On the Complexity and Approximation of Syntenic Distance

B. DasGupta[*] and **T. Jiang**[†] and **S. Kannan**[‡] and **M. Li**[§] and **Z. Sweedyk**[¶]

Abstract

The paper studies the computational complexity and approximation algorithms for a new evolutionary distance between multi-chromosomal genomes introduced recently by Ferretti, Nadeau and Sankoff. Here, a chromosome is represented as a set of genes and a genome is a collection of chromosomes. The syntenic distance between two genomes is defined as the minimum number of translocations, fusions and fissions required to transform one genome into the other. We prove that computing the syntenic distance is NP-hard and give a simple approximation algorithm with ratio 2. The question of how to improve the approximation ratio is also considered, and a tight connection between finding large "balanced" independent sets in bipartite graphs and such an improvement is shown. For the case when an upper bound d on the syntenic distance is known, we show that an an optimal syntenic sequence can be found in $O(n^2 \cdot 2^{O(d^2)})$ time. Next, we show that if the set of operations for transforming a genome is significantly restricted, we can nevertheless find a solution that performs at most $O(\log d)$ additional moves, where d is the number of moves performed by the unrestricted optimum. This result should help in the design of approximation algorithms. Finally, we investigate the median problem: Given three genomes, construct a genome minimizing the total syntenic distance to the three given genomes. The problem has application in the inference of phylogenies based on the syntenic distance. We prove that the problem is NP-hard and design an efficient polynomial time approximation algorithm with ratio $4 + \epsilon$ for any constant $\epsilon > 0$.

1 Introduction

The definition and study of appropriate measures of distance between pairs of species is of great importance in computational biology. Such measures of distance can be used, for example, in phylogeny construction and in taxonomic analysis.

As more and more molecular data becomes available methods for defining distances between species have focused on such data. One of the most popular distance measures is the edit distance between homologous DNA or aminoacid sequences obtained from different species. Such measures focus on point mutations and define the distance between two sequences as the minimum number of these moves required to transform one sequence into another. It has been recognized that the edit-distance may underestimate the distance between two sequences because of the possibility that multiple point

[*]Department of Computer Science, Rutgers University, Camden, NJ 08102. E-mail:bhaskar@crab.rutgers.edu. Work done while the author was at University of Waterloo and supported by a CGAT (Canadian Genome Analysis and Technology) grant.

[†]Department of Computer Science, McMaster University, Hamilton, Ontario L8S 4K1, Canada. E-mail: jiang@maccs.mcmaster.ca. Supported in part by NSERC Operating Grant OGP0046613, CGAT and a JSPS fellowship. Work done while visiting at University of Washington and Gunma University.

[‡]Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104, E-mail: kannan@cis.upenn.edu

[§]Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong. On leave from University of Waterloo. E-mail: mli@math.uwaterloo.ca. Supported by the NSERC operating grant OGP0046506 and a CGAT grant.

[¶]Department of Computer and Information Sciences, University of Pennsylvania, Philadelphia, PA. E-mail: zzz@saul.cis.upenn.edu.

mutations occurring at the same locus will be accounted for simply as one mutation. The problem is that the probability of a point mutation is not low enough to rule out this possibility.

Recently, there has been a spate of new definitions of distance that try to treat rarer, macrolevel mutations as the basic moves. For example, if we know the order of genes on a chromosome for two different species, we can define the *reversal* distance between the two species to be the number of reversals of portions of the chromosome to transform the gene order in one species to the gene order in the other species. The question of finding the reversal distance was first explored in the computer science context by Kececioglu and Sankoff and by Bafna and Pevzner and there has been significant progress made on this question by Bafna, Hannenhalli, Kececioglu, Pevzner, Ravi, Sankoff and others [1, 2, 7, 10, 11]. Other moves besides reversals have been considered as well. Breaking off a portion of the chromosome and inserting it elsewhere in the chromosome is referred to as a *transposition* and one can similarly define the transposition distance[3]. Similarly allowing two chromosomes (viewed as strings of genes) to exchange suffixes (or sometimes a suffix with a prefix) is known as a *translocation* and this move can also be used to define an appropriate measure of distance between two species for which much of the genome has been mapped [9].

Ferretti et. al.[5] proposed a distance measure that is at an even higher level of abstraction. Here even the order of genes on a particular chromosome of a species is ignored/ presumed to be unknown. It is assumed that the genome of a species is given as a collection of sets. Each set in the collection corresponds to a set of genes that are on one chromosome and different sets in the collection correspond to different chromosomes. In this scenario one can define a move to be either an exchange of genes between two chromosomes, the fission of one chromosome into two, or the fusion of two chromosomes into one. The *syntenic distance* between two species has been defined by Sankoff et. al.[5] to be the number of such moves required to transform the genome of one species to the genome of the other.

Notice that any recombination of two chromosomes is permissible in this model. By contrast, the set of legal translocations (in the translocation distance model) is severely limited by the order of genes on the chromosomes being translocated. Furthermore, the transformation of the first genome into the second genome does not have to produce a specified order of genes in the second genome. The underlying justification of this model is that the exchange of genes between chromosomes is a much rarer event than the movement of genes within a chromosome and hence a distance function should measure the minimum number of such exchanges needed.

Ferretti et. al. provide a heuristic that attempts to compute the syntenic distance and provide empirical evidence of the value of this distance measure. In this paper we attempt to put the notion of syntenic distance on more formal foundations. To wit, we show the following results.

- The syntenic distance is, in fact, a distance.
- An optimal sequence of moves can be assumed to occur in a canonical order with fusions preceding translocations, preceding fissions.
- The problem of computing the syntenic distance is NP-hard.
- There is an approximation algorithm that achieves a factor of 2 approximation to syntenic distance.
- Computing this distance is fixed parameter tractable.
- When the set of moves is significantly restricted, there is nevertheless an optimal sequence of restricted moves whose length is not much more than the length of the unrestricted optimal sequence.
- Any constant approximation of the "balanced" independent set problem for bipartite graphs leads to an approximation of the syntenic distance with ratio better than 2.
- The problem of computing the median genome, for a given set of 3 genomes, is NP-hard and admits an approximation with ratio $4+\epsilon$ for any constant $\epsilon > 0$.

These results will be described in the sections that follow. Due to space limitations some proofs are omitted; they can be found in [4].

2 Notation and Preliminaries

For the purpose of this paper, a genome is a collection of k subsets (called synteny sets or chromosomes) of a set of n objects (called genes). A genome mutates by one of three simple *moves*; these are the *translocation, fusion,* and *fission.*

Definition 2.1 *Let S_1, S_2, T_2, T_2 be sets such that at most one is empty and such that $T_1 \cup T_2 = S_1 \cup S_2$.*

(a) *If S_1, S_2, T_1, T_2 are non-empty then $(S_1, S_2) \longrightarrow (T_1, T_2)$ is called a* translocation of S_1 and S_2.

(b) *If S_2 is empty then $S_1 \longrightarrow (T_1, T_2)$ is called a* fission of S_1.

(c) *If T_2 is empty then $(S_1, S_2) \longrightarrow T_1$ is called a* fusion of S_1 and S_2.

