Efficient Computational Design of Tiling Arrays Using a Shortest Path Approach

Alexander Schliep[1] and Roland Krause[1,2]

[1] Max Planck Institute for Molecular Genetics, 14195 Berlin, Germany
schliep@molgen.mpg.de
[2] Max Planck Institute for Infection Biology, 10117 Berlin, Germany

Abstract. Genomic tiling arrays are a type of DNA microarrays which can investigate the complete genome of arbitrary species for which the sequence is known. The design or selection of suitable oligonucleotide probes for such arrays is however computationally difficult if features such as oligonucleotide quality and repetitive regions are to be considered.

We formulate the minimal cost tiling path problem for the selection of oligonucleotides from a set of candidates, which is equivalent to a shortest path problem. An efficient implementation of Dijkstra's shortest path algorithm allows us to compute globally optimal tiling paths from millions of candidate oligonucleotides on a standard desktop computer. The solution to this multi-criterion optimization is spatially adaptive to the problem instance. Our formulation incorporates experimental constraints with respect to specific regions of interest and tradeoffs between hybridization parameters, probe quality and tiling density easily. Solutions for the basic formulation can be obtained more efficiently from Monge theory.

1 Introduction

Tiling arrays. DNA microarrays can be manufactured today with the high density required to comprehensively sample the complete genomic sequence of an organism. These chips are typically referred to as genomic or tiling arrays and are widely used in transcriptome analysis. Representing a genome completely opens new routes, such as transcription profiles that do not rely on prior gene prediction, discovery of microRNAs or chromatin-immunoprecipitation-chip studies for the detection of transcriptional regulation.

Depending on the type of question, the probe selection for the array needs to be customized, which is economically feasible as several providers offer customer designed arrays. The determination of suitable probe sequences, typically contiguous subsequences of the genomic sequence, however is computationally demanding even for relatively small genome sizes of bacteria if the quality of probes is considered in the design.

R. Giancarlo and S. Hannenhalli (Eds.): WABI 2007, LNBI 4645, pp. 383–394, 2007.

Oligonucleotide quality. There are many aspects determining the quality of oligonucleotides probes, which arise from the specifics of the hybridization reaction between an oligonucleotide probe immobilized on the chip surface and the region of genomic DNA it reacts with. Prominent aspects are the melting temperature or binding energy of the duplex formed and the cross-hybridization potential. The latter reflects the availability of forming stable duplexes with genomic regions other than the intended and is approximated heuristically through sequence similarity. Kane *et al* [1] measured the amount of cross-hybridization and summarized the results into two criteria for unique probe selection for oligonucleotides probes of length 50, or 50mers. The first criterion is satisfied if the probe has no match of global sequence identity exceeding 75% in another region in the target genome. The second criterion requires that there is no exact match of 15 bases or more within unintended matches of a sequence identity between 50% to 75%. Later efforts tried to extend Kane's work to larger sets of genes, oligonucleotide lengths and sequence composition [2,3]. Other aspects are under discussion, for instance hairpin structures or the occurrence of guanine or cytosin on the 5' or 3' end of the oligonucleotide. Recently, a disillusioning study showed that designing for optimal oligonucleotide quality is ill-understood and that thermodynamic considerations do not necessarily improve results [4]. The major body of work to assess hybridization was done on Affymetrix microarrays, which typically employ 25mers and particular experimental arrangements to assess the quality. It is unclear how to make best use of this work in the context of other array designs with oligonucleotides of different lengths and different hybridization conditions.

Prior work. One possible approach to the design is the selection of evenly spaced oligonucleotide probes which provide dense coverage of a genome. Selinger *et al* [5] constructed a dense microarray for the 4MBb genome of *Escherichia coli* with a density of one 25mer for each 30 base window. The design of such an array is straight-forward but suffers from the problem that cross-hybridization between probes and several genomic regions is to be expected. While the analysis of such an array can yield important results as to whether particular genomic regions are expressed, it is compromised due to the high error rates of unspecific probes and, furthermore, if relative expression levels are to be quantified and compared. The question of finding specific probes is further complicated by the vast amount of repetitive sequence in higher organism, which is often handled by repeat detection and subsequent masking.

