Scalable Visual Comparison of Biological Trees and Sequences

Tamara Munzner
University of British Columbia
Department of Computer Science
Outline

• Accordion Drawing
  – information visualization technique
• TreeJuxtaposer
  – tree comparison
• SequenceJuxtaposer
  – sequence comparison
• PRISAD
  – generic accordion drawing framework
Accordion Drawing

- rubber-sheet navigation
  - stretch out part of surface, the rest squishes
  - borders nailed down
  - Focus+Context technique
    - integrated overview, details
  - old idea
    - [Sarkar et al 93], [Robertson et al 91]

- guaranteed visibility
  - marks always visible
  - important for scalability
  - new idea
    - [Munzner et al 03]
Guaranteed Visibility

- marks are always visible
- easy with small datasets
Guaranteed Visibility Challenges

- hard with larger datasets
- reasons a mark could be invisible
Guaranteed Visibility Challenges

- hard with larger datasets
- reasons a mark could be invisible
  - outside the window
    - AD solution: constrained navigation
Guaranteed Visibility Challenges

- hard with larger datasets
- reasons a mark could be invisible
  - outside the window
    - AD solution: constrained navigation
  - underneath other marks
    - AD solution: avoid 3D
Guaranteed Visibility Challenges

- hard with larger datasets
- reasons a mark could be invisible
  - outside the window
    - AD solution: constrained navigation
  - underneath other marks
    - AD solution: avoid 3D
  - smaller than a pixel
    - AD solution: smart culling
Guaranteed Visibility: Small Items

• Naïve culling may not draw all marked items
Guaranteed Visibility: Small Items

• Naïve culling may not draw all marked items

Guaranteed visibility of marks

No guaranteed visibility
Outline

• Accordion Drawing
  – information visualization technique
• TreeJuxtaposer
  – tree comparison
• SequenceJuxtaposer
  – sequence comparison
• PRISAD
  – generic accordion drawing framework
Phylogenetic/Evolutionary Tree

Common Dataset Size Today

Future Goal: 10M node Tree of Life

You are here

Animals

Plants

Protists

Fungi

Paper Comparison: Multiple Trees

focus

context
TreeJuxtaposer

- side by side comparison of evolutionary trees
- [video]
  - video/software downloadable from http://olduvai.sf.net/tj
TJ Contributions

• first interactive tree comparison system
  – automatic structural difference computation
  – guaranteed visibility of marked areas
• scalable to large datasets
  – 250,000 to 500,000 total nodes
  – all preprocessing subquadratic
  – all realtime rendering sublinear
• scalable to large displays (4000 x 2000)
• introduced
  – guaranteed visibility, accordion drawing
Matching Leaf Nodes

- rayfinned fish
  - salamander
  - frog
  - mammal
  - bird
  - crocodile
  - lizard
  - snake
  - turtle
  - lungfish

- rayfinned fish
  - lungfish
  - salamander
  - frog
  - turtle
  - snake
  - lizard
  - crocodile
  - mammal
  - bird
Matching Leaf Nodes

- rayfinned fish
  - salamander
    - frog
  - mammal
    - bird
    - crocodile
    - lizard
    - snake
    - turtle
  - lungfish

- rayfinned fish
  - lungfish
  - salamander
    - frog
    - turtle
    - snake
    - crocodile
    - mammal
    - bird
Matching Interior Nodes

Left:
- rayfinned fish
  - salamander
    - frog
  - mammal
    - bird
    - crocodile
    - lizard
    - snake
    - turtle
  - lungfish

Right:
- rayfinned fish
  - lungfish
  - salamander
    - frog
  - turtle
  - snake
  - lizard
  - crocodile
  - mammal
  - bird
Matching Interior Nodes

- rayfinned fish
  - salamander
  - frog
  - mammal
    - bird
      - crocodile
      - lizard
      - snake
    - turtle
  - lungfish

- rayfinned fish
  - lungfish
  - salamander
    - frog
  - mammal
    - turtle
      - snake
      - lizard
    - mammal
      - bird
Matching Interior Nodes
Matching Interior Nodes