Given two genomes $\mathcal{G}_1$ and $\mathcal{G}_2$ over some gene set Σ, the *syntenic distance from* $\mathcal{G}_1$ *to* $\mathcal{G}_2$, denoted $D(\mathcal{G}_1, \mathcal{G}_2)$, is the minimum number of moves needed to transform $\mathcal{G}_1$ into $\mathcal{G}_2$.

Proposition 2.1 $D(\mathcal{G}_1, \mathcal{G}_2) = D(\mathcal{G}_2, \mathcal{G}_1)$.

Proof: Given an optimal sequence of moves from $\mathcal{G}_1$ to $\mathcal{G}_2$, it is easy to reverse every move (the reverse of a fusion is a fission and vice versa) to get an optimal sequence of moves transforming $\mathcal{G}_2$ to $\mathcal{G}_1$. ■

It follows from Proposition 2.1 that $\mathcal{D}$ defines a metric over the set of genomes over Σ (reflexivity and triangle inequality of $\mathcal{D}$ are obvious).

Lemma 2.1 *Let* $\mathcal{G}_1, \mathcal{G}_2$ *be an instance of syteny. Then there is a sequence of moves* $\sigma = (\sigma_1, \ldots, \sigma_m)$ *such that* $m = D(\mathcal{G}_1, \mathcal{G}_2)$ *and every fission occurs after every translocation and fusion.*

Proof: Let $\sigma = (\sigma_1, \sigma_2, \ldots \sigma_m)$ be an optimal move sequence. If every fission occurs after every translocation or fusion we are done; so assume not. Let $i < m$ be the largest index such that σ_i is a fission preceding a translocation or fusion. We give a new optimal sequence $(\sigma_1, \ldots, \sigma_{i-1}, \sigma'_i, \sigma'_{i+1}, \sigma_{i+2}, \ldots, \sigma_m)$ where σ'_i is a translocation or fusion and σ'_{i+1} is a fission. Repeating the argument eventually yields the desired sequence.

Assume that σ_i is the fission $S_1 \cup S_2 \longrightarrow (S_1, S_2)$ and σ_{i+1} is either the fusion $(T_1, T_2) \longrightarrow T_1 \cup T_2$ or the translocation $(T_1, T_2) \longrightarrow (T'_1, T'_2)$. If neither T_1 nor T_2 is created by σ_i we can simply swap σ_i and σ_{i+1} to yield the desired sequence. Thus we need only consider the case where, without loss of generality, $S_1 = T_1$. Then we claim that σ_{i+1} is a translocation; otherwise we could replace σ_i and σ_{i+1} by $(S_1 \cup S_2, T_2) \longrightarrow (S_2, S_1 \cup T_2)$, reducing the number of move by 1, which contradicts the optimality of σ. Finally, since σ_{i+1} is a translocation, we can replace σ_i and σ_{i+1} by $\sigma'_i : (S_1 \cup S_2, T_2) \longrightarrow (T'_1 \cup S_2, T'_2)$ and $\sigma'_{i+1} : T'_1 \cup S_2 \longrightarrow (T'_1, S_2)$ to yield the desired sequence. ■

Note that the number of translocations, fusions and fissions is preserved in construction of the previous proof. Thus we get the following corollary.

Corollary 2.1 *Let* $\mathcal{G}_1, \mathcal{G}_2$ *be an instance of synteny. If there is an optimal move sequence with* m_1 *translocations,* m_2 *fusions, and* m_3 *fissions, then there is an optimal move sequence with* m_1 *translocations,* m_2 *fusions, and* m_3 *fissions in which all fissions come after all translocations and fusions.*

Lemma 2.2 *Let* $\mathcal{G}_1, \mathcal{G}_2$ *be an instance of synteny. Then there is a sequence of moves* $\sigma = (\sigma_1, \ldots, \sigma_m)$ *such that* $m = D(\mathcal{G}_1, \mathcal{G}_2)$ *and such that all fusions come before all translocations which come before all fissions.*

Proof: Let $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_m)$ be an optimal move sequence. If there are no translocations or fusions we are done. If not, by the Lemma 2.1 we may assume that all translocations and fusions occur before all fissions. Let i be the index of the last non-fission in the sequence and let $\mathcal{G}'$ be the collection of sets after the i-th move. Since σ is optimal $D(\mathcal{G}_1, \mathcal{G}') = i$. Using Proposition 2.1 and Corollary 2.1 (on the problem of transforming $\mathcal{G}'$ to $\mathcal{G}_1$) there is a move sequence that transforms $\mathcal{G}_1$ into $\mathcal{G}'$ consisting solely of fusions and translocations in which the fusions occur first. Concatenating this sequence with $\sigma_{i+1}, ..., \sigma_m$ yields the desired move sequence. ■

2.1 The Compact Representation of Synteny

For our subsequent proofs it is easier to deal with the *compact representation* of the synteny problem as described in [5]. Assume that the genomes $\mathcal{G}_1$ and $\mathcal{G}_2$ contain n and k sets, respectively. Then, the compact representation of $\mathcal{G}_2$ with respect to $\mathcal{G}_1$ is defined as follows: replace the i^{th} set $G_{1,i}$ of $\mathcal{G}_1$ by the set $\{i\}$ for $1 \leq i \leq n$, and for every set S occurring in $\mathcal{G}_2$, replace S by the set $\cup_{i \in S}\{j \mid i \in G_{1,j}\}$. Let $\mathcal{G}'_1$ and $\mathcal{G}'_2$ be the two modified genomes. It is easy to see that $D(\mathcal{G}_1, \mathcal{G}_2) = D(\mathcal{G}'_1, \mathcal{G}'_2)$. Hence, we can alternatively define the synteny problem using the compact representations of genomes as follows.

Definition 2.2 *Given a collection* $\mathcal{S}(n,k)$ *of* k *(not necessarily distinct) sets* $S_1, \ldots, S_k$, $S_i \subseteq \{1, 2, \ldots, n\}$, *the* synteny problem *is to compute the minimum number of mutations, denoted by* $D(\mathcal{S}(n,k))$, *to transform* $\mathcal{S}$ *to the collection* $\{\{1\}, \{2\}, \ldots, \{n-1\}, \{n\}\}$.

The *dual* of $\mathcal{S}(n,k) = (\{a_1, a_2, \ldots, a_n\}; S_1, \ldots, S_k)$ is $\mathcal{S}'(k,n) = S'_1, \ldots, S'_n$, where S_i is a subset of $\{1, \ldots, k\}$ and $j \in S'_i \Leftrightarrow i \in S_j$. The goal in the dual problem is to produce the collection $\{1\}, \ldots, \{k\}$.

Proposition 2.2 follows from Proposition 2.1.

Proposition 2.2 *Let* $\mathcal{S}'(k,n)$ *be the dual of the synteny problem* $\mathcal{S}(n,k)$. *Then* $D(\mathcal{S}(n,k)) = D(\mathcal{S}'(k,n))$.

Henceforth, unless otherwise mentioned, by synteny problem we refer to the compact representation of the synteny problem. By Proposition 2.2, it is sufficient to consider an instance $\mathcal{S}(n,k)$ of the synteny problem with $n \geq k$, since otherwise we can solve the dual problem.

