Chapter 12
Set Systems and the Syntenic Distance

Yi-Hean Lin

1/7/2013

12.1 Introduction

A genome π is a partition of the set of genes into h non-empty chromosomes,

12.1 Introduction

A genome π is a partition of the set of genes into h non-empty chromosomes, that is, $\pi = \{C_1, \ C_2, \ldots, C_h.\}$

.

12.1 Introduction

A genome π is a partition of the set of genes into h non-empty chromosomes, that is, $\pi = \{C_1, C_2, \ldots, C_h.\}$
We will refer to the number h of chromosomes in π as its size.

.

12.1 Introduction

A genome π is a partition of the set of genes into h non-empty chromosomes, that is, $\pi = \{C_1, C_2, \ldots, C_h.\}$
We will refer to the number h of chromosomes in π as its size.
$\pi_1 = \{\{a, b, e, f, h\}, \{c, g\}, \{d, i, j\}\}$,
$\pi_2 = \{\{a, b, c, d\}, \{e, f\}, \{g, h, i\}, \{j\}\}$.

A genome mutates by one of three simple moves:
fusion, fission, and (reciprocal) translocation.

A genome mutates by one of three simple moves:
fusion, fission, and (reciprocal) translocation.

Definition (Syntenic Model)

Let C_i, C_j, C_p, C_q be sets of chromosomes, such that (1)at most one is empty, and (2) $C_i \bigcup C_j = C_p \bigcup C_q$.

A genome mutates by one of three simple moves:
fusion, fission, and (reciprocal) translocation.

Definition (Syntenic Model)

Let C_i, C_j, C_p, C_q be sets of chromosomes, such that (1)at most one is empty, and (2) $C_i \bigcup C_j = C_p \bigcup C_q$.

(a) *If C_q is empty then $\{C_i, C_j\} \to \{C_p\}$ is called a fusion of C_i and C_j.*

A genome mutates by one of three simple moves:
fusion, fission, and (reciprocal) translocation.

Definition (Syntenic Model)

Let C_i, C_j, C_p, C_q be sets of chromosomes, such that (1)at most one is empty, and (2) $C_i \bigcup C_j = C_p \bigcup C_q$.

(a) *If C_q is empty then $\{C_i,\ C_j\} \to \{C_p\}$ is called a fusion of C_i and C_j.*
$\{\{a, e, f\},\ \{b, h\}\} \to \{\{a, b, e, f, h\}\}$.

A genome mutates by one of three simple moves:
fusion, fission, and (reciprocal) translocation.

Definition (Syntenic Model)

Let C_i, C_j, C_p, C_q be sets of chromosomes, such that (1)at most one is empty, and (2) $C_i \bigcup C_j = C_p \bigcup C_q$.

(a) *If C_q is empty then $\{C_i,\ C_j\} \to \{C_p\}$ is called a fusion of C_i and C_j.*
$\{\{a, e, f\},\ \{b, h\}\} \to \{\{a, b, e, f, h\}\}$.

(b) *If C_j is empty then $\{C_i\} \to \{C_p,\ C_q\}$ is called a fission of C_i.*

A genome mutates by one of three simple moves:
fusion, fission, and (reciprocal) translocation.

Definition (Syntenic Model)

Let C_i, C_j, C_p, C_q be sets of chromosomes, such that (1)at most one is empty, and (2) $C_i \bigcup C_j = C_p \bigcup C_q$.

(a) *If C_q is empty then $\{C_i,\ C_j\} \to \{C_p\}$ is called a fusion of C_i and C_j.*
$\{\{a, e, f\},\ \{b, h\}\} \to \{\{a, b, e, f, h\}\}$.

(b) *If C_j is empty then $\{C_i\} \to \{C_p,\ C_q\}$ is called a fission of C_i.*
$\{\{a, b, e, f, h\}\} \to \{\{a, e, f\},\ \{b, h\}\}$.

A genome mutates by one of three simple moves:
fusion, fission, and (reciprocal) translocation.

Definition (Syntenic Model)

Let C_i, C_j, C_p, C_q be sets of chromosomes, such that (1)at most one is empty, and (2) $C_i \bigcup C_j = C_p \bigcup C_q$.

(a) *If C_q is empty then $\{C_i,\ C_j\} \to \{C_p\}$ is called a fusion of C_i and C_j.*
$\{\{a, e, f\},\ \{b, h\}\} \to \{\{a, b, e, f, h\}\}$.

(b) *If C_j is empty then $\{C_i\} \to \{C_p,\ C_q\}$ is called a fission of C_i.*
$\{\{a, b, e, f, h\}\} \to \{\{a, e, f\},\ \{b, h\}\}$.

(c) *If C_i, C_j, C_p, C_q are non-empty then $\{C_i,\ C_j\} \to \{C_p,\ C_q\}$ is called a translocation of C_i and C_j.*

A genome mutates by one of three simple moves:
fusion, fission, and (reciprocal) translocation.