- rayfinned fish
  - salamander
  - frog
  - mammal
    - bird
    - crocodile
    - lizard
    - snake
    - turtle
  - lungfish
- rayfinned fish
  - lungfish
  - salamander
    - frog
    - turtle
    - snake
    - crocodile
    - mammal
    - bird
Previous Work

• tree comparison
  – RF distance [Robinson and Foulds 81]
  – perfect node matching [Day 85]
  – creation/deletion [Chi and Card 99]
  – leaves only [Graham and Kennedy 01]
Similarity Score: $S(m,n)$

$L(m) = \{E, F\}$

$L(n) = \{D, E, F\}$

$$S(m,n) = \frac{|L(m) \cap L(n)|}{|L(m) \cup L(n)|} = \frac{|\{E, F\}|}{|\{D, E, F\}|} = \frac{2}{3}$$
Best Corresponding Node

- $BCN(m) = \arg\max_{v \in T_2} (S(m, v))$
  - computable in $O(n \log^2 n)$
  - linked highlighting

$BCN(m) = n$
Marking Structural Differences

- Nodes for which $S(v, \text{BCN}(v)) \neq 1$
  - Matches intuition
Outline

• Accordion Drawing
  – information visualization technique

• TreeJuxtaposer
  – tree comparison

• SequenceJuxtaposer
  – sequence comparison

• PRISAD
  – generic accordion drawing framework
Genomic Sequences

- multiple aligned sequences of DNA
- now commonly browsed with web apps
  - zoom and pan with abrupt jumps
  - previous work
    - Ensembl [Hubbard 02], UCSC Genome Browser [Kent 02], NCBI [Wheeler 02]
- investigate benefits of accordion drawing
  - showing focus areas in context
  - smooth transitions between states
  - guaranteed visibility for globally visible landmarks
SequenceJuxtaposer

- comparing multiple aligned gene sequences
- provides searching, difference calculation
- [video]
  - video/software downloadable from http://olduvai.sf.net/tj
Searching

- search for motifs
  - protein/codon search
  - regular expressions supported
- results marked with guaranteed visibility
Differences

• explore differences between aligned pairs
  – slider controls difference threshold in realtime
• results marked with guaranteed visibility
SJ Contributions

• fluid tree comparison system
  – showing multiple focus areas in context
  – guaranteed visibility of marked areas
    • thresholded differences, search results

• scalable to large datasets
  – 2M nucleotides
  – all realtime rendering sublinear
Outline

• Accordion Drawing
  – information visualization technique
• TreeJuxtaposer
  – tree comparison
• SequenceJuxtaposer
  – sequence comparison
• PRISAD
  – generic accordion drawing framework
Goals of PRISAD

• generic AD infrastructure
  – tree and sequence applications
    • PRITree is TreeJuxtaposer using PRISAD
    • PRISeq is SequenceJuxtaposer using PRISAD

• efficiency
  – faster rendering: minimize overdrawning
  – smaller memory footprint

• correctness
  – rendering with no gaps: eliminate overculling
PRISAD Navigation

• generic navigation infrastructure
  – application independent
  – uses deformable grid
  – split lines
    • Grid lines define object boundaries
  – horizontal and vertical separate
    • Independently movable
Split line hierarchy

- data structure supports navigation, picking, drawing
- two interpretations
  - linear ordering
  - hierarchical subdivision
PRISAD Architecture

world-space discretization
• preprocessing
  • initializing data structures
  • placing geometry

screen-space rendering
• frame updating
  • analyzing navigation state
  • drawing geometry
World-space Discretization

interplay between infrastructure and application
Laying Out & Initializing

- application-specific layout of dataset
  - non-overlapping objects
- initialize PRISAD split line hierarchies
  - objects aligned by split lines
Gridding

• each geometric object assigned its four encompassing split line boundaries
Mapping

- PRITree mapping initializes leaf references
  - bidirectional $O(1)$ reference between leaves and split lines
Screen-space Rendering

control flow to draw each frame
Partitioning

• partition object set into bite-sized ranges
  – using current split line screen-space positions
    • required for every frame
  – subdivision stops if region smaller than 1 pixel
    • or if range contains only 1 object

\[
\{ [1,2], [3,4], [5] \}
\]

