Using KOJAK Link Discovery Tools to Solve the Cell Phone Calls Mini Challenge

Hans Chalupsky*
USC Information Sciences Institute

ABSTRACT

We present a brief summary of the process and tools employed to generate a submission to the VAST-08 Cell Phone Calls Mini Challenge. The primary system used was KOJAK, which is an integrated suite of link discovery tools that support group detection [1], anomaly detection [4], pattern matching and graph simplification. KOJAK generally operates on data represented by semantic graphs where nodes model typed entities such as persons, organizations, events, etc. and links represent different kinds of relationships between entities. Graphs might be stored explicitly, or implicitly as views over relational data stored in a database. Optionally, graphs can be augmented with background ontologies and logic-based inferencing via the PowerLoom knowledge representation and reasoning system [3]. KOJAK is not a visualization tool per se, yet its components can provide powerful support for visual analytics tasks. In particular, the KOJAK Simplifier allows the reduction and abstraction of large data graphs for purposes of visualization, allowing an analyst to see the most important nodes and connections of a complex network. Based on these data reduction techniques we generated network visualizations that led to useful hypotheses and insights and allowed us to uncover the hidden group structure.

Index Terms: H.2.8 [Database Management]: Database Applications—Data Mining; I.2.3 [Artificial Intelligence]: Deduction and Theorem Proving—Inference Engines

ANALYSIS PROCESS SUMMARY

The main motivation for our participation in the VAST-08 Phone Calls Mini Challenge was to test and, hopefully, demonstrate that a link discovery system such as KOJAK could usefully support an interactive and exploratory visual analytics task. In particular, we were interested whether the recently developed KOJAK Simplifier graph simplification component could generate visualizations that were useful for social network analysis. The process described here is very much in contrast to evaluations in the past, where KOJAK was used fully automatically, batch-processing large amounts of data (10^6 nodes, 10^8 links) with minimal user intervention.

The data contains about 10,000 records of phone calls between 400 different numeric caller IDs over a 10-day period (generating about 900 distinct links). Each call has associated *from*, *to*, *date/time*, *duration* and *cell tower* fields. A map gives approximate locations for cell towers and a terse one-paragraph background story mentions a few key players. We translated the data into a CSV format to load it into a KOJAK graph and a logic format to allow PowerLoom queries and inference.

After some initial exploration, we started to experiment with the KOJAK UNICORN tool [4] to find anomalous nodes in the connection graph. UNICORN operates on semantic graphs and works best if there are different types of relations between nodes represented by different edge labels. It combines these labels in all possible ways (to a certain depth) and then computes correlations between a particular sequence of labels and a node. These correlation values are then combined into feature vectors called *semantic profiles* that characterize the graph neighborhood of a node. Once we have a semantic profile for each node, we use standard outlier detection to find nodes with abnormal profiles. KOJAK operations can be controlled via a Lisp-based command syntax allowing various options for each command. Commands are combined into scripts which can be configured and run from the KOJAK GUI. Here is the result of the first UNICORN run on the connection graph between callers:

```
|= (find-abnormal-nodes :graph the-graph
                        :ignore-edge-direction? true
                        :min-path-length 1 :max-path-length 4
                        :top-n 20 :k 1 :strong-outliers? true)
((<0>   0.99) (<306> 0.98) (<1>   0.90) (<5>  0.89)
 (<309> 0.88) (<23>  0.73) (<14>  0.71) (<2>  0.64)
 (<3>   0.63) (<8>   0.62) (<107> 0.62) (<19> 0.58)
 (<41>  0.58) (<200> 0.57) (<13>  0.53) (<52> 0.46)
 (<145> 0.42) (<360> 0.33) (<21>  0.32) (<11> 0.31))
```

The result list above shows the 20 most abnormal nodes in the connection graph ignoring call direction and frequencies. Each entry is a *(<caller ID> <abnormality score>)* pair. Edges around each node were explored to depth 4. The list contains node 200 which was the only ID mentioned in the background scenario. This gave us some confidence that UNICORN was picking up something useful in this data, such as unusual connectivity patterns (high/low path frequencies) and their combinations. To give UNICORN more label combinations to work with, we mapped call frequencies between parties onto symbolic high/medium/low ranges which seemed to work well.