Definition (Syntenic Model)

Let C_i, C_j, C_p, C_q be sets of chromosomes, such that (1)at most one is empty, and (2) $C_i \bigcup C_j = C_p \bigcup C_q$.

(a) *If C_q is empty then $\{C_i,\ C_j\} \to \{C_p\}$ is called a fusion of C_i and C_j.*
$\{\{a, e, f\},\ \{b, h\}\} \to \{\{a, b, e, f, h\}\}$.

(b) *If C_j is empty then $\{C_i\} \to \{C_p,\ C_q\}$ is called a fission of C_i.*
$\{\{a, b, e, f, h\}\} \to \{\{a, e, f\},\ \{b, h\}\}$.

(c) *If C_i, C_j, C_p, C_q are non-empty then $\{C_i,\ C_j\} \to \{C_p,\ C_q\}$ is called a translocation of C_i and C_j.*
$\{\{a, b, e, f, h\},\ \{d, i, j\}\} \to \{\{a, e, h, i, j\},\ \{b, d, f\}\}$.

Definition (Syntenic Distance)

The syntenic distance between two genomes π_1 and π_2, denoted by $syd(\pi_1, \pi_2)$,

Definition (Syntenic Distance)

The syntenic distance between two genomes π_1 and π_2, denoted by $syd(\pi_1, \pi_2)$, is the minimum number of fusions, fissions, and translocations needed to transform π_1 and π_2.

Definition (Syntenic Distance)

The syntenic distance between two genomes π_1 and π_2, denoted by $syd(\pi_1, \pi_2)$, is the minimum number of fusions, fissions, and translocations needed to transform π_1 and π_2.

Example

$\pi_1 = \{\{a, b, e, f, h\}, \{c, g\}, \{d, i, j\}\}$,
$\pi_2 = \{\{a, b, c, d\}, \{e, f\}, \{g, h, i\}, \{j\}\}$
$syd(\pi_1, \pi_2)=3$

Definition (Syntenic Distance)

The syntenic distance between two genomes π_1 and π_2, denoted by $syd(\pi_1, \pi_2)$, is the minimum number of fusions, fissions, and translocations needed to transform π_1 and π_2.

Example

$\pi_1 = \{\{a, b, e, f, h\}, \{c, g\}, \{d, i, j\}\}$,
$\pi_2 = \{\{a, b, c, d\}, \{e, f\}, \{g, h, i\}, \{j\}\}$
$syd(\pi_1, \pi_2)=3$

Proposition

$syd(\pi_1, \pi_2) = syd(\pi_2, \pi_1)$

12.2 Compact Representation

Definition ((12.2)Compact Representation)

The compact representation of a genome π_1 with respect to a genome π_2 is

12.2 Compact Representation

Definition ((12.2)Compact Representation)

The compact representation of a genome π_1 with respect to a genome π_2 is obtained by replacing each C_i' in π_2 with its index i, and each C_j in π_1 with the set of indices of the chromosomes on which each gene of C_j lies in π_2.

12.2 Compact Representation

Definition ((12.2)Compact Representation)

The compact representation of a genome π_1 with respect to a genome π_2 is obtained by replacing each C_i' in π_2 with its index i, and each C_j in π_1 with the set of indices of the chromosomes on which each gene of C_j lies in π_2.

Example

We consider the following two genomes

$\pi_1 = \{\{a, b, e, f, h\},\ \{c, g\},\ \{d, i, j\}\},$

$\pi_2 = \{\{a, b, c, d\},\ \{e, f\},\ \{g, h, i\},\ \{j\}\}$

Then the compact representation of π_1 with respect to π_2 consists of

12.2 Compact Representation

Definition ((12.2)Compact Representation)

The compact representation of a genome π_1 with respect to a genome π_2 is obtained by replacing each C_i' in π_2 with its index i, and each C_j in π_1 with the set of indices of the chromosomes on which each gene of C_j lies in π_2.

Example

We consider the following two genomes

$\pi_1 = \{\{a, b, e, f, h\},\ \{c, g\},\ \{d, i, j\}\}$,

$\pi_2 = \{\{a, b, c, d\},\ \{e, f\},\ \{g, h, i\},\ \{j\}\}$

Then the compact representation of π_1 with respect to π_2 consists of

$\pi_1 = \{\{1, 2, 3\},\ \{1, 3\},\ \{1, 3, 4\}\}$,

$\pi_2 = \{\{1\},\ \{2\},\ \{3\},\ \{4\}\}$.

Note that, in the compact representation, fissions or translocations can possibly create duplicates.

Note that, in the compact representation, fissions or translocations can possibly create duplicates.

Example

$$\{\{a, b, c, d\},\ \{e, f\},\ \{g, h, i\},\ \{j\}\} = \{\{1\},\ \{2\},\ \{3\},\ \{4\}\}$$

Note that, in the compact representation, fissions or translocations can possibly create duplicates.

