



Computer Science Department
Technion – Israel Institute of Technology

Length-sensitive Algorithms for Sorting by Reversals: a Phylogeny-based Evaluation

Ron Y. Pinter

October 23, 2003

Outline

- The model
 - family of **cost functions**
- Two length-sensitive algorithms for **Sorting by Reversals (SBR)**
 - summary of optimality/approximation
- Phylogeny based **evaluation**
 - issues and workarounds
- Results

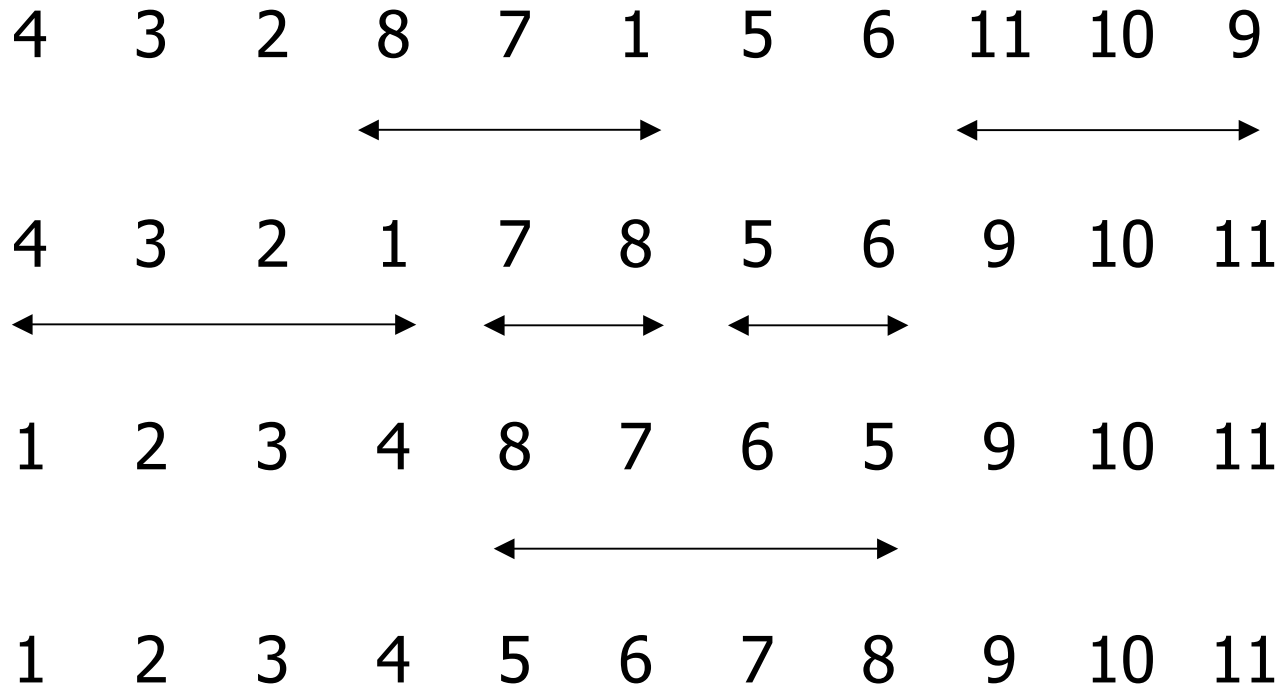
Genome Rearrangement

- events
 - duplication
 - translocation
 - reversal (inversion)
- occur primarily during reproduction
- allow large-scale genomic comparisons

Sorting by Reversals

- genome represented as a permutation on $1, 2, \dots, n$
 - $n = \#$ orthologous **genes** among species
- assumptions
 - can **identify** genes
 - genes are **distinct**
- operation: **reversal** of a subsequence (of genes)
 - models **inversion** (occurs during crossover)
- one of the permutations can be $1, 2, \dots, n$
 - appropriately relabel others

Example



- 6 reversal (3 phases)
- in our model (for $f(l) = l$): cost = 18

Our Model

- unsigned
- **cost** of reversal of subsequence of length l is $f(l)$
- total sorting cost (or **distance**) is

$$\sum_{\substack{S_j \text{ are} \\ \text{reversed} \\ \text{subsequences}}} f(\text{length}(s_j))$$

