Positional Sequencing by Hybridization

Sridhar Hannenhalli
Department of Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802
hannenha@cse.psu.edu

William Feldman
Department of Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802
feldman@cse.psu.edu

Herbert F. Lewis
Department of Applied Mathematics
State University of New York
Stony Brook, NY 11794-3600
hlewis@ams.sunysb.edu

Steven S. Skiena*
Department of Computer Science
State University of New York
Stony Brook, NY 11794-4400
skiena@sbcs.sunysb.edu

Pavel A. Pevzner†
Departments of Mathematics and Computer Science
University of Southern California
Los Angeles, CA 90089-1113
ppevzner@hto.usc.edu

*This work was partially supported by NSF Research Initiation Award CCR-9109289, New York State Science and Technology Foundation Grant RDG-90171, and ONR award 400x116yip01.

†This work is supported by NSF Young Investigator Award, NSF grant CCR-9308567, NIH grant 1R01 HG00987 and DOE grant DE-FG02-94ER61919.

Abstract

Sequencing By Hybridization (SBH) is a promising alternative to the classical DNA sequencing approaches. However, the resolving power of SBH is rather low: with 64 Kb sequencing chips unknown DNA fragments only as long as 200 bp can be reconstructed in a single SBH experiment. To improve the resolving power of SBH Broude *et. al.*, 1994 recently suggested *positional SBH* (PSBH) allowing (with additional experimental work) to measure approximate positions of every l-tuple in a target DNA fragment.

We study the *positional eulerian path* problem motivated by PSBH. The input to the positional eulerian path problem is an eulerian graph $G(V, E)$ in which every edge has an associated range of integers and the problem is to find an eulerian path $e_1, \ldots, e_{|E|}$ in G such that the range of e_i contains i. We show that positional eulerian path problem is NP-complete even when the maximum out-degree (in-degree) of any vertex in the graph is 2. On the positive note we present polynomial algorithms to solve a special case of PSBH (bounded PSBH), where the range of the allowed positions for any edge is bounded by a constant (it corresponds to accurate experimental measurements of positions in PSBH). Moreover, if the positions of every l-tuple in an unknown DNA fragment of length n are measured with $O(\log n)$ error, then our algorithm runs in polynomial time. We also present an estimate of the resolving power of PSBH for more realistic case when positions are measured with $\Theta(n)$ error.

1 Introduction

Sequencing by hybridization (SBH) [1] [5] [10] [18] is a promising approach to DNA sequencing which offers the potential of reduced cost and higher throughput over traditional gel-based approaches. The basic procedure is to attach a large set of single-stranded oligonucleotides to substrate, forming a *sequencing chip*. A solution of radioactive or fluorescent labeled single-stranded target DNA fragments is exposed to the chip. These fragments hybridize with complementary oligonucleotides on the chip, and the hybridization is detected using a nuclear or spectroscopic detector. Under the assumption of an ideal *stringent* hybridization, this experiment provides information about ℓ-tuple composition of an unknown DNA fragment. Pevzner and Lipshutz [13] and Chetverin and Kramer [4] give a survey of the current state of the art in sequencing by hybridization, both technologically and algorithmically.

Biologists originally proposed to use SBH with chip $C(k)$, consisting of all 4^k strings of length k in $\{A, C, T, G\}$ alphabet. In this case, SBH sequence reconstruction is reduced to *eulerian* path problem1 in a subgraph of the *de Bruijn* graph. Every eulerian path in this subgraph corresponds to a possible sequence reconstruction [15]. Clearly, the reconstruction is not unique unless the subgraph has unique eulerian path. To enforce uniqueness of the reconstruction, large sequencing chips are needed to reconstruct relatively short strands of DNA. For example, the classical chip $C(8)$, consisting of all octanucleotides, suffices to reconstruct 200 nucleotide long sequences in only 94 of 100 cases [14], even in an ideal SBH experiment.

Recently, modifications of classical SBH have been proposed to reduce ambiguities in sequence reconstruction ([2], [3], [7], [8],[9], [11], [17], [12]). In particular, *positional sequencing by hybridization* [2] aims at resolving ambiguities by approximate measurement of the positions of each ℓ-tuple in the unknown DNA fragment. Thus for each edge in the subgraph of the de Bruijn graph, we know approximately where it occurs in the eulerian path corresponding to the unknown sequence.