Example

$$\{\{a,b,c,d\},\ \{e,f\},\ \{g,h,i\},\ \{j\}\} = \{\{1\},\ \{2\},\ \{3\},\ \{4\}\}$$

(fissions) $\{\{a,b,e,f,h\}\} \to \{\{a,e,f\},\ \{b,h\}\}$

$\Rightarrow \{1,2,3\} \to \{\{1,2\},\ \{1,3\}\}.$

Note that, in the compact representation, fissions or translocations can possibly create duplicates.

Example

$$\{\{a,b,c,d\},\ \{e,f\},\ \{g,h,i\},\ \{j\}\} = \{\{1\},\ \{2\},\ \{3\},\ \{4\}\}$$

(fissions) $\{\{a,b,e,f,h\}\} \to \{\{a,e,f\},\ \{b,h\}\}$

$\Rightarrow \{1,2,3\} \to \{\{1,2\},\ \{1,3\}\}.$

(translocations) $\{\{a,b,e,f,h\},\ \{d,i,j\}\} \to \{\{a,e,h,i,j\},\ \{b,d,f,i\}\}$

$\Rightarrow \{1,2,3\},\ \{1,3,4\} \to \{\{1,2,3,4\},\ \{1,2,3\}\}.$

According to definition 12.2 there are two ways to construct the compact representation of two genomes π_1 and π_2 of respective sizes h and k $(h \geq k)$:

According to definition 12.2 there are two ways to construct the compact representation of two genomes π_1 and π_2 of respective sizes h and k $(h \geq k)$:
either π_1 becomes $\pi_1' = \{\{1\}, \{2\}, \ldots, \{h\}\}$ and π_2 is modified accordingly, or

According to definition 12.2 there are two ways to construct the compact representation of two genomes π_1 and π_2 of respective sizes h and k $(h \geq k)$:
either π_1 becomes $\pi_1' = \{\{1\}, \{2\}, \ldots, \{h\}\}$ and π_2 is modified accordingly, or
π_2 becomes $\pi_2' = \{\{1\}, \{2\}, \ldots, \{k\}\}$ and π_1 is modified accordingly

According to definition 12.2 there are two ways to construct the compact representation of two genomes π_1 and π_2 of respective sizes h and k $(h \geq k)$:
either π_1 becomes $\pi_1' = \{\{1\}, \{2\}, \ldots, \{h\}\}$ and π_2 is modified accordingly, or
π_2 becomes $\pi_2' = \{\{1\}, \{2\}, \ldots, \{k\}\}$ and π_1 is modified accordingly

Definition (Syntenic Problem)

Given a collection $S(h, k)$ of k subsets $S_1, \ldots, S_k$ of $\{1, \ldots, h\}$,

According to definition 12.2 there are two ways to construct the compact representation of two genomes π_1 and π_2 of respective sizes h and k $(h \geq k)$:
either π_1 becomes $\pi_1' = \{\{1\}, \{2\}, \ldots, \{h\}\}$ and π_2 is modified accordingly, or
π_2 becomes $\pi_2' = \{\{1\}, \{2\}, \ldots, \{k\}\}$ and π_1 is modified accordingly

Definition (Syntenic Problem)

Given a collection $S(h, k)$ of k subsets $S_1, \ldots, S_k$ of $\{1, \ldots, h\}$, the stntenic problem seeks to compute the minimum number of fusions, fissions, and translocations needed to transform $S(h, k)$ into the collection $\{\{1\}, \{2\}, \ldots, \{h\}\}$.

According to definition 12.2 there are two ways to construct the compact representation of two genomes π_1 and π_2 of respective sizes h and k $(h \geq k)$:
either π_1 becomes $\pi_1' = \{\{1\}, \{2\}, \ldots, \{h\}\}$ and π_2 is modified accordingly, or
π_2 becomes $\pi_2' = \{\{1\}, \{2\}, \ldots, \{k\}\}$ and π_1 is modified accordingly

Definition (Syntenic Problem)

Given a collection $S(h, k)$ of k subsets $S_1, \ldots, S_k$ of $\{1, \ldots, h\}$, the stntenic problem seeks to compute the minimum number of fusions, fissions, and translocations needed to transform $S(h, k)$ into the collection $\{\{1\}, \{2\}, \ldots, \{h\}\}$.
This number is denoted by $syd(S(h, k))$ and is called the syntenic distance of $S(h, k)$.

Finally, DasGupta et al.[1] showed the existence of a canonical form for the syntenic distance.

Finally, DasGupta et al.[1] showed the existence of a canonical form for the syntenic distance.

Lemma ([1])

There always exists an optimal sequence of moves between two sets in which all fusions occur first, .

Finally, DasGupta et al.[1] showed the existence of a canonical form for the syntenic distance.

Lemma ([1])

There always exists an optimal sequence of moves between two sets in which all fusions occur first, then all translocations, .

Finally, DasGupta et al.[1] showed the existence of a canonical form for the syntenic distance.

Lemma ([1])

There always exists an optimal sequence of moves between two sets in which all fusions occur first, then all translocations, and finally all fissions.

12.3 Lower Bounds

Definition (Synteny Graph)

The synteny graph of an instance $S(h, k)$, denoted by SG(S(h,k)),

12.3 Lower Bounds

Definition (Synteny Graph)

The synteny graph of an instance $S(h, k)$, denoted by SG(S(h,k)), contains a vertex for each set of $S(h, k)$.

