Evaluating and Understanding View Navigation

Dr. Margaret-Anne Storey
Dept of Computer Science, UVic

Outline

- Effective View Navigation

Effective View Navigation (Furnas)

- *View navigation* – describes how a user moves about an information structure by **selecting** something in the current view of the structure
- Furnas looks at the fundamental requirements for effective view navigation
  - Views must be “small”
  - Moving around must not take too many steps
  - Route to any target must be easy to discover
- We can use these fundamental requirements to help us analyze and compare existing approaches for view navigation

Note: These techniques are more applicable to static large information structures

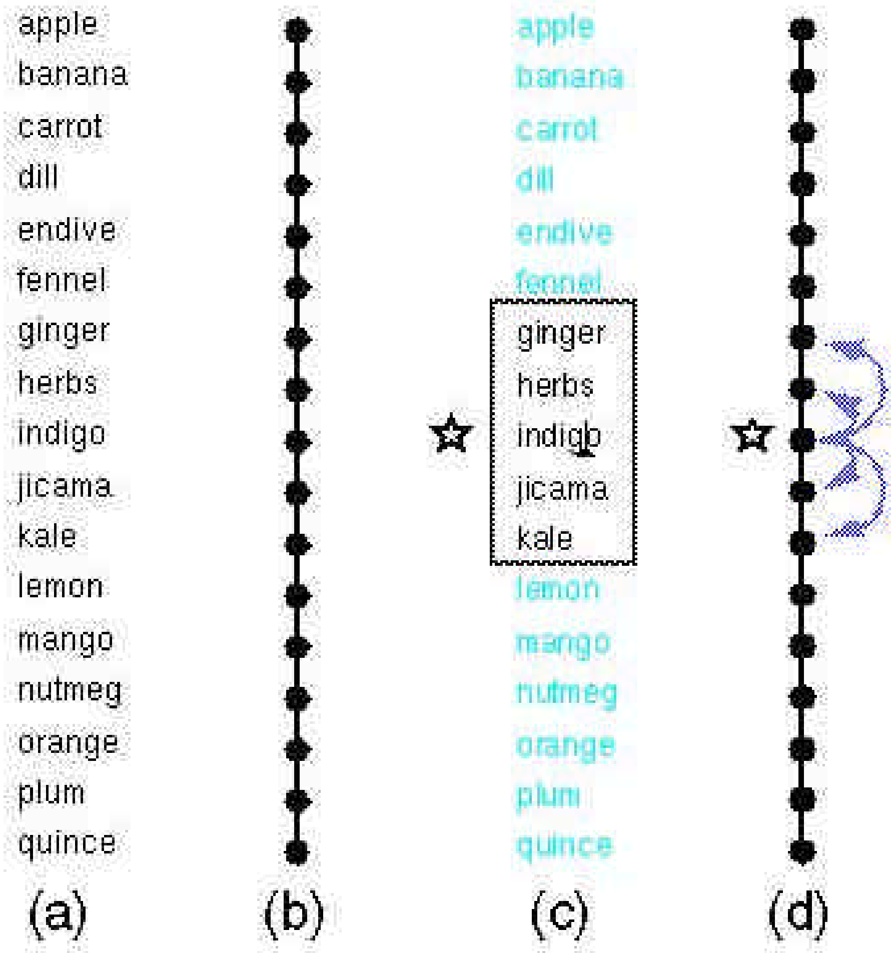
Effective View Navigation (2)

- *View traversal* – underlying iterative process of viewing, selecting something seen, moving to it, thereby forming a path in the structure
- *View navigation* – encompasses view traversal but also includes the process of how to decide where to go next (i.e. how to choose a good route from the available selections)

Logical Structure Graph and Viewing Graph

- *Logical Structure Graph* -- Assume that elements in an information space are organized in a logical structure as dictated by the semantics of the domain
- *Viewing Graph* – It has a node for each node in the logical structure, and there is a directed link between a pair of nodes (*i*,*j*) if the view from *i* includes *j*