Given an instance $\mathcal{S}(n,k)$ of the synteny problem, the graph $G(\mathcal{S}(n,k))$ includes a vertex for each set of of $\mathcal{S}(n,k)$. Two vertices are connected by an edge if and only if their corresponding sets in $\mathcal{S}(n,k)$ have a non-empty intersection. If $G(\mathcal{S}(n,k))$ has $1 \leq p \leq n$ connected components, we will simply say that $\mathcal{S}(n,k)$ has

p components. If $G(\mathcal{S}(n,k))$ is connected we will say that $\mathcal{S}(n,k)$ is connected.

Proposition 2.3 *Let $\mathcal{S}(n,k)$ be a synteny instance with p components. Then, $D(\mathcal{S}(n,k)) \geq n-p$.*

Proof: Let $\sigma = (\sigma_1, \ldots, \sigma_m)$ be an optimal move sequence for $\mathcal{S}(n,k)$. Let $\mathcal{S}_0 = \mathcal{S}$ and let $\mathcal{S}_i$ be the synteny instance obtained after the first i moves. $\mathcal{S}_1$ has p components, $\mathcal{S}_m$ has n components, and $\mathcal{S}_{i+1}$ has at most one more component than $\mathcal{S}_i$. Thus $D(\mathcal{S}(n,k)) = m \geq n-p$. ■

3 NP-hardness of the Synteny Problem

In this section we prove the following theorem.

Theorem 3.1 *Computing the syntenic distance exactly is NP-hard.*

Our reduction will use two problems, the *largest balanced quasi-independent set* (LBQIS) problem and the *largest balanced independent set* (LBIS) problem for bipartite graphs, which are defined as follows:

PROBLEM: Largest balanced independent set (LBIS) problem.

INPUT: A connected bipartite graph $G = (U, V, E)$ with $|U| = |V| = n$ and positive integer k, $1 \leq k \leq n$.

QUESTION: Does there exists $U' \subseteq U$, $V' \subseteq V$, $|U'| = |V'| = k$, such that $(u', v') \notin E$ for any $u' \in U'$ and $v' \in V'$?

PROBLEM: Largest balanced quasi-independent set (LBQIS) problem.

INPUT: A connected bipartite graph $G = (U, V, E)$ with $|U| = |V| = n$ and positive integer k, $1 \leq k \leq n$.

QUESTION: Does there exists $U' \subseteq U$, $V' \subseteq V$, $|U'| = |V'| = k$, such that for some permutation $u'_1, u'_2, \ldots, u'_k$ of the vertices in U' and some permutation $v'_1, v'_2, \ldots, v'_k$ of the vertices in V', $(u'_i, v'_j) \notin E$ for any $1 \leq i \leq k$ and $i > j$?

The LBIS problem is known to be NP-complete[6, page 196]. [1]. Note that an LBIS of size k is also an LBQIS of size k for a graph G, but the converse is not necessarily true. First, we prove the following theorem.

[1] In [6, page 196] the largest *balanced complete bipartite subgraph* problem is shown to be NP-complete, which is same as the largest balanced bipartite independent set on the complement of the graph

Theorem 3.2 *Computing the syntenic distance is NP-hard if the LBQIS problem is NP-hard.*

The proof of Thorem 3.2 is as follows. Given an instance (G, k) of the LBQIS problem as mentioned above, we create an instance $\mathcal{S}(2n-k+1, 2n-k+1)$ of the synteny problem containing the following sets (assume that $U = \{u_1, u_2, \ldots, u_n\}$ and $V = \{v_1, v_2, \ldots, v_n\}$):

(a) $S = \{u_1, u_2, \ldots, u_n, a_1, a_2, \ldots, a_{n-k}, b\}$.

(b) $X_i = \{u_j \mid (u_j, v_i) \in E\} \cup \{b\}$ for $1 \leq i \leq n$.

(c) $Y_i = \{b\}$ for $1 \leq i \leq n-k$.

We refer to the elements $u_1, u_2, \ldots, u_n$ (resp. $a_1, a_2, \ldots, a_{n-k}$) as the u-elements (resp. a-elements) and the sets $X_1, X_2, \ldots, X_n$ (resp. $Y_1, Y_2, \ldots, Y_{n-k}$) as the X-sets (resp. Y-sets). Define $P_0 = S$ and $P_i = P_{i-1} - \{a_i\}$) for $1 \leq i \leq n-k$. Define $Q_k = P_{n-k}$ and $Q_{i-1} = Q_i - \{u'_i\}$ for $1 \leq i \leq k$. Finally, define $R_0 = Q_0$ and $R_i = R_{i-1} - \{u'_{k+i}\}$ for $1 \leq i \leq n-k$. Notice that $R_{n-k} = \{b\}$. The following lemma will complete the proof of Theorem 3.2.

Lemma 3.1 *G has a LBQIS of size k if and only if $D(S(2n-k+1, 2n-k+1)) = 2n-k$.*

The proof of the "only if" part of Lemma 3.1 is as follows. Assume G has a LBQIS (U', V') of size k. Let $U - U' = \{u'_{k+1}, u'_{k+2}, \ldots, u'_n\}$ and $V - V' = \{v'_{k+1}, v'_{k+2}, \ldots, v'_n\}$. An optimal syntenic sequence of $2n-k$ moves consists of the following moves:

- First, for $i = k+1, k+2, \ldots, n$, perform the translocation $(P_{i-k-1}, X_{v'_{k+1}}) \longrightarrow (\{a_{i-k}\}, P_{i-k})$. Notice that after the last move we have created the set $Q_0 = P_{n-k}$.
- Next, for $i = k, k-1, \ldots, 1$, perform the translocation $(Q_i, X_{v'_i}) \longrightarrow (\{u'_i\}, Q_{i-1})$. Notice that after the last move we have the sets $Q_0 = U - U'$, $Y_1, Y_2, \ldots, Y_{n-k}$ still remaining to be processed.
- For $i = 1, 2, \ldots, n-k$, perform the translocation $(R_{i-1}, Y_i) \longrightarrow (\{u'_{k+i}\}, R_i)$.

Before proceeding with the proof of the "if" part of Lemma 3.1, we need a few definitions and results.

Definition 3.1 *A connected instance $\mathcal{S}(n,n)$ of the synteny problem is* exact *if $D(\mathcal{S}(n,k)) = n-1$.*

Definition 3.2 *Let $\mathcal{S}(n,k)$ be an instance of synteny. A move on $\mathcal{S}$ is called a* splitting move *if it increases the number of components of $\mathcal{S}$ by one and it is called a* non-splitting move *otherwise.*

Definition 3.3 *Let $\mathcal{S}(n,n)$ be a connected instance of synteny. A splitting move on $\mathcal{S}(n,n)$ is called a balanced move if it creates two subproblems $\mathcal{S}_1(n_1,n_1)$ and $\mathcal{S}_2(n_2,n_2)$ for some n_1 and n_2.*

A splitting move must be a translocations or fission since fusions cannot increase the number of components. In the case of a translocation, it must operate on sets in the same connected component. A balanced move must be a translocation since fissions increase the total number of sets.

