# A Multi-objective Genetic Algorithm Based Approach to the Optimization of Oligonucleotide Microarray Production Process

Filippo Menolascina[1], Vitoantonio Bevilacqua[1], Caterina Ciminelli[1], Mario Nicola Armenise[1], and Giuseppe Mastronardi[1]

Department of Electrotechnics and Electronics, Technical University of Bari, 70126, Italy
f.menolascina@ieee.org

**Abstract.** Microarrays are becoming more and more utilized in the experimental platform in molecular biology. Although rapidly becoming affordable, these micro devices still have quite high production cost which limits their commercial appeal. Here we present a novel multiobjective evolutionary approach to the optimization of the production process of microarray devices mainly aimed at lowering the number of fabrication steps. In order to allow the reader to better understand what we describe we report herein a detailed description of a real-world study case carried out on the most recent microarray platforms of the market leader in this field. A comparative analysis of the most widely used approaches, main potentialities and drawbacks of the proposed approach are presented.

## 1 Introduction

An oligonucleotide microarray is a piece of glass or plastic material on which single-stranded fragments of DNA, called probes, are placed or synthesized. The chips produced, for instance, can contain more than one million spots (or features) as small as 11 $\mu$m, with each spot accommodating several million copies of a probe. Probes are typically 25 nucleotides long and are synthesized in parallel, on the chip, in a series of repetitive steps. Each step appends the same nucleotide to probes of selected regions of the chip. Selection occurs by exposure to light with the help of a photolithographic mask[1].

Formally, we have a set of probes $P = \{p_1, p_2, \dots p_n\}$ that are produced by a series of masks $M = \{m_1, m_2, \dots m_T\}$, each mask $m_t$ allowing the addition of a particular nucleotide $S_t \in \{A, C, G, T\}$ to be included in a subset of $P$. The nucleotide deposition sequence $S = S_1 S_2 \dots S_T$ corresponding to the sequence of nucleotides added at each masking step is therefore a supersequence of all $p \in P$[10].

In general, a probe can be embedded within $S$ in several ways. The embedding step of $p_k$ can be described as $T$-tuple $\epsilon_k = (e_{k,1}, e_{k,2}, \dots e_{k,T})$ in which $e_{k,t} = 1$ if probe $p_k$ receives nucleotide $S_t$ (at step t), or 0 otherwise. The deposition sequence is often denoted repeated permutation of the alphabet, mainly because

D.-S. Huang et al. (Eds.): ICIC 2008, LNAI 5227, pp. 1039–1046, 2008.

of its regular structure and because such sequences maximize the number of distinct subsequences. We distinguish between *synchronous* and *asynchronous* embeddings. In the first case, each probe has exactly one nucleotide synthesized in every cycle of the deposition sequence; hence, 25 cycles or 100 steps are needed to synthesize probes of length 25. In the case of asynchronous embeddings, probes can have any number of nucleotides synthesized in any given cycle, allowing shorter deposition sequences. All chips manufactured by this producer of can be asynchronously synthesized in 74 steps (18.5 cycles), which is probably due to careful probe selection. The problem of finding the sequence that reduces the number of steps required to accomplish the microarray production process is called SCS (Short Common Supersequence). The SCS problem is well-known to be NP-complete. In this paper, we present a novel approach to the problem of finding the SCS based on a multi-objective genetic algorithm that tries to minimize both the number of steps and the number of mask change in order to minimize the costs related to microarray production.

## 2 The SCS Problem with Applications in Bioinformatics and Nanotechnology

### 2.1 Microarray Techonology

Several microarray technologies are available today, based on a variety of fabrication techniques including printing with fine-pointed pins onto glass slides, ink-jet printing, electrochemistry on microelectrode arrays and photolithography. This paper is mainly concerned with the production of high-density oligonucleotide microarray, also called DNA chips or gene chips, that are fabricated by photolithography. This type of microarray consists of relatively short DNA probes synthesized at specific locations, named features or spots, on a solid surface. Each probe is a single-stranded DNA molecule of 10 to 70 nucleotides that perfectly matches with a specific portion of a target molecule. sequence of nucleotides added in each step is called deposition sequence or synthesis schedule. The selection of which probes receive the nucleotide is achieved by photolithography [1][2]. Figure 1 illustrates this process: The quartz wafer of a GeneChip array is initially coated with a chemical compound topped with a light-sensitive protecting group that is removed when exposed to ultraviolet light, activating the compound for chemical coupling. A lithographic mask is used to direct light and remove the protecting groups of only those positions that should receive the nucleotide of a particular synthesis step. A solution containing adenine (A), thymine (T), cytosine (C) or guanine (G) is then flushed over the chip surface, but the chemical coupling occurs only in those positions that have been previously deprotected. Each coupled nucleotide also bears another protecting group so that the process can be repeated until all probes have been fully synthesized.