12.3 Lower Bounds

Definition (Synteny Graph)

The synteny graph of an instance $S(h, k)$, denoted by SG(S(h,k)), contains a vertex for each set of $S(h, k)$. and two distinct vertices are joined by an edge if the corresponding sets have a nonempty intersection.

12.3 Lower Bounds

Definition (Synteny Graph)

The synteny graph of an instance $S(h,k)$, denoted by SG(S(h,k)), contains a vertex for each set of $S(h,k)$. and two distinct vertices are joined by an edge if the corresponding sets have a nonempty intersection.

Example

$S(h,k) = \{\{1,2,3\},\ \{1,3\},\ \{1,3,4\}\}$.

12.3 Lower Bounds

Definition (Synteny Graph)

The synteny graph of an instance $S(h,k)$, denoted by SG(S(h,k)), contains a vertex for each set of $S(h,k)$. and two distinct vertices are joined by an edge if the corresponding sets have a nonempty intersection.

Example

$S(h,k) = \{\{1,2,3\},\ \{1,3\},\ \{1,3,4\}\}$.

Proposition ((12.3)[1])

For any synteny instance $S(h,k)$, we have

12.3 Lower Bounds

Definition (Synteny Graph)

The synteny graph of an instance $S(h,k)$, denoted by SG(S(h,k)), contains a vertex for each set of $S(h,k)$. and two distinct vertices are joined by an edge if the corresponding sets have a nonempty intersection.

Example

$S(h,k) = \{\{1,2,3\},\ \{1,3\},\ \{1,3,4\}\}$.

Proposition ((12.3)[1])

For any synteny instance $S(h,k)$, we have
$syd(S(h,k)) \geq h - p$,
where p is the number of connected components in $SG(S(h,k))$.

12.3 Lower Bounds

Definition (Synteny Graph)

The synteny graph of an instance $S(h,k)$, denoted by SG(S(h,k)), contains a vertex for each set of $S(h,k)$. and two distinct vertices are joined by an edge if the corresponding sets have a nonempty intersection.

Example

$S(h,k) = \{\{1,2,3\},\ \{1,3\},\ \{1,3,4\}\}$.

Proposition ((12.3)[1])

For any synteny instance $S(h,k)$, we have
$syd(S(h,k)) \geq h - p$,
where p is the number of connected components in $SG(S(h,k))$.

Proof. (idea) Any given move, the number of components in the synteny graph cannot increase by more than 1.

Theorem ([3])

For any instance of synteny $S(h, k)$, we have

Page 15

Theorem ([3])

For any instance of synteny $S(h, k)$, *we have*
$syd(S(h,k)) + log_{4/3}(syd(S(h,k))) \geq$
$h - 1 + max_{1 \leq c < k}\{c - |\{g | count(g) \leq c + 1\}|\}$,

Theorem ([3])

For any instance of synteny $S(h,k)$*, we have*
$syd(S(h,k)) + log_{4/3}(syd(S(h,k))) \geq$
$h - 1 + max_{1 \leq c < k}\{c - |\{g | count(g) \leq c + 1\}|\}$,
where $count(g)$ *is the number of sets of* $S(h,k)$ *in which* g *appears, for any gene* g*.*

Page 15

12.4 Diameter

Theorem (236,300)

For any $h \geq k \geq 4$, the syntenic diameter is $h + k - 4$

Page 15

12.4 Diameter

Theorem (236,300)

For any $h \geq k \geq 4$, the syntenic diameter is $h + k - 4$

It is relatively easy to see that the diameter is tight,

12.4 Diameter

Theorem (236,300)

For any $h \geq k \geq 4$, the syntenic diameter is $h + k - 4$

It is relatively easy to see that the diameter is tight, for example, by $S(h, k) = \{\{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \ldots, \{1, 2, \ldots, h\}\}$.
$k - 4$ times fusions: $\{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}$.

12.4 Diameter

Theorem (236,300)

For any $h \geq k \geq 4$, the syntenic diameter is $h + k - 4$

It is relatively easy to see that the diameter is tight, for example, by $S(h, k) = \{\{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \ldots, \{1, 2, \ldots, h\}\}$.
$k - 4$ times fusions: $\{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}$.
2 times translocations: $\{1, 2\}, \{1, 2\}, \{3, 4, \ldots, h\}, \{3, 4, \ldots, h\}$.

12.4 Diameter

Theorem (236,300)

For any $h \geq k \geq 4$, the syntenic diameter is $h + k - 4$

It is relatively easy to see that the diameter is tight, for example, by
$S(h, k) = \{\{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \ldots, \{1, 2, \ldots, h\}\}$.
$k - 4$ times fusions: $\{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}$.
2 times translocations: $\{1, 2\}, \{1, 2\}, \{3, 4, \ldots, h\}, \{3, 4, \ldots, h\}$.
2 times translocations: $\{1\}, \{2\}, \{3\}, \{4, \ldots, h\}$.

