

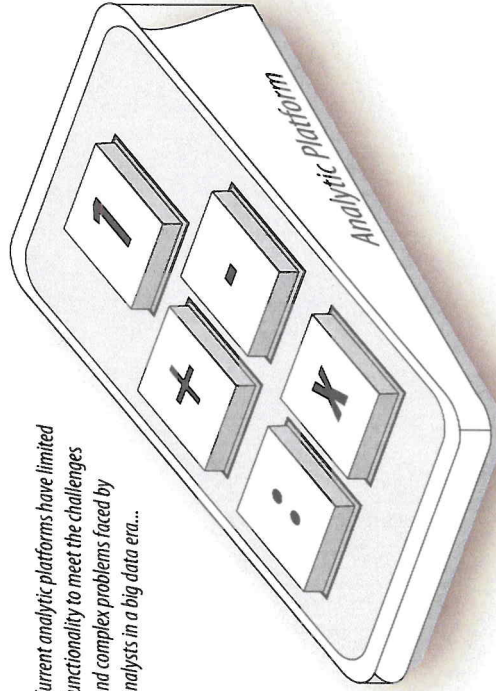
Pathfinder Toolkit (PTk)

A conceptual design for an advanced graph analytics toolkit

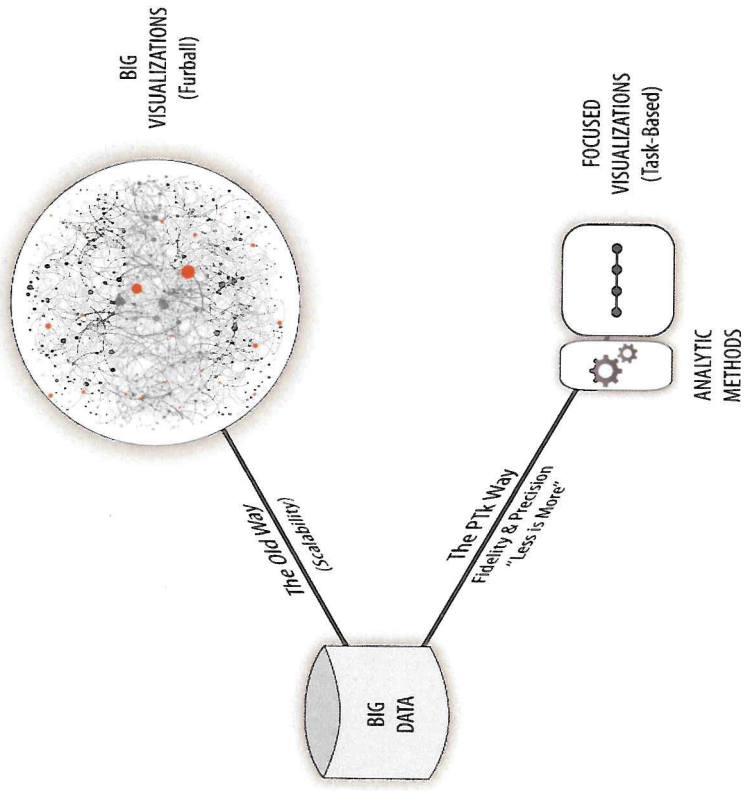
Identity Discovery
Challenge

Challenge to Innovate

Current analytic platforms have limited functionality to meet the challenges and complex problems faced by analysts in a big data era...



Consider this...



Pathfinder Toolkit: Analytic Methods + Task-Based Visualization

Path Finder

Load Data:

Select Start Node(s):

Select Destination Node(s):

Select Via Node(s):

Path Direction and Weights

Use Directionality:

Edge Weights: similarity distance none

Path Filters

Node Properties: includes avoids

Edge Properties: includes avoids

Path Properties: includes avoids

Knock Outs: remove high frequency (SMA) nodes

Path Distance (Min/Max):

Path Segments (Min/Max):

Transitive Closure

Node Merge: merge very similar nodes to one?

Threshold:

Edge Insert: add edges (A->C)

Algorithm: Warshall

Pattern Recognition

Structural Proximity: compute frequent pattern neighbors?

structural loop search (loop)?

frequency descending mining (floop)?

gPrune

Visual Encoders

Layouts: Road & Stack V-Stack Order by: Best-Fit Scale by: Fitting Highlight Shared:

Actions

Node Properties

Field A:

Field B:

Field C:

Field D:

Field E:

Field F:

Field G:

Field H:

Edge Properties

LN-1: LN-2:

FN-1: FN-2:

MN-1: MN-2:

ADDR-1: ADDR-2:

PHONE-1: PHONE-2:

Path / Network Properties

Measures: uniqueness: clusters:

Analytic Methods

What am I looking for?