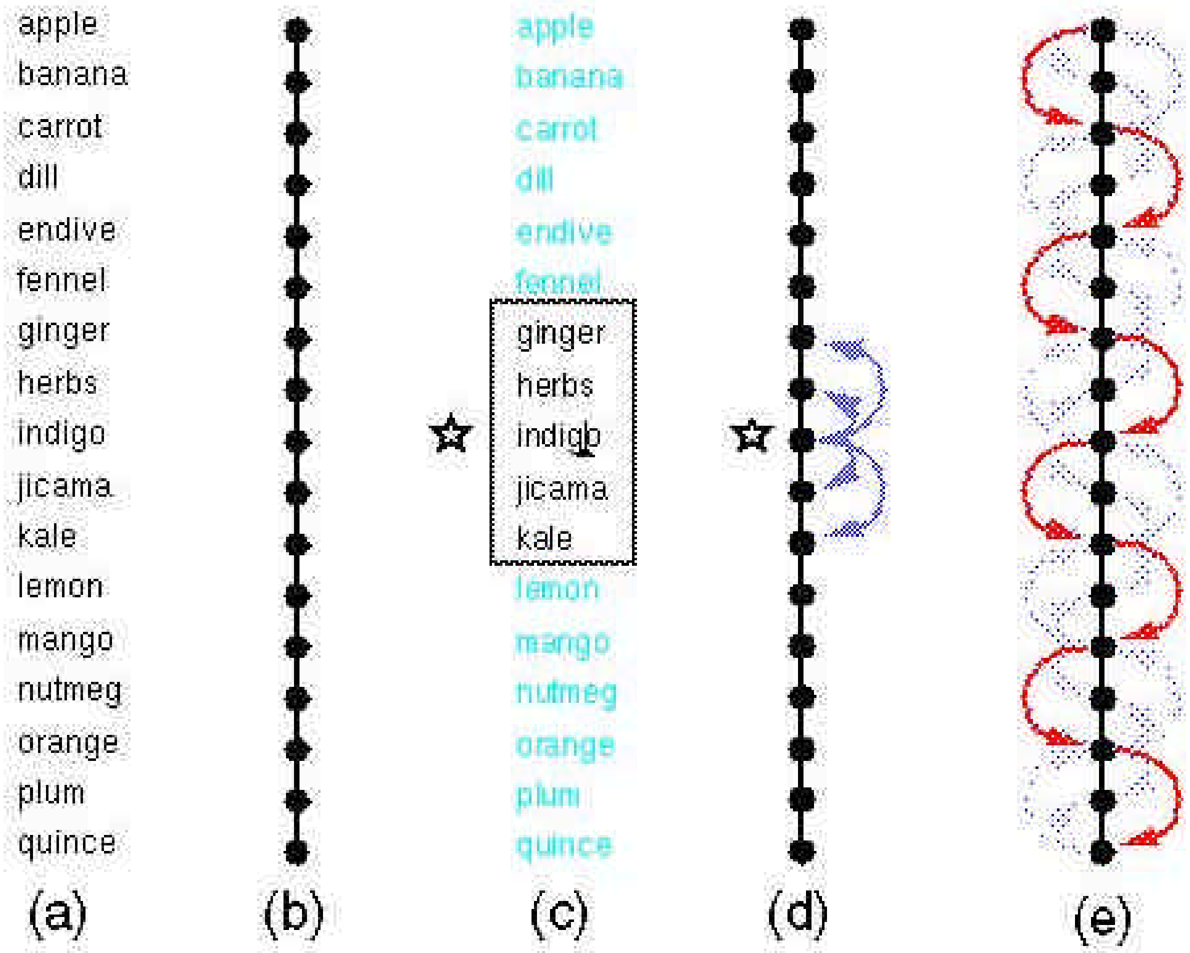
Logical and Viewing graphs



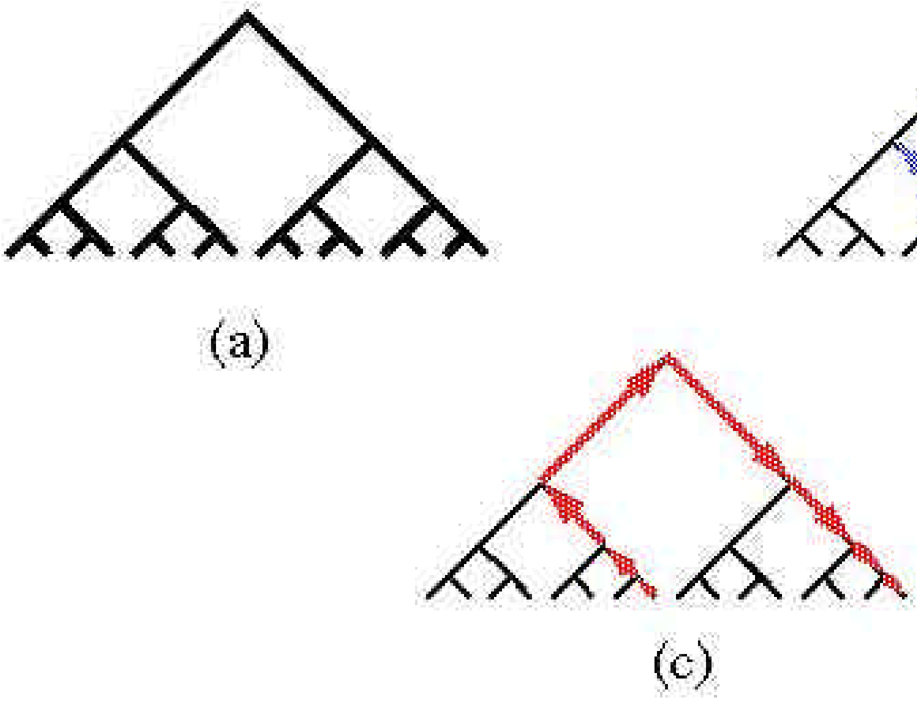
Requirements for Efficient View Traversal

- *Small views:* the number of out-going links in the viewing graph must be “small” compared to the size of the structure
  - Maximal out degree (MOD) will characterize how well this requirement is met
- *Short paths:* The distance (number of links) between all pairs of nodes in the viewing graph must be “small” compared to the size of the structure