12.4 Diameter

Theorem (236,300)

For any $h \geq k \geq 4$, the syntenic diameter is $h + k - 4$

It is relatively easy to see that the diameter is tight, for example, by
$S(h, k) = \{\{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \ldots, \{1, 2, \ldots, h\}\}$.
$k - 4$ times fusions: $\{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}, \{1, 2, \ldots, h\}$.
2 times translocations: $\{1, 2\}, \{1, 2\}, \{3, 4, \ldots, h\}, \{3, 4, \ldots, h\}$.
2 times translocations: $\{1\}, \{2\}, \{3\}, \{4, \ldots, h\}$.
$h - 4$ times fissions: $\{\{1\}, \{2\}, \ldots, \{h\}\}$.

Page 15

12.5 Algorithmic Results

DasGupta et al.[1] proved that computing the syntenic distance is an NP-hard problem, by a transformation from Largest Balanced Independent Set.

12.5 Algorithmic Results

DasGupta et al.[1] proved that computing the syntenic distance is an NP-hard problem, by a transformation from Largest Balanced Independent Set.

Theorem ([1])

There exists a polynomial-time approximation algorithm H for computing $syd(S(h,k))$ whose ratio equals 2.

12.5 Algorithmic Results

DasGupta et al.[1] proved that computing the syntenic distance is an NP-hard problem, by a transformation from Largest Balanced Independent Set.

Theorem ([1])

There exists a polynomial-time approximation algorithm H for computing $syd(S(h,k))$ whose ratio equals 2. Its running time is $O(hkA^{-1}(hk,k))$, where A^{-1} is the inverse of Ackermann's function.

Proof. We consider the synteny graph $SG(S(h,k))$ and let CC_i be its p connected components, for $1 \leq i \leq p$.

12.5 Algorithmic Results

DasGupta et al.[1] proved that computing the syntenic distance is an NP-hard problem, by a transformation from Largest Balanced Independent Set.

Theorem ([1])

There exists a polynomial-time approximation algorithm H for computing $syd(S(h,k))$ whose ratio equals 2. Its running time is $O(hkA^{-1}(hk,k))$, where A^{-1} is the inverse of Ackermann's function.

Proof. We consider the synteny graph $SG(S(h,k))$ and let CC_i be its p connected components, for $1 \leq i \leq p$. Let k_i (resp. n_i) be the number of vertices (resp. number of elements) in CC_i.

12.5 Algorithmic Results

DasGupta et al.[1] proved that computing the syntenic distance is an NP-hard problem, by a transformation from Largest Balanced Independent Set.

Theorem ([1])

There exists a polynomial-time approximation algorithm H for computing $syd(S(h,k))$ whose ratio equals 2. Its running time is $O(hkA^{-1}(hk,k))$, where A^{-1} is the inverse of Ackermann's function.

Proof. We consider the synteny graph $SG(S(h,k))$ and let CC_i be its p connected components, for $1 \leq i \leq p$. Let k_i (resp. n_i) be the number of vertices (resp. number of elements) in CC_i. 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.

12.5 Algorithmic Results

DasGupta et al.[1] proved that computing the syntenic distance is an NP-hard problem, by a transformation from Largest Balanced Independent Set.

Theorem ([1])

There exists a polynomial-time approximation algorithm H for computing $syd(S(h,k))$ whose ratio equals 2. Its running time is $O(hkA^{-1}(hk,k))$, where A^{-1} is the inverse of Ackermann's function.

Proof. We consider the synteny graph $SG(S(h,k))$ and let CC_i be its p connected components, for $1 \leq i \leq p$. Let k_i (resp. n_i) be the number of vertices (resp. number of elements) in CC_i. 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
$\Sigma_{i=1}^{p}(h_i + k_i - 2) = h + k - 2p \leq 2h - 2p$ moves,

12.5 Algorithmic Results

DasGupta et al.[1] proved that computing the syntenic distance is an NP-hard problem, by a transformation from Largest Balanced Independent Set.

Theorem ([1])

There exists a polynomial-time approximation algorithm H for computing $syd(S(h,k))$ whose ratio equals 2. Its running time is $O(hkA^{-1}(hk,k))$, where A^{-1} is the inverse of Ackermann's function.

Proof. We consider the synteny graph $SG(S(h,k))$ and let CC_i be its p connected components, for $1 \leq i \leq p$. Let k_i (resp. n_i) be the number of vertices (resp. number of elements) in CC_i. 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
$\Sigma_{i=1}^{p}(h_i + k_i - 2) = h + k - 2p \leq 2h - 2p$ moves,
By Proposition 12.3, $syd(S(h,k)) \geq h - p$, and hence a performance ratio of 2 is achieved.