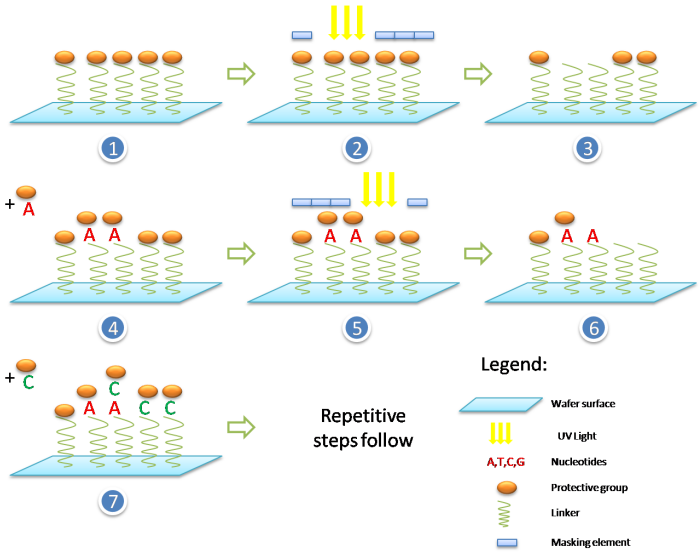


**Fig. 1.** Probe synthesis via photolithographic masks. The chip is coated with a chemical compound and a light-sensitive protecting group; masks are used to direct light and activate selected probes for chemical coupling; nucleotides are appended to deprotected probes; the process is repeated until all probes have been fully synthesized.

### 2.2 SCS Problem

The problem of finding the Shortest Common Supersequence (SCS) of a given set of sequences is a very important problem in computer science, especially in computational molecular biology. The SCS of a set of sequences can be stated as follows: Given two sequences $S = s_1 s_2 \ldots s_m$ and $T = t_1 t_2 \ldots t_n$, over an alphabet set $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$, we say that $S$ is the subsequence of $T$ (and equivalently, $T$ is the *supersequence* of $S$) if for every $s_j$, there is $s_j = t_{i_j}$ for some $1 \leq i_1 < i_2 < \ldots < i_m \leq n$. Given a finite set of sequences $S = \{S_1, S_2, \ldots, S_k\}$, a common supersequence of $S$ is a sequence $T$ such that $T$ is a supersequence of every sequence $S_j (1 \leq j \leq k)$ in $S$. Then, a shortest common supersequence (SCS) of $S$ is a supersequence of $S$ that has minimum length. In this paper, we shall assume that k is the number of sequences in $S$, $n$ is the length of each sequence, and $q = |\Sigma|$ is the size of the alphabet. The SCS problem has applications in many diverse areas, including data compression [3], scheduling [4], query optimization [5], text comparison and analysis, and biological sequence comparisons and analysis [6][7]. As a result, the SCS problem has been very intensively investigated [8][9]. One basic result is that the SCS of two sequences of length n can be computed using dynamic programming in $O(n^2)$ time and $O(n^2)$ space (see, for example, [10]). There are also several papers that reported improvements on the running time and space required for dynamic programming algorithms [9]. For a fixed $k$, the dynamic programming algorithm can be extended to solve the SCS problem for $k$ sequences of length $n$ in $O(n^k)$ time

and space. Clearly, this algorithm is not practical for large $k$. The general SCS problem on arbitrary $k$ sequences of length $n$ is well-known to be NP-hard. In fact, Jiang and Li [10] showed that even the problem of finding a constant ratio approximation solution is also NP-hard.

**Previous Research in SCS problem.** We now present a brief survey of the most popular heuristic algorithms proposed in literature. Let $S$ be any instance of the SCS problem and let $CS_A(S)$ be the supersequence of $S$ identified by a heuristic algorithm $A$. Let $opt(S)$ denote an optimal solution for the instance $S$. Then, we say that A has an approximation ratio of $\lambda$ if $|CSA(S)|/|opt(S)| \leq \lambda$ for all instances $S$.