The large proportion of repetitive sequence in the human genome requires enhanced design considerations to achieve high coverage. One approach [6] concentrates on maximizing the size of segments which can be covered with evenly spaced oligonucleotide probes by post-processing genomic sequences with masked repeats by joining segments of non-repetitive DNA if they are separated by short segments of repetitive DNA as long as the joined segment is sufficiently longer. As probe quality is ignored in the process, probes with a large cross-hybridization potential will often be chosen for the array.

A two-stage approach was proposed by Lipson *et al* [7]. Following computation of candidate probes they propose to choose a whenever possible (WP) ϵ-cover. A WP ϵ-cover is a subset of candidates so that for any chromosomal position x the following holds. Either there are probes i, j, $i \leq x \leq j$ (probes are identified with their chromosomal location) in the cover with $j - i \leq \epsilon$ or there is no candidate between i and j. They provide a greedy algorithm for computation of WP ϵ-covers and minimizing ϵ for a given array size with a log-normal complexity of their entire approach. For a problem instance with candidates at positions $1, 2, \ldots, \epsilon, \epsilon + 1, 2\epsilon + 1, 2\epsilon + 2, \ldots, 3\epsilon + 1, 3\epsilon + 2, 4\epsilon + 2$ the greedy algorithm will arrive at the WP ϵ-cover $\epsilon, \epsilon + 1, 2\epsilon + 1, 3\epsilon + 1, 3\epsilon + 2, 4\epsilon + 2, \ldots$. This is clearly undesirable as one third of the probes are essentially uninformative and for example $\epsilon + 1, 2\epsilon + 1, 3\epsilon + 2, 4\epsilon + 2$ uses fewer probes to cover the same segment with the same resolution. Furthermore, their approach cannot optimize with respect to probe quality.

Computation of candidates. There are also two stages in our approach. First, candidate oligonucleotides are computed for a given genome with one of the several tools available. For this work, we compute candidates with the recently introduced tool `Flog` [8]. It employs a suffix array [9] for the selection of unique probes that satisfy particular experimental conditions such as GC content and melting temperature. It implements a set of filters to satisfy Kane's criteria [1] (see above). We used the margin with which the second criterion was fulfilled as the quality values for probes.

For specific applications it is desirable to use oligos of different lengths to satisfy experimental constraints, in particular the melting temperature T_m. The very tool used in the process is not essential and other tools that satisfy experimental requirements or considerations such as hair-pin formation tendencies can be easily employed. The use of `Flog` removes the need to scan for repetitive regions. If particular repetitive regions are to be covered explicitly, our algorithm can incorporate additional selected oligonucleotides. Note that candidate computation has to be done only once per genome.

Novel contributions. We formulate the multi-criterion optimization problem of selecting an optimal subset of oligonucleotide probes from set of candidates as the *minimal cost tiling path problem*. We show that our formulation is equivalent to a shortest path problem, which can be solved in log-linear time to global optimality using Dijkstra's shortest path algorithm. The optimal solutions adapt to spatial differences within the problem instance and can be controlled globally with respect to tradeoffs between hybridization parameters, probe quality and tiling density. Additionally, the solutions can be locally constrained for example to reflect regions of particular interest to the biologist. We also provide an efficient implementation of Dijkstra's shortest path algorithm. The resulting computational efficiency and memory footprint makes it feasible to solve instances with millions of candidate oligonucleotides on standard desktop computers. Furthermore, optimal solution for the case of homogeneous (position-independent) tradeoffs and design parameters can be obtained from Monge theory in linear time.

2 Minimal Cost Tiling Paths

We formalize the problem of designing a tiling array in the following. The primary input is a collection of n candidate oligonucleotide probes and relevant information about them. Let $i \in \{1, \ldots, n\}$ denote the probes covering a DNA sequence S and $p(i), T_m(i)$ and $q(i)$ the position, the melting temperature and the probe quality of probe i respectively. The melting temperatures $T_m(i)$ for probes selected should be within a narrow range to optimize comparability of probes and thus consistency of the experiments. There are various measures for probe quality mainly reflecting the potential for cross-hybridization. Higher quality probes with a lower cross-hybridization potential not only have a lower false positive rate but also a lower variance of probe intensities, so high quality probes should be used. Here we assume that the $q(i)$ are computed while creating the list of candidates; the details do not matter. To simplify the analysis, which typically relies on homogeneous proximity effects in signal correlation between adjacent probes, it is also desirable to keep the distance $p(j) - p(i)$ between any two adjacent probes close to constant. Further relevant parameters exist and our formulation can be trivially extended to accommodate them.