UNICORN forms the basis of the KOJAK Simplifier tool which was used to generate the network visualizations used in our submission (e.g., see Figure 1). The Simplifier first computes a set of globally abnormal nodes and then a set of locally abnormal nodes connected to the global seeds. It then fills in the graph between the global and local nodes given some user-specified parameters. The result is a simplified graph that contains important nodes in the data with relevant connections between them, which, once visualized, shows important elements and structure of the data useful for analysis. Simplified graphs are generated as follows:

```
|= (simplify-graph :graph the-graph
            :ignore-edge-direction? true
            :min-path-length 1 :max-path-length 4
            :top-global-N 10 :top-local-N 5 :k 1
            :max-connect-length 2 :strong-outliers? true)
```

The options control how and how many global and local nodes are produced, and how edges are filled in in the resulting graph. Here we add all paths up to *max-connect-length* 2 between global and local abnormal nodes. It generally requires some experimentation with these parameters to generate a graph with the right amount of information. In future versions, this should be supported interactively with immediate feedback in the KOJAK GUI.

Initially, simplified graphs were visualized using KOJAK's own visualizer based on the Prefuse toolkit. Later on, we exported

*e-mail: hans@isi.edu

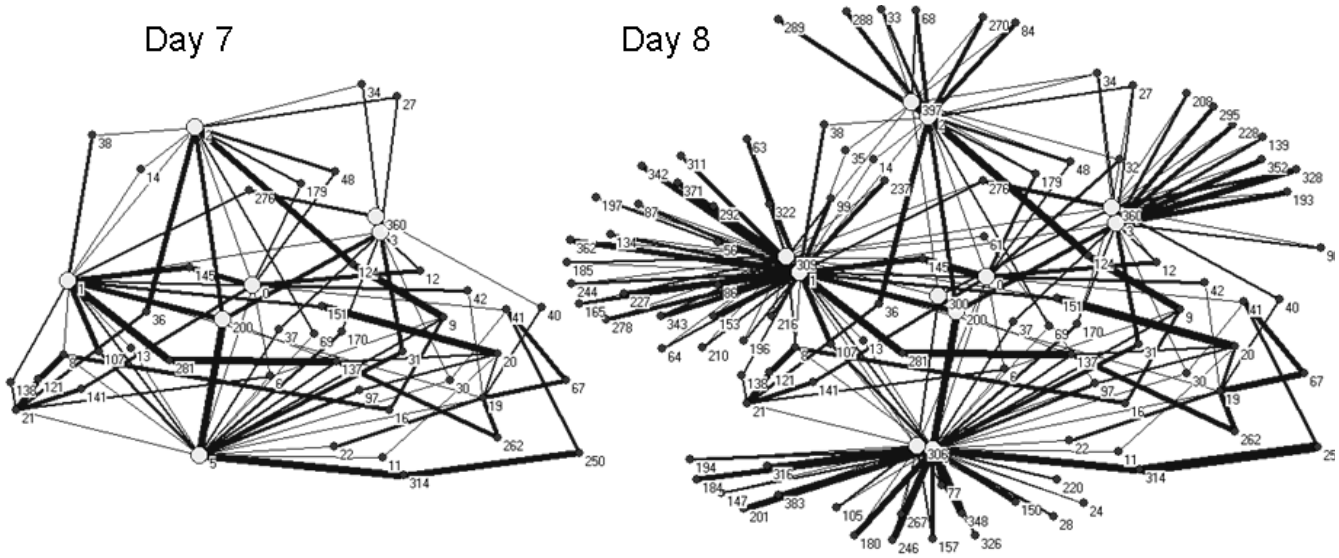

Figure 1: Significant network structure change from Day 7 to Day 8

graphs to Pajek [2] due to its greater choice of layout algorithms and visual markup features. Layouts like the ones shown in Figure 1 were generated automatically using Pajek's energy-based layout algorithms (with minor manual adjustments). Line thickness represents call frequency. Nodes we deemed important (e.g., 0, 200, 300, etc.) were emphasized manually with bigger circles.