*Alphabet Algorithm*
The Alphabet algorithm is a quite simple approach to the problem undest investigation[8]. Let $S$ be a set of sequences of maximum length $n$ over the alphabet $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_q\}$, then the Alphabet algorithm outputs a common supersequence of $(\sigma_1, \sigma_2, \ldots, \sigma_q)^n$. The Alphabet algorithm has an approximation ratio of $q = |\Sigma|$. The time complexity of the Alphabet algorithm is $O(qn)$. There have also been modifications of the Alphabet algorithm that uses information from $S$ to 'remove' redundant characters in $(\sigma_1, \sigma_2, \ldots, \sigma_q)^n$. These methods improve the performance in practice, but not in the worst case approximation ratio of $q$.

*Majority Merge Algorithm*
The Majority-Merge algorithm [10](MM) is a simple, greedy heuristic algorithm. Let's suppose we analyze every sequence from left to right, the frontier is defined as the rightmost characters to be analyzed. Initially, the supersequence $CS$ is empty. At each step, let $s$ be the majority among the 'frontier' characters of the remaining portions of the sequences in $S$. Set $CS = CS||s$ (where $||$ represent concatenation) and delete the 'frontier' $s$ characters from sequences in $S$. Repeat until no sequences are left. This algorithm is the same as the Sum Height algorithm (SH) proposed in [12]. This algorithm does not have any worstcase approximation ratio, but performs very well in practice. The time complexity of the Majority-Merge algorithm is $O(qkn)$.

*Greedy and Tournament algorithms*
The Greedy algorithm (GRDY) and Tournament algorithm (TOUR) studied in [13] are two variations of an iterative scheme based on combining optimal sequence pairs. Given any pair of sequences, $S_i$ and $S_j$, an optimal supersequence of the pair, denoted by $SCS(S_i, S_j)$, can be computed in $O(n^2)$ using dynamic programming. The Greedy algorithm first chooses the 'best' sequence pair the that gives the shortest $SCS(S_i, S_j)$. Without loss of generality, we assume that these two sequences are $S_1$ and $S_2$. The algorithm then replaces the two sequences $S_1$ and $S_2$ by their supersequence, $SCS(S_1, S_2)$. The algorithm proceeds recursively. Thus, we can express it as follows:

$\text{Greedy}(S_1, S_2, ..., S_k) = \text{Greedy}(SCS(S_1, S_2), S_3, \ldots, S_k)$

The Tournament algorithm is similar to the Greedy algorithm. It builds a 'tournament' based on finding multiple best pairs at each round and can be expressed schematically as follows:

$\text{Tournament}(S_1, S_2, \ldots, S_k) = \text{Tournament}(SCS(S_1, S_2), SCS(S_3, S_4), \ldots, SCS(S_{k-1}, S_k))$.

Both Greedy and Tournament algorithms have $O(k^2n^2)$ time complexity and $O(kn + n^2)$ space complexity. Unfortunately, it was shown in [11] that both Greedy and Tournament do not have approximation ratios.

## 3 Multi-objective Genetic Algorithms in Microarray Production Process Optimization

Multi-Objective Genetic Algorithms (MOGAs) are a relatively recent extension of Genetic Algorithms (GAs) that are well established bio-inspired computational optimization approaches with a wide range of applications that spans from finance to medicine. The concept of GA was developed by Holland and his colleagues in the 1960s and 1970s [14]. GA are inspired by the evolutionist theory explaining the origin of species. In nature, weak and unfit species within their environment are faced with extinction by natural selection. The strong ones have greater opportunity to pass their genes to future generations via reproduction. If these changes provide additional advantages in the challenge for survival, new species evolve from the old ones. Unsuccessful changes are eliminated by natural selection. In GA terminology, a solution vector $x \in X$ is called an *individual* or a *chromosome*. Chromosomes are made of discrete units called *genes*. Each *gene* controls one or more features of the chromosome. In the original implementation of GA by Holland, genes are assumed to be binary digits. In later implementations, more varied gene types have been introduced. Normally, a chromosome corresponds to a unique solution $x$ in the solution space. This requires a mapping mechanism between the solution space and the chromosomes.

