Fidelity Probes for DNA Arrays

Earl Hubbell[1,4] and **Pavel A. Pevzner**[1,2,3]

Departments of Mathematics[1], Computer Science[2], and Biology[3]
DRB-155 University of Southern California
Los Angeles, California 90089-1113

Affymetrix[4] 3380 Central Expressway
Santa Clara, California 95051
hubbell@hto.usc.edu ppevzner@hto.usc.edu

Abstract

One current approach to quality control in DNA array manufacturing is to synthesize a small set of test probes that detect variation in the manufacturing process. These fidelity probes consist of identical copies of the same probe, but they are deliberately manufactured using different steps of the manufacturing process. A known target is hybridized to these probes, and those hybridization results are indicative of the quality of the manufacturing process. It is not only desirable to detect variations, but also to analyze the variations that occur, indicating in what process step the manufacture changed. We describe a combinatorial approach which constructs a small set of fidelity probes that not only detect variations, but also point out the manufacturing step in which a variation has occurred. This algorithm is currently being used in mass-production of DNA arrays at Affymetrix.

Quality Control for DNA Arrays

DNA arrays are large collections of oligonucleotides that are produced with a small number of manufacturing steps. An error in a single manufacturing step will lead to synthesis errors in many oligonucleotides. How can one detect whether there is a manufacturing error and identify a faulty step? Since checking all oligonucleotides on the chip is impractical, quality control is an important issue for mass-production of such arrays. We describe an approach based on a small set of test probes (fidelity probes) that allow one to detect and identify errors in any manufacturing step.

The mass production technology uses photolithography to manufacture DNA arrays, by growing many oligonucleotides in parallel, adding one base simultaneously to a large collection of selected oligonucleotides (Fodor et al., 1991, Jacobs and Fodor 1994.) Regions of growth are selected by illuminating specific areas of the array, causing a chemical protectant to be removed in those areas. The array is then exposed to a solution consisting of a single type of nucleotide (itself protected), which binds only to the selected areas. This process may be repeated with different patterns of illumination and differing nucleotide solutions to build a diverse assortment of probes. For each construction step, a photolithographic mask controls the selective illumination of the array surface. Many different chemical processes may be used in array construction to optimize the quality of the probes on the array (Beecher et al., 1996, McGall et al.,1997.) The design of the photolithographic masks including arrangement of probes may also be optimized (Feldman and Pevzner, 1994, Pevzner and Lipshutz, 1994, Lipshutz et al., 1995, Frieze et al., 1999.) However, in a mass production environment, deviations from an ideal procedure may occur, and must be detected.

The photolithographic equipment is in continuous use and although it is extremely accurate, occasionally variations occur in the manufacturing process. Although actual errors are rare, it is often difficult to detect any errors even the most extreme, since a reference array and a defective array may appear very similar in overall behavior. However, even in the case only one chip in a thousand is defective, it may cause serious problems and result in misinterpretation of experimental and diagnostic data. From this perspective, using a defective DNA chip may cause damage comparable to damage caused by a defective Pentium chip. Since these errors may be difficult to detect, a quality control procedure is necessary to assure error-free production of the chips.

As an example of an error, the synthesis of many probes on an array may be affected by a missing or altered chemical synthesis step, but only a few of the (changed) probes may noticeably differ in their hybridization to a target under experimental conditions. In a common tiling array (Chee et al, 1996), the majority of the probes do not hybridize with a particular target, and hence any change in their hybridization behavior may go undetected. It is likely that several strongly hybridizing probes would also be affected by the change. However, if the alteration occurs towards the end of such probes, the hybridization signal will only be slightly reduced. Further, it is not obvious how to disentangle variations in the hybridization behavior caused by changes in the probe manufacturing process

from natural variations occurring within an experiment. For stringent quality control, it is desirable to detect variations in the normal manufacturing process before they stray from acceptable limits and cause actual errors. For this purpose as well, it is important to have a method of analyzing process variation independent of the actual experimental use of the array. It is important to detect that a manufacturing step has varied, and in addition, it is valuable to determine in which particular manufacturing step a variance took place. Detecting the specific step at which a variation occurs allows one to more easily diagnose the source of any manufacturing difficulty.