The question remains how to choose a tuple $T \subset \{1, \ldots, n\}$ of probes, which we will refer to as the *tiling path* such that $p(j) - p(i) = d^\star$ for any two adjacent probes, $T_m(i) = T_m^\star$ and $q(i)$ maximal for all $i \in T$. Here $d^\star$ is the desired tiling distance and $T_m^\star$ the desired melting temperature. The multi-criterion optimization has conflicting criteria. Fulfilling the distance criterion is trivial when choosing arbitrarily bad probes with incorrect melting temperature and vice versa. Which tradeoffs are made has to be decided individually depending on the organism selected, possible density maxima given by the size of the array and sensitivity differences in the melting temperature, which varies in time and stringency between experimental conditions.

To simplify, we first cancel units in probe parameters and bring the different criteria on a common scale. Let

$$d(i,j) := \frac{|d^\star - (p(j) - p(i))|}{d^\star} \tag{1}$$

denote the penalty for choosing adjacent probes i and j in T. Similarly, let

$$pt(j) := \frac{|T_m^\star - T_m(j)|}{T_m^\star} \tag{2}$$

denote the penalty contribution from T_m violations and finally

$$pq(j) := \begin{cases} \frac{q^\star - q(j)}{q^\star} & \text{if } q(j) < q^\star, \text{ and} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

denote the penalty for selecting probes of quality below $q^\star$. Note that this formulation does not distinguish between probe quality exceeding $q^\star$ as this reflects the practice of microarray design.

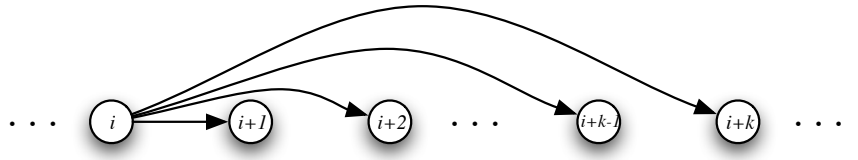

Fig. 1. We show the neighborhood structure for node i. It is adjacent to nodes $i+1, i+2, \ldots, i+k$. Other edges are not shown.

We can compute the cost of a tiling path T now as

$$C(T) = \sum_{i=0}^{|T|} d(T_i, T_{i+1}) + \sum_{i=1}^{|T|} pt(i) + pq(i) \tag{4}$$

or consider a weighted version to allow global control of the tradeoffs

$$C_w(T) = w_d \times \sum_{i=0}^{|T|} d(T_i, T_{i+1}) + \sum_{i=1}^{|T|} w_t \times pt(i) + w_q \times pq(i). \tag{5}$$

Note that $d(0,i)$ and $d(i,|T|+1)$ only penalize probe choices which are too far away from either sequence end; that is $d(0,i) = \frac{p(i)-d^\star}{d^\star}$ for $p(i) > d^\star$ and 0 else; $d(i,|T|+1)$ is analogously defined. This leads us to the definition of the *minimal cost tiling path* problem.

Problem 1. Find a tiling path T of minimal cost $C_w(T)$ given candidate probes $\{1,\ldots,n\}$, probe parameters $p(i), T_m(i)$ and $q(i)$ and design parameters $d^\star, T_m^\star$ and $q^\star$ with criteria weights w_d, w_t and w_q.

