Trapezoid Graphs, Independent Sets, and Whole Genome Comparisons

Raluca Uricaru, Eric Rivals

LIRMM, CNRS Université de Montpellier 2

6 novembre 2009
Summary

1. Whole Genome Comparisons
2. Fragment Chaining
3. Chaining with Proportional Overlaps
4. Conclusion
Summary

1. Whole Genome Comparisons
2. Fragment Chaining
3. Chaining with Proportional Overlaps
4. Conclusion
Problem and Motivation

Align two or several genomic sequences to determine resemblances between them;

- infer knowledge from one sequence to other highly similar sequences;
- identify the conserved parts between sequences indicating common biological components;
- identify differences between sequences responsible for what distinguishes them (e.g., virulence, pathogenicity).
complex problem due to the sizes of the genomic sequences;

3-phase heuristic called the "anchor based strategy":

1. computation of local similarities (fragments) between sequences; using an external method;
2. fragment chaining phase: selects a subset of non-overlapping, collinear anchors giving a maximum weighted chain; non-collinear anchors $\rightarrow$ NP-complete problem;
3. apply recursively the first 2 phases on yet not aligned regions;
Summary

1. Whole Genome Comparisons

2. Fragment Chaining

3. Chaining with Proportional Overlaps

4. Conclusion
Fragment Chaining
Trapezoid Representation

Trapezoid Graphs, Independent Sets, and Whole Genome Comparisons
Fragment Chaining
Box Representation

$w(B_1) = 5$
$w(B_2) = 8$
$w(B_3) = 10$
$w(B_5) = 13$
$w(B_4) = 6$

$W[C_{max}] = w(B_2) + w(B_5) = 8 + 13 = 21$
Fragment Chaining Definition

- \( n \) fragments represented as boxes in a dotplot: \( \{B_1, \ldots, B_n\} \);

- lower corner of \( B_i = l(B_i) \), upper corner of \( B_i = u(B_i) \);

- if \( u(B_i) < l(B_j) \) then \( B_i \ll B_j \);

- the weight of a box \( B_i \): \( w(B_i) = |P_x(B_i)| + |P_y(B_i)| \)
  
  \( P_x \) and \( P_y \) are the projections on the 2 axis;

\( \{B_{i_1}, B_{i_2}, \ldots, B_{i_k}\} \) forms a chain \( C \) if \( B_{i_j} \ll B_{i_{j+1}} \) \( \forall j, 1 \leq j \leq k \), with

\[ W[C] = \sum_{j=1}^{k} w(B_{i_j}). \]

We are looking for the chain \( C_{\text{max}} \) giving the maximum weight.
equivalent to the maximum weighted independent set in the trapezoid graph corresponding to the trapezoid/box representations above;

[S. Felsner et al, *Trapezoid graphs and generalizations, geometry and algorithms*, 1995.]

dynamic programming algorithm in $O(n \log(n))$, using the sweep line paradigm;
Summary

1. Whole Genome Comparisons

2. Fragment Chaining

3. Chaining with Proportional Overlaps

4. Conclusion
Chaining with Overlaps

Box Representation

\[ W[C_{\text{max}}] = w(B_2) + w(B_3) + w(B_5) - O_x(B_2, B_3) - O_x(B_3, B_5) \]
\[ = 28 \]
Chaining with Overlaps

- overlap between $B_i, B_j$ boxes on x-axis: $O_x(B_i, B_j) = |P_x(B_i) \cap P_x(B_j)|$

- maximum allowed overlap on x-axis: $r \ast \min(|P_x(B_i)|, |P_x(B_j)|)$

- if $l(B_i) < l(B_j)$ and $O_x(B_i, B_j) \leq r \ast \min(|P_x(B_i)|, |P_x(B_j)|)$ or $B_i \ll B_j$ then $B_i \ll_{r,x} B_j$. Similarly on y-axis . . .

- if $B_i \ll_{r,x} B_j$ and $B_i \ll_{r,y} B_j$ then $B_i \ll_r B_j$

- total overlap between $B_i, B_j$ boxes: $O(B_i, B_j) = O_x(B_i, B_j) + O_y(B_i, B_j)$

$\{B_{i_1}, B_{i_2}, \ldots B_{i_k}\}$ forms a chain $C$ if $B_{i_j} \ll_r B_{i_{j+1}} \ \forall j, 1 \leq j \leq k$, with $W[C] = \sum_{j=1}^{k} w(B_{i_j}) - \sum_{j=1}^{k-1} O(B_{i_j}, B_{i_{j+1}})$.

We are looking for the chain $C_{max}$ giving the maximum weight.
Remarks on Chaining With Overlaps

- adapted to all types of fragments;
- makes sense from a biological point of view;
- a straightforward but non-practical dynamic programming algorithm in $O(n^2)$, $n = number of fragments$;
- partial order on boxes $\ll_r$ with the sweep line paradigm: novel algorithm in $O(n \log n + nm)$, where $m \leq r \% \max(|P_y|)$. 

Trapezoid Graphs, Independent Sets, and Whole Genome Comparisons
Algorithm for Chaining With Overlaps

**Algorithm 1**: Maximum Weighted Chain with Overlaps

**Data**: $\mathcal{P}$ a set of points corresponding to the $2n + 4$ box corners

$n$ boxes $+$ 2 additional boxes corresponding to $\perp$, $\top$

**Result**: $\text{Prev}$ a vector of previous boxes in the maximum weighted chain

begin

foreach $p \in \mathcal{P}$ in ascending order on $x$-coordinate do

if $\exists$ a box $B_i$, with $p$ the lower corner of $B_i$ then

Compute $B_j$, the best previous box ending before $B_i$;  
/* $B_j$ may overlap $B_i$ on $y$-axis */

else

/* $\exists$ a box $B_i$, with $p$ the upper corner of $B_i$ */

Update the weight and the predecessor of opened boxes;
/* deal with overlaps on $x$-axis */

Update the list of potential predecessors;

end

end

traceback($\text{Prev}[\top]$);
Algorithm 2: Maximum Weighted Chain with Overlaps

Data: $\mathcal{P}$ a set of points corresponding to the $2n + 4$ box corners
$n$ boxes + 2 additional boxes corresponding to $\bot$, $\top$

Result: $\text{Prev}$ a vector of previous boxes in the maximum weighted chain

begin

foreach $p \in \mathcal{P}$ in ascending order on $x$-coordinate do

if $\exists$ a box $B_i$, with $p$ the lower corner of $B_i$ then

    $\triangleright$ Case 1
    Compute $B_j$, the best previous box ending before $B_i$;
    /* $B_j$ may overlap $B_i$ on $y$-axis */

else

    $\triangleright$ Case 2
    Update the weight and the predecessor of opened boxes;
    /* deal with overlaps on $x$-axis */

    Update the list of potential predecessors;

traceback($\text{Prev}[\top]$);

end
Case 1

Compute $B_j$, the best previous box ending before $B_i \ldots$
Case 2

Update the weight and the predecessor of opened boxes …
Summary

1. Whole Genome Comparisons
2. Fragment Chaining
3. Chaining with Proportional Overlaps
4. Conclusion
the chaining with proportional overlaps corresponds to an extended definition of a maximum weighted independent set including a tolerance notion;

the algorithm in $O(n \log n + nm)$ is very efficient in practice;

e.g., on a real case composed of 190000 fragments:
- $O(n^2)$ algorithm: 34 min
- $O(n \log n + nm)$ algorithm: < 2 min;
Support :
CoCoGEN project
http://www.lirmm.fr/~uricaru/CoCoGEN

Thank you for your attention!

Questions?