ has $O(lN)$ vertices. Furthermore, since $s$ has outdegree less than $2N$ and every other vertex has outdegree at most 4, it follows that $G$ has $O(lN)$ arcs.

We claim that, for $c \leq l$, MTSDP$(l|C|1)$ can be reformulated as the problem of finding the maximum number of $s$-$t$ paths in $G$ that collectively visit at most one vertex $v_i^k$ for every $i$. Indeed, let $P$ be an $s$-$t$ path and $v_i^k$ be the vertex following $s$ in $P$. If $k = |c_i|$, we associate to $P$ the tag obtained by concatenating $c_i$ with the last letters of the $c$-tokens corresponding to the subsequently visited vertices, until reaching $t$. Otherwise, we must have $c_i \in \mathcal{C}_2$ and $k = |c_i| + 1$. In this case we associate to $P$ the two tags obtained by concatenating either `A` or `T` with $c_i$ and with the last letters of subsequently visited $c$-tokens. The claim follows by observing that at most one of the tags associated with each path can be used in a feasible solution.

Our ILP formulation can be viewed as a generalized version of the maximum integer flow problem in which unit capacity constraints are imposed on *sets of vertices* of $G$ instead of individual vertices. The formulation uses 0/1 variables $x_v$ and $y_e$ for every vertex $v \in V \setminus \{s,t\}$, respectively arc $e \in E$. These variables are set to 1 if the corresponding vertex or arc is visited by an $s$-$t$ path corresponding to a selected tag. Let $in(v)$ and $out(v)$ denote the set of arcs entering, respectively leaving vertex $v$. The integer program can then be written as follows:

$$\text{maximize} \quad \sum_{v \in V_{first}} x_v \tag{4.1}$$

subject to

$$x_v = \sum_{e \in in(v)} y_e = \sum_{e \in out(v)} y_e, \quad v \in V \setminus \{s,t\} \tag{4.2}$$

$$\sum_{v \in V_i} x_v \leq 1, \qquad 1 \leq i \leq N \tag{4.3}$$

$$x_v, y_e \in \{0,1\}, \qquad v \in V \setminus \{s,t\}, e \in E. \tag{4.4}$$

Constraints (4.2) ensure that variables $y_e$ set to 1 correspond to a set of $s$-$t$ paths, and

that a variable $x_v$ is set to 1 if and only if one of these paths passes through $v$. Antitag-to-tag hybridization constraints ($C$) and $c$-token uniqueness are enforced by (4.3). Finally, the objective (4.1) corresponds to maximizing the number of selected $s$-$t$ paths, since every arc out of $s$ goes to a vertex of $V_{first}$.

For a token $c_i = c_j a \in \mathcal{C}_0$, where $a \in \{\mathtt{A}, \mathtt{T}\}$, let $\widehat{c_i} = c_j \bar{a}$. Since both $c_i$ and $\widehat{c_i}$ contain token $c_j$ as a prefix, it follows that at most one of them can appear in $\mathcal{T}$. Therefore, the following valid inequality can be added to the ILP formulation (4.1)–(4.4) to improve its integrality gap (i.e., the gap between the value of the optimum integer solution and that of the optimal fractional relaxation):

$$\sum_{v \in V_i \cup V_j} x_v \leq 1, \qquad c_i \in \mathcal{C}_0,\ c_j = \widehat{c_i},\ i < j. \tag{4.5}$$

The formulation of MTSDP($h|C|1$) has exactly the same objective and constraints for a slightly modified graph $G$. Let us define the *tail weight* of a $c$-token $c_i$, denoted $tail(c_i)$, as the weight of the last letter of $c_i$. Also, let $h_i = h$ if $c_i$ has a tail weight of 1 and $h_i = h + 1$ if $c_i$ has a tail weight of 2. We will require that every tag ending with token $c_i$ has total weight of at most $h_i$ – it is easy to see that this constraint is not affecting the size of the optimum tag set. The modified graph $G$ has vertex set $V = \{s, t\} \cup \bigcup_{1 \leq i \leq N} V_i$, where $V_i = \{v_i^k \mid w(c_i) \leq k \leq h_i\}$. $G$ contains a directed arc from $v_i^k$ to $v_j^{k+tail(i)}$ for every triple $i, j, k$ with $|c_i| \leq k \leq h_i - tail(c_i)$ for which $c_j$ can be obtained from $c_i$ by appending a single nucleotide and removing the maximal prefix that still leaves a valid $c$-token. Finally, $G$ contains arcs from $s$ to every $v \in V_{first}$, where $V_{first}$ is now equal to $\{v_i^{w(c_i)} \mid c_i \in \mathcal{C} \setminus \mathcal{C}_0\} \cup \{v_i^{w(c_i)+1} \mid c_i \in \mathcal{C}_2\}$, plus arcs from every $v_i^k$ to $t$ for every $1 \leq i \leq N$ and $h_i - tail(c_i) < k \leq h_i$.

# Chapter 5

# Approximation Algorithms for MTSDP$(*|*|1)$

The ILP formulation for MTSDP$(l|C|1)$ (respectively MTSDP$(h|C|1)$) has $O(lN)$ (respectively $O(hN)$) variables and constraints, where $N = \Theta((1+\sqrt{3})^c)$ is the number of $c$-tokens. For small values of $c$ these formulations can be solved to optimality by general purpose optimization packages. However, as shown in chapter 7, even state-of-the-art solvers such as CPLEX require a prohibitive amount of time for values of $c$ greater than 8. In this chapter we present two faster algorithm for computing near-optimal tag sets.