2.1 Shortest Path Solution

The problem can be reformulated as a shortest path problem. Consider the graph in Figure 1. The set of vertices are the probes $\{1,\ldots,n\}$ with special nodes 0 and $n+1$, which are "virtual probes" before the start and after the end of the sequence we want to tile. For an edge (i,j), with $1 \le i,j \le n$ we compute its weight $w(i,j)$ as the terms j contributes to the sum in Eq. 5. That is, $w(i,j) = w_d \times d(i,j) + w_t \times pt(j) + w_q \times pq(j)$. The weights $w(0,i)$ and $w(i,n+1)$ are defined as $d(0,i)$ above. Clearly, the cost of a path $0, i_1, i_2, \ldots, n+1$ is exactly as defined in Eq. 5 and its vertices define the tiling path. Dikstra's shortest path algorithm [10] with a Fibonacci heap based priority queue [11] can compute a shortest path in $O(|E| + |V| \log |V|)$, where $|V|$ and $|E|$ are order and size of the input.

To improve the running time we bound the cardinality of the neighborhoods by k. Theoretically, all possible edges $(i,j), 0 \le i < j \le n+1$ have to be considered. However, a simple analysis of behavior of the cost function reveals that the distance penalty dominates the other penalties for typical choices of

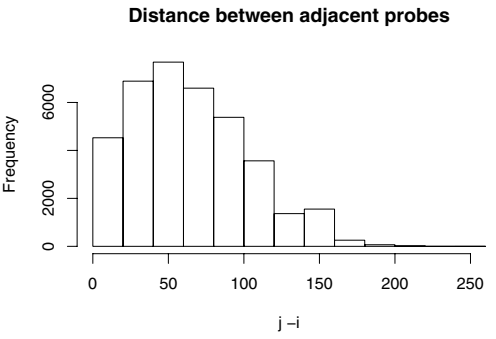

Fig. 2. The graph-theoretic distance of vertices i, j, which are adjacent on the shortest path in the input graph. This experiment with about 40,000 probes selected from 2,000,000 candidates in *Mycobacterium smegmatis* exemplifies the existence of a bound for $j - i$ on real data. Computations were performed with $k = 1000$.

tradeoff weights and instances. As $p(j) - p(i) \geq j - i$, we can see that k depends on $d^\star$ rather than n. Hence larger arrays, with lower $d^\star$ for the same genome length $|S|$, are actually faster to compute. Note, from k independent of n it follows that we can solve the minimal cost tiling path problem with a worst-case complexity of $O(n \times \log(n))$, as in the corresponding graph $|E| < k \times |V|$, which is dominated by $|V| \log |V|$.

We present the derivation for the unweighted cost function $C(T)$; similar results for $C_w(T)$ are straight-forward to obtain.

Proposition 1. *Let T be a minimal unweighted cost tiling path, and let i and j be adjacent probes in T. Then $j-i < 2d^\star+1$, provided* $\max_l pt(l)+\max_l pq(l) < 1$.

Proof. Assume that $j - i \geq 2d^\star + 1$. We show that for $i' = \lfloor \frac{j-i}{2} \rfloor$ the cost of T is lowered when we replace (i, j) by $(i, i'), (i', j)$. That is, T is not minimal. Using Eq. 4 we can rewrite $w(i, j) > w(i, i') + w(i', j)$—note i, i' and j are chosen such that with $p(j) - p(i) > j - i$ the position specific terms are positive—as

$$-d^\star+(p(j) - p(i)) > -d^\star+(p(i') - p(i))+d^\star(pq(i')+pt(i'))-d^\star+(p(j) - p(i')) .$$

Canceling further terms we arrive at

$$0 > -d^\star + d^\star(pq(i') + pt(i'))$$

which finishes the proof.

Note that typically $\max_l pt(l) < 0.3$; the quality penalties are equally bounded. Based on experimental results, cf. Figure 2, and manual inspection of gap size, melting temperature and probe quality distributions we choose $k = 400$ for $d^\star = 160$. In the instances for which we computed tiling paths we never observed two selected probes i and j with $j - i > 200$. Our choice of k clearly leaves room for further improvement, in particular as a k too small could lead to a shortest path which is not a minimal cost tiling path.

2.2 Implementation

As problem instances are quite large—from order 2,000,000 and size 800,000,00 up to order 30,000,000 and size 1,200,000,000 for a small bacterium like *M. tubercolosis* and *Human* chromosome 2 respectively—computation of the shortest path is a veritable challenge. Even state-of-the-art libraries like Boost(`http://www.boost.org`) or LEDA [12] cannot effectively cope with graphs of this size. For $n = 3.200.000$ and $k = 300$ a Boost-based implementation needs over 50GB of memory and over 20 minutes of CPU time for *allocating* the graph. This does not include time for computing edge weights or the shortest path.