In the next two sections we describe the relationship between PSBH and positional eulerian path problem and prove that the latter is NP-complete. In section 4 we present two polynomial time algorithms for a special case of PSBH, *Bounded PSBH (BPSBH)*, where the range of positions for each ℓ-tuple in SBH spectrum is bounded by a small number. Finally in section 5 we use simulations to study the resolving power of PSBH.

2 PSBH and Positional Eulerian Path Problem

For an alphabet Σ and an integer ℓ, the *de Bruijn* graph $G_\ell(\Sigma)$ contains $|\Sigma|^{\ell-1}$ vertices corresponding to all $(\ell-1)$-tuples over Σ. There is an edge from vertex $u = (u_1, \ldots, u_{\ell-1})$ to vertex $v = (v_1, \ldots, v_{\ell-1})$ iff $(u_2, \ldots, u_{\ell-1}) = (v_1, \ldots, v_{\ell-2})$. This edge is labeled by ℓ-tuple $(u_1, \ldots, u_{\ell-1}, v_{\ell-1})$. Any path of length p in the de Bruijn graph corresponds to a sequence of length $(\ell + p - 1)$ over Σ. On the other hand every sequence S over Σ defines a multigraph G_S which is a subgraph of the de Bruijn graph $G_\ell(\Sigma)$ induced by the edges corresponding to ℓ-tuples present in S [15]. The multiplicity of an edge in G_S is number of occurrences of the corresponding ℓ-tuple in S.

PSBH provides extra information about the hybridized oligonucleotides, *viz.* their approximate distances from the beginning of the sequence. Although it makes the reconstruction less ambiguous, polynomial algorithms for PSBH sequence reconstruction are unknown. Clearly, PSBH can be reduced to computing an eulerian path with an additional restriction that the position of any edge in the computed eulerian path should be in the range of positions associated with the edge.

We will denote the position of an edge e in a path P by $\pi(P,e)$. PSBH can be adequately formulated in combinatorial terms as *positional eulerian path* problem stated as follows:

Given a directed multigraph $G(V,E)$ and an interval $I_e = \{l_e, h_e\}$, $l_e \leq h_e$ associated with every edge $e \in E$ the problem is to find an eulerian path P in G such that for all $e \in E$, $l_e \leq \pi(P,e) \leq h_e$.

In the following section we show that *positional eulerian path* problem is NP-complete. We further study the *k-bounded positional eulerian path* problem corresponding to the case when the sizes of all the intervals I_e are bounded by a constant k (*i.e.* $h_e - l_e \leq k$) and devise polynomial algorithms for this problem.

3 Complexity of Positional Eulerian Path Problem

In this section, we demonstrate that the positional eulerian path problem is NP-complete, even with graphs of low degree and when all error intervals are the same size, *i.e.* $h_e - l_e = h_{e'} - l_{e'}$, for all $e, e' \in E$. The second condition implies that we can assign each edge $e \in E$ an estimated position p_e, and that a global error bound of B implies that $h_e = p_e + B$ and $l_e = p_e - B$. This formulation appears natural for modeling experimental data.

Theorem 1 *Positional eulerian path problem is NP-complete, even if each vertex has in-degree and out-degree at most two and intervals associated with edges are of the same size.*

Proof: Clearly, the positional eulerian path problem is in NP. For the hardness proof, we use a reduction from Hamiltonian path in directed graphs of in-degree and out-degree exactly two [16]. Specifically, given a digraph $G(V,E)$ with n vertices and $m = 2n$ edges, we will construct a digraph $G'(V',E')$ with $5n+2$ vertices and $8n+2$ edges, each edge $e \in E'$ assigned a integer position p_e such that G' has a positional eulerian path with each edge $e \in E'$ satisfying $|p_e - \pi(P,e)| \leq B$ if and only if G has a Hamiltonian path.