To generate feasible sets of tags for MTSDP$(*|C|1)$ we employ a simple alphabetic tree search algorithm (see Figure 5.1). A similar algorithm is suggested in [29] for the problem of finding sets of tags that satisfy an unweighted version of constraint (C2). We start with an empty set of tags and an empty tag prefix. In every step we try to extend the current tag prefix $t$ by an additional `A`. If the added letter completes a $c$-token that has been used in already selected tags or in $t$ itself, we try the next letter in the DNA alphabet, or backtrack to a previous position in the prefix when no more letter choices are left. Whenever we succeed generating a complete tag, we save it and backtrack to the last letter of its first $c$-token. For MTSDP$(*|\bar{C}|1)$, we not only verify if the added letter completes an unavailable $c$-token, but also if it completes the complement of an unavailable $c$-token.

The alphabetic tree search algorithm guarantees that the set $\mathcal{T}$ of selected tags is *maximal*, i.e., there is no tag $t$ such that $\mathcal{T} \cup \{t\}$ remains feasible for MTSDP$(l|C|1)$. Hence, every tag of an optimal solution must share at least one $c$-token with tags in $\mathcal{T}$.

Since every tag of $\mathcal{T}$ has at most $l-c/2+1$ $c$-tokens, it follows that the size of $\mathcal{T}$ is within a factor of $l - c/2 + 1$ of the size of an optimum MTSDP($l|C|1$) solution. Similarly, the approximation factor of the algorithm when applied to MTSDP($h|C|1$) is no more than $h - c/2 + 2$.

The second algorithm is based on an equivalent ILP formulation of MTSDP($*|C|1$) using "path" instead of "arc" variables. Let $\mathcal{P}$ be the set of all $s$-$t$ paths in the auxiliary graph $G$ defined as in the previous chapter. Using a 0/1 variable $x_p$ for every path $p \in \mathcal{P}$, MTSDP($*|C|1$) can be formulated as follows:

$$\text{maximize} \quad \sum_{p \in \mathcal{P}} x_p \tag{5.1}$$

subject to

$$\sum_{p \in \mathcal{P}} |p \cap V_i| x_p \leq 1, \qquad 1 \leq i \leq N \tag{5.2}$$

$$x_p \in \{0, 1\}, \qquad p \in \mathcal{P}. \tag{5.3}$$

The fractional relaxation of ILP (5.1)-(5.3) is obtained by replacing integrality constraints (5.3) with

$$x_p \geq 0, \qquad p \in \mathcal{P}. \tag{5.4}$$

The optimum solution of the fractional relaxation can be efficiently approximated within any desired accuracy using the algorithm in Figure 5.2, which is a specialization of the approximation algorithm for packing linear programs in [17]. Briefly, the algorithm starts by assigning a small weight $y_i = \delta$ to every set $V_i$. Then, the algorithm repeatedly computes minimum-weight $s$-$t$ paths in $G$, where the weight of a node is given by the weight $y_i$ of the corresponding set $V_i$. For every minimum-weight path $p$, the $y_i$'s corresponding to visited sets $V_i$ are multiplied by a factor of $(1 + \epsilon \frac{|p \cap V_i|}{\max_i |p \cap V_i|})$. Finally, the algorithm stops when the weight of every $s$-$t$ path is greater than or equal to 1.

Since the auxiliary graph $G = (V, E)$ is directed and acyclic, the minimum weight path can be computed in $O(|V| + |E|)$ time. Therefore, using the fact that $G$ has $O(lN)$ vertices and arcs for MTSDP($l|C|1$), and $O(hN)$ vertices and arcs for MTSDP($h|C|1$), Theorem 3.1 of [17] gives:

```
Input: Positive integers c and l, c ≤ l
Output: Feasible MTSDP(l|C|1) solution 𝒯
---
Mark all c-tokens as available
For every i ∈ {1, 2, ..., l}, B_i ← A
𝒯 ← ∅; Finished ← 0; pos ← 1
While Finished = 0 do
    While the weight of B_1 B_2 ... B_pos < c do
        pos ← pos + 1
    EndWhile
    If the c-token ending B_1 B_2 ... B_pos is available then
        Mark the c-token ending at position pos as unavailable
        If pos = l then
            𝒯 ← 𝒯 ∪ {B_1 B_2 ... B_l}
            pos ← [the position where the first c-token of B_1 B_2 ... B_l ends]
            I ← {i | 1 ≤ i ≤ pos, B_i ≠ G}
            If I = ∅ then
                Finished ← 1
            Else
                pos ← max{I}
                B_i ← A for all i ∈ {pos + 1, ..., l}
                B_pos ← nextbase(B_pos)
            EndIf
        Else
            pos ← pos + 1
        EndIf
    Else
        I ← {i | 1 ≤ i ≤ pos, B_i ≠ G}
        If I = ∅ then
            Mark all the c-tokens in B_1 B_2 ... B_{pos-1} as available
            Finished ← 1
        Else
            prevpos ← pos
            pos ← max{I}
            Mark all the c-tokens in B_pos ... B_{prevpos-1} as available
            B_i ← A for all i ∈ {pos + 1, ..., l}
            B_pos ← nextbase(B_pos)
        EndIf
    EndIf
EndWhile
```

Figure 5.1: The alphabetic tree search algorithm for MTSDP($l|C|1$). The *nextbase*$(\cdot)$ function is defined by *nextbase*(A) = T, *nextbase*(T) = C, and *nextbase*(C) = G.

```
Input: ε > 0
Output: Feasible solution (x_p)_{p∈P} to the fractional relaxation of ILP (5.1)-(5.3)
```

---

For every $p \in \mathcal{P}$, $x_p \leftarrow 0$
$\delta \leftarrow (1+\epsilon)((1+\epsilon)N)^{-1/\epsilon}$
For every $i \in \{1,\ldots,N\}$, $y_i \leftarrow \delta$
Find a minimum weight $s$-$t$ path $p$ in $G$, where $weight(v) = y_i$ for every $v \in V_i$, $i \in \{1,\ldots,N\}$
**While** $weight(p) < 1$ **do**
    $M \leftarrow \max_i |p \cap V_i|$
    $x_p \leftarrow x_p + \frac{1}{M}$
    For every $i$, $y_i \leftarrow y_i(1 + \epsilon\frac{|p \cap V_i|}{M})$
    Find a minimum weight $s$-$t$ path $p$ in $G$, where $weight(v) = y_i$ for every $v \in V_i$, $i \in \{1,\ldots,N\}$
**EndWhile**
For every $p \in \mathcal{P}$, $x_p \leftarrow x_p / \left(\log_{1+\epsilon} \frac{1+\epsilon}{\delta}\right)$

Figure 5.2: The Garg and Könemann algorithm.

