Supporting the Pattern Development Cycle in Intelligence Gathering

Michael Wolverton, Ian Harrison, John Lowrance, Andres Rodriguez, and Jerome Thomere
SRI International
Menlo Park, CA 94025
<lastname>@ai.sri.com

Abstract

To deal with noisy and incomplete data sets, analysts need tools that support an intelligence gathering *cycle*. In this cycle, the analyst (1) creates an initial pattern corresponding to his information need, (2) retrieves a collection of matching episodes in the data, (3) revises the pattern based on the shortcomings of the matches, and (4) repeats the process until the revised pattern is returning the right data. This paper discusses the cycle through a use case of the Link Analysis Workbench (LAW), a tool for discovering and analyzing situations of interest in large relational data sets.

1. Introduction

Intelligence analysts work with incomplete and noisy data. Their information needs are often hard to express, and almost impossible to get right the first time. Their information gathering is rarely a one-shot operation. Instead, the process is generally an evolutionary *cycle*, where the pattern of interest is constructed and then repeatedly refined based on results returned from the data. The analyst is heavily involved at all stages of the cycle. Supporting this cycle poses technical challenges for the tool developer, both in designing a pattern language flexible enough to describe both very specific and very general match criteria, and in producing a system that allows the analyst to define and refine patterns and interpret results quickly.

The next section describes the pattern development cycle as supported by the LAW system (Wolverton *et al.* 03). LAW features several characteristics that are important for this cycle:

- An intuitive pattern language based on *semantic graphs*.
- A simple *similarity metric*, based on graph edit distance (Bunke 97), which supports the retrieval and ranking of inexact matches.
- A pattern editor that supports easy editing of patterns, and a pattern library that allows users to construct complex hierarchical patterns out of simpler, previously defined ones.
- A match display interface designed to allow the user to understand at a glance the quality and content of a match to a pattern.

2. Pattern Development Cycle in LAW

In our hypothetical LAW use case, an analyst is searching for situations of interest within a very large dataset assembled from HUMINT, OSINT, and other sources. To stand in for an actual intelligence data set, we will use a data set created by the EAGLE program's PE Lab simulator (Schrag 2005). This simulator creates an abstract, artificial world in which individuals belonging to organizations (ThreatGroups and NonThreatGroups) perform various actions that, collectively, may constitute an attack. Many of the relations and actions are unobservable—that is, contained in the simulation's ground truth, but not in the data set. The data set in this example contains approximately 130,000 nodes and 330,000 links, and all the results discussed below represent the results of actual runs of the LAW matcher against this data set.

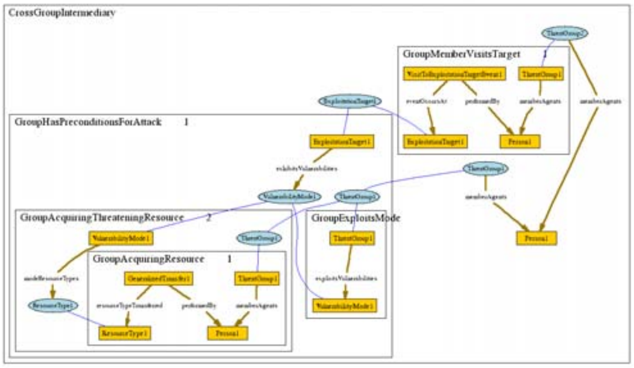
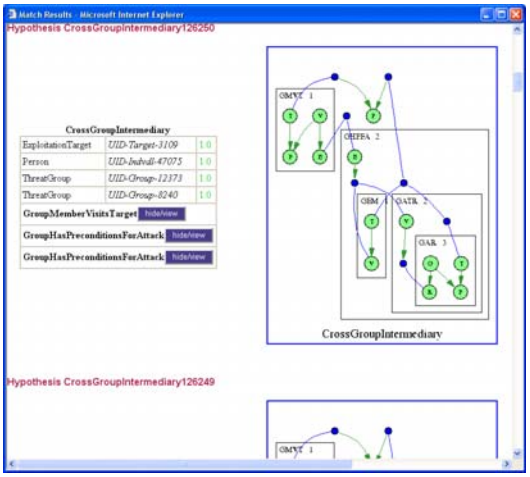


Figure 1: Initial pattern representing two groups cooperating through an intermediary

To complement his search for threatening group activity, the analyst gets the idea of looking for a new type of scenario: one where two threat groups cooperate to carry out different portions of an attack. The idea he has is that the two groups will cooperate through an *intermediary*—a person who is a member of both groups. He uses LAW's pattern editor and the pattern library to construct a new pattern, hierarchically, out of primitive graphs he has already used for other searches. The resulting hierarchical pattern is shown in Figure 1. It represents two groups potentially cooperating—one group makes one or more visits to a target, and the other acquires two or more resources necessary to attack that target via a mode that it is known to exploit—while sharing a member in common. While the details in the figure are too small to read, it shows how LAW's pattern language supports hierarchical graphs and relations between subgraphs.