As Dikstra's shortest path algorithm explores the neighborhood of minimal vertices removed from the priority queue, we simply compute the neighborhood when needed instead of precomputing all neighbors and storing them as a graph. This quite obvious optimization of shortest path computations was implemented for example in [13]. Our method is implemented in Python (`http://www.python.org` using the numpy (`http://numpy.scipy.org/`) package for linear algebra and a priority queue implementation from `http://py.vaults.ca/apyllo.py/514463245.769244789.44776582.`

Figure 3 shows the growth of running time and memory usage for increasing problem sizes. The memory usage is minimized as we can use large arrays to store probe position, quality, and hybridization parameters, which can be allocated in constant time (neglecting the priority queue). Neighborhood computations are performed by vector-valued operations using numpy, where the vectors are *slices*. A slice is a vector consisting of a consecutive number of elements of another vector, for example the positions of candidate oligonucleotides $i, i+1, \ldots, i+k$ is a slice in the vector of all candidate positions. Numpy utilizes either vendor supplied basic linear algebra system (BLAS) libraries or self-optimizing BLAS such as Atlas [14] which are tuned to specific hardware.

2.3 Local Constraints in the Formulation

There is considerable interest in custom tiling arrays for specific applications. Novel, complete genome sequences are released almost daily now and many different experimental applications exist for tiling arrays. Theoretically, oligonucleotide probes should be evenly spaced, but often the experimental questions warrants a particular interest in specific genes or their upstream regions and prefer higher coverage (or better regions) for particular regions of the chromosome.

– *Inclusion of obligatory oligonucleotides.* The algorithm can easily incorporate the selection of specific oligonucleotides, such as positioned at the start of a gene or exon to ensure specific coverage and optimal tiling density. Another potential constraint is selection of probes that span exon boundaries [15]. For comparison with previous results and standardization, it might be helpful to use oligonucleotide probes from previous experimental designs. This constraint is trivially implemented for a obligatory probe o by removal of edges (i, j), $i < o < j$.

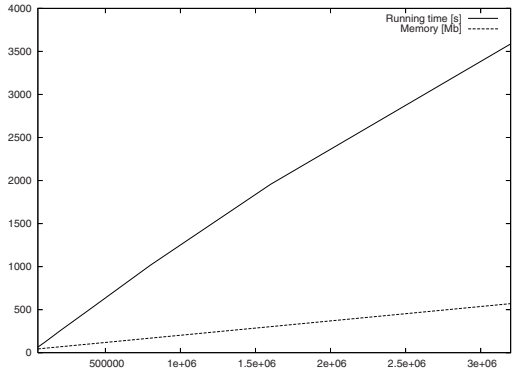

Fig. 3. Memory usage and running time of the algorithm for segments of *human* chromosome 2. In this experiments, n denotes the number of probe candidates, $k = 400$ denotes the neighborhood cardinality. Further parameters were $d^\star = 150$, $T_m^\star = 87.5$.

- *Heterogeneous probe density.* To improve the density in gene-rich or other regions of interest, one can specify a shorter spacing of the oligonucleotides by a position-specific definition of the $d^\star(p)$, where p designates the chromosomal location. g
- *Position-dependent penalty weights.* Likewise, to achieve a more even spacing for particular regions, the algorithm can relax criteria locally by using position-specific weights $w_d(p)$ instead of global weights, similarly for melting temperature and quality.

2.4 Optimal Solutions from Monge Theory

In the case of homogeneous (position-independent) weights w_d, w_t and w_q and design parameters $d^\star$, optimal solutions can be obtained in linear time from the theory of Monge matrices. See Burkard *et al* [16] for a review.