How can I configure the finder for my specific scenario?

How can I reduce the noise and enhance the signal with filters and interference detection?

Are there patterns which help to discover meaningful structural or topological characteristics to support the sense-making process?

Path Finder

Load Data:

Select Start Node(s):

Select Destination Node(s):

Select Via Node(s):

Path Direction and Weights

Use Directionality:

Edge Weights: similarity distance none

Path Filters

Node Properties: includes avoids

Edge Properties: includes avoids

Path Properties: includes avoids

Knock Outs: remove high frequency (SMA) nodes

Path Distance (Min/Max):

Path Segments (Min/Max):

Transitive Closure

Node Merge: merge very similar nodes to one?

Threshold:

Edge Insert: add edges (A -> C)

Algorithm: Warshall

Pattern Recognition

Structural Proximity: compute frequent pattern neighbors?

structural leap search (sLeap)?

frequency descending mining (fLeap)?

gPrune

How should path finder interpret the relations as represented in the data?

Is there a way to simplify a complex path by merging nodes or inserting edges?

Show me the paths...

Path Finder

Load Data:

Select Start Node(s):

Select Destination Node(s):

Select Via Node(s):

We see this module as similar to a GPS or Directions mapping interface.

Start Node: SEED-1
Destination Node: SEED-2

Via Nodes would be useful for more complex scenarios where intermediate points are known.

We know there are algorithms for shortest path analysis, but how can they be extended to this analytic use case?

Path Direction and Weights

Use Directionality:

Edge-Weights: similarity distance none

The challenge problem is comprised of edges derived from an entity resolution process.

They are not directional, however they are weighted.

A score of 0 to 1 indicates "similarity" of records, e.g. 1=Exact Match

These controls would be useful to parameterize the algorithm's treatment of weights

Show me only...

← Path Filters

Node Properties: includes avoids

Edge Properties: includes avoids

Path Properties: includes avoids

Knock Outs: remove high frequency (SNA) nodes

Path Distance (Min/Max):

Path Segments (Min/Max):

Path filters are intended to fine-tune path features (e.g. avoid highways) and reduce noise by eliminating possibilities.

Includes and avoids by property reduce graph complexity.

Knock Outs is a conceptual concept to represent the possible use of SNA Centrality measures to detect nodes which negatively (and erroneously) affect path structure.

For example, entity resolution often highlights the high frequency of common entities whose linkages are not relevant to the analytic objective.

Path Distance (or Similarity) can be used to filter the strength of the path (e.g. show only highest scores)

Path Segments can be used to filter the complexity of the path. (e.g. show only paths of 5 or more segments)

Simplify my paths...

Transitive Closure

Node Merge: merge very similar nodes to one?

Threshold:

Edge Insert: add edges (A -> C)

Algorithm:

Transitive Closure offers possibilities to simplify a complex graph structure by merging nodes (based on criteria) and inserting edges (shortening paths).

Node merge is based on a decision criteria such as:

*if name A is similar to name B by score of 0.8 or above
AND
address A is similar to address B by a score of 0.71 or above,
then MERGE*

Edge inserts may also be employed based on a logical criteria or use of algorithms such as Warshall and PageRank (are there others?).

These possible methods are analogous to cartographic use of simplification methods such as the Douglas-Peucker algorithm which "splits and merges" complex vector paths.

However, these methods are very sensitive and could affect the analytic story (verbosity).

Show me interesting patterns...

Pattern Recognition

- Structural Proximity:
- compute frequent pattern neighbors?
 - structural leap search (sl Leap)?
 - frequency descending mining (fLeap)?
 - yPrune

We know very little about the possibilities of graph-based pattern recognition.

However, research suggests some advancements in structural / topological pattern discovery approaches.

Are they applicable?

Do methods exist to cluster, classify, associate, or otherwise detect "interesting" features of a graph structure?

We don't know the answers to these questions. Do you?

Task-Based Visualization: Path Mapper (LoD 0: Overview First)

Based on a path finder configuration, how are the parameters effecting the results — what is the big picture?

Level of Detail (LOD) 0
should provide a bird's eye view, to include:

- Path Depth (Segments)
- Path Breadth (Candidates)
- Edge Sensitivity (Weak Links)
- Edge Strength (Strong links)
- Node Centrality (Dominance)
- Connectedness (Shared nodes/edges)
- Distinctness (Unique paths)
- Path Distance

Any new ideas for an effective visualization approach?

*I'm programmed to be a treasure hunter,
not a caveman*

— KITT, Knightrider