Lemma 3.2 *Every move in any optimal move sequence for an exact instance of synteny is a balanced move on a connected component.*

Proof: Let $\mathcal{S}(n,n)$ be an exact instance of synteny and let $\sigma = (\sigma_1,\ldots,\sigma_{n-1})$ be an optimal move sequence. Since $\mathcal{S}(n,n)$ is connected, each move of σ must be a splitting move. Assume σ_1 splits $\mathcal{S}(n,n)$ into two subproblems $\mathcal{S}_1(n_1,k_1)$ and $\mathcal{S}_2(n_2,k_2)$, where $n_1+n_2 = k_1+k_2 = n$. (Note these problems have disjoint alphabets.) Since each subsequent move must act on a connected component of the current problem, we can partition $(\sigma_2,\ldots,\sigma_{n_1})$ into two subsequences that solve, respectively, $\mathcal{S}_1$ and $\mathcal{S}_2$. By optimality of σ, these subsequences must be optimal move sequences so

$$D(\mathcal{S}_1(n_1,k_1))+D(\mathcal{S}_2(n_2,k_2)) = D(\mathcal{S}(n,n))-1 = n-2.$$

By Proposition 2.3 and the fact that $\mathcal{S}_i(n_i,k_i)$ is connected, $D(\mathcal{S}_i(n_i,k_i)) \geq \max(n_i,k_i)-1$. Thus, we get $n_i = k_i$. ■

Notice that the instance of the synteny problem created in Theorem 3.2 is exact. Proof of Lemma 3.1 is complete if we can prove the following lemma.

Lemma 3.3 *If $D(S(2n-k+1,2n-k+1)) = 2n-k$. then G has a LBQIS of size k.*

Proof: Let $\sigma = (\sigma_1,\sigma_2,\ldots,\sigma_{2n-k})$ be any optimal move sequence. By Lemma 3.2, every move is a balanced translocation.

First, we claim that, after a possible reordering of the indices of the a-elements, the move σ_j, for $1\leq j\leq n-k$, must be a translocation $(P_{j-1},X_\ell) \longrightarrow (\{a_j\},P_j)$ for some $\ell \in \{v_1,v_2,\ldots,v_n\}$. For contradiction, assume this is not the case and rearrange the indices of the a-elements, if necessary, so that $j\leq n-k$ is the least index such that σ_j violates the condition. Since σ_j must be a splitting translocation, it cannot translocate two X-sets, an X-set with a Y-set, or an Y-set with P_{j-1}. Hence, σ_j must translocate an X-set with P_{j-1}. Let $\sigma_j = (P_{j-1},X_\ell) \rightarrow (P',P'')$. Assume that $b\in P''$ (and, hence $b\notin P'$). Since σ_j must be a balanced move, P' must contain at most 1 a-element and at most 1 u-element. If P' does not contain any a-element, σ_j does not violate the condition. Otherwise, P' contains exactly one u-element u_t and no a-element. Then, modify σ_j such that the two elements u_t and a_j exchange their places in P' and P'' and then σ_j satisfies our condition.

Hence, after the move σ_{n-k}, we have the set $Q_k = P_{n-k} = \{u_1,u_2,\ldots,u_n,b\}$, the sets $Y_1,Y_2,\ldots,Y_k$ and some k X-sets, say $X_{v_1},x_{v_2},\ldots,X_{v_k}$. Then, by essentially the same reasoning as before, after a possible rearrangement of the indices, the move σ_j, for $n-k+1\leq j\leq n$, must be the translocation $(Q_{n-j+1},X_{v_{n-j+1}}) \longrightarrow (\{u_{n-j+1}\},Q_{n-j})$. This implies that $(u_i,v_j)\notin E$ for $1\leq i\leq k$ and $i>j$. ■

This completes the proof of Theorem 3.2. To complete our proof of Theorem 3.1, it is sufficient to prove the following theorem.

Theorem 3.3 *The LBQIS problem for bipartite graphs is NP-complete.*

We will reduce LBIS to the LBQIS problem. Assume that we are given an instance (G,k) of the LBIS problem, where $G=(U,V,E)$, $U=V=\{1,2,\ldots,n\}$. We create an instance (G',k') of the LBQIS problem, where $k'=k^2+k$, $G'=(U',V',E')$, $U'=V'=\{[i,j] \mid 1\leq i\leq k+1,\ 1\leq j\leq n\}$, and $E'=E_1\cup E_2$ consists of the following edges:

$$\begin{aligned} E_1 &= \{([i,k],[j,l]) \mid i<j\} \\ E_2 &= \{([i,k],[j,l]) \mid i\geq j,\ (k,l)\in E\} \end{aligned}$$

Intuitively, we use the amplification technique (see, for example, [12, page 428–429]) and "blow up" the graph G by using $k+1$ copies of it with some additional edges. We will prove the following lemma showing the correctness of our reduction.

Lemma 3.4 *G has an LBIS of size k if and only if G' has an LBQIS of size k'.*

The proof of the "only if" part of Lemma 3.4 is easy. Let $U_1\subseteq U$ and $V_1\subseteq V$ be an LBIS of G of size k. Assume, wlog, that $U_1=V_1=\{1,2,\ldots,k\}$. Let $U_1'=V_1'$ be the following permutation of a subset of k^2+k vertices of G':

$$[1,1],[1,2],\ldots,[1,k],[2,1],[2,2],\ldots,[2,k],\ldots,[k+1,1],$$
$$[k+1,2],\ldots,[k+1,k]$$

Then, U_1' and V_1' induces an LBQIS of size k' in G'.

The proof of the "if" part of Lemma 3.4 is more involved. Let σ_1 and σ_2 be a permutation of the vertices in U' and V', respectively, which realizes an LBQIS of size $k' = k^2 + k$. One crucial step in the proof is the following lemma which says that σ_1 and σ_2 can be decomposed in $k+1$ modules.

Lemma 3.5 (Rearrangement lemma) *There exist integers $p_1, p_2, \ldots, p_{k+1} \geq 0$, $p_1 + p_2 + \cdots + p_{k+1} = k^2 + k$, such that σ_1 and σ_2 may be assumed to be of the following forms:*

$$\begin{aligned}\sigma_1 &= (\ [1, x_1^1], \ldots, [1, x_1^{p_1}], [2, x_2^1], \ldots, [2, x_2^{p_2}], \\ &\quad \ldots, [k+1, x_{k+1}^1], \ldots, [k+1, x_{k+1}^{p_{k+1}}]\ ) \\ \sigma_2 &= (\ [1, y_1^1], \ldots, [1, y_1^{p_1}], [2, y_2^1], \ldots, [2, y_2^{p_2}], \\ &\quad \ldots, [k+1, y_{k+1}^1], \ldots, [k+1, y_{k+1}^{p_{k+1}}]\ )\end{aligned}$$

where $x_i^j, y_i^j \in \{1, 2, \ldots, n\}$ and $p_i = 0$ means that that sequence $[p_i, y_{p_i}^1], [p_i, y_{p_i}^2], \ldots, [p_i, y_{p_i}^{p_i}]$ is absent.