For each vertex $v_i \in V$, $1 \leq i \leq n$, we introduce 3 new vertices in V', v_{i1}, v_{i2}, and v_{i3} (Fig.1). We also introduce an additional $2n+2$ vertices w_i, $1 \leq i \leq 2n+2$ in V'. We set $B = 4n$. E' consists of the following three classes of edges:

- The first class E_1' consists of $n-1$ directed edges, (v_{i1}, v_{i2}), $1 \leq i \leq n-1$ and $2n+3$ directed edges, $\{(v_{n1}, w_1), (w_1, w_2), \ldots, (w_{2n+1}, w_{2n+2}), (w_{2n+2}, v_{n2})\}$. Each of the $3n+2$ edges in this class is assigned the position $\{1\}$. The window of allowable visit times is thus $\{1, 4n+1\}$.

- The second class E_2' represents the edges in E. For each directed edge (v_i, v_j) in E, introduce the edge (v_{i2}, v_{j1}) in E_2'. Each of the $2n$ edges in this class is assigned the position $\{4n+2\}$. The window of allowable visit times is thus $\{2, 8n+2\}$.

- The third class of edges E_3' ensures that G' is connected and that for all vertices the in-degree equals the out-degree. Together this implies that G' is eulerian [6]. For each i, $1 \leq i \leq n$ introduce the pair of directed edges (v_{i1}, v_{i3}) and (v_{i3}, v_{i2}) in E_3'. Finally introduce the n directed edges $\{(v_{13}, v_{23}), (v_{23}, v_{33}), \ldots, (v_{n3}, v_{13})\}$ in E_3'. Each of the $3n$ edges in this class is assigned the position $\{8n+2\}$. The window of allowable visit times is thus $\{4n+2, 8n+2\}$.

If G has a Hamiltonian path, we claim that G' must contain a positional eulerian path satisfying the time window constraints. We can map any sequence of n vertices of G forming a Hamiltonian path to $4n+1$ edges of G'. These edges form a path in G' which starts and ends with edges in E_1' and alternates between edges in E_1' and E_2' with the exception of the chain between vertices v_{n1} and v_{n2} which includes all the w_i vertices. This path includes all $3n+2$ edges in E_1' and exactly $n-1$ edges in E_2'. Thus the edges in E_1' are visited between $\{1, 4n+1\}$ and since the path starts with an edge from E_1', no edge in E_2' is visited before time 2 as desired. If we remove the edges of the path from G' we are left with a connected graph in which all but the starting and ending vertices of the path have equal in-degree and out-degree. There are no time window restrictions since the remaining edges in E_2' and all the edges in E_3' can be visited between $\{4n+2, 8n+2\}$. An eulerian path can always be found since all the vertices (except the starting and ending vertices) are of balanced degree.

If G' contains an eulerian path satisfying the time window constraints, we claim that G must contain a Hamiltonian path. Take the first $4n+1$ edges of the eulerian path. Notice that all the edges of class E_1' are among the first $4n+1$ edges. Remove the chain between vertices v_{n1} and v_{n2} (*i.e. all the w_i vertices*) and connect v_{n1} directly to v_{n2}. The resulting path alternates between edges in E_1' and E_2'. The ordering of the class E_1' edges defines a directed Hamiltonian path in G.

The reduction (shown in Fig. 1) can clearly be done in polynomial time, giving the result. ∎

4 Algorithms for Positional Eulerian Path Problem

In this section we present two simple polynomial time algorithms to solve k-bounded PSBH. The first one is a *divide and conquer* algorithm with running time $O(n^{2\cdot k+\log(2\cdot k)})$ for a fixed constant k. Later we prove that the running time of an breadth-first-search algorithm is linear for constant k and polynomial for $k = O(\log\ n)$.