**Theorem 3** *The algorithm shown in Figure 5.2 computes a $(1-\epsilon)^2$-approximation to the fractional relaxation in $O(lN^2\lceil\frac{1}{\epsilon}\log_{1+\epsilon}N\rceil)$ (respectively $O(hN^2\lceil\frac{1}{\epsilon}\log_{1+\epsilon}N\rceil)$) time for MTSDP($l|C|1$) (respectively MTSDP($h|C|1$)).*

The fractional solution computed by the Garg and Könemann algorithm is then used to construct a feasible set of tags using a simple method that has been shown in [13] to work better in practice than classical randomized rounding [33], particularly when starting from poor approximate solutions such as those obtained by running the algorithm in Figure 5.2 with a large value of $\epsilon$. We simply save the list of $s$-$t$ paths selected as minimum-weight paths by the Garg and Könemann algorithm (excluding minimum-weight paths that visit some set $V_i$ more than once, since such paths do not correspond to valid tags) and then, traversing the list in reverse order, we sequentially pick tags that correspond to paths visiting only sets $V_i$ not yet appearing in the already picked tags. Finally, we mark all $c$-tokens of picked tags as unavailable, and augment the set $\mathcal{T}$ of picked tags with the additional tags found by running the alphabetic tree search algorithm described in chapter 3.

# Chapter 6

# Algorithms for MTSDP$(*|*|multiple)$

In this chapter we describe two algorithms for MTSDP($l|C|multiple$); both algorithms can be easily adjusted to handle the other MTSDP($*|*|multiple$) variants. The first algorithm (see Figure 6.1 for a detailed pseudocode) is similar to the alphabetic tree search algorithm described for MTSDP($l|C|1$). The algorithm performs an alphabetical traversal of a 4-ary tree representing all $4^l$ possible tags, skipping over subtrees rooted at internal vertices that correspond to tag prefixes including unavailable $c$-tokens. The difference from the MTSDP($l|C|1$) algorithm in chapter 3 lies in the strategy used to mark $c$-tokens as unavailable. While the algorithm given in chapter 3 marks a $c$-token $C$ as unavailable as soon as it incorporates it in the current tag prefix (changing $C$'s status back to "available" when forced to backtrack past $C$'s tail), the algorithm in Figure 6.1 marks a $c$-token as unavailable only when a complete tag is found.

We call a tag $t$ *periodic* if $t$ is the length $l$ prefix of an infinite string $x^\infty$, where $x$ is a DNA string with $|x| < |t|$. (Note that a periodic tag $t$ is not necessarily the concatenation of an integer number of copies of its period $x$ as in the standard definition of string periodicity [27].)

The following lemma shows that tag set design algorithms can restrict the search to two simple classes of tags.

**Lemma 4** *For every $c$ and $l$, there exists an optimal tag set $\mathcal{T}$ in which every tag has the uniqueness property or is periodic. (Note that the two classes of tags are not disjoint, as there are tags that are both periodic and posses the uniqueness property.)*

```
Input: Positive integers c and l, c ≤ l
Output: Feasible MTSDP(l|C|multiple) solution 𝒯
---
Mark all c-tokens as available
For every i ∈ {1, 2, ..., l}, B_i ← A
𝒯 ← ∅; Finished ← 0; pos ← 1
While Finished = 0 do
    While the weight of B_1 B_2 ... B_pos < c do
        pos ← pos + 1
    EndWhile
    If the c-token ending B_1 B_2 ... B_pos is available then
        If pos = l then
            𝒯 ← 𝒯 ∪ {B_1 B_2 ... B_l}
            Mark all the c-tokens of B_1 B_2 ... B_l as unavailable
            pos ← [the position where the first c-token of B_1 B_2 ... B_l ends]
            I ← {i | 1 ≤ i ≤ pos, B_i ≠ G}
            If I = ∅ then
                Finished ← 1
            Else
                pos ← max{I}
                B_i ← A for all i ∈ {pos + 1, ..., l}
                B_pos ← nextbase(B_pos)
            EndIf
        Else
            pos ← pos + 1
        EndIf
    Else
        I ← {i | 1 ≤ i ≤ pos, B_i ≠ G}
        If I = ∅ then
            Finished ← 1
        Else
            pos ← max{I}
            B_i ← A for all i ∈ {pos + 1, ..., l}
            B_pos ← nextbase(B_pos)
        EndIf
    EndIf
EndWhile
```

Figure 6.1: The alphabetic tree search algorithm for MTSDP($l|C|$*multiple*). The *nextbase*$(\cdot)$ function is defined by *nextbase*(A) = T, *nextbase*(T) = C, and *nextbase*(C) = G.