Being a population-based approach, GA are well suited to solve multi-objective optimization problems. A generic single-objective GA can be modified to find a set of multiple non-dominated solutions in a single run. The ability of GA to simultaneously search different regions of a solution space makes it possible to find a different set of solutions for difficult problems with non-convex, discontinuous, and multi-modal solutions spaces. Most multi-objective GA do not require the user to prioritize, scale, or weigh objectives. Therefore, GA have been the most popular heuristic approach to multi-objective design and optimization problems. Jones et al. [15] reported that 90% of the approaches to multiobjective optimization aimed to approximate the true Pareto front for the underlying problem. A majority of these used a meta-heuristic technique, and 70% of all metaheuristics approaches were based on evolutionary approaches. From this perspective it

could be easily intended how MOGAs can be used in order to carry out an optimization that aims at pursuing a minimization in terms of the number of steps required for manufacturing a microarray and, contemporary, to minimize set up times costs associated to the change in base to base to be deposited on the surface of the so far assembled biochip. In the next paragraph we will show how this provblem has been addressed using a MOGA and we will expose computational results relating to a specific instance of this problem.

## 4 Novel Optimisation Procedure

In the proposed experimental design we evaluated the performance of the MOGA approach (Pareto front in Fig. 2, i.e. the times required. In order to make our evaluation as close to a real case as possible we used 4 oligonucleotide sequences taken from the most advanced chip for gene expression evaluation from the market leader. It is well known that the manufacturing process consists of depositing nucleotides in a step-by-step flavor, so as to reach the 25 oligonucleotide length for each of the features on the array. We selected the sequences:

8146645 gaagactcgcctgttgggacagcgc
8054479 gcatgtggctacttagtaaatagta
8154660 gcttagaaaacaggtcctcagcaca
8162631 ggtagcaaccgtcacaatctggatg

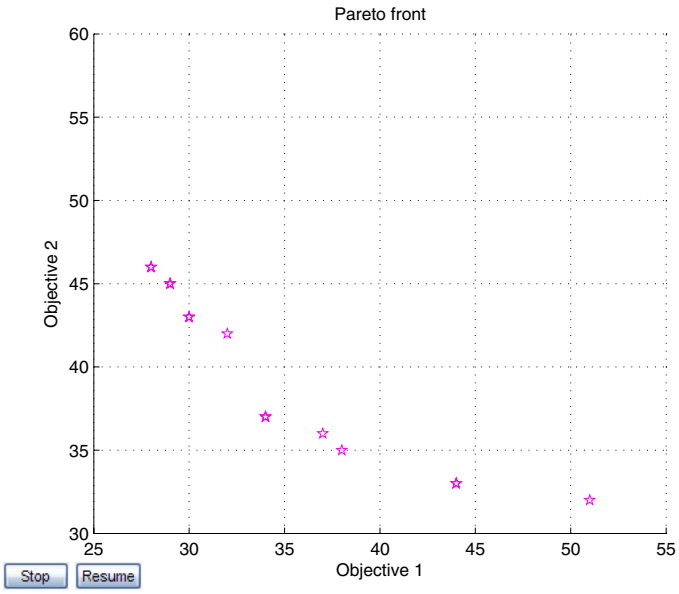


**Fig. 2.** Pareto front of the solutions found by the MOGA

**Table 1.** The deposition sequence and the corresponding embedding matrix

| | G | G | C | T | A | C | G | T | G | C | T | G | A | C | A | G | C | T | A | G | C | G | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 |
| ATCGAAGCGCGA | - | - | - | - | A | - | - | T | - | C | - | G | A | - | A | G | C | - | - | G | C | G | A |
| CCGTCCAGCTAA | - | - | C | - | C | G | T | - | C | - | - | - | - | C | A | G | C | T | A | - | A | - | - |
| GTACTTAGCTAC | G | - | - | T | A | C | - | T | - | - | T | - | A | - | G | C | T | A | - | C | - | - | - |
| GGATGGAGCTAC | G | G | - | - | A | - | - | T | G | - | - | G | A | - | - | G | C | T | A | - | C | - | - |
| | - | - | - | - | 1 | - | - | 1 | - | 1 | - | 1 | 1 | - | 1 | 1 | 1 | - | - | 1 | 1 | 1 | 1 |
| Embedding | - | - | 1 | - | 1 | 1 | 1 | - | 1 | - | - | - | - | 1 | 1 | 1 | 1 | 1 | 1 | - | 1 | - | - |
| Matrix | 1 | - | - | 1 | 1 | 1 | - | 1 | - | - | 1 | - | 1 | - | 1 | 1 | 1 | 1 | - | 1 | - | - | - |
| | 1 | 1 | - | - | 1 | - | - | 1 | 1 | - | - | 1 | 1 | - | - | 1 | 1 | 1 | 1 | - | 1 | - | - |