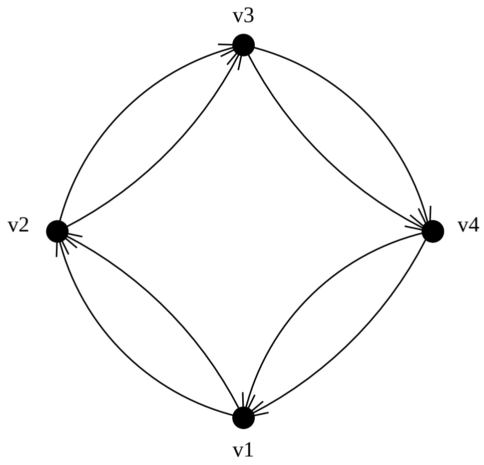
4.1 A Divide and Conquer Algorithm

Partition the entire set of edges E into the following three subsets: $L = \{e:\ h_e < \frac{|E|}{2}\}$, $R = \{e:\ l_e > \frac{|E|}{2}\}$ and $M = \{e:\ l_e \leq \frac{|E|}{2} \leq h_e\}$. Clearly, $|M| \leq 2k-1$ (otherwise there can not be a solution). This observation leads to a divide and conquer algorithm, since we can solve the original k-bounded positional eulerian path problem by solving, for all partitioning of M (a constant number) into M_L and M_R, two smaller problems (over $L \cup M_L$ and $R \cup M_R$).

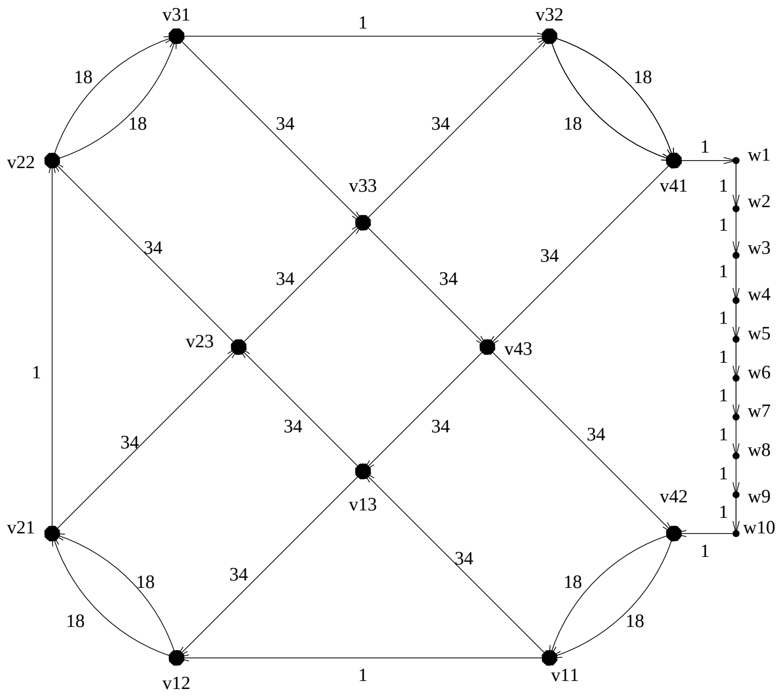
Given a set $S \subseteq E$, edges $b, f \in S$ and an integer sp, a path P is (S, b, f, sp)-*compatible* if:

1. P is an eulerian path in $G(V, S)$ starting with edge b and ending in edge f.
2. $\forall e \in S, l_e \leq (sp + \pi(P, e) - 1) \leq h_e$.

The second condition in the above definition implies that P could be a subpath starting at position sp, of some positional eulerian path over the entire graph $G(V, E)$. Clearly, a $(E, b, f, 1)$-compatible path corresponds to a positional eulerian path in G. Define $\mathcal{C}_i$ to be the set of edges whose range contain i, *i.e.* $\mathcal{C}_i = \{e : l_e \leq i \leq h_e\}$. Fig. 2 shows the algorithm *Divide_Conquer* which detects the existence of a positional eulerian path in a graph $G(V, E)$, based on a divide and conquer strategy. The algorithm *Divide_Conquer* can be easily modified to find a positional eulerian path (not just verify its existence). Let $\psi(n)$ denote the running time required for exhaustive search of a (S, b, f, sp)-*compatible* path in a graph where $|S| = n$. Clearly, $|L \cup M_L \cup \{m\}|$, $|R \cup M_R \cup \{m\}| \leq |S|/2 + 1$. In *Compatible*, steps 6, 7 and 8 can be computed in $O(n)$ time. Step 9 is executed at most $2k-1$ times and step 11 is executed at most 2^{2k-2} times. Every execution of step 11



(a)



(b)

Figure 1: The hardness reduction applied to a doubled 4-cycle. (a) Instance of Hamiltonian Path Problem G is a doubled 4-cycle; sequence $(v1, v2, v3, v4)$ of vertices gives a hamiltonian path. (b) Instance of bounded Eulerian Path Problem corresponding to G; Each edge is labeled with its position; sequence $(v11, v12, v21, v22, v31, v32, v41, w1, w2, w3, w4, w5, w6, w7, w8, w9, w10, v42, v11, v13, v23, v33, v43, v13, v12, v21, v23, v22, v31, v33, v32, v41, v43, v42, v11)$ of vertices describes an eulerian path.