Proof: We may first assume without loss of generality that the permutations are

$$\begin{aligned}\sigma_1 &= (\ [1, x_1^1], \ldots, [1, x_1^{p_1}], [2, x_2^1], \ldots, [2, x_2^{p_2}], \\ &\quad \ldots, [k+1, x_{k+1}^1], \ldots, [k+1, x_{k+1}^{p_{k+1}}]\ ) \\ \sigma_2 &= (\ [1, y_1^1], \ldots, [1, y_1^{q_1}], [2, y_2^1], \ldots, [2, y_2^{q_2}], \\ &\quad \ldots, [k+1, y_{k+1}^1], \ldots, [k+1, y_{k+1}^{q_{k+1}}]\ )\end{aligned}$$

Then clearly $q_1 \leq p_1$, $q_1 + q_2 \leq p_1 + p_2$, ..., $q_1 + \ldots + q_k \leq p_1 + \ldots + p_k$ because of the edges in E_1. Since $\sum_{i=1}^{k+1} q_i = \sum_{i=1}^{k+1} p_i = k^2 + k$, $q_{k+1} \geq p_{k+1}$. Define $A = \{i \mid [j, i] \in \sigma_1\}$.

If $|A| \geq q_{k+1}$, then we can modify the suffix $[i, x_i^j], [i, x_i^{j+1}], \ldots, [k+1, x_{k+1}^{p_{k+1}}]$ of σ_1 with length q_{k+1} so that $x_i^j, ..., x_{k+1}^{p_{k+1}}$ are all distinct elements of A. Hence we can replace it with the sequence $[k+1, x_i^j], [k+1, x_i^{j+1}], \ldots, [k+1, x_{k+1}^{p_{k+1}}]$. The proof is then completed by induction.

Otherwise, suppose $|A| < q_{k+1}$. Then $q_{k+1} > p_1, \ldots, p_{k+1}$. In particular, $q_{k+1} > p_1$. Now we can modify the prefix $[1, y_1^1], [1, y_1^2], \ldots, [i, y_i^j]$ of σ_2 with length p_1 so that $y_1^1, \ldots, y_i^j$ are all distinct elements of $\{y_{k+1}^1, \cdots, y_{k+1}^{q_{k+1}}\}$. Hence we can replace it with the sequence $[1, y_1^1], [1, y_1^2], \ldots, [1, y_i^j]$. The proof is again completed by induction. ■

Now, to complete the proof of Lemma 3.4, note that we have the following two cases.

Case 1. There exists $i > j$ such that $p_i \geq k$ and $p_j \geq k$. Then, there is no edge between the vertices $[p_i, x_{p_i}^1], [p_i, x_{p_i}^2], \ldots, [p_i, x_{p_i}^{p_i}]$ and $[p_j, y_{p_j}^1], [p_j, y_{p_j}^2], \ldots, [p_j, y_{p_j}^{p_j}]$. Since $i > j$, by our construction of G', G must have an LBIS of size at least k consisting of the vertices $U_1 = \{x_{p_i}^1, x_{p_i}^2, \ldots, x_{p_i}^{p_i}\} \subseteq U$ and $V_1 = \{y_{p_j}^1, y_{p_j}^2, \ldots, y_{p_j}^{p_j}\} \subseteq V$.

Case 2. There are no such pair of indices as in Case 1. Let $t \geq 2$ be the largest integer such that $p_t \geq k$. Now, we have two cases:

(a) There is no such t. In this case,

$$p_1 = k^2 + k - (\sum_{i=2}^{k+1} p_i) \geq 2k$$

(b) Otherwise, since $p_i < k$ for $i \neq t$, $p_t \geq 2k$.

Hence, in either case, there exists an index j such that $p_j \geq 2k$. Then, the vertices

$$U_1 = \{x_j^{p_j}, x_j^{p_j - 1}, \ldots, x_j^{p_j - k + 1}\} \subseteq U$$

and

$$V_1 = \{y_j^1, y_j^2, \ldots, y_j^k\} \subseteq V$$

form an LBIS of size k for G. This completes the proof of Lemma 3.4.

4 A Simple Approximation Algorithm for the Synteny Problem

In this section, we describe a polynomial time approximation algorithm for the synteny problem with performance ratio 2.

Lemma 4.1 *Let $\mathcal{S}(n, k)$ be an instance of the synteny problem. Then, it is possible to approximate $D(\mathcal{S}(n, k))$ with a performance ratio of 2 in $O(nk)$ time.*

Proof: Assume, without loss of generality, that $n \geq k$. Assume $\mathcal{S}(n, k)$ has p components and n_i (resp. k_i) be the number of elements (resp. number of sets) in the i^{th} connected component of $G(S)$. Our simple fusion-fission algorithm is as follows. First, find the connected components of $G(S)$. Then, for each connected component, repeatedly use fusion until only one set is remaining and then repeatedly use fission to separate the remaining elements from the set. In all, we perform $\sum_{i=1}^{p}(n_i + k_i - 2) = n + k - 2p \leq 2n - 2p$ moves. By Proposition 2.3, $D(\mathcal{S}(n, k)) \geq n - p$, and hence a performance ratio of 2 is achieved. Note that the approximation algorithm uses no translocations and can easily be implemented in $O(nk)$ time using standard data structures (the running time is dominated by the step necessary to build the graph $G(S)$). ■

Remark 4.1 *The performance ratio 2 of the above heuristic is tight. Let the instance $S(n, n-1)$ consist of the*

$n-1$ sets $\{1\},\{1,2\},\{1,2,3\},\ldots,\{1,2,\ldots,n\}$. Then, $D(S(n,n))=n-1$, whereas our heuristic takes $2n-2$ moves. It is possible to use $n+k-3p$ moves instead of $n+k-2p$ moves if we replace the last fusion in our heuristic by a translocation which separates one of the elements from the rest.

5 Linear Synteny

The move sequences used in the NP-completeness proof and (without loss of generality) produced by the approximation algorithm have a particular form. There is a merging set Δ that is initially one of the input sets. The first $k-1$ moves are either fusions or very restricted translocations between Δ and an input set. The restriction on translocations is that only translocations that produce a singleton set $\{j\}$ such that j does not occur in any other set are allowed. The remaining moves are fissions on Δ that create singleton sets. In this section we study this restricted problem.

Let $\mathcal{S}(n,k)=\{S_1,\ldots,S_k\}$ be a connected instance of synteny and let π be a permutation of $[1,\ldots,k]$. The *linear move sequence* σ_π for $S(n,k)$ is defined as follows.

1. Let $\Delta_1 = S_{\pi_1}$.
2. For $i<k$
   (a) If there is $j\in\Delta_i\cup S_{\pi_{i+1}}$ that is not in $\cup_{\ell=i+2}^{k} S_{\pi_\ell}$ then choose the smallest such j and set $\sigma_i=(\Delta_i,S_{\pi_{i+1}})\longrightarrow(\Delta_{i+1},\{j\})$.
   (b) Otherwise $\sigma_i=(\Delta_i,S_{\pi_{i+1}})\longrightarrow\Delta_{i+1}$.
3. For $i=k,\ldots,k+||\Delta_k||-1$, let j be the smallest element in Δ_i and set $\sigma_i=\Delta_i\longrightarrow(\Delta_{i+1},\{j\})$.