**Proof.** Let $\mathcal{T}$ be an optimal tag set. Assume that $\mathcal{T}$ contains a tag $t$ that does not have the uniqueness property, and let $c_{i_1}, \ldots, c_{i_k}$ be the sequence of $c$-tokens occurring in $t$, in left to right order. Since $t$ does not have the uniqueness property, there exist indices $1 \le j < j' \le i_k$ such that $c_{i_j} = c_{i_{j'}}$. Let $t'$ be the tag formed by taking the first $l$ letters of the infinite string with $c$-token sequence $(c_{i_j}, \ldots, c_{i_{j'-1}})^\infty$; note that $t'$ is a periodic tag. Since $c$-tokens $c_{i_j}, \ldots, c_{i_{j'-1}}$ do not appear in the tags of $\mathcal{T} \setminus \{t\}$, it follows that $(\mathcal{T} \setminus \{t\}) \cup \{t'\}$ is also optimal. Repeated application of this operation yields the lemma. □

Note that a periodic tag whose shortest period has length $p$ contains as substrings

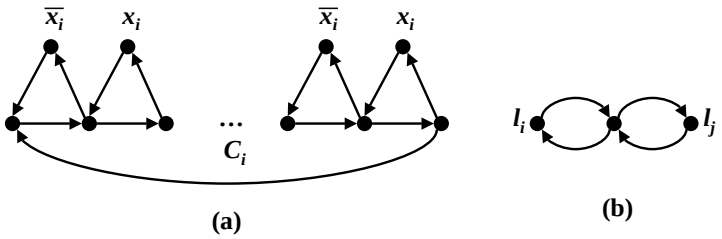


Figure 6.2: Vertices and arcs added to $G(\phi)$ for (a) variable $x_i$, and (b) clause $l_i \vee l_j$.

exactly $p$ $c$-tokens, while tags with the uniqueness property contain between $l - c + 1$ and $l - c/2 + 1$ $c$-tokens. Therefore, of the two classes of tags in Lemma 4, periodic tags (particularly those with short periods) make better use of the limited number of available $c$-tokens.

Each periodic tag corresponds to a directed cycle in the graph $H_c$ which has $\mathcal{C}$ 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. Clearly, a vertex-disjoint packing of $n$ cycles in $H_c$ yields a feasible solution for MTSDP($l|C|$*multiple*) consisting of $n$ tags, since we can extract at least one tag of length $l$ from each cycle, and tags extracted from different cycles do not have common $c$-tokens. This motivates the following:

Maximum Vertex-Disjoint Directed Cycle Packing Problem: Given a directed graph $G$, find a maximum number of vertex-disjoint directed cycles in $G$.

The next theorem shows that Maximum Vertex-Disjoint Directed Cycle Packing in arbitrary graphs is unlikely to admit a polynomial approximation scheme. A stronger inapproximability result was established for arbitrary graphs by Salavatipour and Verstraete [35], who proved that there is no $O(\log^{1-\varepsilon} n)$-approximation for Maximum Vertex-Disjoint Directed Cycle Packing unless $NP \subseteq DTIME(2^{polylogn})$. On the positive side, Salavatipour and Verstraete showed that Maximum Vertex-Disjoint Directed Cycle Packing can be approximated within a factor of $O(\sqrt{n})$ via linear programming techniques, matching the best approximation factor known for the arc-disjoint version of the problem [25].

**Theorem 4** MAXIMUM VERTEX-DISJOINT DIRECTED CYCLE PACKING *is APX-hard even for regular directed graphs with in-degree and out-degree of 2.*

**Proof.** We use a reduction from the MAX-2-SAT-3 problem, similar to the one in [11]. An instance $\phi$ of MAX-2-SAT-3 consists of a set $\{c_1, \ldots, c_m\}$ of disjunctive clauses over a ground set $\{x_1, \ldots, x_n\}$ of variables. Each clause consists of at most 2 literals (variables or negations of variable), and each variable appears in at most 3 clauses, counting both negated and non-negated occurrences. The objective is to find a truth assignment that satisfies as many of the clauses as possible. It is known that MAX-2-SAT-3 is APX-hard [4, 7].

Let $m_i$ denote the number of occurrences of variable $x_i$ in a given instance of MAX-2-SAT-3. We construct, in polynomial time, a directed graph $G(\phi)$ as follows. For each variable $x_i$ we add to $G$ a directed cycle $C_i$ of length $4m_i$, plus $2m_i$ additional vertices alternatively labeled by $x_i$ and $\bar{x}_i$, used to close a directed cycle of length 3 with each arc of $C_i$, as in Figure 6.2(a). For each unary clause we pick a distinct vertex labeled by the *negation* of the respective literal and attach a loop to it. Finally, for each 2-literal clause $c$ we pick 2 vertices labeled by the negations of the literals of $c$, again without reusing labeled vertices between clauses, and use a new vertex to connect them via two length-2 cycles as in Figure 6.2(b). Note that, for every $i$, at least $2\sum_{i=1}^{n} m_i$ of the labeled vertices remain incident to a single cycle; we will refer to these as "free" labeled vertices.

We claim that every truth assignment that makes $k$ clauses of $\phi$ true can be converted in polynomial time into a set of $k + 2\sum_{i=1}^{n} m_i$ vertex disjoint cycles of $G(\phi)$, and vice-versa. Indeed, for a given truth assignment, select (1) the $2m_i$ length-3 cycles passing through nodes labeled by $\bar{x}_i$ for every variable $x_i$ that is set to true, (2) the $2m_i$ length-3 cycles passing through nodes labeled by $\bar{x}_i$ for every variable $x_i$ that is set to false, and (3) the loop or length-2 cycle passing through a labeled node corresponding to a *false* literal. It is easy to verify that these cycles are vertex-disjoint.

Conversely, let $\mathcal{C}$ be a set of $k + 2\sum_{i=1}^{n} m_i$ vertex disjoint cycles of $G(\phi)$. If any of the cycles $C_i$ is in $\mathcal{C}$, we replace it by the length-3 cycle passing through a free labeled vertex. Similarly, if any of the cycles in $\mathcal{C}$ visits two of the arcs of a 3-cycle (or one

of the arcs of a 2-cycle), we replace it by the 3-cycle (respectively 2-cycle) itself. After this transformation we have a set of $k + 2\sum_{i=1}^{n} m_i$ vertex-disjoint loops, 2-cycles, and 3-cycles. We say that a set of cycles is *consistent* if only one of the labels $x_i$, $\bar{x}_i$ appear in $\mathcal{C}$ for every $i$. If $\mathcal{C}$ is consistent, we choose a truth assignment that makes all literals corresponding to labels in $\mathcal{C}$ true. It is easy to see that at least $k$ of the cycles in $\mathcal{C}$ must be loops and 2-cycles, and clauses corresponding to these cycles are satisfied by the above truth assignment.