Divide_Conquer$(G(V,E))$

Forall $b \in \mathcal{C}_1$

Forall $f \in \mathcal{C}_{|E|}$

If *Compatible*$(E,\ b,\ f,\ 1)$ returns **Success**

Return **Success**

Return **Failure**

Compatible$(S,\ b,\ f,\ sp)$

If $|S| \leq 2k$

Search $G(V,S)$ exhaustively for a (S,b,f,sp)-compatible path

If such a path is found

return **Success**

else return **Failure**

$L \leftarrow \{e \in S : h_e < sp + \frac{|S|}{2}\}$

$R \leftarrow \{e \in S : l_e > sp + \frac{|S|}{2}\}$

$M \leftarrow S \backslash (L \cup R)$

Forall $m \ \in \ M$

/* Assume that edge m occupies the position $sp + \frac{|S|}{2}$ */

Forall $M_L \subseteq M \backslash \{m\}$ such that $|M_L| = \frac{|S|}{2} - |L|$

$M_R \leftarrow \{M \backslash \{m\}\} \backslash M_L$

/* $M_L(M_R)$ corresponds to the edges before(after) the edge m */

/* in a (S,b,f,sp)-compatible path */

Compatible$(L \cup M_L \cup \{m\},\ b,\ m,\ sp)$

Compatible$(R \cup M_R \cup \{m\},\ m,\ f,\ sp + \frac{|S|}{2})$

If both of the above calls return **Success**

Return **Success**

Return **Failure**

Figure 2: A Divide and Conquer algorithm for PSBH

requires $O(k)$ time to construct the sets M_L and M_R. Let $T(n)$ denote the running time of $Compatible(S, b, f, sp)$. $T(n)$ satisfies the following recursion.

$$T(n) \leq \begin{cases} (2k-1) \cdot 2^{2k-2} \cdot 2 \cdot T(n/2+1) + \alpha \cdot n + \beta \cdot (2k-1) \cdot 2^{2k-2} \cdot k & \text{if } n \geq 2k+1 \\ \psi(n) & \text{otherwise} \end{cases}$$

for some constants α and β. Hence

$$T(n) = O(\psi(2k) \cdot n^{\log((2k-1) \cdot 2^{2k-2})}) = O(\psi(2k) \cdot n^{2 \cdot k + log(2 \cdot k)})$$

Therefore the running time of the algorithm *Divide_Conquer* is $O(k^2 \cdot \psi(2k) \cdot n^{2 \cdot k + \log(2 \cdot k)})$ since $|\mathcal{C}_1|$ and $|\mathcal{C}_{|E|}|$ are at most k.

We show in the following that a simple breadth first search of a positional eulerian path turns out to be computationally more efficient than the previous algorithm by exploiting the fact that the de Bruijn graph $G_\ell(\Sigma)$ is of bounded degree.

4.2 Breadth-First Search Algorithm

Recall that an instance of PSBH is a multigraph with the multiplicity of an edge equal to the number of occurrences of corresponding ℓ-tuple. In the following we differentiate between different instances of a multiple edge (since they may have different intervals associated with them).

We use the term *edge* to refer to the edges of the multigraph without distinguishing between the multiple copies of an edge and we use the term *edge-instance* to distinguish between the multiple copies of the same edge. Every edge $\mathbf{e}$ of multiplicity r represents the set of r edge-instances. Let the number of edges be m, the edges being $(\mathbf{e_1}, \mathbf{e_2}, \ldots, \mathbf{e_m})$ with respective multiplicities $(n_1, n_2, \ldots, n_m)$. Clearly, $\sum_{i=1}^{m} n_i = n$, where n is the number of edge-instances. Represent the edge containing an edge-instance f by $\hat{f}$. Let $P = (f_1, f_2, \cdots, f_n)$ be a positional eulerian path (of edge-instances) and let $\hat{P} = (\hat{f}_1, \hat{f}_2, \cdots, \hat{f}_n)$ be the corresponding path of edges. Clearly, any edge $\mathbf{e_i}, 1 \leq i \leq m$ occurs n_i times in an eulerian path $\hat{P}$. In search for a positional eulerian path if we consider only paths of edges (as opposed to path of edge-instances), different possibilities of extending a path by an edge is bounded by $|\Sigma|$ (4 for DNA sequences). At the same time we are required to map various occurrences of an edge along the path to its instances. In the following we observe some properties of such paths allowing us to compute such a mapping efficiently.