and we evaluated the performances of the algorithm using the following protocol: we started taking 12 bases from each of sequence and running the optimization algorithm 100 times on the same problem. We proceeded by adding each time one base to the previous sequence and re-evaluating the performances of the algorithm, until the end of the sequences has been reached. At each step we recorded the time required for each optimization task, the number of steps required by each solution and the reason why the optimization algorithm ended. For each of the 14 tests carried out we extracted the mean and the confidence intervals for both optimization times and optimization results; this was done in order to extract main estimators of the performances of the proposed algorithm under the hypothesis that the process under observation is ergodic (so that the mean estimation is independent on the specific realization and that it tends to the real value for $n$, number of observations, $n \to \infty$). Computational time analysis of the optimization task revealed that times required by the proposed algorithm can be adequately fitted with a relatively low order polynomial that seems to enforce the thesis that states that the computational complexity of this approach can be approximated to $O(x^6)$. This algorithm was able to complete the optimation task proposed in the previous section using only 23 deposition setps are reported in Tab. 1. As it can be observed no deposition step can be suppressed without affecting the whole process. This suggest that the necessary condition for optimality is at least satisfied. The results reported herein seem to confirm the robustness and versatility of the proposed algorithm and push the need for further research in this field.

## 5 Discussion

In this paper, we have proposed a novel Multi-Objective approach for reduction of manufacturing steps required for microarray assembly. The algorithm is built on a MOGA that firstly generates random templates, and the Evolution process to reduce templates from template pool to get shorter and less expensive result. These processes are shown to be powerful for solving the SCS problem. Comparing the performance of our approach with the industry gold standard we can state

that the proposed system is able to outperform alternative approaches under predefined conditions. The proposed solution results to be quite interesting in terms of result optimality; however it should be noticed that computational complexity of the algorithm under investigation is not negligible and it requires many optimization tasks before completion. However much research effort must be spent on border conflicts minimization in microarray production due to the specific technological limitations that characterize the photolitographic processes.

## References

1. Fodor, S., Read, J., Pirrung, M., Stryer, L., Lu, A., Solas, D.: Light-directed, Spatially Addressable Parallel Chemical Synthesis. Science 251, 767–773 (1991)
2. Hannenhalli, S., Hubell, E., Lipshutz, R., Pevzner, P.A.: Combinatorial Algorithms for Design of DNA Arrays. Advances in Biochemical Engineering Biotechnology 77, 1–9 (2002)
3. Storer, J.A.: Data Compression: Methods and Theory. Computer Science Press (1988)
4. Foulser, D.E., Li, M., Yang, Q.: Theory and Algorithms for Plan Merging. Artificial Intelligence 57(2), 143–181 (1992)
5. Sellis, T.K.: Multiple-query Optimization. ACM Transactions on Database Systems (TODS) 13(1), 23–52 (1988)
6. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT Press/McGraw-Hill (2001)
7. Sankoff, D., Kruskal, J.: Time Warps, String Edits and Macromolecules: the Theory and Practice of Sequence Comparisons. Addison Wesley, Reading (1983)
8. Barone, P., Bonizzoni, P., Vedova, G.D., Mauri, G.: An Approximation Algorithm for the Shortest Common Supersequence Problem: an Experimental Analysis. In: Symposium on Applied Computing, Proceedings of the 2001 ACM symposium on Applied computing, pp. 56–60 (2001)
9. Gusfield, D.: Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology. Cambridge University Press, New York (1997)
10. Jiang, T., Li, M.: On the Approximation of Shortest Common Supersequences and Longest Common Subsequences. SIAM Journal of Computing 24(5), 1122–1139 (1995)
11. Timkovsky, V.G.: On the Approximation of Shortest Common Non-subsequences and Supersequences. Technical report (1993)
12. Kasif, S., Weng, Z., Derti, A., Beigel, R., DeLisi, C.: A Computational Framework for Optimal Masking in the Synthesis of Oligonucleotide Microarrays. Nucleic Acids Research 30(20) (2002)
13. Irving, R.W., Fraser, C.: On the Worst-Case Behaviour of Some Approximation Algorithms for the Shortest Common Supersequence of k Strings. In: Proceedings of the 4th Annual Symposium on Combinatorial Pattern Matching, pp. 63–73 (1993)
14. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)
15. Jones, D.F., Mirrazavi, S.K., Tamiz, M.: Multiobjective Meta-heuristics: An Overview of the Current State-of-the-art. Eur. J. Oper. Res. 137(1), 1 (2002)