  - The Diameter (DIA, longest connecting path required between any pair of nodes) characterizes how well this requirements is met
- A viewing graph is efficient if it meets both of these requirements
  - Scrolling list – MOD = $O(1)$, DIA = $O(n)$
  - But trees – MOD = O (log n), DIA = $O(\log n)$
- But even for information with poor logical structures, we can craft the viewing graph to have an improved EVT

Diameter of a scrolling list…



Efficiency of a tree structure



(a)

(b)

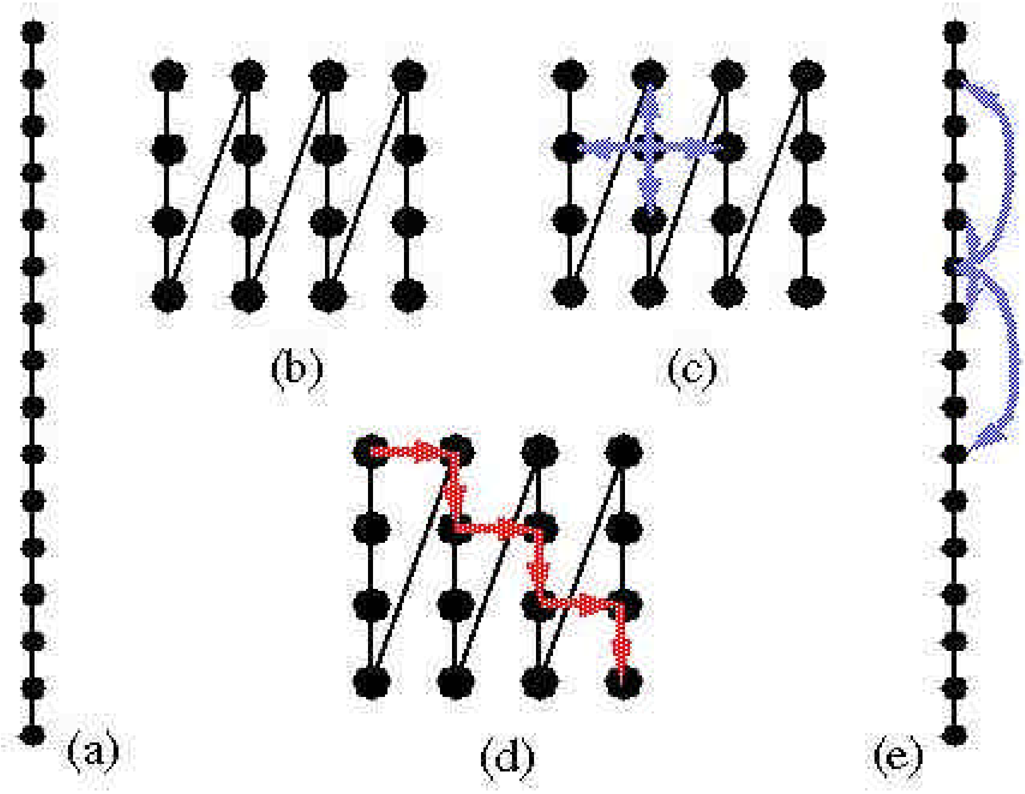
(c)

