Hierarchy Vis cs533c 2005

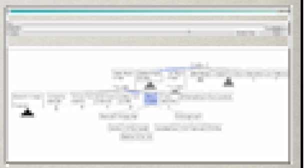
By
Andrew A Carbonetto

Papers

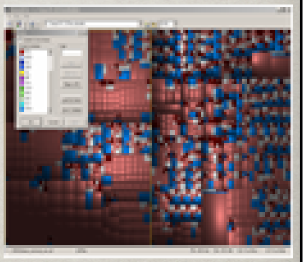
- Multitrees: Enriching and reusing hierarchical structures. George W. Furnas and Jeff Zacks, SIGCHI 1994 , pp 330-336.
- Polyarchy Visualization: Visualizing multiple intersecting hierarchies George G. Robertson, Kim Cameron, Mary Czerwinski, and Daniel Robbins. Information Visualization, 1(1), p.50-65, 2002

- Why Hierarchy Vis is first?

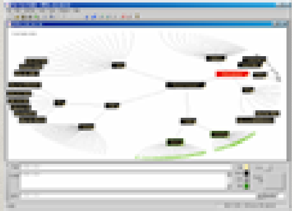
- Space-Tree
- SequoiaView (or Cushion TreeMap)
- H3
- TreeViewer
- Star-Tree



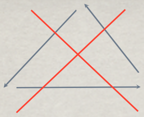
- Space-Tree
- SequoiaView (or Cushion TreeMap)
- H3
- TreeViewer
- Star-Tree



- Space-Tree
- SequoiaView (or Cushion TreeMap)
- H3
- TreeViewer
- Star-Tree

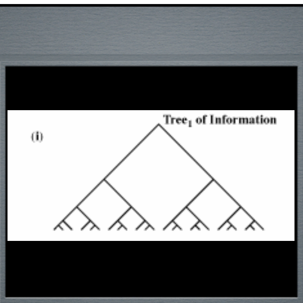


- Trees are easily visualized.
- Trees are not versatile enough

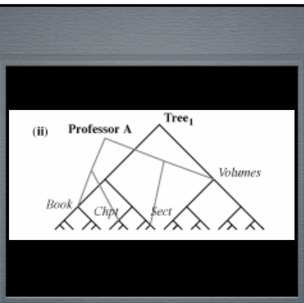


- Hierarchical Vis: Concerned with DAGs
- But there are DAGs that cannot be easily visualized (eg some DAGs cannot be put on a plane without crossovers)

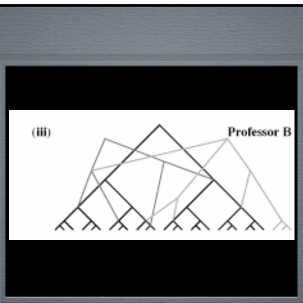
- Multitrees:
  - Is a DAG and not a tree
  - Easy reuse of data



Creating a Multitree



Add another Tree On top of the Old

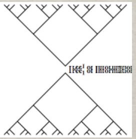


Continue...

- Proposition I: The following properties are equivalent:
  - The DAG can be constructed by adding new tree structure above existing (or newly added) disjoint complete subtrees.
  - The DAG is diamond free
  - The descendants of any node form a tree.
  - The ancestors of any node form an inverted tree

- Proposition I: The following properties are equivalent:
  - The DAG can be constructed by adding new tree structure above existing (or newly added) disjoint complete subtrees.
  - The DAG is diamond free
  - The descendants of any node form a tree.
  - The ancestors of any node form an inverted tree

- Proposition I: The following properties are equivalent:
  - The DAG can be constructed by adding new tree structure above existing (or newly added) disjoint complete subtrees.
  - The DAG is diamond free
  - The descendants of any node form a tr
  - The ancestors of any node form an inverted tree



- At any one node, we have a *topological tree* (t-tree)
- Actually, this can be extended to a set of points along a path from *x* to *y*, where *x <= y*

- Proposition II: Consider any two nodes *x>=y* in a multitree, and the necessarily unique path connecting them. The union of all the ancestors of this path and all the descendants of this path is a topological tree.

FishEye View
Similar:Space-Tree?

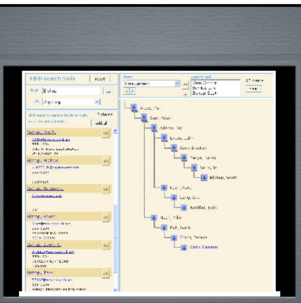
- Multitrees are Great
  - More options then trees
  - Reuse of data
  - Ability to view Ancestors & Descendants in a tree-like fashion

- Multitrees are Bad:
  - Diamonds are Forever... (and local Multitrees)
  - Cannot view the whole Multitree
  - Reused data is static
  - Difficult construction

- Animated Visualization of Multiple Intersecting Hierarchies

- Multiple Intersecting Hierarchy : Polyarchy
  - Data is replicated at several nodes among several hierarchies
- Metadirectory
  - Intersecting data is organized into a metadirectory for *easy* referencing

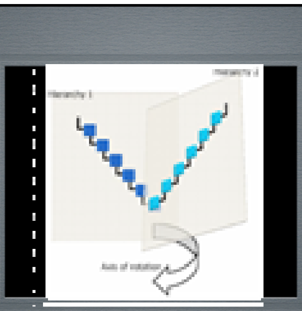
- Polyarchy Visualization:
  - Viewing points that are distributed amongst several hierarchies



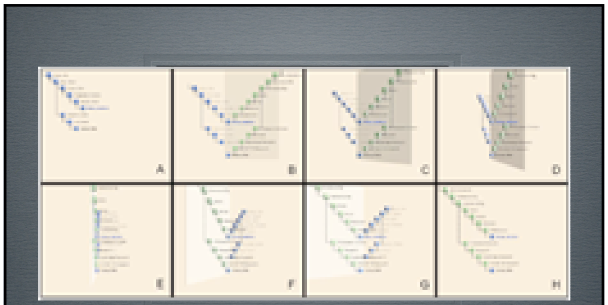
Polyarchy

Stacked View

- Goals:
  - Show how instances in each database (hierarchy) relate to each other
  - Simple transition from one hierarchy view to another
  - Help understand the relationships between several hierarchy views



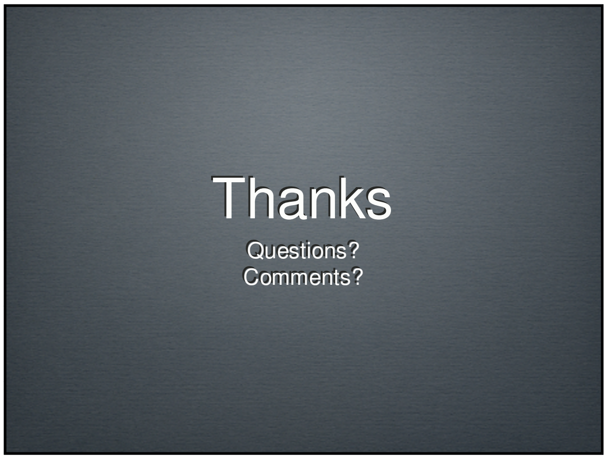
Pivot Point



Horizontal Animation

- Pros
  - User study to determine best approach
  - Comprehensive visualization of a complicated structure
  - Searchable
  - Superset of Multitrees...

- Cons
  - No order to databases
  - Pivots only around one point
  - Text gets cluttered during animation
  - No general browse option
  - Doesn’t exploit any other infovis sources.



Thanks

Questions?
Comments?