If $\mathcal{S}(n,k)$ is not connected, a linear move sequence is a partition of the connected components of $\mathcal{S}(n,k)$ and a linear move sequence for each.[2] We let $\tilde{D}(\mathcal{S}(n,k))$ denote the length of the shortest linear move sequence for $\mathcal{S}(n,k)$.

Since the NP-Completeness proof uses linear move sequences, $\tilde{D}(\mathcal{S}(n,k))$ is hard to compute. Since the approximation algorithm (without loss of generality) produces a linear move sequence and $D(\mathcal{S}(n,k))\le\tilde{D}(\mathcal{S}(n,k))$, this algorithm gives a 2-approximation of $\tilde{D}(\mathcal{S}(n,k))$. Note as well that the optimal move sequence for the example given in Remark 4.1 is a linear move sequence so, as in the general case, the 2-approximation bound is tight.

It remains open whether one can approximate linear synteny by a factor better than 2, but this problem seems easier to analyze than the general synteny problem. The following theorem says, in fact, it suffices to improve the approximation bound for linear synteny since any such algorithm yields a better approximation for the general problem.

[2]If for example an input is $\{1\},\{2\},\ldots,\{n\}$ then no moves are required in either the original or linear versions of synteny.

Theorem 5.1 *If linear synteny can be approximated within a factor of c in polynomial time then for any $\epsilon>0$, for sufficiently large input, general synteny can be approximated within a factor of $c+\epsilon$ in polynomial time.*

This theorem follows directly from Lemma 5.1 since $D(\mathcal{S}(n,k))$ is $\Omega(n,k)$.

Lemma 5.1 *Let $\mathcal{S}(n,k)$ be an instance of synteny. Then*

$$\tilde{D}(\mathcal{S}(n,k))\le D(\mathcal{S}(n,k))+O(\log D(\mathcal{S}(n,k))).$$

To prove this lemma we need the following definitions.

Definition 5.1 *Let $\mathcal{S}(n,k)$ be an instance of synteny and let σ be an arbitrary move sequence for $\mathcal{S}(n,k)$. The* move digraph *$G_M(\mathcal{S},\sigma)$ contains a vertex for each move in σ. If σ_i creates a set S that is input to σ_j then G_M has an edge from σ_i to σ_j.*

We point out that G_M implies a partial order on the moves in σ and any consistent total order yields a move sequence for $\mathcal{S}(n,k)$. If σ is optimal, each total order yields an optimal move sequence for $\mathcal{S}(n,k)$. Note that G_M is directed, acyclic and each node has in-degree and out-degree at most 2. A directed graph is *weakly connected* if it is connected when its edges are considered in an undirected sense.

Definition 5.2 *Let G be a weakly connected, directed acyclic graph on n nodes. An $f(n)$ directed biseparator of G is a non-empty subset of edges A that partition G into two weakly connected components G_1 and G_2 such that each has between $f(n)$ and $n-f(n)$ nodes. Further, for every $<u,v>inA$, $u\in G_1$ and $v\in G_2$.*

The proof of Lemma 5.1 uses the following graph-theoretic lemma whose proof can be found in [8].

Lemma 5.2 *Let G be a weakly connected, directed, acyclic graph on n nodes where the in-degree and out-degree of each node are each at most 2. Then G has a $\frac{n}{4}$ directed biseparator.*

Proof of Lemma 5.1: Because of the form of Lemma 5.2, all logarithms in this proof are to the base 4/3. Let σ be an arbitrary move sequence for $\mathcal{S}(n,k)=\{S_1,\ldots,S_k\}$ of length d. To prove the bound it suffices to prove the case where σ consists solely of translocations. To see this, notice first that we may assume that fissions occur after all translocations which occur after all fusions. At the end of the fusion/translocation stages, the current sets $T_1, \ldots, T_\ell$ are disjoint. Create a new instance of synteny by renaming each j in the current set T_i as a_i in the original instance. Thus the fusion/translocation stages of σ solves the new problem. Suppose σ_π is a linear move sequence that solves the new problem and has length $\leq d' + \log d'$, where d' is the length of the fusion/translocation stage of σ. Then running σ_π on the original problem requires $d - d'$ additional fissions and has length $\leq d + \log d' \leq d + \log d$. So assume that σ consists of fusions followed by translocations. Consider the synteny instance $T_1, \ldots, T_\ell$ created by the last fusion. Suppose σ_π is a linear move sequence that solves this problem and has length $\leq d' + \log d'$, where d' is the number of translocations. For each T_{π_i}, let π'_i be an arbitrary ordering of the sets that were fused to create T_{π_i} and let $\pi' = \pi'_1 \cdot \pi'_2 \cdots \pi'_\ell$. Then $\sigma_{\pi'}$ has length at most $d + \log d' \leq d + \log d$.

Thus we'll assume σ consists solely of translocations. Notice in this case that $n = k$. If $d = 1$ then σ is already a linear move sequence so assume $d \geq 2$. First consider the case where $G_M(\mathcal{S}(n,k), \sigma)$ is connected. Note that G_M has d nodes.

By Lemma 5.2, there exists a $n/4$ directed biseparator A of $G_M(\mathcal{S}(n,k), \sigma)$. Let G_1 and G_2 be the two weakly connected components created by removing A. Assume G_i has d_i nodes; note $d_1 + d_2 = d$. We construct two new synteny instances as follows.

$\mathcal{S}_2$**:** Each edge e of A corresponds to a set T_e that is passed to a node of G_2. The instance $\mathcal{S}_2$ consist of these sets $T_e, e \in A$, plus any of the input sets of $\mathcal{S}(n,k)$ that are input to G_2. Notice that any move sequence implied by G_2 (i.e. consistent total order on its nodes) is a move sequence that solves $\mathcal{S}_2$. Since each move is a translocation, $\mathcal{S}_2$ has the same number of sets as elements; let n_2 be this number.

$\mathcal{S}_1$**:** Initially let $\mathcal{S}_1$ consist of the input sets of $\mathcal{S}(n,k)$ that are input to nodes of G_1. A move sequence implied by G_1 does not typically solve $\mathcal{S}_1$ because the sets $T_e, e \in A$, may not be disjoint singleton sets. To fix this, let us first rename an element j that occurs in the set S_ℓ of $\mathcal{S}_1$ as $[j, \ell]$. Carry the renaming through the moves of σ. Then for each T_e create a new *dummy* name a_e. For each $[j, \ell] \in T_e$, rename $[j, \ell]$ as a_e in $\mathcal{S}_1$. With this renaming a move sequence implied by G_1 solves $\mathcal{S}_1$. As above, $\mathcal{S}_1$ has the same number of sets as elements; let n_1 be this number. Let $W_1 = \{a_e, e \in A\}$.