One method of performing this diagnosis is to compare a test set of probes of identical sequence that are produced using different manufacturing steps. If this set is sufficiently diverse, every manufacturing step error should have a unique signature of affected test probes. The computational challenge in fidelity probe generation is to find a small, yet diverse, collection of ways to synthesize an oligonucleotide sequence using different manufacturing steps. We have solved this problem by finding a small and diverse set of paths in an acyclic graph similar to the dynamic programming graph for sequence alignment. Random generation of such paths does not produce a small set meeting our requirements for diversity, and so we propose a deterministic approach that works well in practice. To the best of our knowledge, problems of this type have not been examined before in computational biology and the algorithm below outlines one solution as described in the resulting patent application.

Fidelity Probe Generation

The manufacturing process is described by a manufacturing protocol, $S = S_1 S_2 \ldots S_s$, which is a string over the alphabet, A,T,G,C, with S_i corresponding to the nucleotide added in the ith manufacturing step. Any subsequence of S may be synthesized at a given address using the photolithographic process. Usually, many different subsequences of the protocol correspond to a given sequence $p = p_1 p_2 \ldots p_t$. We call these different subsequences *p-subsequences* in S. For example, given the protocol $S = ACGTTACGT$, and the desired probe sequence $p = ACGT$, the six possible p-subsequences in S are { $S_1 S_2 S_3 S_4$, $S_1 S_2 S_3 S_5$, $S_1 S_2 S_3 S_9$, $S_1 S_2 S_8 S_9$, $S_1 S_7 S_8 S_9$, $S_6 S_7 S_8 S_9$ }. In each p-subsequence, the monomers A,C,G,T are added in that order to produce the probe ACGT, but different manufacturing steps are used.

Since all p-subsequences construct the same oligomer p, addresses corresponding to different p-subsequences should have similar hybridization behavior. If similar behavior is not seen, then the manufacturing protocol has varied in some step. Therefore, we may use an appropriately chosen set of p-subsequences to detect variation in the manufacturing protocol. Given a set of p-subsequences, FP, define the subset of p-subsequences that use the letter S_i to be P_i. This subset consists of those p-subsequences that are affected by variations in the step i. If these subsets P_i are each unique, a change in any single step i of the manufacturing protocol will have a unique signature of affected p-subsequences which allows one to find the erroneous step. In practice, we want to design p-subsequences to fill a reserved set of N addresses, so that every step in the protocol S is represented several times (required for good statistics.) Therefore we have a chosen parameter, *MinSize*, determining the minimum number of p-subsequences affected by each manufacturing step. To ensure that every step has a unique signature (set of addresses that respond when it varies), we require a parameter *MinDiff*, indicating the minimum number of differences between every pair of sets P_i and P_j to count them as 'unique'.

Fidelity Probe Generation Problem. Given a protocol S, a probe p, a number N, and constraints *MinSize*, *MinDiff*, find N different p-subsequences in S such that

- $| P_i | \geq MinSize$ for $1 \leq i \leq s$
- $| P_i \setminus P_j | \geq MinDiff$ for $1 \leq i \neq j \leq s$

While the fidelity probe generation problem could be formulated as a minimization problem in N, we prefer the formulation above since in practice the number N is a fixed constant. Note that this formulation of the problem assumes that every S_i can be present in some set of p-subsequences. This is false in general, and in this case, the algorithm reports that there is no answer.

At first glance, it seems that a simple approach that generates p-subsequences randomly would satisfy the constraints. Although for sufficiently large N such a naive strategy may succeed, for practical values of N, *MinSize*, and *MinDiff*, our computational experiments demonstrate that at least one of our constraints is violated under this approach. Therefore a more sophisticated approach is required.