Ackermann's function

Theorem ([3])

Any algorithm that works only within connected components of the synteny graph cannot have an approximation ratio better than 2

Theorem ([3])

Any algorithm that works only within connected components of the synteny graph cannot have an approximation ratio better than 2

Example

Let $S(h, h) = \{\{1, 2, \ldots, h-1\}, \{h\}, \{h\}, \ldots, \{h\}\}$

Theorem ([3])

Any algorithm that works only within connected components of the synteny graph cannot have an approximation ratio better than 2

Example

Let $S(h, h) = \{\{1, 2, \ldots, h-1\}, \{h\}, \{h\}, \ldots, \{h\}\}$
It is easily seen that solving the following instance using only intracomponent moves requires at least $2h - 4$ *moves.*

Theorem ([3])

Any algorithm that works only within connected components of the synteny graph cannot have an approximation ratio better than 2

Example

Let $S(h, h) = \{\{1, 2, \ldots, h-1\}, \{h\}, \{h\}, \ldots, \{h\}\}$
It is easily seen that solving the following instance using only intracomponent moves requires at least $2h - 4$ *moves.*
However, this instance can be solved in $h - 1$ *moves.*

12.5.2 Easy Cases

Restricted versions:

12.5.2 Easy Cases

Restricted versions:

(1) Linear synteny

12.5.2 Easy Cases

Restricted versions:

(1) Linear synteny

(2.1) Linear nested synteny

(2.2) Exact nested synteny

12.5.2 Easy Cases

Restricted versions:

(1) Linear synteny

(2.1) Linear nested synteny

(2.2) Exact nested synteny

(3) Uncovering synteny

12.5.2 Easy Cases

Restricted versions:

(1) Linear synteny

(2.1) Linear nested synteny

(2.2) Exact nested synteny

(3) Uncovering synteny

(4) Fusions and Fissions only

12.5.2.1 Linear Synteny

Definition (Linear Synteny)

The linear synteny problem as the synteny problem in which all move sequences are constrained as follows:

12.5.2.1 Linear Synteny

Definition (Linear Synteny)

The linear synteny problem as the synteny problem in which all move sequences are constrained as follows:

(1) *The first $k-1$ moves must be either fusions or translocations, and*

12.5.2.1 Linear Synteny

Definition (Linear Synteny)

The linear synteny problem as the synteny problem in which all move sequences are constrained as follows:

(1) *The first $k-1$ moves must be either fusions or translocations, and*

(2) *all following moves must be fissions that create singletons.*

12.5.2.1 Linear Synteny

Definition (Linear Synteny)

The linear synteny problem as the synteny problem in which all move sequences are constrained as follows:

(1) *The first $k-1$ moves must be either fusions or translocations, and*

(2) *all following moves must be fissions that create singletons.*

Theorem ([3])

For any instance of synteny $S(h,k)$, we have

12.5.2.1 Linear Synteny

Definition (Linear Synteny)

The linear synteny problem as the synteny problem in which all move sequences are constrained as follows:

(1) *The first $k-1$ moves must be either fusions or translocations, and*

(2) *all following moves must be fissions that create singletons.*

Theorem ([3])

For any instance of synteny $S(h,k)$, we have $lsyd(S(h,k)) \geq h-1+max_{1\leq c<k}\{c-|\{g|count(g)\leq c+1\}|\}$. Theorem 12.1

12.5.2.1 Linear Synteny

Definition (Linear Synteny)

The linear synteny problem as the synteny problem in which all move sequences are constrained as follows:

(1) *The first $k-1$ moves must be either fusions or translocations, and*

(2) *all following moves must be fissions that create singletons.*

Theorem ([3])

For any instance of synteny $S(h,k)$, we have $lsyd(S(h,k)) \geq h-1+max_{1\leq c<k}\{c-|\{g|count(g)\leq c+1\}|\}$. Theorem 12.1

Theorem ([3])

For any $h \geq 4$, the maximal linear syntenic distance between two h-sized genomes is $2h-3$. Theorem 12.2

12.5.2.2 Linear Nested Synteny

Definition (Linear Nested Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be nested if, for all* $1 \leq i \neq j \leq k$, *one of the three following cases occurs:*

12.5.2.2 Linear Nested Synteny

Definition (Linear Nested Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be nested if, for all* $1 \leq i \neq j \leq k$, *one of the three following cases occurs:*

(1) $S_i \bigcap S_j = \emptyset$,

12.5.2.2 Linear Nested Synteny

Definition (Linear Nested Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be nested if, for all* $1 \leq i \neq j \leq k$, *one of the three following cases occurs:*

(1) $S_i \bigcap S_j = \emptyset$,

(2) $S_i \subseteq S_j$, *or*

12.5.2.2 Linear Nested Synteny

Definition (Linear Nested Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be nested if, for all* $1 \leq i \neq j \leq k$, *one of the three following cases occurs:*

(1) $S_i \bigcap S_j = \emptyset$,

(2) $S_i \subseteq S_j$, *or*

(3) $S_j \subseteq S_i$.

12.5.2.2 Linear Nested Synteny

Definition (Linear Nested Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be nested if, for all* $1 \leq i \neq j \leq k$, *one of the three following cases occurs:*

(1) $S_i \bigcap S_j = \emptyset$,

(2) $S_i \subseteq S_j$, *or*

(3) $S_j \subseteq S_i$.