Cost Functions

- additive

$$f(x+y) = f(x) + f(y)$$

- subadditive

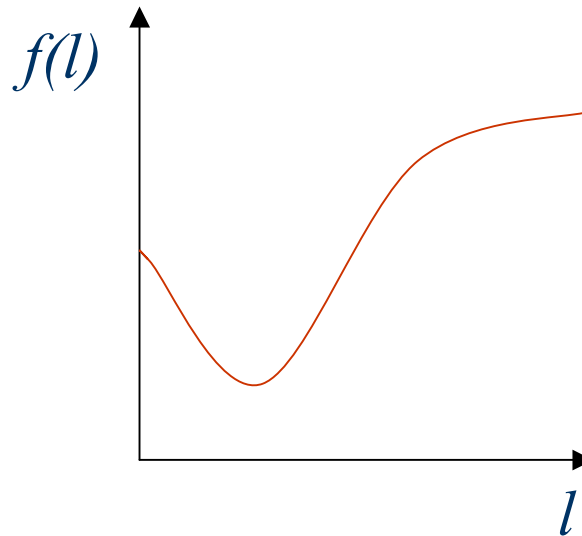
$$f(x+y) < f(x) + f(y)$$

- superadditive

$$f(x+y) > f(x) + f(y)$$

- other

– e.g. bitonic



Family of Cost Functions

- Consider $f(l) = l^\alpha$
 - based on initial feedback
 - ignores declining portion
- Captures additivity issues
 - $\alpha=0$: unit-cost
 - $0<\alpha<1$: sub-additive
 - $\alpha=1$: additive
 - $\alpha>1$: super-additive

Problems

- algorithm to sort **any** permutation
 - worst-case min cost
- approximate min cost for a **given** permutation

Extremal Costs

- highly subadditive: *e.g.* unit cost, $f(l) = 1$
 - NP complete [Caprara, '97]
 - series of approximation ratios:
 - 2 [Kececioglu and Sankoff, 1993]
 - 1.75 [Bafna and Pevzner, 1996]
 - 1.375 [Berman, Hannenhalli and Karpinski, 2002]
- highly superadditive: $f(l) > l^2$
 - essentially bubblesort