MOD = O (1), DIA = $O(\log n)$

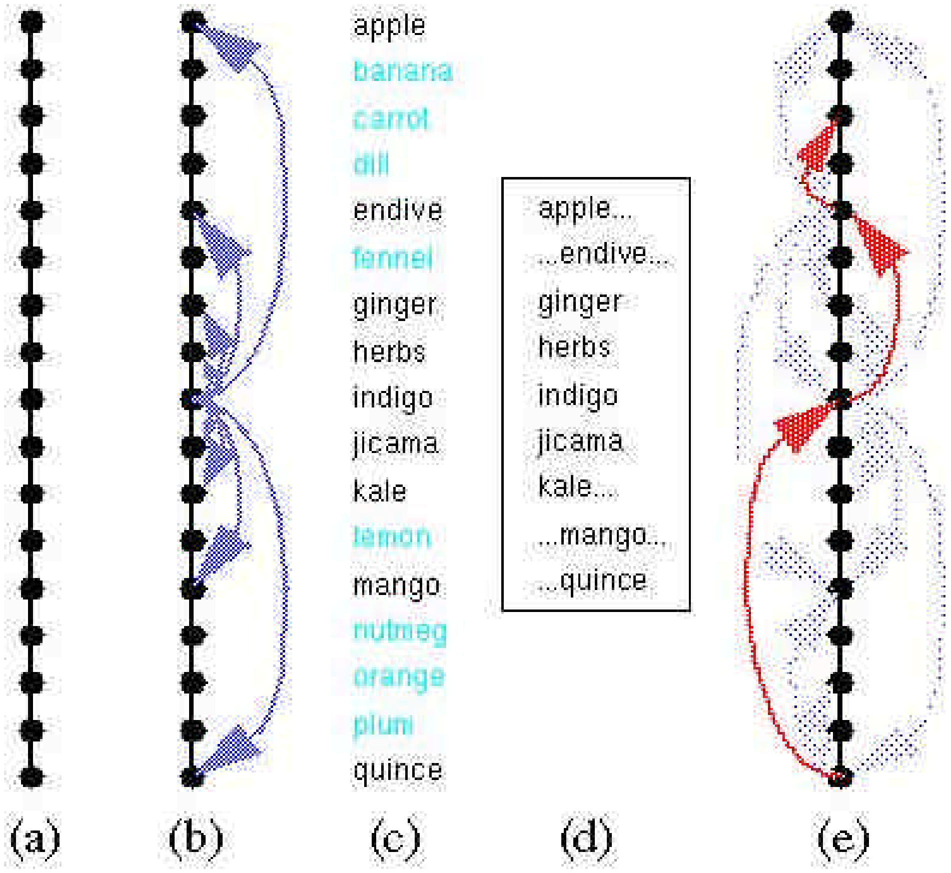
Fixing Non-EVT Structures

- Add an extra dimension to the links
  - E.g. for the list example, fold it into two dimensions, making a multi-column list
    - The outdegree (MOD) is still constant, but the DIA is now sublinear (sqrt(*n*))
  - Fisheye sampling
    - Nodes can be viewed using geometric sampling
    - MOD is *O*(log *n*), DIA is *O*(log *n*)
  - Tree augmentation (adding a tree structure to a list)
    - MOD is *O*(1), DIA is *O*(log *n*)
  - Note the use of a zooming interface changes diameter of a space from *O*(sqrt(*n*)) to *O*(log *n*)