2004, Liben-Nowell et al. [250] give an $O(hk^2 + pk^2 log\ k)$ algorithm for solving the linear nested synteny problem.

12.5.2.2 Linear Nested Synteny

Definition (Linear Nested Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be nested if, for all* $1 \leq i \neq j \leq k$, *one of the three following cases occurs:*

(1) $S_i \bigcap S_j = \emptyset$,

(2) $S_i \subseteq S_j$, *or*

(3) $S_j \subseteq S_i$.

2004, Liben-Nowell et al. [250] give an $O(hk^2 + pk^2 log\ k)$ algorithm for solving the linear nested synteny problem.
2002, Pisanti et al. [300] give an $O(hk + k^2 + pklog^2\ k)$ algorithm for solving the linear nested synteny problem.

12.5.2.3 Exact Nested Synteny

Definition (Exact Nested Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be exact if* $syd(S(h,k)) = h - 1$.

12.5.2.3 Exact Nested Synteny

Definition (Exact Nested Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be exact if* $syd(S(h,k)) = h - 1$. *The exact synteny problem is thus the problem of determining whether a general synteny instance is exact.*

12.5.2.3 Exact Nested Synteny

Definition (Exact Nested Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be exact if* $syd(S(h,k)) = h - 1$. *The exact synteny problem is thus the problem of determining whether a general synteny instance is exact.*

1998,DasGupta et al.[1] proved that computing the exact syntenic problem is NP-hard.

12.5.2.3 Exact Nested Synteny

Definition (Exact Nested Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ is said to be exact if $syd(S(h,k)) = h - 1$. The exact synteny problem is thus the problem of determining whether a general synteny instance is exact.

1998,DasGupta et al.[1] proved that computing the exact syntenic problem is NP-hard.
2002, Pisanti et al. [300] give an $O(hk + k^2 + klog\ k)$ algorithm for solving the exact nested synteny problem.

12.5.2.4 Uncovering Synteny

Definition (Uncovering Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be uncovering if there exists an ordering of the sets* S_j, *for* $1 \leq j \leq k$, *such that*

12.5.2.4 Uncovering Synteny

Definition (Uncovering Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be uncovering if there exists an ordering of the sets* S_j, *for* $1 \leq j \leq k$, *such that* $S_i \backslash (S_{i+1} \bigcup S_{i+2} \bigcup \cdots \bigcup S_k) \neq \emptyset$ *for all* $1 \leq i \leq k-3$.

12.5.2.4 Uncovering Synteny

Definition (Uncovering Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be uncovering if there exists an ordering of the sets* S_j, *for* $1 \leq j \leq k$, *such that* $S_i \backslash (S_{i+1} \bigcup S_{i+2} \bigcup \cdots \bigcup S_k) \neq \emptyset$ *for all* $1 \leq i \leq k-3$.

2006, Ting et al.[358] give an $O(hk)$ algorithm for solving the uncovering synteny problem.

12.5.2.4 Uncovering Synteny

Definition (Uncovering Synteny)

An instance $S(h,k) = \{S_1, S_2, \ldots, S_k\}$ *is said to be uncovering if there exists an ordering of the sets* S_j, *for* $1 \leq j \leq k$, *such that* $S_i \backslash (S_{i+1} \bigcup S_{i+2} \bigcup \cdots \bigcup S_k) \neq \emptyset$ *for all* $1 \leq i \leq k-3$.

2006, Ting et al.[358] give an $O(hk)$ algorithm for solving the uncovering synteny problem. Actually, for any uncovering instance, they show that either $syd(S(h,k)) = h - p$ or $syd(S(h,k)) = h - p + 1$.

12.5.2.5 Fusions and Fissions only

Theorem ((139))

For any $h \geq k$, $syd_{ff}(S(h,k)) = h + k - 2p$.

Ackermann's function

12.5.2.5 Fusions and Fissions only

Theorem ((139))

For any $h \geq k$, $syd_{ff}(S(h,k)) = h + k - 2p$.

Theorem ((139))

For any $h \geq k$ and any syntenic instance $S(h,k)$, there exists an $O(h^2 + hkA^{-1}(hk,k))$ algorithm, where A^{-1} is the inverse of Ackermann's function,

Ackermann's function

12.5.2.5 Fusions and Fissions only

Theorem ((139))

For any $h \geq k$, $syd_{ff}(S(h,k)) = h + k - 2p$.

Theorem ((139))

For any $h \geq k$ and any syntenic instance $S(h,k)$, there exists an $O(h^2 + hkA^{-1}(hk,k))$ algorithm, where A^{-1} is the inverse of Ackermann's function, that finds an optimal sequence of $syd_{ff}(S(h,k)) = h + k - 2p$ fusions and fissions to slove this instance.

Ackermann's function

12.6 Conjectures and Open Problems

Problem

Devise an approximation algorithm with ratio strictly smaller than 2 for computing the syntenic distance,

12.6 Conjectures and Open Problems

Problem

Devise an approximation algorithm with ratio strictly smaller than 2 for computing the syntenic distance, or show that this ratio cannot be improved.