Otherwise, we make $\mathcal{C}$ consistent by repeating the following transformation. Let $i$ be an index for which both $x_i$ and $\bar{x}_i$ appear in $\mathcal{C}$. Without loss of generality, assume that $x_i$ appears in only one clause of $\phi$ (recall that, together, $x_i$ and $\bar{x}_i$ can appear in at most 3 clauses). It follows that there is a single loop or 2-cycle $C \in \mathcal{C}$ visiting a vertex labeled by $\bar{x}_i$ – all other vertices labeled by $\bar{x}_i$ are free. Since the $x_i$'s and $\bar{x}_i$'s alternate around $C_i$, the cycles going through vertices labeled by $\bar{x}_i$ can be replaced by at least the same number of 3-cycles going through vertices labeled by $x_i$.

To complete the proof of the theorem, notice that the optimum number of satisfiable clauses, $k_{opt}$, is at least $m/2$, since we can repeatedly assign a variable such that at least half of the clauses containing it are satisfied. Hence, $\sum_{i=1}^{n} m_i \leq 2m \leq 4k_{opt}$. If there exists a polynomial time algorithm with an approximation factor of $\frac{1}{1-\varepsilon}$ for Maximum Vertex-Disjoint Directed Cycle Packing, we can run it on $G(\phi)$ to get a set $\mathcal{C}$ of at least $k + 2\sum_{i=1}^{n} m_i \geq \frac{1}{1-\varepsilon}(k_{opt} + 2\sum_{i=1}^{n} m_i)$ vertex disjoint cycles, and then convert $\mathcal{C}$ as above into a truth assignment satisfying $k \geq \frac{1+8\varepsilon}{1-\varepsilon} k_{opt}$ clauses of $\phi$. □

We use a simple greedy algorithm to solve Maximum Vertex-Disjoint Directed Cycle Packing for the graph $H_c$: we enumerate possible tag periods in pseudo-lexicographic order, and check for each period if all $c$-tokens are available for the resulting tag. We refer to this algorithm as the *greedy cycle packing algorithm*, since it is equivalent to packing cycles greedily in order of length.

# Chapter 7

# Experimental Results

Tables 7.1 and 7.2 give empirical results for the MTSDP($l|C|1$) and MTSDP($h|C|1$) problems respectively. We give the number of selected tags and runtimes for the following three algorithms:

- the tree search (TS) algorithm in chapter 3,
- the Garg-Könemann based algorithm described in chapter 5 (denoted LP approx) which we ran with $\epsilon = 0.5$, and
- the CPLEX 9.0 commercial solver applied to ILP (4.1)–(4.4).

All compared algorithms were run using a single CPU of a dual 2.8 GHz Dell PowerEdge 2600 Linux server with 4Gb of main memory. Missing LP/ILP entries did not complete in 10 hours.

Optimum tag sets are found by CPLEX for small values of $c$. However, even computing the optimum fractional relaxation of ILP (4.1)–(4.4) is impractical for $c$ greater than 8. In contrast, the Garg-Könemann based algorithm is much more scalable than CPLEX, and generally produces better solutions than running the tree search algorithm alone, although we run it with a very large value for $\epsilon$.

To help assessing the quality of the compared algorithms when the optimum solution is not available, we also include in Tables 7.1 and 7.2 the $c$-token count upper bound established for MTSDP($l|C|1$) in chapter 3 and the tail-weight upper bound established for MTSDP($h|C|1$) in [6], as well as the value of the fractional (LP) relaxation of ILP (4.1)–(4.4). For all cases where the optimum ILP solution could be computed, the difference between the optimal fractional and integer solution values is smaller than

Table 7.1: ILP results for MTSDP($l|C|1$), i.e., tag set design with specified tag length $l$, antitag-to-tag hybridization constraints, and a unique copy of each $c$-token allowed in a tag.

| $l$ | $c$ | # Selected Tags | | | Upper Bounds | | CPU Seconds | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | TS | LP approx | ILP | LP | $c$-token count | TS | LP approx | LP | ILP |
| 10 | 4 | 7 | 7 | 8 | 8.57 | 9 | 0.00 | 0.27 | 0.13 | 0.71 |
| | 5 | 23 | 25 | 28 | 28.00 | 29 | 0.00 | 0.32 | 2.27 | 5.85 |
| | 6 | 67 | 79 | 85 | 85.60 | 96 | 0.00 | 0.57 | 11.40 | 98.25 |
| | 7 | 196 | 232 | 259 | 259.67 | 328 | 0.00 | 3.11 | 86.70 | 586.67 |
| | 8 | 655 | 793 | 853 | 853.33 | 1194 | 0.00 | 63.01 | 552.74 | 4321.66 |
| | 9 | 2359 | 2703 | – | – | 4896 | 0.01 | 841.64 | – | – |
| | 10 | 9072 | 10144 | – | – | 26752 | 0.04 | 11019.64 | – | – |
| 20 | 4 | 3 | 3 | 3 | 3.53 | 3 | 0.00 | 0.32 | 1.05 | 58.46 |
| | 5 | 9 | 9 | 10 | 10.50 | 11 | 0.00 | 0.40 | 13.72 | 381.33 |
| | 6 | 26 | 26 | 29 | 29.87 | 32 | 0.00 | 1.26 | 182.96 | 12448.61 |
| | 7 | 75 | 75 | – | 88.00 | 93 | 0.00 | 4.79 | 2675.68 | – |
| | 8 | 213 | 220 | – | 257.23 | 275 | 0.00 | 49.21 | 134525.81 | – |
| | 9 | 600 | 641 | – | – | 816 | 0.00 | 624.16 | – | – |
| | 10 | 1667 | 1854 | – | – | 2432 | 0.04 | 7717.13 | – | – |

1, indicating that the LP solution is a very tight upper bound. Furthermore, ILP results confirm the high quality of the upper bound established for MTSDP($h|C|1$) in [6]; the upper bound established in chapter 3 for MTSDP($l|C|1$) appears to be somehow weaker.