Algorithm for Fidelity Probe Generation Problem

The problem with the naive "random choice" approach is that each additional p-subsequence generated only makes a small contribution towards satisfying our *MinSize* and *MinDiff* constraints, and thus leads to an excessive number of fidelity probes N. Thus, we wish to bias the selection procedure so that each additional p-subsequence makes a large contribution towards satisfying our constraints. A compact method of representing all possible p-subsequences is to construct a directed acyclic graph similar to the dynamic programming graph for sequence alignment. The dynamic programming matrix for p-subsequences in S is set up between the probe, p, and the manufacturing protocol, S (Fig.1). The corresponding dynamic programming graph contains all horizontal edges, and diagonal edges from $(i-1, j-1)$ to (i, j) if $p_i = S_j$. Every path from $(0, 0)$ to (t, s) in this directed acyclic graph corresponds to a possible p-subsequence. Since generation of p-subsequences by finding random paths in

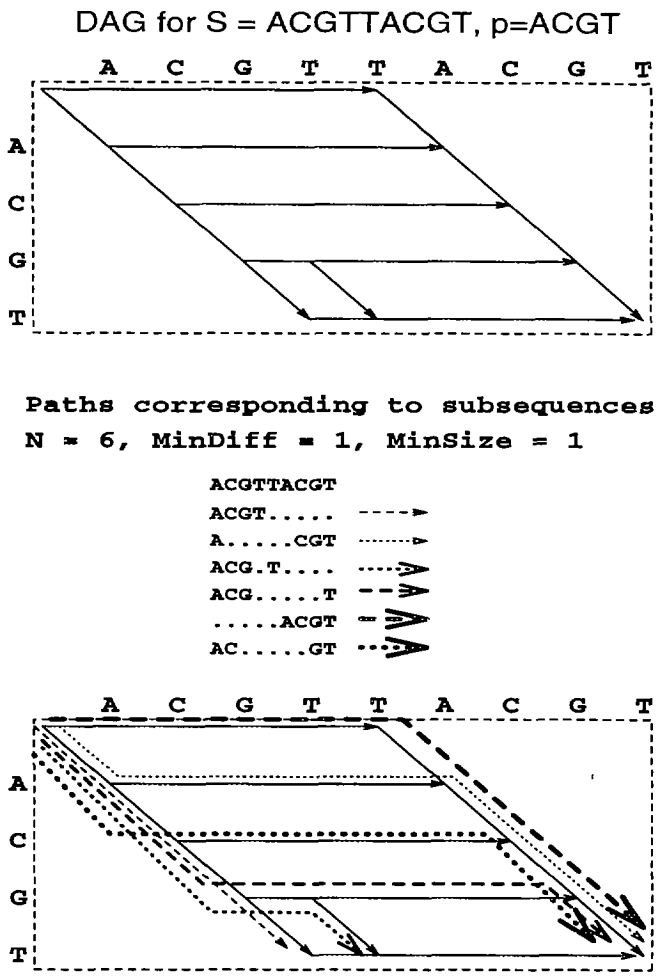

Figure 1: Diverse paths in DAG generate fidelity probes

this DAG does not produce a feasible solution, we suggested the idea of "reweighting" this graph. Given a set of p-subsequences that partially satisfy the constraints, we want to weight the edges of this graph so that the shortest such path produces a new p-subsequence that helps us satisfy the constraints. The newly generated p-subsequence is added to our set, and the DAG is reweighted. Each iteration of this procedure should add a p-subsequence that satisfies new aspects of our constraints. Thus, we must find a method for choosing weights that causes the shortest path to differ from previous p-subsequences.

Weighting the graph has some subtle aspects. A natural strategy to satisfy the *MinSize* constraint is to weight edges in the graph with the number of times each edge is used in p-subsequences. Repeated generation of p-subsequences corresponding to a shortest path in the DAG with the DAG reweighted every time a p-subsequence is generated should produce a diverse set of p-subsequences. However, this natural strategy does not produce a small set of p-subsequences that satisfy our conditions, because many different edges may correspond to the same synthesis step. It is necessary to explicitly include our conditions in the weighting strategy. For example, weighting the edges corresponding to manufacturing steps S_i that have been used more than *MinSize* times automatically causes the shortest path to pass through a maximal number of unweighted edges, and hence constructs a p-subsequence that assists us towards satisfying our *MinSize* constraint. An improved strategy is to weight the edges based on the number of times the corresponding manufacturing step S_i is used. Using this improved strategy, repeated generation of p-subsequences will gradually satisfy the *MinSize* condition.

