Comparative Genomics

Family-free Comparative Genomics

D. Doerr, J. Stoye, S. Böcker, and K. Jahn. (2014) Identifying gene clusters by discovering common intervals in indeterminate strings. BMC Genomics, 15(Suppl 6):S2.

D. Doerr (2016) PhD thesis, Bielefeld University.

So far

Genome = permutation / sequence of IDs ($\Leftarrow$ gene families)

So far

Genome = permutation / sequence of IDs ($\Leftarrow$ gene families)

Problems

familes based on seqence similarity

- $\rightarrow$ threshold?
- $\rightarrow$ ambiguities/inconsistencies
- $\rightarrow$ errors

Family-free (FF) approach – idea

- Each genome has individual set of genes.
- $\mathcal{A}$ = set of all genes in all genomes.

Family-free (FF) approach – idea

- Each genome has individual set of genes.
- $\mathcal{A}$ = set of all genes in all genomes.
- $\sigma : \mathcal{A} \times \mathcal{A} \longrightarrow [0, 1]$ = normalized similarity score
  e.g., *relative reciprocal BLAST score*:

Family-free (FF) approach – idea

- Each genome has individual set of genes.
- $\mathcal{A}$ = set of all genes in all genomes.
- $\sigma : \mathcal{A} \times \mathcal{A} \longrightarrow [0, 1]$ = normalized similarity score
  e.g., *relative reciprocal BLAST score*: $\frac{\text{bitscore}(s,t)+\text{bitscore}(t,s)}{\text{bitscore}(s,s)+\text{bitscore}(t,t)}$

Family-free (FF) approach – idea

- Each genome has individual set of genes.
- $\mathcal{A}$ = set of all genes in all genomes.
- $\sigma : \mathcal{A} \times \mathcal{A} \longrightarrow [0, 1]$ = normalized similarity score
  e.g., *relative reciprocal BLAST score*: $\frac{\text{bitscore}(s,t)+\text{bitscore}(t,s)}{\text{bitscore}(s,s)+\text{bitscore}(t,t)}$
- *Gene connection graph*: k-partite, verteces = genes,
  edge between a and b in differnet genomes if $\sigma(s, b) > 0$.

Family-free (FF) approach – idea

- Each genome has individual set of genes.
- $\mathcal{A}$ = set of all genes in all genomes.
- $\sigma : \mathcal{A} \times \mathcal{A} \longrightarrow [0, 1]$ = normalized similarity score
  e.g., *relative reciprocal BLAST score*: $\frac{\text{bitscore}(s,t)+\text{bitscore}(t,s)}{\text{bitscore}(s,s)+\text{bitscore}(t,t)}$
- *Gene connection graph*: k-partite, verteces = genes,
  edge between a and b in differnet genomes if $\sigma(s, b) > 0$.
- *Gene similarity graph*: edges weighted by σ.

FF-adjacencies

Find matching $\mathcal{M}$ such as to maximize:

$$\alpha \cdot \mathsf{adj}(\mathcal{M}) + (1 - \alpha) \cdot \mathsf{edg}(\mathcal{M})$$

FF-adjacencies

Find matching $\mathcal{M}$ such as to maximize:

$$\alpha \cdot \mathsf{adj}(\mathcal{M}) + (1 - \alpha) \cdot \mathsf{edg}(\mathcal{M})$$

$$\mathsf{edg}(\mathcal{M}) = \sum_{(a,b) \in \mathcal{M}} \sigma(a, b)$$

FF-adjacencies

Find matching $\mathcal{M}$ such as to maximize:

$$\alpha \cdot \text{adj}(\mathcal{M}) + (1 - \alpha) \cdot \text{edg}(\mathcal{M})$$

$$\text{edg}(\mathcal{M}) = \sum_{(a,b) \in \mathcal{M}} \sigma(a, b)$$

$$\text{adj}(\mathcal{M}) = \sum_{\substack{(a,b) \in \mathcal{M} \\ (c,d) \in \mathcal{M} \\ \text{forming} \\ \text{conserved} \\ \text{adjacency} \\ \text{in } A^{\mathcal{M}} \text{ and } B^{\mathcal{M}}}} \sqrt{\sigma(a, b) \cdot \sigma(c, d)}$$