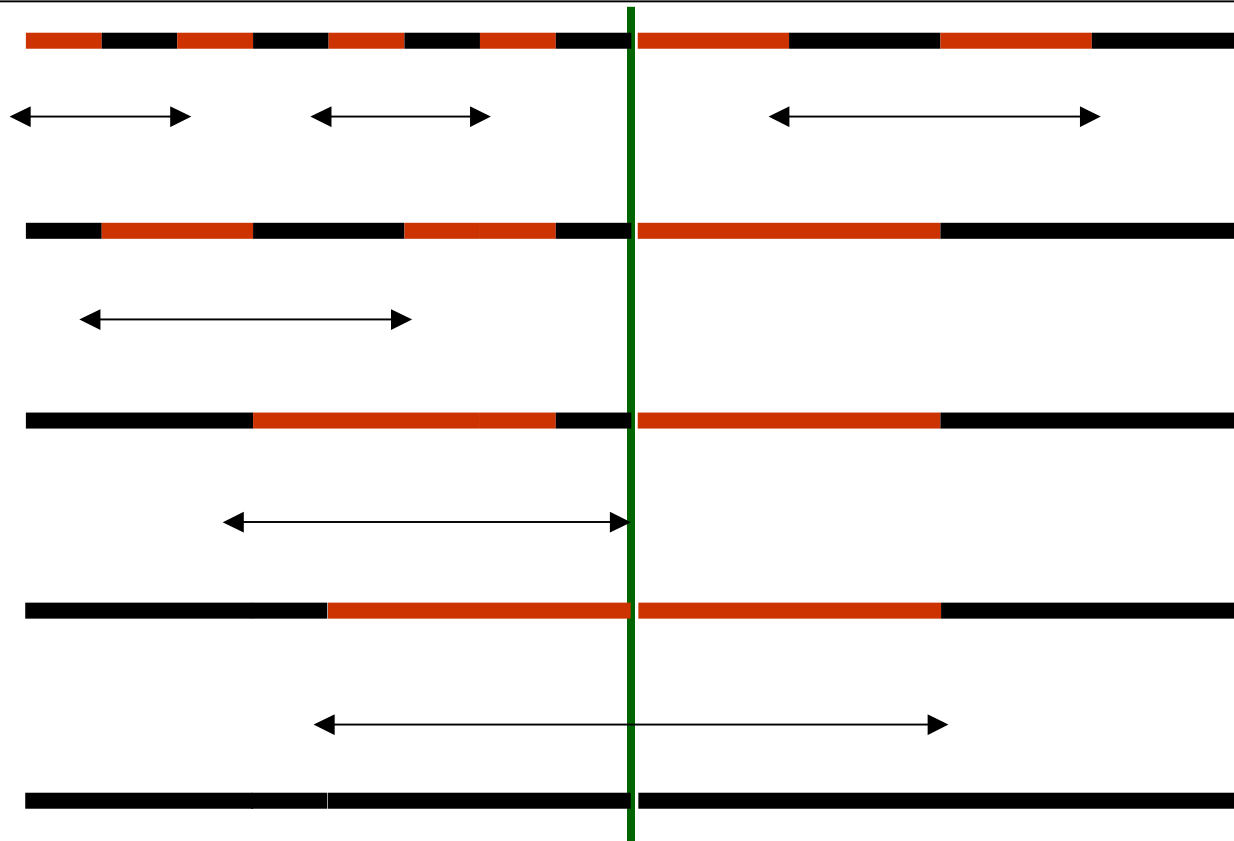
Our Algorithmic Workhorses

- QuickSort-like algorithm
 - recursive application of MedianEject (*next foil*)
 - time complexity: $O(n \lg n)$
- Dynamic programming scheme
 - use sorting of 0-1 sequences as subroutine
 - time complexity: $O(n^3)$

MedianEject(a,b)

- find r maximal blocks of **wrong-sided** elements with respect to median
- for $lg r$ do: flip every **other pair** of blocks of wrong-sided and adjacent blocks
- move wrong-sided blocks to **median** boundary
- reverse left and right blocks

Sample Run



cost: $O(f(b-a) \lg r)$

ReversalSort(a,b)

MedianEject (a,b);

ReversalSort (a, $\lfloor \frac{b-a}{2} \rfloor$);

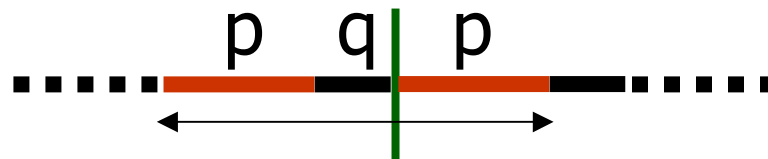
ReversalSort ($\lceil \frac{b-a}{2} \rceil$, b);

Cost

$$\begin{aligned} T(n) &= 2 \cdot T\left(\frac{n}{2}\right) + O(f(n) \lg n) = O(f(n) \lg^2 n) \\ &= O(n \lg^2 n) \quad \text{for } f(n) \sim n \end{aligned}$$

Algorithmic Improvements

- simplify "short" phases
- merge 2 last steps of MedianEject



when possible ($2p+q$ vs. $3p+q$)

- "soft" median

Dynamic Programming Scheme

- $S = w_1 w_2 \dots w_{2l}$
 - odd – above median
 - even – below median
- for all i, j , $j > i$, i odd, j even
 - sort $w_1 \dots w_{i-1}$ and $w_{j+1} \dots w_{2l}$
 - reverse sort $w_i \dots w_j$
 - reverse middle
- pick best solution

Summary of Results: Bounds on Costs and Approximation Ratios

α Value	Lower Bounds	Upper Bounds		Approximation Ratio	
		Permutations	0/1's	Permutations	0/1's
$0 < \alpha < 1$	$\Omega(n)$	$O(n \lg n)$	$\Theta(n)$		$O(1)$
$\alpha = 1$	$\Omega(n \lg n)$	$O(n \lg^2 n)$	$\Theta(n \lg n)$	$O(\lg n)$	1
$1 < \alpha < 2$	$\Omega(n^\alpha)$	$\Theta(n^\alpha)$	$\Theta(n^\alpha)$	$O(\lg n)$	$O(1)$
$\alpha \geq 2$	$\Omega(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	2	1