Tables 7.3 and 7.4 give the results obtained for MTSDP($*|*|$*multiple*) by the alphabetic tree search algorithm in Figure 6.1 respectively by the greedy cycle packing algorithm (in our implementation, we impose an upper bound of 15 on the length of the cycles that we try to pack) followed by running the alphabetic tree search algorithm with the $c$-tokens occurring in the selected cycles already marked as unavailable. Performing cycle packing significantly improves the results compared to running the alphabetic tree search algorithm alone; as shown in the tables, most of the resulting tags are found in the cycle packing phase of the combined algorithm.

Across all instances, the combined algorithm increases the number of tags by at least 40% compared to the best available MTSDP($*|*|1$) algorithm –the improvement is much higher for smaller values of $c$. Quite notably, although the number of tags is increased, the tag sets found by the combined algorithm use a *smaller* total number of $c$-tokens. Thus, these tag sets are less likely to cross-hybridize to the primers used in the reporter probes, enabling higher tag utilization rates during tag assignment [5].

Table 7.2: ILP results for MTSDP($h|C|1$), i.e., tag set design with specified minimum tag weight $h$, antitag-to-tag hybridization constraints, and a unique copy of each $c$-token allowed in a tag.

| $h$ | $c$ | # Selected Tags | | | Upper Bounds | | CPU Seconds | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | TS | LP approx | ILP | LP | tail-weight | TS | LP approx | LP | ILP |
| 15 | 4 | 6 | 5 | 7 | 7.00 | 7 | 0.00 | 0.34 | 0.45 | 9.04 |
| | 5 | 18 | 18 | 21 | 21.09 | 21 | 0.00 | 0.38 | 5.66 | 117.62 |
| | 6 | 47 | 52 | 63 | 63.20 | 63 | 0.00 | 0.89 | 54.43 | 2665.39 |
| | 7 | 149 | 155 | 192 | 192.00 | 192 | 0.00 | 5.49 | 544.95 | 3644.85 |
| | 8 | 460 | 480 | – | 588.00 | 590 | 0.00 | 99.21 | 7153.87 | – |
| | 9 | 1197 | 1608 | – | – | 1842 | 0.00 | 1788.07 | – | – |
| | 10 | 3669 | 4947 | – | – | 5872 | 0.07 | 24223.92 | – | – |
| 28 | 4 | 3 | 3 | 3 | 3.30 | 3 | 0.00 | 0.46 | 1.88 | 132.78 |
| | 5 | 8 | 8 | 9 | 9.67 | 9 | 0.00 | 0.60 | 34.66 | 1137.21 |
| | 6 | 22 | 22 | 27 | 27.48 | 27 | 0.00 | 1.26 | 392.42 | 18987.09 |
| | 7 | 64 | 63 | – | 78.55 | 78 | 0.00 | 8.89 | 7711.41 | – |
| | 8 | 175 | 182 | – | 224.76 | 224 | 0.00 | 111.63 | 850642.82 | – |
| | 9 | 531 | 515 | – | – | 644 | 0.00 | 1606.85 | – | – |
| | 10 | 1428 | 1491 | – | – | 1854 | 0.02 | 26728.47 | – | – |

Table 7.3: Results for MTSDP($*|C|multiple$), i.e., tag set design with antitag-to-tag hybridization constraints and multiple copies of a $c$-token allowed in a tag.

| $l/h$ | $c$ | One $c$-token copy | | Multiple $c$-token copies | | | | |
|---|---|---|---|---|---|---|---|---|
| | | LP approx | | Tree search | | Cycle packing + Tree search | | |
| | | tags | $c$-tokens | tags | $c$-tokens | tags | $c$-tokens | % cyclic |
| $l = 20$ | 4 | 3 | 51 | 14 | 59 | 17 | 40 | 100.0 |
| | 5 | 9 | 146 | 31 | 165 | 40 | 140 | 100.0 |
| | 6 | 26 | 402 | 53 | 433 | 72 | 293 | 98.6 |
| | 7 | 75 | 1096 | 124 | 1179 | 178 | 928 | 99.4 |
| | 8 | 220 | 3014 | 281 | 3095 | 383 | 2411 | 97.1 |
| | 9 | 641 | 8322 | 711 | 8230 | 961 | 7102 | 96.9 |
| | 10 | 1854 | 22693 | 1835 | 21400 | 2344 | 19691 | 95.1 |
| $h \geq 28$ | 4 | 3 | 58 | 14 | 61 | 17 | 40 | 100.0 |
| | 5 | 8 | 151 | 32 | 174 | 40 | 140 | 100.0 |
| | 6 | 22 | 391 | 44 | 432 | 72 | 300 | 98.6 |
| | 7 | 63 | 1083 | 118 | 1200 | 178 | 934 | 99.4 |
| | 8 | 182 | 2996 | 239 | 3037 | 379 | 2405 | 96.6 |
| | 9 | 515 | 8025 | 632 | 8622 | 943 | 6969 | 96.5 |
| | 10 | 1491 | 22183 | 1570 | 22145 | 2260 | 19270 | 94.1 |

Table 7.4: Results for MTSDP($*|\bar{C}|$*multiple*), i.e., tag set design with both antitag-to-tag and antitag-to-antitag hybridization constraints and multiple copies of a $c$-token allowed in a tag.