$A^{\mathcal{M}} =$ subsequence of A that only contains genes that are part of $\mathcal{M}$

FF-DCJ

adjacency graph $\Rightarrow$ *weighted* adjacency graph

FF common intervals

string $\Rightarrow$ *indeterminate sting*:
sequence of sets of characters

obtained, e.g., from gene connection graph

Definitions

Character set: $\mathcal{C}(S) := \bigcup_{i=1}^{n} S[i]$

Definitions

Character set: $\mathcal{C}(S) := \bigcup_{i=1}^{n} S[i]$

Given two indeterminate strings S and T,
two intervals $[i, j]$ in S and $[k, \ell]$ in T are

strict common intervals if $\mathcal{C}(S[i, j]) = \mathcal{C}(T[k, \ell])$

Definitions

Character set: $\mathcal{C}(S) := \bigcup_{i=1}^{n} S[i]$

Given two indeterminate strings S and T,
two intervals $[i, j]$ in S and $[k, \ell]$ in T are

strict common intervals if $\mathcal{C}(S[i, j]) = \mathcal{C}(T[k, \ell])$

weak common intervals

with common character set $C = \mathcal{C}(S[i, j]) \cap \mathcal{C}(T[k, \ell])$

if $C \cap \mathcal{C}(S[x]) \neq \emptyset$ for all $i \leq x \leq j$, and
$C \cap \mathcal{C}(T[y]) \neq \emptyset$ for all $k \leq y \leq \ell$.

(at least one char. per position must be in common char. set)

Definitions

left/right maximality

Given indeterminate string S and character set C,
interval $[i, j]$ is *C-closed*
if $i = 1$ or $S[i-1] \cap C = \emptyset$, and
$j = n$ or $S[j+1] \cap C = \emptyset$.

Definitions

left/right maximality

Given indeterminate string S and character set C,
interval $[i, j]$ is *C-closed*
if $i = 1$ or $S[i-1] \cap C = \emptyset$, and
$j = n$ or $S[j+1] \cap C = \emptyset$.

A pair of intervals $[i, j]$ in S and $[k, \ell]$ in T are *mutually closed*
if $[i, j]$ is $\mathcal{C}(T[k, \ell])$-closed, and
$[k, \ell]$ is $\mathcal{C}(S[i, j])$-closed.

Getting prepared...

Any pair of indeterminate strings can be formed into $1, \ldots, n$ and indeterminate string:

Getting prepared...

Any pair of indeterminate strings can be formed into $1, \ldots, n$ and indeterminate string:

index string of S: $I_S := \{1\}\{2\}\cdots\{n\}$

Getting prepared...

Any pair of indeterminate strings can be formed into $1, \ldots, n$ and indeterminate string:

index string of S: $I_S := \{1\}\{2\}\cdots\{n\}$

index mapping of S onto T:

$$T_S[y] = \begin{cases} \{\infty\} & \text{if } T[y] \cap \mathcal{C}(S) = \emptyset \\ \{x \mid x = 1..n, S[x] \cap T[y] \neq \emptyset\} & \text{otherwise.} \end{cases}$$

Getting prepared...

Any pair of indeterminate strings can be formed into $1, \ldots, n$ and indeterminate string:

index string of S: $I_S := \{1\}\{2\}\cdots\{n\}$

index mapping of S onto T:

$$T_S[y] = \begin{cases} \{\infty\} & \text{if } T[y] \cap \mathcal{C}(S) = \emptyset \\ \{x \mid x = 1..n, S[x] \cap T[y] \neq \emptyset\} & \text{otherwise.} \end{cases}$$

DING no. 1

$$[i,j] \text{ in } S \text{ and } [k,\ell] \text{ in } T \text{ are weak common intervals}$$
$$\Leftrightarrow$$
$$[i,j] \text{ in } I_S \text{ and } [k,\ell] \text{ in } T_S \text{ are weak common intervals.}$$

Detection of mutualy closed weak common intervals

similar to Didier's algorithm: For all left positions i in S: ...

Detection of mutualy closed weak common intervals

similar to Didier's algorithm: For all left positions i in S: ...

Pos $\rightarrow$ array of lists:
$Pos[j] :=$ positions in T_S containing char. j.