Inductively assume that there is a linear move sequence σ_{i,π_i} for $\mathcal{S}_i$ of length $\leq d_i + g(d_i)$. Let π be the order on the sets of $\mathcal{S}$ induced by π_1 followed by π_2. We claim that σ_π has length at most $d + \max(g(d_1), g(d_2)) + 1$. The lemma follows since

$$g(n) \leq 1 + g(\frac{3n}{4}) \leq \log n.$$

For the accounting we'll modify σ_π slightly to create its singleton sets in a way we can count. In the following $\Delta_{1,j}$, $\Delta_{2,j}$ and Δ_j denote the current merging set of, respectively, σ_{π_1}, σ_{π_2} and σ_π before their j-th move. During the first $n_1 - 1$ merges σ_π matches the moves of σ_{π_1}. By this we mean that when σ_{π_1} creates a singleton $\{j\}$, σ_π creates the same singleton provided $j \notin W_1$. If $j \in W_1$ then σ_π simply performs a fusion at that step. The n_1-th move of σ_π is a fusion of Δ_{n_1} with the first set in the ordering for $\mathcal{S}_2$ unless this first set is a dummy set T_e, in which case this move is skipped. In the remaining moves it matches the first $n_2 - 1$ moves of σ_{π_2}, except those involving sets $T_e, e \in A$, during which σ_π makes no move. Let W_2 be the set of singletons created by σ_{π_2} in translocations with the sets $T_e, e \in A$. Notice that $\Delta_{n_1} = \Delta_{1,n_1} \cup_{e \in A} T_e \setminus W_1$ and $\Delta_n = \Delta_{1,n_1} \cup \Delta_{2,n_2} \cup W_2 \setminus W_1$. Let $r_1 = ||\Delta_{1,n_1} \setminus W_1||$ and $r_2 = ||\Delta_{2,n_2}||$.

As described σ_π is somewhat wasteful. No element of $\Delta_{1,n_1} \setminus W_1$ exists in an $\mathcal{S}_2$ set. If there are f fusions in the last $n - n_1$ moves of σ_π, we can replace $\min(r_1, f)$ of them by translocations creating singletons from $\Delta_{1,n_1} \setminus W_1$. The sequence σ_{π_2} performs $r_2 - 1$ fusions of which $||A|| - ||W_2||$ involve sets $T_e, e \in A$. The remaining have corresponding fusions in σ_π. In addition, move n_1 of σ_π is a fusion. Since Δ_{1,n_1} and Δ_{n_2} are disjoint, the modified move sequence ends with a set Δ_n where

$$\begin{aligned} ||\Delta_n|| &= r_1 + r_2 - \min(r_1, r_2 - ||A|| + ||W_2||) \\ &= \max(r_1, r_2 - ||A|| + ||W_2||) + ||A|| - ||W_2|| \end{aligned}$$

An additional $||\Delta_n|| - 1$ fissions are needed. So the function $g(d)$ must satisfy

$$\begin{aligned} d + g(d) &\geq n + \max(r_1, r_2 - ||A|| + ||W_2||) + ||A|| \\ &\quad -||W_2|| - 2 \\ &= n_1 + n_2 + \max(r_1, r_2 - ||A|| + ||W_2||) \\ &\quad -||W_2|| - 2 \end{aligned}$$

since $n = n_1 + n_2 - ||A||$. If $r_1 \geq r_2 - ||A|| + ||W_2||$ then using the facts that $n_1 + r_1 - 2 \leq n_1 + ||\Delta_{1,n_1}|| - 2 = d_1 + g(d_1)$ and $n_2 \leq d_2 + 1$ we get

$$d + g(d) \geq d_1 + d_2 + g(d_1) + 1 = d + g(d_1) + 1.$$

If $r_2 - ||A|| + ||W_2|| > r_1$ then using the facts that $n_1 \leq d_1 + 1$ and $n_2 + r_2 - 2 = d_2 + g(g_2)$ we get that

$$d + d(g) \geq d_1 + d_2 + g(d_2) + 1 = d + g(d_2) + 1.$$

Thus $d+g(d) \geq d+\max(g(d_1), g(d_2))+1$ and the choice of $g(d) = \log d$ suffices. ■

6 Optimal Syntenic Sequence When the Distance is Bounded

In practice, it may be the case sometimes, that the synteny distance between two genomes is bounded, and one is interested in finding the optimal sequence of synteny moves between the two genomes. The following theorem states our result in this regard. Notice that the algorithm mentioned below takes polynomial time provided $d = O(\sqrt{\log(nk)})$.

Theorem 6.1 *Let $\mathcal{S}(n,k)$ be an instance of the synteny problem with $D(\mathcal{S}(n,k)) \leq d$. Then, an optimal sequence of synteny moves for $\mathcal{S}(n,k)$ can be computed in $O(nk \cdot 2^{O(d^2)})$ time.*

We need to prove a few results before proving Theorem 6.1. As usual, we may assume $n \geq k$ without any loss of generality.

Lemma 6.1 *If $D(\mathcal{S}(n,k)) \leq \frac{n}{3} - 1$, then $\mathcal{S}(n,k)$ has one connected component containing just the set $\{a_i\}$ for some $1 \leq i \leq n$.*

Proof: Assume $G(S)$ has $p \geq 1$ connected components $C_1, \ldots, C_p$ and let $n_i \geq 1$ (resp. $k_i \geq 1$) be the number of elements (resp. number of sets) in C_i. By assumption, $n_i + k_i \geq 3$ for all i. Thus, $n + k = \sum_{i=1}^{p}(n_i + k_i) \geq 3p$, implying $p \leq \frac{2n}{3}$. But, by Proposition 2.3, $p \geq n - D(\mathcal{S}(n,k)) \geq \frac{2n}{3} + 1$. But, this is a contradiction! ■

Lemma 6.2 *Assume that a given instance $\mathcal{S}(n,k)$ has a connected component containing only the set $\{a_i\}$. Let $T(n-1,k-1)$ be the instance obtained by removing the element a_i and the set $\{a_i\}$ from $\mathcal{S}(n,k)$. Then, $D(\mathcal{S}(n,k)) = D(T(n-1,k-1))$.*

Proof: Obviously, $D(\mathcal{S}(n,k)) \leq D(T(n-1,k-1))$. Conversely, assume that an optimal sequence for $\mathcal{S}(n,k)$ translocates the set $\{a_i\}$ with some other set. We do not perform this translocation, but proceed with the remaining moves assuming that the element a_i is carried to subsequent sets. Finally, we must have a translocation or fission separating a_i from other elements. If it is a translocation, we replace it by a fusion, whereas if it is a fission, we do nothing. So, in fact we save moves by not translocating a_i. Hence, $D(T(n-1,k-1)) \leq D(\mathcal{S}(n,k))$. ■

We now proceed with the proof of Theorem 6.1. By Proposition 6.2, given an instance $\mathcal{S}(n,k)$, we can derive an instance $T(n',k')$ such that $n' \leq n$, $k' \leq k$, $n' \geq k'$ and $D(\mathcal{S}(n,k)) = D(T(n',k')) \geq \frac{n'}{3}$. Hence, it is sufficient to prove the theorem when $n \geq k$ and $d \geq \frac{n}{3}$. By Lemma 2.2, we know that it is sufficient to do at most α_1 fusions first, at most α_2 translocations next and at most α_3 fissions at the end, for every choice of $\alpha_1, \alpha_2, \alpha_3 \geq 0$ such that $\alpha_1 + \alpha_2 + \alpha_3 \leq d$.