(to appear in SODA'04)

Evaluation Framework

- Interpret **costs** as **distances**
- For a set of genomes, **build** a **phylogenetic tree** using distance matrix
- Compare computed tree to **reference tree**
- Find α with best match

Issues

- “Good” reference dataset
- Gene identification
 - duplicate genes
- Number of common genes
- Size of partial genome sets
- Combining partial matches
 - is reconstruction necessary?
- Tree similarity scores
- Coping with approximation (of SBR)

Data Sets

- 10-20 genomes
 - Related organisms
 - Highly curated
 - mitochondrial or chloroplast
 - Reference tree
-
- 15 related plants (incl. *A. thaliana*)
 - Chloroplast genome, reference tree
 - Martin *et al.*, PNAS, Sept 2002

Number of Common Genes

- All SBRs must use the same number of genes
 - Intersection of all genomes
- Often too low (26)
- Long gaps

Genome Subsets

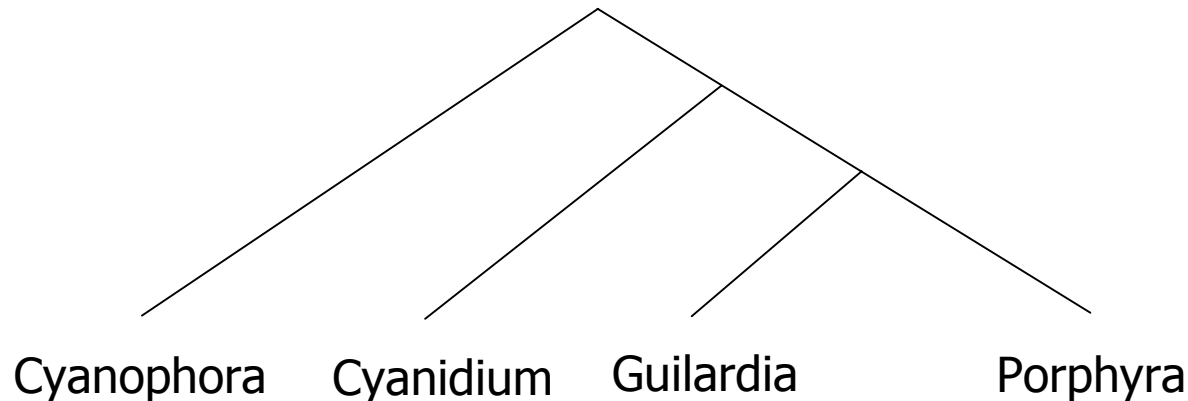
- Increase **common** gene count
- Generate **meaningful** subtrees

quartets!

- Not too many (**1365**)
- Know how to puzzle

Early “Validation”

- Used our cost (= **distance**) to build several phylogenetic trees, e.g.



- 4 plants (chloroplastic genes)
- consistent with [Martin et al., PNAS Sept '02]

Reconstruction

- Quartet puzzling
- Use only quartets with high enough gene count?
 - “partial” puzzling
- (Sample) subsets of other size
- Match subtrees, accumulate similarity scores

Tree Similarity Measures

- Nearest-neighbor interchange distance
 - NP-hard
- Maximal agreement subtree
- **Bipartition counting** [Robinson-Foulds, 1981]

Approximation etc.

- Symmetrization
- Eliminate duplicates
- Break circularity
- Signs...