LAW's display of the matches it found, shown in Figure 2, allows the analyst to see the mappings between the pattern and the matching data in detail. Tables on the left side of the screen describe the pattern-node-to-data-node mappings.[1] The table contains the mappings of the top-level nodes in the pattern, and contains buttons for each subgraph that expand into full mapping display tables for them if pressed. On the right of each table is a graphical summary of the match that shows, via color-coding, which pattern nodes and links were matched in the data, and which were missing.



Hypothesis CrossGroupIntermediary126250

CrossGroupIntermediary

ExploitationTarget	*UID-Target-3109*	1.0
Person	*UID-Indvdl-47075*	1.0
ThreatGroup	*UID-Group-12373*	1.0
ThreatGroup	*UID-Group-8240*	1.0
GroupMemberVisitsTarget hide/view		
GroupHasPreconditionsForAttack hide/view		
GroupHasPreconditionsForAttack hide/view		

Hypothesis CrossGroupIntermediary126249

Figure 2: LAW's display of pattern matches

For this pattern in this data set, LAW finds 23 matches in 3½ minutes. These represent a good starting collection of scenarios that merit further investigation. But the analyst feels that this set of results still does not cover all the group-cooperation possibilities he should be investigating. In particular, because information about threat group membership is often sketchy and incomplete, the pattern's requirement that the intermediary be a known member of both threat groups seems overly restrictive. The analyst wonders what kind of results he would get if, instead of requiring known group membership, he used repeated communication with members of the group as a surrogate. He uses the pattern editor to modify the pattern to include this new condition. Now the graph represents a situation where the intermediary is linked to each of the two groups in two ways: (1) directly, through a membership link, and (2) indirectly, through a TwoWayCommunicateWGroup subpattern, which specifies that the person initiates two or more communications (e.g., phone calls) with known members of the group. Additionally, the analyst changes the maximum allowable cost on the top-level pattern from 0 to 2. Since each node and link in the top-level pattern has a cost of 1 (LAW's default), this effectively makes it so that the two group membership links are optional.

For the new pattern, LAW now returns 40 matches (in 4 minutes). They are presented to the analyst in order, best match to worst. In this case, that effectively means that the candidate intermediaries with the strongest known ties to the two groups—both known group membership and repeated communication with group members—are presented first, and the ones with the weakest ties are presented last. These represent a better candidate set for further investigation, where that further investigation can involve either using the matches as catalysts for searches for information in other formats (e.g., text, video, discussions with colleagues), or continuing the pattern cycle.

3. Conclusion

This brief example highlights only some of the ways that LAW is designed to support the pattern development cycle. And many open issues remain—allowing more analyst control of the hypothesis management problem, developing more efficient approaches to inexact graph matching, better handling of uncertainty in the link analysis process, and so on. Addressing these problems, hardening LAW and other related tools, and putting them into the hands of operational end users should help give analysts the flexibility, speed, and coverage they need for the growing challenges they face.

Acknowledgments

This research was supported under Air Force Research Laboratory (AFRL) contract number F30602-00-C-0193.

References

Bunke, H. 1997. On a relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters* 18:689-694.

Schrag, R. 2005. A performance evaluation laboratory for threat detection technologies. Submitted for review.

Wolverton, M.; Berry, P.; Harrison, I.; Lowrance, J.; Morley, D.; Rodriguez, A.; Ruspini, E.; and Thomere, J. 2003. LAW: A workbench for approximate pattern matching in relational data. In *The Fifteenth Innovative Applications of Artificial Intelligence Conference (IAAI-03).*

[1] In the simulated domain, entities are associated only with machine-generated IDs, so the mappings shown in Figure 2 are not terribly informative. In more realistic test domains we have used, mapped nodes in the data are described by more user-friendly names or textual summaries.