A visualization of a simplified graph derived from all ten days of data (very similar to the one on the right in Figure 1) provided some key hypotheses and direction for further analysis. The graph showed very interesting structure with nodes 0, 200 and 300 in the middle and the double wheel patterns around 1, 2, 3 and 5 which made us suspect the centers to be aliases of the same person (note that this structure is *not* visible when looking at a layout of the unsimplified data). We further investigated these hypotheses by comparing connectivity patterns, and all the double wheel centers turned out to be very similar. Moreover, we started to consider that 0, 200 and 300 might all be aliases as well. Since similarity was only a somewhat weak indicator for aliases, we tried to see whether it was possible to refute them. For example, suspected aliases should not call each other, and they should never be at different places at the same time (unless phones can be shared).

Using cell tower location information and times between calls and a bit of PowerLoom modeling, we checked these constraints, which, unfortunately, were not satisfied even for a single phone assuming driving as the fastest way of transportation. As a side-effect, however, when we compared the calls of 1 and 309 using a PowerLoom query, we discovered that they occupied disjoint time ranges which in turn very much supported the initial alias hypotheses. The same was true for 200 and 300 (with 300 exclusively calling on days 8-10). While 200 and 300 didn't have any connectivity overlap, using the alias hypotheses based on strong similarity and a little bit of alias rule modeling in PowerLoom, we could easily see that they are connected to the same suspected core network.

Triggered by this, we visualized the temporal progression of the simplified Day-10 network based on its nodes (and layout) and filling in more and more links as days progressed. As expected, there is a big change between Day 7 and 8 as shown in Figure 1. All of the suspected aliases (except 360) appear the first time on Day 8. We do not have enough background information to produce a good explanation for this, except that maybe some external event forced the inner circle of the Catalano network to change to different phones, maybe to avoid detection. Another possible explanation is the complete takeover of the group by different players, but the alias hypothesis seemed more plausible to us. At this point, we had enough information to hypothesize assignments of names mentioned in the background story to phone IDs. Assuming we found the players of the core group, we then employed the GroupFinder to try to find the complete extent of the relevant Catalano network, by looking for those with the strongest connection to the inner circle.

In summary, network visualizations produced by first simplifying the data using KOJAK's graph simplifier and then visualizing the result using standard energy-based layout techniques produced important insights and hypotheses, such as, potential key players and aliases or double group structure. Moreover, time sliced visualizations could clearly demonstrate a significant change in group structure from Day 7 to 8. However, the visualizations by themselves were not sufficient, additional querying, modeling and what-if analysis was necessary to strengthen the analysis. Also, analysis was not always linear, e.g., temporal disjointness was discovered as a side-effect of a query and then further ascertained by creating supporting visualizations. KOJAK as it exists now provides much of the needed core functionality for this type of analysis, however, manual notes-keeping of all the different analysis threads, hypotheses, potential interactions, etc. is required and quite challenging. This provides interesting directions for future versions of the tool.

Acknowledgements

This work was funded in part by the Department of Homeland Security, ONR Grant number N00014-07-1-0149. Large portions of KOJAK's algorithms, code and good ideas were contributed by Jafar Adibi, Shou-de Lin, Eric Melz, Tom Russ and Andre Valente.

References

[1] J. Adibi, H. Chalupsky, E. Melz, and A. Valente. The KOJAK Group Finder: Connecting the dots via integrated knowledge-based and statistical reasoning. In *Proceedings of IAAI-04*, pages 800–807, 2004.

[2] V. Batagelj and A. Mrvar. Pajek – Program for Large Network Analysis. http://vlado.fmf.uni-lj.si/pub/networks/pajek.

[3] H. Chalupsky, R. MacGregor, and T. Russ. *PowerLoom Manual*. Available at http://www.isi.edu/isd/LOOM/PowerLoom.

[4] S. Lin and H. Chalupsky. Discovering and explaining abnormal nodes in semantic graphs. *IEEE Transactions on Knowledge and Data Engineering*, 20(8):1039–1052, 2008.