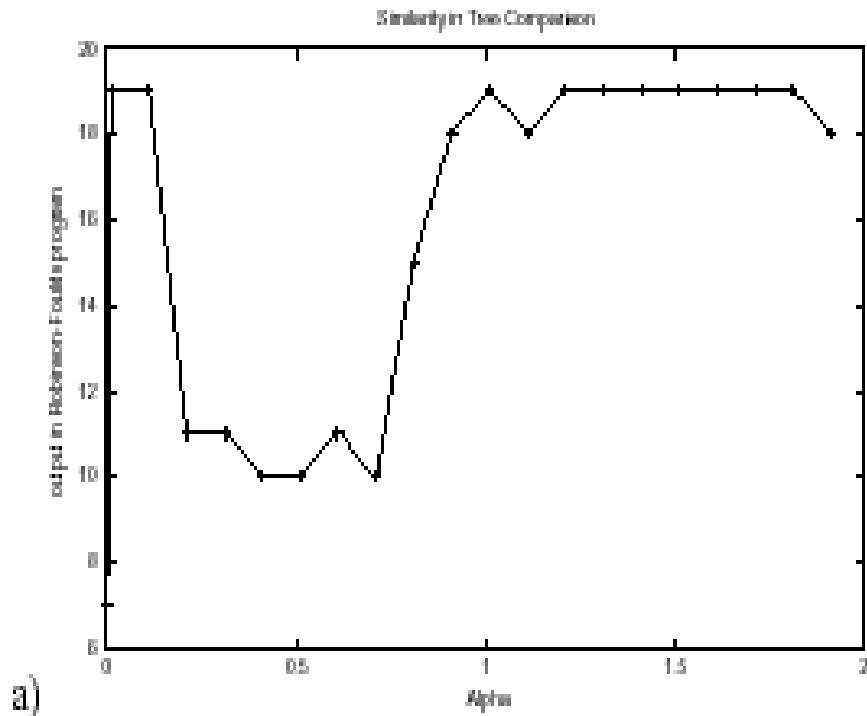
Finding α

- for $\alpha=0.0$ to 1.9 by 0.1
 - for all distinct $s_i, s_j \in S$, $d_{ij} := \text{dist}_\alpha(s_i, s_j)$
 - for all i, j , $d_{ij} := \min(d_{ij}, d_{ji})$
 - build T_α based on $[d_{ij}]$
 - $\text{tree-dist}(\alpha) := \text{RF}(T_\alpha, T_{\text{ref}})$
- Pick α with smallest tree-dist

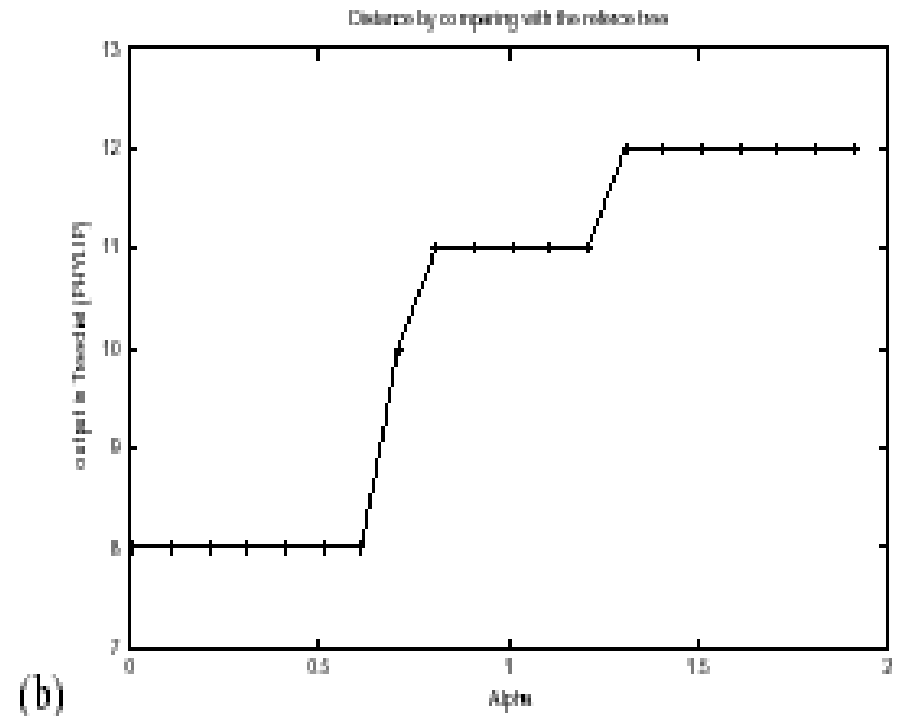
Finding α with puzzling

- for $\alpha=0.0$ to 1.9 by 0.1
 - for all distinct $s_i, s_j, s_k, s_l \in S$
 - for all $p, q \in \{i, j, k, l\}$, $d_{pq} := \text{dist}_\alpha^*(s_p, s_q)$
 - for all p, q , $d_{pq} := \min(d_{pq}, d_{qp})$
 - build $T_\alpha(i, j, k, l)$ based on $[d_{pq}]$
 - quartet puzzle T_α from $T_\alpha(i, j, k, l)$
 - $\text{tree-dist}(\alpha) := \text{RF}(T_\alpha, T_{\text{ref}})$
- Pick α with smallest tree-dist