Queue of ranges
Seeding

- reordering range queue result from partition
  - marked regions get priority in queue
    - drawn first to provide landmarks
Drawing Single Range

• each enqueued object range drawn according to application geometry
  – selection for trees
  – aggregation for sequences
PRITree Range Drawing

• select suitable leaf in each range
• draw path from leaf to the root
  – ascent-based tree drawing
  – efficiency: minimize overplotting
• only draw one path per range

```
{ [3,4], [5], [1,2] }
```

```
[3,4]
```

![Diagram of a tree with ranges and labels]
Rendering Dense Regions

- correctness: eliminate overculling
  - bad leaf choices would result in misleading gaps
- efficiency: maximize partition size to reduce rendering
  - too much reduction would result in gaps

Intended rendering

Partition size too big
Rendering Dense Regions

- correctness: eliminate overculling
  - bad leaf choices would result in misleading gaps
- efficiency: maximize partition size to reduce rendering
  - too much reduction would result in gaps

Intended rendering

Partition size too big
PRITree Skeleton

- guaranteed visibility of marked subtrees during progressive rendering

  first frame: one path per marked group

  full scene: entire marked subtrees
PRISeq Range Drawing: Aggregation

- aggregate range to select box color for each sequence
  - random select to break ties
PRISeq Range Drawing

- collect identical nucleotides in column
  - form single box to represent identical objects
    - attach to split line hierarchy cache
    - lazy evaluation

- draw vertical column

{ A:[1,1], T:[2,3] }
PRISAD Performance

- PRITree vs. TreeJuxtaposer (TJ)
- synthetic and real datasets

  - complete binary trees
    - lowest branching factor
    - regular structure

  - star trees
    - highest possible branching factor
InfoVis Contest Benchmarks

- two 190K node trees
- directly compare TJ and PT
OpenDirectory benchmarks

- two 480K node trees
- too large for TJ
PRITree Rendering Time Performance

TreeJuxtaposer renders all nodes for star trees:
• branching factor $k$ leads to $O(k)$ performance

InfoVis 2003 Contest dataset:
• 5x rendering speedup leads to $O(k)$ performance
Detailed Rendering Time Performance

PRITree handles 4 million nodes in under 0.4 seconds.

- TreeJuxtaposer takes twice as long to render 1 million nodes.
Memory Performance

1GB difference for both applications comparison
- marked range storage changes improve scalability
- linear memory usage for both applications
- 4-5x more efficient for synthetic datasets
Performance Comparison

• PRITree vs. TreeJuxtaposer
  – detailed benchmarks against identical TJ functionality
    • 5x faster, 8x smaller footprint
    • handles over 4M node trees

• PRISeq vs. SequenceJuxtaposer
  – 15x faster rendering, 20x smaller memory size
  – 44 species * 17K nucleotides = 770K items
  – 6400 species * 6400 nucleotides = 40M items
Future Work

• future work
  – editing and annotating datasets
  – PRISAD support for application specific actions
    • logging, replay, undo, other user actions
  – develop process or template for building applications
PRISAD Contributions

- infrastructure for efficient, correct, and generic accordion drawing
- efficient and correct rendering
  - screen-space partitioning tightly bounds overdrawing and eliminates overculling
- first generic AD infrastructure
  - PRITree renders 5x faster than TJ
  - PRISeq renders 20x larger datasets than SJ
Joint Work

• TreeJuxtaposser
  – François Guimbretière, Serdar Taşiran, Li Zhang, Yunhong Zhou
    • SIGGRAPH 2003

• SequenceJuxtaposser
  – James Slack, Kristian Hildebrand, Katherine St.John
    • German Conference on Bioinformatics 2004

• PRISAD
  – James Slack, Kristian Hildebrand
    • IEEE InfoVis Symposium 2005
Open Source

• software freely available from http://olduvai.sourceforge.net
  – SequenceJuxtaposer
    olduvai.sf.net/sj
  – TreeJuxtaposer
    olduvai.sf.net/tj
  – requires Java and OpenGL
    • GL4Java bindings now, JOGL version coming soon

• papers, talks, videos also from http://www.cs.ubc.ca/~tmm