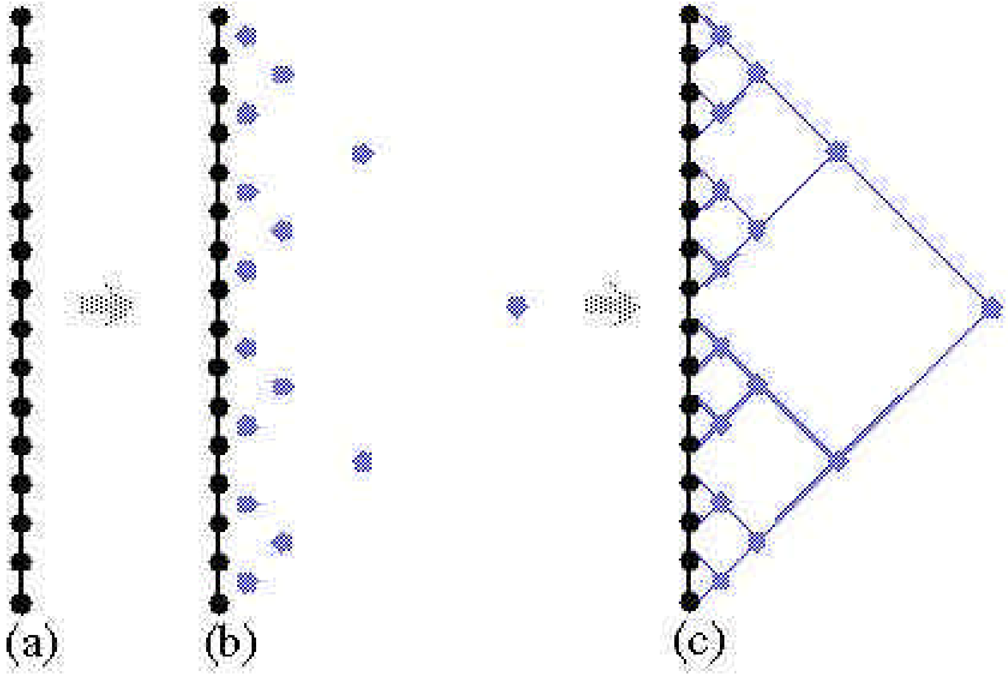
Folding a scrolling list



Fisheye sampling of a scrolling list



Adding a tree structure



EVT Remarks

- Always consider putting a traversable infrastructure on an otherwise unruly information structure
  - Done all the time for webpages!
- Jump-and-show – arbitrary navigation steps can be very powerful (e.g. search)
- But EVT is not enough, if a user can't find the right path to take…
  - We need to be able to read the structure to find the right path, the structure needs to be *View Navigable*