Unfortunately, our *MinDiff* condition may not be satisfied by this procedure. In order to make progress towards satisfying our *MinDiff* condition, two manufacturing steps S_i and S_j are chosen for which the condition is not satisfied. The subroutine of the algorithm *SplitSteps*, reweights two such manufacturing steps to encourage one and discourage the other. It is important that the revised weights are not reweighted to have the same sum as before, because then the shortest path could use both edges and pay no penalty. A further difficulty is that often there are correlations between steps so that using one step S_i in a p-subsequence forces the use of another step S_j. To avoid this difficulty, if the previously generated set of p-subsequences has an even number of members, S_i is encouraged, and if it has an odd number of members, S_j is encouraged. In this manner, even if S_j is discouraged, but S_i forces the use of S_j so that progress is not made towards the *MinDiff* condition, the next iteration of the algorithm will encourage the use of S_j and discourage S_i and so make progress towards satisfying the *MinDiff* constraint. If the *MinDiff* condition is satisfied for all pairs of steps, then *SplitSteps* chooses a step that has been used in the fewest p-subsequences and encourages that step, while discouraging a step that has been used in the most p-subsequences. In the code, the manufacturing step that is encouraged has the weights of the corresponding edges reduced by $2 \times LARGE$ and the discouraged manufacturing step has the corresponding weights increased by $LARGE$, where $LARGE$ is a large constant.

Surprisingly, under several reweighting strategies, repeated generation of shortest paths often produces paths duplicating a previous path. Since we are looking for a set of distinct p-subsequences, a simple reweighting scheme *TabuSearch* is used to guarantee the uniqueness of each chosen p-subsequence. In this reweighting strategy, those manufacturing steps used by the duplicate p-subsequence are increased in weight, and a replacement p-subsequence is generated. This procedure is iterated until a distinct p-subsequence is created, or $LOOPMAX$ iterations have occurred. If $LOOPMAX$ iterations occur, the subroutine returns a duplicate p-subsequence, and hence the set of p-subsequences produced will not have distinct members and the algorithm will return no solution found.

Specifically, a set of fidelity probes meeting our conditions are constructed by *FidelityProbe*, the following algorithm. At every iteration, we start with the set FP consisting of the p-subsequences already constructed, and produce the p-subsequence PR. P_i is the subset of FP which uses the manufacturing step S_i. The directed acyclic graph DAG is constructed as described above. The set of edges E_j corresponding to the manufacturing step S_j are those edges $(i-1, j-1)$ to (i, j) in the jth column of the DAG where $p_i = S_j$. Because each edge in this set E_j corresponds to the same manufacturing step, each such edge has the same weight assigned to it as all other edges in E_j. These weights are tracked by the variable $Weight[j]$, corresponding to the identical weight assigned to each edge within E_j.

```
FidelityProbe(p,S,N,MinDiff,MinSize)
    FP ← ∅
    for k ← 1 to s
        P_k ← ∅
    generate DAG corresponding to p and S
    while | FP |< N
        for k ← 1 to s
            Weight[k] ←| P_k |
        SplitSteps(DAG,FP)
        FP ← FP∪ { TabuSearch(DAG,FP) }
    if FP satisfies fidelity probe requirements
        output FP
    else
        output " no solution found "

SplitSteps(DAG,FP)
    if exists i ≠ j such that| P_i \ P_j |< MinDiff
        if | FP | is even
            Weight[i] ← Weight[i] + LARGE
            Weight[j] ← Weight[j] − 2 * LARGE
        else
            Weight[j] ← Weight[j] + LARGE
            Weight[i] ← Weight[i] − 2 * LARGE
    else
        Find i ≠ j such that | P_i | is minimal
            and | P_j | is maximal
        Weight[i] ← Weight[i] − 2 * LARGE
        Weight[j] ← Weight[j] + LARGE
return

TabuSearch(DAG, FP)
    c ← 0
    repeat
        PR ← ShortestPath(DAG)
        if PR ∈ FP
            for k ← 1 to s
            if S_k ∈ PR
                Weight[k] ← Weight[k] + 1
        c ← c + 1
    until PR ∉ FP or c ≥ LOOPMAX
    for k ← 1 to s
        if PR uses step k
            P_k ← P_k ∪ {PR}
return PR
```

Conclusion

This algorithm has several helpful properties. First, it is deterministic, so that it produces the same set of fidelity probes for identical manufacturing protocols. Second, the *TabuSearch* routine may be easily modified to avoid any pre-specified set of p-subsequences. The first property is valuable because it allows fidelity probes to be consistent between different DNA arrays, and the second is valuable because it allows several different sets of fidelity probes to be generated and compared.

