Locating Optimal Destabilization Strategies

Il-Chul Moon
PhD student
School of Computer Science
Carnegie Mellon University
Jun/13/2007

Center for Computational Analysis of Social and Organizational Systems
http://www.casos.cs.cmu.edu/
Problem statement

- Network destabilization is an important tactic.
  - Counter terrorism – destabilize a terrorist network to disrupt its plan
  - Network centric warfare – destabilize a C2 structure to disrupt information diffusion
  - Computer network security – destabilize a computer network to disrupt its function
- However, we don’t have complete answers to the following questions.
  - How to find an efficient network destabilization strategy (or scenario)?
    - Minimum intervention, maximum destabilization effect
  - If we remove a node (possibly, agent, resource or knowledge), which node to target?
    - Agents with many resources and knowledge vs. Agents at the center of an agent-to-agent network
  - When to remove the node?
    - Earlier removal of hub agents and later removal of information-control agents vs. Later removal of hub agents and earlier removal of information-control agents
  - How to assess the located strategy under dynamically changing conditions?
    - Big damage, but still able to recover
    - Or, small damage, but unable to recover
    - Or, big damage and unable to recover
Introduction

- We limit ourselves to
  - Destabilization of an organization represented in a network structure
  - Only agent removal strategic intervention
  - Only one agent removal for a single intervention
  - Limited number of interventions
- We develop a framework
  - Dynamic network analysis on the target network to reveal its vulnerabilities
  - Automatic generation of (optimal) destabilization scenario by using machine learning technique and network analysis results
  - Assess the scenarios by utilizing a multi-agent network simulation model, Dynet, as a test-bed for the developed scenarios
- We expect to see
  - Better destabilization result from automatically generated scenarios compared to random destabilization scenarios
  - An implied trend of the generated destabilization scenarios
A terrorist network from the U.S. Embassy bombing incident in Tanzania
The network has 16 Agents, 4 knowledge pieces, 4 resources (5 tasks, too, but not used for this analysis)
Only 16 agents will be the target of removal, and each scenario has 10 removal chances.
Overall Framework Description

- Integration of three different components
  - Dynamic Network Analysis
    - Calculate network analysis measures
  - Multi-Agent Simulation Model
    - Assess the effect of the scenario with a simulation
  - Machine Learning
    - Train the algorithm based on random scenario results
    - Generate the scenario based on the training results

Random Scenario Generator
- Randomly synthesize a removal scenario

Dynet & Near Term Analysis
- Assess the effect of a scenario with a simulation

Target Network
Located Optimal Destabilization Scenario
- Assess and compare the effectiveness to the random generation case
In this presentation, a destabilization scenario is equivalent to an isolation (removal) sequence for agents.

- Ten isolations and one agent removal for each isolation
  - The test dataset has 16 agents
- The first isolation happens at time 2, and the next isolation happens after a gap of two time periods.
  - Start at time 2 and end at time 20
- i.e. Random scenario generation
  - Randomly pick an agent for each intervention in a scenario

First intervention, isolate al-Owahali at time-step 2

Last intervention, isolate sadiq-odeh at time-step 20
Dynet and Near-Term Analysis: a multi-agent simulation for assessing the sequence

- Dynet (a.k.a. Construct)
  - Multi-agent simulation
    - Agent interact based on probability of interaction which is determined by agent-to-agent network, relative similarity, relative expertise, etc.
  - Able to simulate node removals in the middle of simulation
  - Various performance metrics, such as knowledge diffusion, task accuracy, etc.

- Near-Term Analysis
  - A wrapping function for Dynet
    - GUI front-end for Dynet and callable for ORA (a dynamic network analysis tool)
  - Provide a function to setup a sophisticated strategic intervention scenario
  - Easy control of parameters for Dynet
Evaluation criteria for destabilization events

- We use a knowledge diffusion measure to see the performance changes.
- Three classes of events:
  - Suppression:
    - Diffusion rate goes up, but not as much as without intervention.
  - Damage:
    - Diffusion rate goes down, but can recover in the next time point.
  - Break:
    - Diffusion rate goes down, and the damage sustained for multiple time points.

\[ KD = \frac{\sum_{i=0}^{N} \sum_{j=0}^{K} AK_{ij}}{NK} \]
Dynamic Network Analysis measures

- Calculate the target network’s network-level and node-level metrics based on dynamic network analysis
- Metrics are responsible for
  - Training the learning algorithm with random isolation sequence
  - Eventually the generation of optimized isolation sequence
- Metrics are calculated by ORA

<table>
<thead>
<tr>
<th>Used measures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network measure (27 measures)</td>
</tr>
<tr>
<td>Node measure (11 measures)</td>
</tr>
</tbody>
</table>
Generation of Optimal Isolation Sequence: machine learning approach with DNA measures