12.6 Conjectures and Open Problems

Problem

Devise an approximation algorithm with ratio strictly smaller than 2 for computing the syntenic distance, or show that this ratio cannot be improved.

Problem

What is the complexity of the syntenic distance problem using only translocations?

12.6 Conjectures and Open Problems

Problem

Devise an approximation algorithm with ratio strictly smaller than 2 for computing the syntenic distance, or show that this ratio cannot be improved.

Problem

What is the complexity of the syntenic distance problem using only translocations?

Problem

Is it possible to characterize those syntenic instances that can be optimally solved by tanslocations only?


B. DasGupta and T. Jiang and S. Kannan and M. Li and E. Sweedyk, On the Complexity and Approximation of Syntenic Distance, (1998)


V. Ferretti, J.H. Nadeau and D. Sankoff, Original Synteny, (1996)


D. Liben-Nowell, On the Structure of Syntenic Distance, (2001)


D. Liben-Nowell, Gossip is synteny incomplete gossip and syntenic distance between genomes, (2002)

Ackermann's function.

Definition

$$A(m,n) = \begin{cases} n+1 & if\ \ m = 0, \\ A(m-1,1) & if m > 0\ and\ \ n = 0, \\ A(m-1,A(m,n-1)) & if m > 0\ and\ \ n > 0. \end{cases}$$

Ackermann's function.

Definition

$$A(m,n) = \begin{cases} n+1 & if\ \ m = 0, \\ A(m-1,1) & if m > 0\ and\ \ n = 0, \\ A(m-1, A(m,n-1)) & if m > 0\ and\ \ n > 0. \end{cases}$$

Values of $A(m, n)$

$m \backslash n$	0	1	2	3	4
0	0+1	1+1	2+1	3+1	4+1
1	A(0,1)	A(0,A(1,0))	A(0,A(1,1))	A(0,A(1,2))	A(0,A(1,3))
2	A(1,1)	A(1,A(2,0))	A(1,A(2,1))	A(1,A(2,2))	A(1,A(2,3))
3	A(2,1)	A(2,A(3,0))	A(2,A(3,1))	A(2,A(3,2))	A(2,A(3,3))
4	A(3,1)	A(3,A(4,0))	A(3,A(4,1))	A(3,A(4,2))	A(3,A(4,3))
5	A(4,1)	A(4,A(5,0))	A(4,A(5,1))	A(4,A(5,2))	A(4,A(5,3))
6	A(5,1)	A(5,A(6,0))	A(5,A(6,1))	A(5,A(6,2))	A(5,A(6,3))

Ackermann's function.

Definition

$$A(m,n) = \begin{cases} n+1 & if\ \ m = 0, \\ A(m-1,1) & if m > 0\ and\ \ n = 0, \\ A(m-1, A(m,n-1)) & if m > 0\ and\ \ n > 0. \end{cases}$$

Values of $A(m, n)$

$m \backslash n$	0	1	2	3	4
0	0+1	1+1	2+1	3+1	4+1
1	A(0,1)	A(0,A(1,0))	A(0,A(1,1))	A(0,A(1,2))	A(0,A(1,3))
2	A(1,1)	A(1,A(2,0))	A(1,A(2,1))	A(1,A(2,2))	A(1,A(2,3))
3	A(2,1)	A(2,A(3,0))	A(2,A(3,1))	A(2,A(3,2))	A(2,A(3,3))
4	A(3,1)	A(3,A(4,0))	A(3,A(4,1))	A(3,A(4,2))	A(3,A(4,3))
5	A(4,1)	A(4,A(5,0))	A(4,A(5,1))	A(4,A(5,2))	A(4,A(5,3))
6	A(5,1)	A(5,A(6,0))	A(5,A(6,1))	A(5,A(6,2))	A(5,A(6,3))

Ackermann's function.

Definition

$$A(m,n) = \begin{cases} n+1 & \text{if } m = 0, \\ A(m-1,1) & \text{if } m > 0 \text{ and } n = 0, \\ A(m-1, A(m,n-1)) & \text{if } m > 0 \text{ and } n > 0. \end{cases}$$

Page 12 Page 19

Values of $A(m, n)$

$m \backslash n$	0	1	2	3	4	n
0	1	2	3	4	5	$n+1$
1	2	3	4	5	6	$n+2 = 2+(n+3)-3$
2	3	5	7	9	11	$2n+3 = 2\cdot(n+3)-3$
3	5	13	29	61	125	$2^{(n+3)}-3$
4	13 $=2^{2^2}-3$	65533 $=2^{2^{2^2}}-3$	$2^{65536}-3$ $=2^{2^{2^{2^2}}}-3$	$2^{2^{65536}}-3$ $=2^{2^{2^{2^{2^2}}}}-3$	$2^{2^{2^{65536}}}-3$ $=2^{2^{2^{2^{2^{2^2}}}}}-3$	$\underbrace{2^{2^{\cdot^{\cdot^{\cdot^2}}}}}_{n+3}-3$