Clearly, there are at most $\binom{d+3}{d} < 2^{3d/2}$ choices for the α_i's. For every such choice, we count the total number of possible alternative moves we may have to perform. First, we count the total number f of fusions we need to look at. Clearly,

$$f \leq \Pi_{j=1}^{\alpha_1}\binom{n-j+1}{2} < \binom{n}{2}^{\alpha_1} = 2^{O(d\log d)}$$

Next, we count the total number t of translocations we need to look at. Since there are at most $2^n - 1$ ways to translocate two sets,

$$t \leq \left[\binom{n}{2}\right]^d \cdot 2^{n\alpha_1} < \binom{n}{2}^{\alpha_1} \cdot 2^{n\alpha_1} = 2^{O(d^2)}$$

Since all the fissions are done at the end, there is only one unique way of doing them. Hence, overall our algorithm takes $O(nk \cdot 2^{O(d^2)})$ time.

7 Synteny and the Balanced Bipartite Independent Set Problem

In Section 3 we used the balanced bipartite independent set (LBIS) problem to show that computing the syntenic distance exactly is NP-hard. In this section we investigate the relationship between approximation of the syntenic distance with a performance ratio better than 2 and a good approximation of the LBIS problem. We omit the complicated proof of the following theorem.

Lemma 7.1 *Assume that the LBIS problem for bipartite graphs can be approximated, in polynomial time, with a performance ratio of ϵ for some constant $\epsilon > 1$. Then, the syntenic distance can be approximated, in polynomial time, with a constant performance ratio $\theta < 2$.*

8 The Median Problem

The median problem arises in connection with the phylogenetic inference problem[5] and defined as follows. Given three genomes $\mathcal{G}_1$, $\mathcal{G}_2$ and $\mathcal{G}_3$, we are required to construct a genome $\mathcal{G}$ such that the *median distance* $\alpha_{\mathcal{G}} = \sum_{i=1}^{3} D(\mathcal{G}, \mathcal{G}_i)$ is minimized. Without any additional constraints, this problem is trivial, since we can take $\mathcal{G}$ to be empty (and then $\alpha_{\mathcal{G}} = 0$). In the context of syntenic distance, any one of the following three constraints seem relevant:

(c1) $\mathcal{G}$ must contain all genes present in all the three given genomes.

(c2) $\mathcal{G}$ must contain all genes present in at least two of the three given genomes.

(c3) $\mathcal{G}$ must contain all genes present in at least one of the three given genomes.

Lemma 8.1 *The median problem is NP-hard with any one of the three constraints* **(c1), (c2) or (c3)**.

Proof: We reduce the synteny problem to this problem. Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be the two genomes of any instance of the synteny problem and let $d = D(\mathcal{G}_1, \mathcal{G}_2) > 0$. The NP-hardness reduction of Section 3 shows that we may assume that both $\mathcal{G}_1$ and $\mathcal{G}_2$ contain the same set of genes. Let $\mathcal{G}_1$, $\mathcal{G}_1$ and $\mathcal{G}_2$ be the three genomes for the corresponding median problem. Assume that $\mathcal{G}$ is the solution of the centroid problem (under any one of the constraints). If $\mathcal{G} \neq \mathcal{G}_1$, then $\alpha_{\mathcal{G}} \geq d + D(\mathcal{G}, \mathcal{G}_1) > d$; but if $\mathcal{G} = \mathcal{G}_1$, then $\alpha_{\mathcal{G}} = d$. Hence, $\alpha_{\mathcal{G}}$ is precisely the syntenic distance between $\mathcal{G}_1$ and $\mathcal{G}_2$ and determining $\alpha_{\mathcal{G}}$ determines d also. ■

Lemma 8.2 *We can approximate the median problem in polynomial time (under any one of the constraints (c1), (c2) or (c3)) with ratio* $4 + \epsilon$ *for any constant* ϵ.

9 Conclusion

In this paper, we have proved several results concerning the complexities of efficient exact and approximate computations of the syntenic distance between genomes. The NP-hardness proof for computing the synteny distance uses a cascade of reductions from the LBIS problem for bipartite graphs. We also showed that any constant approximation of the LBIS problem leads to an approximation of the synteny distance with ratio better than 2.

The following problems still remain open:

- Can we approximate the synteny distance in polynomial time with a ratio better than 2? Does a PTAS for this problem exist? Our results seem to indicate that the answer to this question is closely related to finding a large "balanced" independent set in bipartite graphs.
- When the synteny distance is bounded, can we improve the time complexity further to compute an optimal move sequence?

References

[1] V. Bafna and P. Pevzner. Genome Rearrangements and Sorting by Reversals. In *34th IEEE Symp. on Foundations of Computer Science*, 1993, pp. 148-157.

[2] V. Bafna and P. Pevzner. Sorting by Reversals: Genome Rearrangements in Plant Organelles and Evolutionary History of X Chromosome. *Mol. Biol. and Evol.*, 12, 1995, pp. 239-246.

[3] V. Bafna and P. Pevzner. Sorting by Transpositions. In *Proc. of 6th Ann. ACM-SIAM Symp. on Discrete Algorithms*, 1995, pp. 614-623.

[4] B. DasGupta, T. Jiang, S. Kannan, M. Li and Z. Sweedyk. On the Complexity and Approximation of Syntenic Distance. University of Pennsylvania Technical Report MS-CIS-96-21.

[5] V. Ferretti, J.H. Nadeau and D. Sankoff. Original Synteny. In *Proc. of 7th Ann. Symp. on Combinatorial Pattern Matching,* 1996, pp. 159–167.

[6] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., 1979.

[7] S. Hannenhalli and P. Pevzner. Transforming Cabbage into Turnip (polynomial algorithm for sorting signed permutations by reversals). In *Proc. of 27th Ann. ACM Symp. on Theory of Computing*, 1995, pp. 178-189.

[8] S. Kannan and Z Sweedyk. A Separator Theorem for Directed Acyclic Graphs. University of Pennsylvania Tehnical Report MS-CIS-96-20.

[9] J. Kececioglu and R. Ravi. Of Mice and Men: Evolutionary Distances Between Genomes under Translocation. In *Proc. of 6th Ann. ACM-SIAM Symp. on Discrete Algorithms*, 1995, pp. 604-613.

[10] J. Kececioglu and D. Sankoff. Exact and Approximation Algorithms for the Inversion Distance between Two Permutations. In *Proc. of 4th Ann. Symp. on Combinatorial Pattern Matching,* Lecture Notes in Computer Science 684, Springer Verlag, 1993, pp. 87-105.

[11] J. Kececioglu and D. Sankoff. Efficient Bounds for Oriented Chromosome Inversion Distance. In In *Proc. of 5th Ann. Symp. on Combinatorial Pattern Matching,* Lecture Notes in Computer Science 807, Springer Verlag, 1994, pp. 307-325.

[12] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1982.