- We create a training set by brief searching in the possible sequence space
  - Record the result of intervention, metrics for node positions, metrics for network topology
- We train a machine learning algorithm, a variant of Support Vector Machine
  - Result of intervention is a dependent variable
  - Metrics for nodes and networks are an independent variables
- We use the trained learning algorithm and create possible sequences
  - Get estimates for result by supplying the node and network metrics
  - Synthesize the sequence by choosing the agents with the highest damage estimates

<table>
<thead>
<tr>
<th>Training set (Random sequence)</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Randomly generated isolation sequences</td>
</tr>
<tr>
<td>- 1024 random cases and 10 isolations for each case</td>
</tr>
<tr>
<td>- Features for training include isolation timing, network level statistics and node level statistics from social network analysis.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Machine Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Used a variant of Support Vector Machine</td>
</tr>
<tr>
<td>- Applied a non-linear kernel, RBF</td>
</tr>
<tr>
<td>- Accept training instances with network measures and a boolean value for the success of suppression.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test set (Selected sequence)</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Test all the nodes with the same measures in the training set.</td>
</tr>
<tr>
<td>- Get the estimate value from the learning algorithm</td>
</tr>
<tr>
<td>- Select the top two nodes showing the high estimates for the isolation of the time.</td>
</tr>
</tbody>
</table>
Result (1): average destabilization performance

- Randomly generated isolation sequence vs. learning algorithm generated isolation sequence
- The learning algorithm generated sequences show more destabilization events and lower overall knowledge diffusion rates.
- High level comparison of two isolation sequence generation schemes
Result (2)

: average over time destabilization result

- Baseline, a case without intervention, shows highest knowledge diffusion rate.
- Random isolation sequence shows somewhat damaged diffusion rate.
- Learning algorithm shows very lower diffusion rate.
- This is the average across 1024 scenario of the random and optimized cases.

Smooth information diffusion curve: fail to destabilize the information flow

- Almost no difference between no-intervention and random interventions
- Big difference between average results from random interventions and optimized interventions
- Some damage events: relative success in preventing information diffusion
Result (3)

: best over time destabilization result

- Baseline, a case without intervention, shows highest knowledge diffusion rate.
  - Same to the previous slide
- Random isolation sequence shows pretty damaged diffusion rate, but the organization is still able to recover.
  - Also, notice the big variance between the best case and the average case
- Learning algorithm shows total break-down of the organization in terms of knowledge diffusion.
Result (4)

: a trend about who to target and when

- Beginning waves of isolations
  - Target nodes with high-degree centrality, clique count, betweenness centrality, etc

- Next waves of isolations
  - Target nodes with high betweenness and low degree, meaning connecting nodes

- Isolations of agents with exclusive knowledge are not the first priority.
  - It happens after initial isolation of high degree centrality agents

Average the node-level measures of the first selected agents

Simulation timeline
Simulation Time Point (Range : 0 ~ 52 ), Simulation case name : Isolation Mohammed Rashed Daoud al-Owhali at 2, Isolation Khalfan Khantis Mohamed at 4, Isolation Mohar

Optimized (or random) destabilization scenarios

1024

- cognitive demand
- total degree centrality
- clique count
- betweenness centrality
- high betweenness and low degree
- task exclusivity
- knowledge exclusivity
- resource exclusivity
- workload
Conclusion

• We demonstrated that
  • Machine learning based destabilization scenario creation
  • Destabilization scenario test result based on a multi-agent simulation
  • Better destabilization performance compared to random isolations

• We examined and found out that
  • Trained learning algorithm have a certain preference in choosing the target
    • Initial attacks, target nodes at the center of the network
    • Last attacks, target nodes at bridging points
    • Isolation of agents with exclusive knowledge may not be a priority, and they can be isolated after the nodes with high degree centrality.

• This tendency implies that
  • Destabilize the network first
  • Isolate the exclusive knowledge or resource later
Limitation & Future work

- Too small dataset, need extensive tests
- Need to find out the performance changes when we limit the initial training set size.
- Need to test the robustness of this framework when the network is not fully uncovered.
- Need to test the scalability in terms of computation time
- Any improvements in three related areas will enhance the performance of this framework
  - Better social network metrics to represent the network structure accurately
  - Better multi-agent models with better usability, confidence, validation, etc.
  - Better machine learning technique
Acknowledgements

This work was supported in part by the Office of Naval Research (ONR N0001140210973-NAVY, N000140610921 and N00014-06-0104), the National Science Foundation (SES-0452487), the Army Research Lab, and the AirForce Office of Sponsored Research (MURI: Cultural Modeling of the Adversary, 600322) and the Department of Defense for research in the area of dynamic network analysis. Additional support was provided by CASOS - the center for Computational Analysis of Social and Organizational Systems at Carnegie Mellon University. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Office of Naval Research, the National Science Foundation, the Army Research Lab or the U.S. government.