Definition 1. *A matrix C is called an upper triangular Monge matrix, if for all integers i, r, j, s with $1 \leq i < r \leq j < s \leq n$ the following condition holds:*

$$c_{ij} + c_{rs} \leq c_{is} + c_{rj}. \tag{6}$$

Lemma 1. *The matrix $W = \{w(i,j)\}$ is an upper triangular Monge matrix.*

Proof. Let i, r, j, s be integers such that $1 \leq i < r \leq j < s \leq n$ holds. We can substitute our definition of edge weights into the Monge condition (6):

$$\begin{aligned} & w_d \times d(i,j) + w_t \times pt(j) + w_q \times pq(j) \\ & + w_d \times d(r,s) + w_t \times pt(s) + w_q \times pq(s) \\ & \leq w_d \times d(i,s) + w_t \times pt(s) + w_q \times pq(s) \\ & + w_d \times d(r,j) + w_t \times pt(j) + w_q \times pq(j). \end{aligned} \tag{7}$$

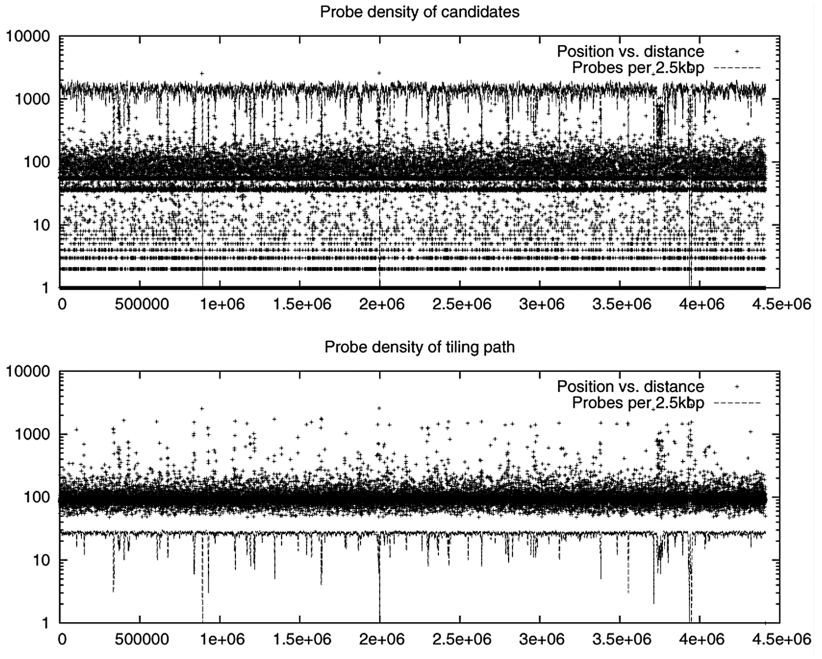

Fig. 4. Distance between adjacent candidates and the density of candidate probes in 2.5kb windows for *M. tuberculosis* (top). The low density regions are caused by known repetitive regions.On the bottom we show distance and density for a minimal tiling path computed for $d^\star = 90$ and stringent conditions on oligonucleotide quality, causing gaps. Note, the there is little deviation from $d^\star$ towards smaller values.

Note that the quality and temperature penalties are identical on both sides and hence cancel. We multiply both sides by $d^\star/w_d$ to simplify the remaining distance penalties. We obtain

$$\begin{aligned} &|d^\star - (p(j) - p(i))| + |d^\star - (p(s) - p(r))| \\ &\leq |d^\star - (p(s) - p(i))| + |d^\star - (p(j) - p(r))| \,. \end{aligned} \tag{8}$$

To check that the inequality holds we have to consider cases depending on the signs of the individual terms, which is tedious but fully elementary. If, for example, $p(j) - p(r) \geq d^\star$, it follows that also $p(j) - p(i) \geq d^\star$, $p(s) - p(r) \geq d^\star$ and $p(s) - p(i) \geq d^\star$. Then the inequality (8) is equivalent to $-d^\star + p(j) - p(i) - d^\star + p(s) - p(r) \leq -d^\star + p(s) - p(i) - d^\star + p(j) - p(r)$where all terms cancel. The other cases are left to the reader.

This is relevant, as linear time algorithms [17,18,19] exist, which are on-line variants of the SMAWK-algorithm of Aggarwal *et al* [20], and which solve the shortest path problem for upper triangular Monge matrices. Moreover, also efficient algorithms to compute shortest paths with a prescribed number of edges have been proposed; e.g. [21].