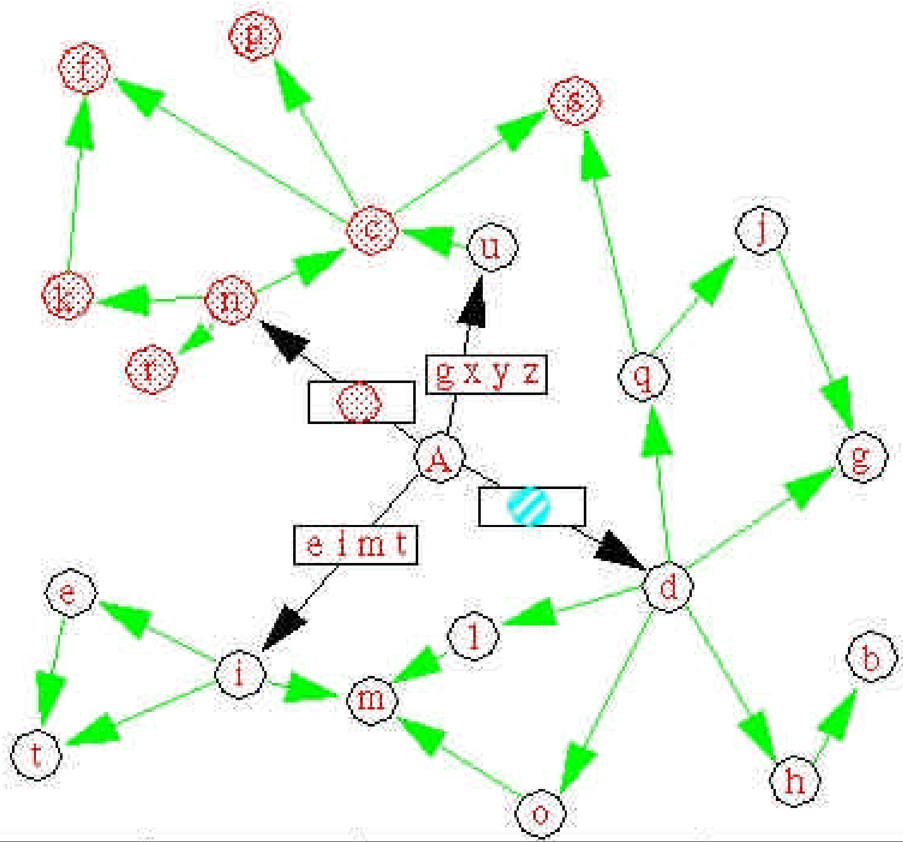
Navigability

- *Strong navigability* requires that the structure and its outlink-info allow one to
  - To find the shortest path to the target
  - Without error
  - And by looking only at what is visible at the current node (i.e. not using history information)

Definitions

- *to-set* (link) = all the targets that the link efficiently leads to
- *inferred-to-set (*link) = all the target nodes that the associated-outlink-info indicates is down that path

Strong Navigability



Link	A-->n	A-->u	A-->i	A-->d
to-set	c,f,k,n,p,r,s	c,f,p,s,u	e,i,m,t	b,d,g,h,j,l,o,q,s
outlink-info	●	g x y z	e i m t	▨
inferred to-set	c,f,k,n,p,r,s	g,x,y,z	e,i,m,t	?

Definitions (2)

- The outlink-info of a node is well-matched with respect to a target if
  - its outlink-info is not misleading with respect to the location of the target and
  - the target is in the inferred-to-set of at least one outlink
- The outlink-info of a node is well-matched *iff* it is well-matched with respect to all possible targets

Navigability Requirements (1)

Note: A user is always guaranteed to find the shortest paths to all targets *iff* the outlink-info is well-matched everywhere

Requirement 1 for Navigability:

The outlink-info must be everywhere well matched

Therefore, the outlink-info of a link must somehow describe the whole set of nodes it links to, not just the next node

The *Scent* of Information

- Instead of considering a user trying to find a target, consider instead the target trying to pull the user towards it

Requirement 1 for Navigability restated:

Every node must have good residue (scent) at every other node

but:

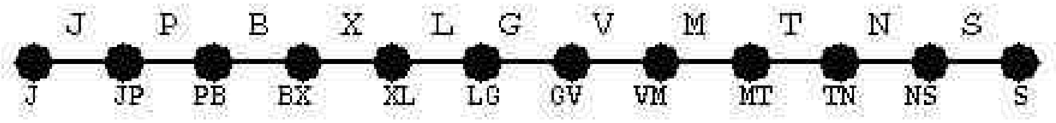
Requirement 2 for Navigability:

Outlink-info must be "small"

The *Scent* of Information (2)

- We can’t enumerate every target for each outlink-info, but we can use more conceptually complex representations such as attributes (red) and abstractions (living things)
    - Classification again – leads us back to trees!
- Unfortunately it is not always possible to classify things….
- However, the notions of traversability and navigability for trees shows us the importance of the ontologies used by Yahoo and other information rich resources

What navigating the web could be compared to….



Effective View Navigability

- Recall: *View navigation* – encompasses view traversal but also includes the process of **how to decide** where to go next
- Therefore we need both the mechanics of EVT on diameter and outdegree and the residue constraints of VN to hold

Empirical observations

- User interface experiment showed multiple windows were slower to complete tasks than a non-windowed system
  - Window management distracts users from their tasks and uses up time
  - More effective window management doesn't receive enough attention… unfortunately
  - Structural relations between windows are not exploited to help provide between window placement strategies
- Lack of user studies (still) -- We need more empirical observations for different domains/tasks, and for the different techniques
- Things to measure:
  - Ease of learning
  - Ease of use
  - Task completion times

References

- Effective View Navigation, see the textbook