A *path-prefix* $P_i = \{e_1, e_2, \ldots, e_i\}$ is a prefix of length i of some positional eulerian path. For a path-prefix P_i, $1 \leq i \leq n$ and an edge $\mathbf{e}$, define the set of *active* instances $\mathcal{A}(P_i, \mathbf{e})$ as the set of instances of $\mathbf{e}$ which are not present in P_i and whose intervals contain the position $i+1$, *i.e.* $\mathcal{A}(P_i, \mathbf{e}) = \{f : \hat{f} = \ \mathbf{e}, f \notin P_i, i+1 \in I_f\}$. Also define $\mathcal{A}(P_i)$ as $\bigcup_{\mathbf{e} \in E} \mathcal{A}(P_i, \mathbf{e})$. Positional eulerian path $P = \{e_1, e_2, \ldots, e_n\}$ is *ordered*, if for every $1 \leq i \leq n$, e_i minimizes h_{e_i} among all the instances in $\mathcal{A}(P_{i-1}, \hat{e_i})$.

Lemma 1 *For an instance G of k-bounded PSBH, there exists a positional eulerian path in G iff there exists an ordered positional eulerian path in G.*

Lemma 1 suggests that, while searching for a positional eulerian path, we can restrict the search space to ordered paths only, leading to an efficient algorithm. In the following, by positional eulerian path, we mean an ordered positional eulerian path. Define $next(P_i, \mathbf{e})$ as an edge-instance f of $\mathbf{e}$ such that $f \in \mathcal{A}(P_i, \mathbf{e})$ and f minimizes h_f among all the edge-instances in $\mathcal{A}(P_i, \mathbf{e})$ (ties are broken arbitrarily if there are more than one such edge-instances).

Starting with the set of all path-prefixes of length one (edges which may occur in the first position), we keep extending the paths by an edge until we have a path of length n or no such path exists. A path-prefix $P_i = \{e_1, e_2, \ldots, e_i\}$ can be *extended* to $P_{i+1} = \{e_1, e_2, \ldots, e_i, next(P_i, \mathbf{e})\}$ by an edge $\mathbf{e}$, if $\mathbf{e}$ follows edge $\hat{e_i}$. Notice that there are only $|\Sigma|$ of such edges. Also, P_i can be extended by edge $\mathbf{e}$ only if $next(P_i, \mathbf{e})$ exists. Moreover, lemma 1 suggests that it suffices to consider only $next(P_i, \mathbf{e})$ while extending path P_i by edge $\mathbf{e}$.

Let S_i be the set of all pairs $(P_i, \mathcal{A}(P_i))$. We can construct S_{i+1} from S_i by extending the path-prefix P_i using the information provided by $\mathcal{A}(P_i)$ for every $(P_i, \mathcal{A}(P_i)) \in S_i$. Notice that the set of active edge instances can be easily updated for the extended path-prefix. Define $\mathcal{L}_i = \{f : f$ is an edge instance and $l_f = i\}$ and $\mathcal{H}_i = \{f : f$ is an edge instance and $h_f = i\}$. Algorithm "BFS" shown in Fig. 3 outputs all positional eulerian paths for a given graph G.