In practice, a DNA array is designed to reflect a biological problem and a manufacturing protocol S is constructed for the synthesis of that array. A preselected 20-mer quality control target, and the four possible 17-mer probes that hybridize perfectly to that target are used in the fidelity algorithm. Obviously, if a given manufacturing protocol begins with the addition of a monomer that is not the same as the initial monomer of the desired probe, there are some cycles in the manufacturing protocol that cannot be present in any p-subsequence. With four different probes, each one starting with a different one of the four possible monomers and ending with a different one of the four possible monomers, a greater variety of manufacturing protocols may be handled. For each of these four probes, $N = 32$ p-subsequences are generated, utilizing all cycles in S that can be covered by that probe. The final set of 128 p-subsequences are placed into the reserved addresses in the array.

This algorithm satisfies the technological requirements. A typical manufacturing protocol S is 60-100 steps, usually ACGT ... ACGT, or a short edit distance from such a string. The algorithm produces a set of fidelity probes satisfying our requirements from a typical 20-mer quality control target. A number of theoretical questions remain. For example, given a typical set of manufacturing protocols, what is a target that can be used for fidelity testing on any of the protocols? Is there a simple algorithm that solves the problem over a wider range of protocols and probes than this algorithm does?

Acknowledgement

The authors are grateful to David Smith for discussions on the practical aspects of array quality control, and to Sridhar Hannenhalli for discussions on the early versions of the algorithm for fidelity probe generation.

References

Beecher, Jody E., McGall, Glenn H., Goldberg, Martin J. Chemically Amplified Photolithography for the Fabrication of High Density Oligonucleotide Arrays. *Polym. Mater. Sci. Eng.* 76:597-598,1997.

Chee, M., Yang, R., Hubbell, E., Berno, A., Huang, X.C., Stern, D., Winkler, J., Lockhart, D.J., Morris, M.S. and Fodor, S.P.A. Accessing genetic information with high-density DNA arrays. *Science* , 274:610-614, 1996.

Feldman, W., Pevzner, P.A. Gray code masks for Sequencing by Hybridization. *Genomics*, 23:233-235, 1994

Fodor, S.P.A., Read, J.L., Pirrung, M.C., Stryer, L., Tsai Lu, A. and Solas, D., Light-directed, spatially addressable parallel chemical synthesis. *Science*, 251:767-773, 1991.

Frieze, A.M., Preparata, F. P., Upfal, E., On the Power of Universal Bases in Sequencing by Hybridization. *Third Annual International Conference on Computational Molecular Biology*, Lyon, France, 1999.

Jacobs, J., Fodor, S. Combinatorial Chemistry - Applications of Light-Directed Chemical Synthesis. *Trends In Biotechnology* 12:19-26,1994.

Pevzner, P.A. and Lipshutz, R.J. Towards DNA sequencing chips. *19th Symp Math on Foundations in Computer Science, Springer-Verlag Lecture Notes in Mathematics*, 841, 143-158. 1994.

Lipshutz, R.J., Morris, M.S., Chee, M., Hubbell, E., Kozal, M.J., Shah, N., Shen, N., Yang, R. Fodor, S.A. Using Oligonucleotide Probe Arrays to Access Genetic Diversity. *BioTechniques*. 19:442-447, 1995.

McGall, G., Labadie, J. Brock, P., Wallraff, G., Nguyen, T., Hinsberg, W. Light-Directed Synthesis of High-Density Oligonucleotide Arrays Using Semiconductor Photoresists. *Proc. Natl. Acad. Sci. U.S.A.* 93:13555-13560,1996.