RF(T_α, T_{ref}) for Matrin *et al.*



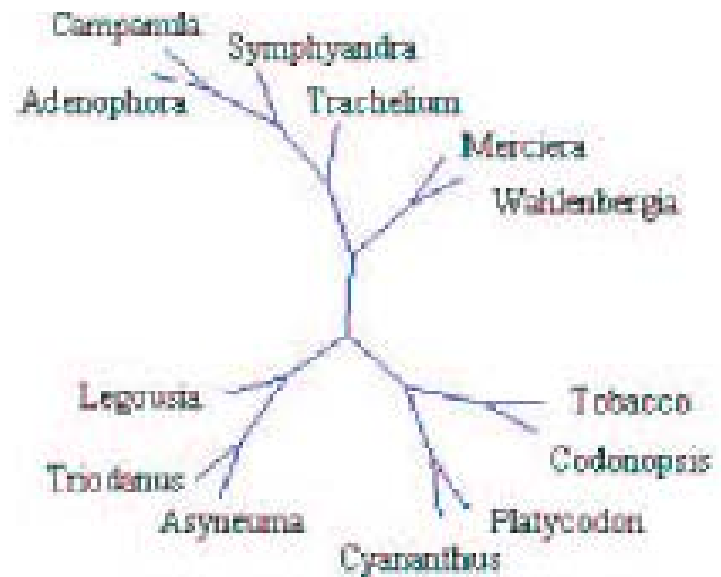
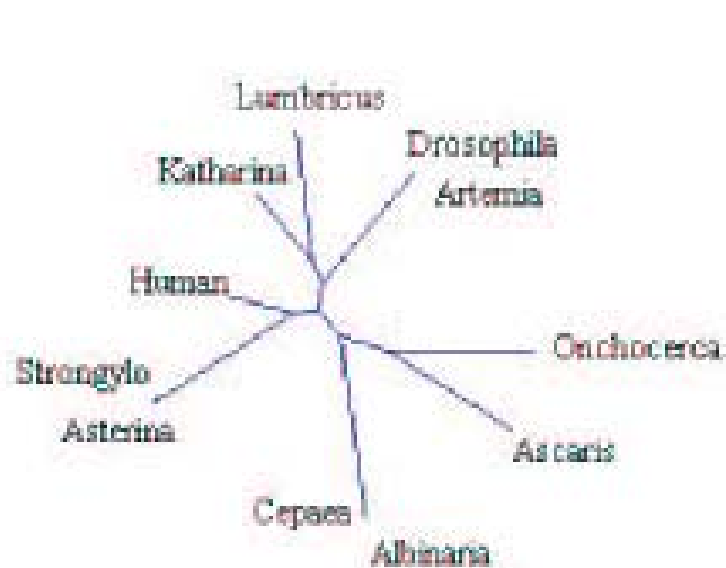
a)



(b)

Prediction*

- With $\alpha=0.4 \dots 0.7$



* no puzzling necessary

Open Problems: Algorithmic

- weighted genes
- model gaps
- tighter approximation ratios
- better algorithms
 - with **constant** ratio?
- symmetric cost
- other cost functions (incl. bitonic)
- the **signed** case

Open Problems: Modeling

- chromosomal ordering
- homology level
- what is the **right** cost function?
- validation criteria
 - “trace” of reversals
- combine with **constraint-based** models
 - restricted regions
 - “undesired” reversal sequences
- deal with duplication and translocation events

Future Work

- Solve open problems...
- Acquire more datasets
 - e.g. the **Fungi** project, Whitehead (MIT)
- Integrate model into DCM-GRAPPA and validate with appropriate simulations
- Set up web server
 - possibly jointly with GRIMM

Acknowledgements

- **SUNY Stony Brook**

- Michael A. Bender
- Dongdong Ge
- Simai He
- Haodong Hu
- Steven Skiena

- **Technion**

- Yaniv Berliner
- Michael Shmoish
- Meir Shoham
- Firas Swidan