Notice that $|S_i| \leq (2 \cdot k - 1) \cdot 2^{2 \cdot k - 1}, 1 \leq i \leq |E|$ since the maximum number of path-prefixes (which is bounded by the maximum number of instances which could occur at position i) is $2 \cdot k - 1$ and maximum number of different sets of active edges at any position is $2^{2 \cdot k - 1}$. As the preprocessing step we construct in $O(n \cdot k)$ time, a list of edge-instances allowed at

```
BFS(G(V,E))
1.    Construct S_1
2.    For i = 2 to |E|
3.        S_{i+1} ← Φ
4.        Forall (P_i, A(P_i)) ∈ S_i
              /* Let e_i be the last edge of P_i */
5.            Forall edges e following e_i
6.                If next(P_i, e) exists
7.                    P_{i+1} ← P_i extended by next(P_i, e)
8.                    A(P_{i+1}) ← (A(P_i)\{next(P_i, e)})\H_{i+1} ∪ L_{i+2}
9.                    S_{i+1} ← S_{i+1} ∪ {(P_{i+1}, A(P_{i+1}))}
10.   For all (P, A(P)) ∈ S_{|E|} /* A(P) is empty */
11.       Output P
```

Figure 3: Breadth first search of positional eulerian paths

position $i, 1 \leq i \leq n$. Clearly, S_1 can be constructed in $O(n)$ time. Step 4 is executed at most $O(k \cdot 4^k)$ times in a particular iteration of step 2. $next(P_i, \mathbf{e})$ in step 6 and $\mathcal{A}(P_{i+1})$ in step 8 can be computed in $O(k)$ time. Step 9 involves checking for duplications in S_{i+1} which can be done in $O(k^2 \cdot \log\ k)$ time. Hence overall time complexity of the above algorithm is $O(n \cdot k^3 \cdot \log\ k \cdot 4^k)$. Observing that, in i^{th} iteration we only need the set of active edges from S_{i-1}, the space complexity of BFS is $O(n \cdot k + k^2 \cdot 4^k)$.

Notice that even when $k = O(\log\ n)$, BFS still runs in polynomial time.

In the following section we study the computational limitations of PSBH.

5 Resolving Power of PSBH

In this section we consider a less restrictive model, where the ℓ-tuple locations are known to within a certain fractional error f, *i.e.* if the experimentally defined position of the ℓ-tuple is x we assume that $l_e = x(1-f)$ and $h_e = x(1+f)$.

Although *PSBH* provides an additional information as compared to classical SBH, it might lead to combinatorial problems in sequence reconstruction. The defect, $def(C, n)$, of a chip C on sequences of length n is defined as fraction of all the sequences of length n

for which the reconstruction with C is ambiguous. Clearly, the chip defect grows with the sequence length. From this perspective, we can use the maximum sequence length for which the ambiguity of sequence reconstruction is below a threshold, as a measure of effectiveness of chip. Formally, the effectiveness of a chip is the maximum length n such that $def(C,n) \leq 1-p$ for a given fraction p. We call this quantity *max_len(p,f)*, where f is allowed fractional error in the positions. Since *max_len(p,f)* is hard to compute analytically, we use Monte-Carlo simulations to come up with approximation.

We used a branch-and-bound program to determine whether a randomly generated DNA fragment has an ambiguous reconstruction. This program was run 500 times on randomly generated strings of a given length, assuming a given fraction f of positional error and tuple-length l. The sequence length was increased until the fraction of strings with ambiguous reconstructions reached the value p. We will denote the value of *max_len(f,p)* obtained experimentally as $L(f,p,l)$.

Figure 4 shows the values of $L(f,p,l)$ for sequencing chips with varying tuple-lengths and positional errors. The resulting sequence lengths represent improvements of one to two orders of magnitude over conventional SBH.

References

[1] W. Bains and G. Smith. A novel method for nucleic acid sequence determination. *J. Theor. Biol.*, 135:303–307, 1988.

[2] N. Broude, T. Sano, C. Smith, and C. Cantor. Enhanced DNA sequencing by hybridization. *Proc. of National Academy of Sciences*, 91:3072–3076, 1994.

[3] A. Chetverin and F. Kramer. Sequencing of pools of nucleic acids on oligonucleotide arrays. *Bio Systems*, 30:215–232, 1993.

[4] A. Chetverin and F. Kramer. Oligonucleotide arrays: New concepts and possibilities. *BioTechnology*, 12:1093–1099, 1994.

[5] R. Drmanac, I.Labat, I. Brukner, and R.Crkvenjakov. Sequencing of megabase plus dna by hybridization: theory of the method. *Genomics*, 4:114–128, 1989.