| $l/h$ | $c$ | One $c$-token copy | | Multiple $c$-token copies | | | | |
|---|---|---|---|---|---|---|---|---|
| | | Tree search | | Tree search | | Cycle packing + Tree search | | |
| | | tags | $c$-tokens | tags | $c$-tokens | tags | $c$-tokens | % cyclic |
| $l = 20$ | 4 | 1 | 17 | 10 | 35 | 10 | 25 | 100.0 |
| | 5 | 4 | 65 | 17 | 83 | 23 | 85 | 100.0 |
| | 6 | 13 | 200 | 30 | 241 | 41 | 171 | 97.6 |
| | 7 | 37 | 537 | 68 | 585 | 97 | 512 | 99.0 |
| | 8 | 107 | 1480 | 147 | 1619 | 202 | 1268 | 98.0 |
| | 9 | 300 | 3939 | 362 | 4124 | 512 | 3799 | 96.3 |
| | 10 | 844 | 10411 | 934 | 10869 | 1204 | 10089 | 95.8 |
| $h \geq 28$ | 4 | 1 | 22 | 10 | 36 | 10 | 25 | 100.0 |
| | 5 | 4 | 74 | 17 | 84 | 23 | 85 | 100.0 |
| | 6 | 12 | 213 | 29 | 238 | 41 | 178 | 97.6 |
| | 7 | 32 | 559 | 64 | 586 | 97 | 518 | 99.0 |
| | 8 | 90 | 1489 | 135 | 1632 | 199 | 1238 | 98.0 |
| | 9 | 263 | 4158 | 329 | 4314 | 504 | 3760 | 95.8 |
| | 10 | 714 | 10837 | 809 | 11250 | 1163 | 9937 | 93.6 |

# Chapter 8

# Conclusions

In this thesis we proposed new solution methods for designing optimal and near-optimal tag sets for universal DNA arrays. Most notably, we have shown that the use of periodic tags leads to over 40% more tags compared to best previous methods. Our algorithms use simple combinatorial ideas and greedy strategies that can be easily extended to handle more sophisticated hybridization models such as the near-neighbor model of [36], and can incorporate additional practical design constraints, such as preventing the formation of hairpin secondary structures, or disallowing specific nucleotide sequences such as runs of 4 identical nucleotides [29].

In ongoing work we seek to extend our methods to emerging applications of universal tag arrays in microfluidics-based labs-on-a-chip, as well as DNA-mediated assembly of nanoscale devices such as carbon-nanotube-based field-effect transistors [20]. An interesting open problem is to find tight upper bounds and exact methods for the MTSDP($*|*|$*multiple*) formulations. Settling the approximation complexity of Maximum Vertex-Disjoint Directed Cycle Packing is another interesting problem.

# Bibliography


[1] Affymetrix, Inc. Geneflex tag array technical note no. 1, available online at `http://www.affymetrix.com/support/technical/technotes/genflex_technote.pdf`. 2001.

[2] Agilent Technologies, Inc. `http://www.agilent.com`.

[3] U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, , and A.J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues by oligonucletide arrays. *Proc. Natl. Acad. of Sci. USA*, 96:6745–6750, 1999.

[4] G. Ausiello, M. Protasi, A. Marchetti-Spaccamela, G. Gambosi, P. Crescenzi, and V. Kann. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer-Verlag, New York, 1999.

[5] A. Ben-Dor, T. Hartman, B. Schwikowski, R. Sharan, and Z. Yakhini. Towards optimally multiplexed applications of universal DNA tag systems. In *Proceedings of the 7th Annual International Conference on Research in Computational Molecular Biology*, pages 48–56, 2003.

[6] A. Ben-Dor, R. Karp, B. Schwikowski, and Z. Yakhini. Universal DNA tag systems: a combinatorial design scheme. *Journal of Computational Biology*, 7(3-4):503–519, 2000.

[7] P. Berman and M. Karpinski. On some tighter inapproximability results. In *Proceedings of the 26th International Colloquium on Automata, Languages and Programming*, pages 200–209, 1999.

[8] A.P. Blanchard and L. Hood. Sequence to array: probing the genomes secrets. *Nature Biotechnology*, 14:1649, 1996.

[9] A. Brenneman and A. Condon. Strand design for biomolecular computation. *Theor. Comput. Sci.*, 287(1):39–58, 2002.

[10] S. Brenner. Methods for sorting polynucleotides using oligonucleotide tags. *US Patent 5,604,097*, 1997.

[11] A. Caprara, A. Panconesi, and R. Rizzi. Packing cycles in undirected graphs. *Journal of Algorithms*, 48(1):239–256, 2003.

[12] J. DeRisi, V. Iyer, and P. Brown. Exploring the metabolic genetic control of gene expression on a genomic scale. *Science*, 278:680–686, 1997.

[13] F.F. Dragan, A.B. Kahng, I.I. Măndoiu, S. Muddu, and A.Z. Zelikovsky. Provably good global buffering by generalized multiterminal multicommodity flow approximation. *IEEE Transactions on Computer-Aided Design*, 21(3):263–274, 2002.

[14] R. Drmanac, G. Lennon, S. Drmanac, I. Labat, R. Crkvenjakov, and H. Lehrach. Partial sequencing by oligohybridization: Concept and applications in genome analysis. In *Proceedings of the first international conference on electrophoresis supercomputing and the human genome*, pages 60–75. World Scientific, 1991.

[15] D.G. Wang et al. Large-scale identification, mapping and genotyping of single nucleotide polymorphisms in the human genome. *Science*, 280:1077–1082, 1998.

[16] A.G. Frutos, Q. Liu, A.J. Thiel, A.M.W. Sanner, A.E. Condon, L.M. Smith, and R.M. Corn. Demonstration of a word design strategy for DNA computing on surfaces. *Nucleic Acids Research*, 25:4748–4757, 1997.

[17] N. Garg and J. Konemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In *Proceedings of the 39th IEEE Annual Symposium on Foundations of Computer Science*, pages 300–309, 1998.