Detection of mutualy closed weak common intervals

similar to Didier's algorithm: For all left positions i in S: ...

Pos $\rightarrow$ array of lists:
$Pos[j] :=$ positions in T_S containing char. j.

rank $\rightarrow$ *i-reduced string:*
$T_S^i[y] := \min\{c | c \in T_S[y] \cup \{\infty\}, c \geq i\}$

Detection of mutualy closed weak common intervals

similar to Didier's algorithm: For all left positions i in S: ...

Pos $\rightarrow$ array of lists:
$Pos[j] :=$ positions in T_S containing char. j.

rank $\rightarrow$ *i-reduced string:*
$T_S^i[y] := \min\{c | c \in T_S[y] \cup \{\infty\}, c \geq i\}$

rank distance $\rightarrow$ *min-rank distance:*
$d^i_{T_S}(k, \ell) := \max\{ T_S^i[p] | p = k..\ell\}$

Detection of mutualy closed weak common intervals

similar to Didier's algorithm: For all left positions i in S: ...

Pos $\rightarrow$ array of lists:
$Pos[j] :=$ positions in T_S containing char. j.

rank $\rightarrow$ *i-reduced string:*
$T_S^i[y] := \min\{c | c \in T_S[y] \cup \{\infty\}, c \geq i\}$

rank distance $\rightarrow$ *min-rank distance:*
$d^i_{T_S}(k, \ell) := \max\{T_S^i[p] | p = k..\ell\}$

rank interval $\rightarrow$ *min-rank interval* in T_S of char. j for position y:
largest interval $[k, \ell]$ around y where each position contains character from $S[i, j]$

Detection of mutualy closed weak common intervals

similar to Didier's algorithm: For all left positions i in S: ...

Pos $\rightarrow$ array of lists:
$Pos[j] :=$ positions in T_S containing char. j.

rank $\rightarrow$ *i-reduced string:*
$T_S^i[y] := \min\{c | c \in T_S[y] \cup \{\infty\}, c \geq i\}$

rank distance $\rightarrow$ *min-rank distance:*
$d^i_{T_S}(k, \ell) := \max\{T_S^i[p] | p = k..\ell\}$

rank interval $\rightarrow$ *min-rank interval* in T_S of char. j for position y:
largest interval $[k, \ell]$ around y where each position contains character from $S[i, j]$

DING no. 2! min-rank intervals in T_S
$=$ Didier's rank intervals in T_S^i.

Algorithm: modified Didier

1: **for** each left bound i **do**
2: **for** each min-rank nearest successor **do**
3: **if** outermost positions visited on the way
$\subseteq$ current min-rank interval $[k, \ell]$ in T_S
then
4: $\Rightarrow$ weak common interval
5: **if** $i - 1$ not in $\mathcal{C}(T_S[k, \ell])$
and next successor not in current interval
then
6: $[k, \ell]$ mutually closed

Algorithm: modified Didier

1: **for** each left bound i **do**
2: **for** each min-rank nearest successor **do**
3: **if** outermost positions visited on the way
$\subseteq$ current min-rank interval $[k, \ell]$ in T_S
then
4: $\Rightarrow$ weak common interval
5: **if** $i - 1$ not in $\mathcal{C}(T_S[k, \ell])$
and next successor not in current interval
then
6: $[k, \ell]$ mutually closed

$\Rightarrow O(n \cdot ||T_S||)$ time

Exercise

Consider the following indeterminate strings:

$A = [\{d\}\{g, b\}\{a\}\{p, s\}\{n\}\{a, b\}\{f, m, w\}\{e, w\}\{i\}\{q, y\}\{h\}\{c, r\}\{z\}]$
$B = [\{g\}\{b, p\}\{x\}\{n, p\}\{d, o, s\}\{a, z\}\{e, n, w\}\{f\}\{l, v\}\{h, u, z\}\{h, r\}\{k\}\{m, q, y\}]$

1. Determine the corresponding index string I_A and index mapping B_A.
2. Determine the i-reduced string B_A^i for $i = 4$.
3. Determine the min-rank intervals for $i = 4$.
4. Follow the rank-nearest successors of all elements in $Pos[4]$ to determine all weak common intervals starting at position $i = 4$ in A.
5. Which of the above intervals are reported as mutually-closed weak common intervals?