3 Application

We have designed a tiling array for the genome of *Mycobacterium tuberculosis* H37Rv comprising 44,000 spots [22], a desired $T_m^\star$ of 87.5 °C, desired distance of $d^\star = 150bp$ using oligomers of length 50 to 60. Mycobacteria have GC-rich genomes and contain several repetitive regions of varying degree of similarity. Instead of masking known biases, we explored the maximal possible tiling path defined by sequence properties only. Figure 4 shows that regions without coverage have no candidate oligonucleotides. As manual inspection shows, the gaps in the coverage (spikes) are typically caused by repeated regions such as insertion sequences, which results in 2kb regions without placements of oligonucleotides. The largest regions (6kb) without oligo placement correspond to proteins belonging to the families of PE and PPE genes, which are highly repetitive structures with low sequence complexity. These regions are of no particular interest for our experimental settings and can safely be ignored.

Note, any naive equidistant tiling with $d^\star = 150$ selects more than 75% of probes from the over 5.3 million oligonucleotides filtered out by `Flog`.

4 Discussion

We formulate a multi-criterion optimization problem in the design of oligonucleotide tiling or genome arrays. The minimal cost tiling path problem lends itself to a reformulation as a shortest path problem and a solution with Dijkstra's algorithm. Due to the structure of our cost function we can bound the neighborhood cardinality in the graph by a constant independent of n, the number of candidate probes, leading to a $O(n \times \log n)$ complexity. We demonstrate on real data that our efficient implementation of Dijkstra's algorithm allows us to solve problem instances with millions of candidates on standard desktop computers. Solutions can be controlled globally by criteria weights and locally to incorporate different constraints with respect to solution specifics and even extend partial solutions. A case study on a bacterial genome shows the effectiveness in covering genomes without prior handling of repetitive regions, in contrast to [6], which also does not optimize for oligonucleotide quality, and in obtaining balanced solutions of high-quality oligonucleotides with low cross-hybridization potential.

The size of the final design m cannot be specified directly. Rather the genome size divided by the desired distance between probe start position gives the expected number of probes, that is $m \approx |S|/d^\star$. Depending on parameter choices this may or may not be realized and consequently m can possibly not take on arbitrary values. Nevertheless a parameter exploration starting with $d^\star = |S|/m$ quickly provides reasonable solutions in practice. A prior approach [7] of the same log-normal complexity as ours allows specification of m directly but produces solutions with possibly large variations in probe distances and without discrimination of probe quality.

Further improvements concern the inclusion of non-unique probes [23], probes which hybridize very well in two or more genomic positions but otherwise do not

cross-hybridize, in the candidate generation which will facilitate closing of gaps due to repetitive regions. A spatially adaptive quality measure [7] for candidates can easily be incorporated. We will also explore an implementation of the Monge linear time algorithms in practice.

An implementation of our method is available from `http://algorithmics.molgen.mpg.de/Tileomatic`.

Acknowledgments. Thanks to Jörg Schreiber at the MPI for Infection Biology for helpful discussions, to Stefan Bienert for making Flog [8] available to us and to Janne Grunau for computational experiments. We would also thank one of the reviewers for helpful comments regarding Monge theory.