[18] N.P. Gerry, N.E. Witowski, J. Day, R.P. Hammer, G. Barany, and F. Barany. Universal DNA microarray method for multiplex detection of low abundance point mutations. *J. Mol. Biol.*, 292(2):251–262, 1999.

[19] J.G. Hacia. Resequencing and mutational analysis using oligonucleotide microarrays. *Nature Genetics*, 21(1):42–47, 1999.

[20] M. Hazani, D. Shvarts, D. Peled, V. Sidorov, and R. Naaman. Self-assembled carbon-nanotube-based field-effect transistors. *Applied Physics Letters*, 85:5025–5027, 2004.

[21] J.N. Hirschhorn, P. Sklar, K. Lindblad-Toh, Y.-M. Lim, M. Ruiz-Gutierrez, S. Bolk, B. Langhorst, S. Schaffner, E. Winchester, and E. Lander. SBE-TAGS: An array-based method for efficient single-nucleotide polymorphism genotyping. *PNAS*, 97(22):12164–12169, 2000.

[22] L. Kaderali. *Selecting Target Specific Probes for DNA Arrays*. PhD thesis, Köln University, 2001.

[23] K.R. Khrapko, A. Khorlin, I.B. Ivanov, B.K. Chernov, Y.P. Lysov, S. Vasilenko, V. Florenyev, and Mirzabekov. Hybridization of dna with oligonucleotides immobilized in gel: A convenient method for detecting single base substitutions. *Molecular Biology*, 25(3):581–591, 1991.

[24] M. Kozal, N. Shah, N. Shen, R. Fucini, R. Yang, T. Merigan, D.D. Richman, M.S. Morris, E. Hubbell, M. Chee, and T.R. Gingeras. Extensive polymorphisms observed in hiv-1 clade b protease gene using high density oligonucleotide arrays: implications for therapy. *Nature Medicine*, 7:753–759, 1996.

[25] M. Krivelevich, Z. Nutov, and R. Yuster. Approximation algorithms for cycle packing problems. In *Proc. ACM-SIAM Annual Symposium on Discrete Algorithms*, pages 556–561, 2005.

[26] C.Y. Lin, K.H. Hahnenberger, M.T. Cronin, D. Lee, N.M. Sampas, and R. Kanemoto. A method for genotyping cyp2d6 and cyp2c19 using genechip probe array hybridization. In *ISSX Meeting*, 1996.

[27] M. Lothaire. *Combinatorics on Words*, volume 17 of *Encyclopedia of Mathematics and Its Applications*, pages xix+238. Addison-Wesley, 1983.

[28] M.S. Morris, D.D. Shoemaker, R.W. Davis, and M.P. Mittmann. Methods and compositions for selecting tag nucleic acids and probe arrays. *European Patent Application 97302313*, 1997.

[29] M.S. Morris, D.D. Shoemaker, R.W. Davis, and M.P. Mittmann. Selecting tag nucleic acids. *U.S. Patent 6,458,530 B1*, 2002.

[30] I.I. Măndoiu, C. Prăjescu, and D. Trincă. Improved tag set design and multiplexing algorithms for universal arrays. In *Proceedings of the 5th International Conference on Computational Science (ICCS 2005)/ 2005 International Workshop on Bioinformatics Research and Applications (IWBRA), May 22–25, 2005, Atlanta, GA, USA*, volume 3515, pages 994–1002. Lecture Notes in Computer Science, Springer-Verlag. Extended version appeared in *LNCS Transactions on Computational Systems Biology*, volume 3680, pages 124–137, 2005, Springer-Verlag; also available as ACM Computing Research Repository report cs.DS/0502054.

[31] I.I. Măndoiu and D. Trincă. Exact and approximation algorithms for dna tag set design. In *Proceedings of 16th Annual Symposium on Combinatorial Pattern Matching, June 19-22, 2005, Jeju Island, Korea*, volume 3537, pages 383–393. Lecture Notes in Computer Science, Springer-Verlag. Extended version accepted to *Journal of Computational Biology*.

[32] Oxford Gene Technology. `http://www.ogt.co.uk`.

[33] P. Raghavan and C.D. Thomson. Randomized rounding. *Combinatorica*, pages 365–374, 1987.

[34] Rosetta Biosoftware, Inc. `http://www.rosettabio.com`.

[35] M.R. Salavatipour and J. Verstraete. Disjoint cycles: Integrality gap, hardness, and approximation. In *Proceedings of the 11th Conference on Integer Programming and Combinatorial Optimization (IPCO), June 8–10, 2005, Berlin, Germany*, pages 51–65. Lecture Notes in Computer Science, Springer-Verlag.

[36] J. SantaLucia. A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbor thermodynamics. *Proc. Natl. Acad. Sci. USA*, 95:1460–1465, 1998.

[37] D.D. Shoemaker, D.A. Lashkari, D. Morris., M. Mittmann, and R.W Davis. Quantitative phenotypic analysis of yeast deletion mutants using a highly parallel molecular bar-coding strategy. *Nature Genetics*, 4:450–456, 1996.

[38] D. Solas, A.C. Pease, E.J. Sullivan, M.T. Cronin, C.P. Holmes, and S.P.A. Fodor. Oligonucleotide arrays for rapid dna sequence analysis. *Proc. Natl. Acad. of Sci. USA*, 91:5022–5026, 1994.

[39] Texas Instruments, Inc. `http://www.ti.com`.

[40] R.B. Wallace, J. Shaffer, R.F. Murphy, J. Bonner, T. Hirose, and K. Itakura. Hybridization of synthetic oligodeoxyribonucleotides to phi chi 174 DNA: the effect of single base pair mismatch. *Nucleic Acids Res.*, 6(11):6353–6357, 1979.

[41] J.D. Watson, M. Gilman, J. Witkowski, and M. Zoller. *Recombinant DNA*. Scientific American Books, 1996.