[6] R. Gould. *Graph Theory.* Benjamin/Cummings, Inc., Menlo Park CA, 1988.

p=1	f=1	f=3	f=5	f=10
l=5	L : 520	220	150	90
l=6	670	315	255	155
l=7	1300	590	410	260
l=8	2980	1250	770	470
l=9	5250	2290	1580	910
l=10	10000	4700	3130	1900
l=11	21900	9400	6200	3600

Fig. 4.5a

p=3	f=1	f=3	f=5	f=10
l=5	L : 635	260	160	95
l=6	960	470	315	195
l=7	1810	830	560	360
l=8	3500	1540	1020	710
l=9	7050	3060	2110	1370
l=10	13500	6150	4120	2530
l=11	28600	12300	8320	4700

Fig. 4.5b

p=5	f=1	f=3	f=5	f=10
l=5	L : 645	280	180	115
l=6	1080	510	360	215
l=7	2010	950	680	400
l=8	4200	1760	1160	730
l=9	8000	3500	2280	1490
l=10	14200	7000	4800	2770
l=11	32800	14000	9400	5420

Fig. 4.5c

p=10	f=1	f=3	f=5	f=10
l=5	L : 740	300	210	140
l=6	1290	600	420	270
l=7	2510	1090	780	530
l=8	4900	2170	1520	970
l=9	9700	4310	2790	1840
l=10	19300	8330	5680	3590
l=11	38900	16800	12100	7200

Fig. 4.5d

Figure 4: $L(f,p,l)$ for varying f, p and tuple length l. f and p are expressed in %

[7] K. Khrapko, Y. Lysov, A. Khorlin, I. Ivanov, G. Yershov, and S. Vasilenko. A method for DNA sequencing by hybridization with oligonucleotide matrix. *DNA Sequencing*, 1:375–388, 1991.

[8] K. Khrapko, Y. Lysov, A. Khorlin, V. Shik, V. Florent'ev, and A. Mirzabekov. An oligonucleotide hybridization approach to DNA sequencing. *FEBS Letters*, 256:118–122, 1989.

[9] Y. Lysov, A. Chernyi, A. Balaeff, K. Beattie, and A. Mirzabekov. DNA sequencing by hybridization to oligonucleotide matrix. Calculation of continuous stacking hybridization efficiency. *J. of Biom. Structure and Dynamics*, 11(4):797–812, 1994.

[10] Y. Lysov, V. Florent'ev, A. Khorlin, K. Khrapko, V. Shik, and A. Mirzabekov. DNA sequencing by hybridization with oligonucleotides. *Dokl. Acad. Sci. USSR*, 303:1508–1511, 1988.

[11] D. Margaritis and S. Skiena. Reconstring strings from substrings in rounds. In *36 ACM Symposium on Foundations of Computer Science*, pages 143–158, 1995.

[12] M. Gelfand O. Razgulyaev, A. Rubinov and M. Gelfand. Sequencing potential of nested strand hybridization. *J. of Comp. Biology*, 2:383–395, 1995.

[13] P. Pevzner and R. Lipshutz. Towards DNA sequencing chips. In *19th Int. Conf. Mathematical Foundations of Computer Science*, volume 841, pages 143–158, Lecture Notes in Computer Science, 1994.

[14] P. Pevzner, Y. Lysov, K. Khrapko, A. Belyavski, V. Florentiev, and A. Mirzabekov. Improved chips for sequencing by hybridization. *J. of Biomol. Structure and Dynamics*, 9:399–410, 1991.

[15] P.A. Pevzner. l-tuple DNA sequencing: computer analysis. *J. of Biomol. Structure and Dynamics*, 7:63–73, 1989.

[16] J. Plesnik. The NP-completeness of the hamiltonian cycle problem in planar digraphs with degree bound two. *Info. Proc. Letters*, 8:199–201, 1979.

[17] A. Rubinov and M. Gelfand. Reconstructing strings from substrings precedence data. *J. of Comp. Biology*, 2:371–382, 1995.

[18] E. Southern, U. Maskos, and R. Elder. Hybridization with oligonucleotide arrays. *Genomics*, 13:1008–10017, 1992.