References

1. Kane, M.D., Jatkoe, T.A., Stumpf, C.R., Lu, J., Thomas, J.D., Madore, S.J.: Assessment of the sensitivity and specificity of oligonucleotide (50mer) microarrays. Nucleic Acids Res. 28(22), 4552–4557 (2000)
2. Matveeva, O., Shabalina, S., Nemtsov, V., Tsodikov, A., Gesteland, R., Atkins, J., Journals, O.: Thermodynamic calculations and statistical correlations for oligoprobes design. Nucleic Acids Research 31(14), 4211–4217 (2003)
3. He, Z., Wu, L., Li, X., Fields, M., Zhou, J.: Empirical Establishment of Oligonucleotide Probe Design Criteria. Applied and Environmental Microbiology 71(7), 3753–3760 (2004)
4. Pozhitkov, A., Noble, P.A., Domazet-Loso, T., Nolte, A.W., Sonnenberg, R., Staehler, P., Beier, M., Tautz, D.: Tests of rrna hybridization to microarrays suggest that hybridization characteristics of oligonucleotide probes for species discrimination cannot be predicted. Nucleic Acids Res. 34(9) (2006)
5. Selinger, D.W., Cheung, K.J., Mei, R., Johansson, E.M., Richmond, C.S., Blattner, F.R., Lockhart, D.J., Church, G.M.: Rna expression analysis using a 30 base pair resolution escherichia coli genome array. Nat. Biotechnol. 18(12), 1262–1268 (2000)
6. Bertone, P., Trifonov, V., Rozowsky, J.S., Schubert, F., Emanuelsson, O., Karro, J., Kao, M.Y., Snyder, M., Gerstein, M.: Design optimization methods for genomic dna tiling arrays. Genome Res. 16(2), 271–281 (2006)
7. Lipson, D., Yakhini, Z., Aumann, Y.: Optimization of probe coverage for high-resolution oligonucleotide acgh. Bioinformatics 23(2), 77–83 (2007)
8. Bienert, S.: Flexible combination of filters for oligodesign. Diplomathesis, Center for Bioinformatics, Universität Hamburg (2006)
9. Abouelhoda, M., Kurtz, S., Ohlebusch, E.: Replacing suffix trees with enhanced suffix arrays. Journal of Discrete Algorithms 2(1), 53–86 (2004)
10. Dijkstra, E.W.: A note on two problems in connexion with graphs. In: Numerische Mathematik, vol. 1, pp. 269–271. Mathematisch Centrum, Amsterdam, The Netherlands (1959)
11. Fredman, M.L., Tarjan, R.E.: Fibonacci heaps and their uses in improved network optimization algorithms. J. ACM 34(3), 596–615 (1987)
12. Leda: `http://www.algorithmic-solutions.com/`
13. Schliep, A.: The software GADAR and its application to extremal graph theory. In: Proceedings of the Twenty-fifth Southeastern International Conference on Combinatorics, Graph Theory and Computing, Boca Raton, FL, vol. 104, pp. 193–203 (1994)

14. Automatically tuned linear algebra software (atlas): http://math-atlas.sourceforge.net/
15. Shai, O., Morris, Q., Blencowe, B., Frey, B.: Inferring global levels of alternative splicing isoforms using a generative model of microarray data. Bioinformatics 22(5), 606 (2006)
16. Burkard, R.E., Klinz, B., Rudolf, R.: Perspectives of monge properties in optimization. Discrete Applied Mathematics 70(2), 95–161 (1996)
17. Wilber, R.: The concave least-weight subsequence problem revisited. J. Algorithms 9(3), 418–425 (1988)
18. Eppstein, D.: Sequence comparison with mixed convex and concave costs. J. Algorithms 11(1), 85–101 (1990)
19. Galil, Z., Park, K.: A linear-time algorithm for concave one-dimensional dynamic programming. Inf. Process. Lett. 33(6), 309–311 (1990)
20. Aggarwal, A., Klawe, M., Moran, S., Shor, P., Wilber, R.: Geometric applications of a matrix searching algorithm. In: SCG '86: Proceedings of the second annual symposium on Computational geometry, New York, NY, USA, pp. 285–292. ACM Press, New York (1986)
21. Aggarwal, A., Schieber, B., Tokuyama, T.: Finding a minimum weight k-link path in graphs with monge property and applications. In: SCG '93: Proceedings of the ninth annual symposium on Computational geometry, New York, NY, USA, pp. 189–197. ACM Press, New York (1993)
22. Cole, S., Brosch, R., Parkhill, J., Garnier, T., Churcher, C., Harris, D., Gordon, S., Eiglmeier, K., Gas, S., Barry III, C., et al.: Deciphering the biology of Mycobacterium tuberculosis from the complete genome sequence. Nature 393, 537–544 (1998)
23. Schliep, A., Torney, D., Rahmann, S.: Group testing with DNA chips: generating designs and decoding experiments. In: Proceedings of the 2nd IEEE Computer Society Bioinformatics conference, pp. 84–93. IEEE Computer Society Press, Los Alamitos (2003)