

Regex-Based Linkography Abstraction Refinement

Abhiram Kothapalli (UIUC)

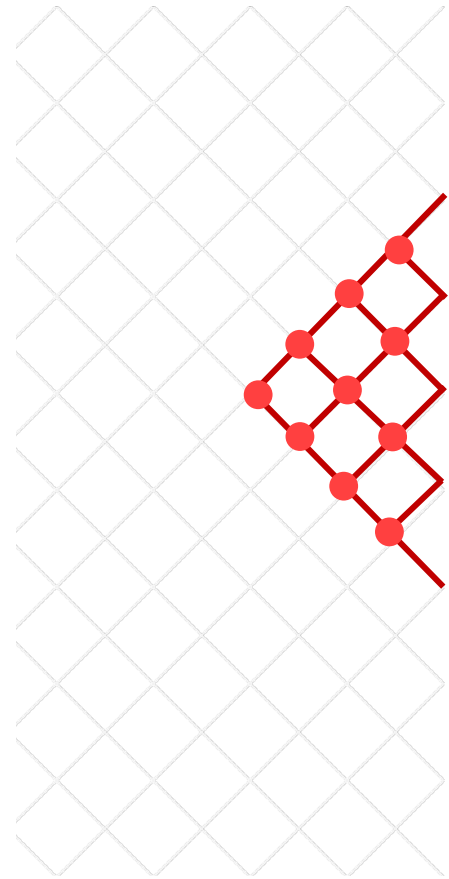
Robert Mitchell (Virginia Tech)

Linkograph: Cause and Effect Graph



How can we use linkographs to study the behavioral patterns of remote attackers of cyber systems in a highly automated fashion?

A Human Created Linkograph



```
copy alef.txt upload  
copy bet.txt upload  
ipconfig >>upload\gimel.txt  
type dalet.txt >upload\dalet.txt  
copy.exe he.txt upload
```

Automated Linkograph: Abstraction

```
Move: (copy) (.) *
```

```
Exec: (.) * (\.exe)
```

```
Look: (type) (.) * | (ipconfig) (.) *
```



```
copy alef.txt upload
```

```
copy bet.txt upload
```

```
ipconfig >>upload\gimel.txt
```

```
type dalet.txt >upload\dalet.txt
```

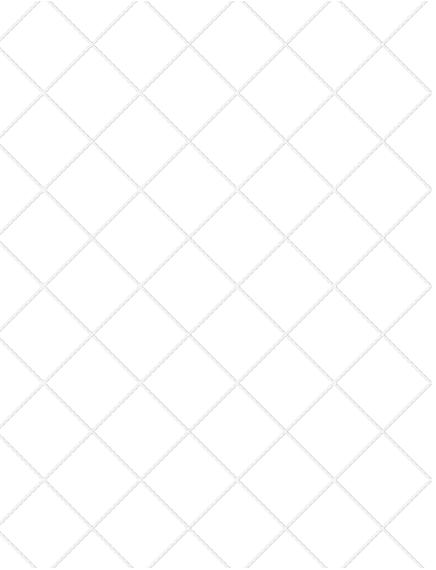
```
copy.exe he.txt upload
```

Automated Linkograph: Abstraction

```
Move: (copy) (.) *
```

```
Exec: (.) * (\.exe)
```

```
Look: (type) (.) * | (ipconfig) (.) *
```



```
(Move) copy alef.txt upload
```

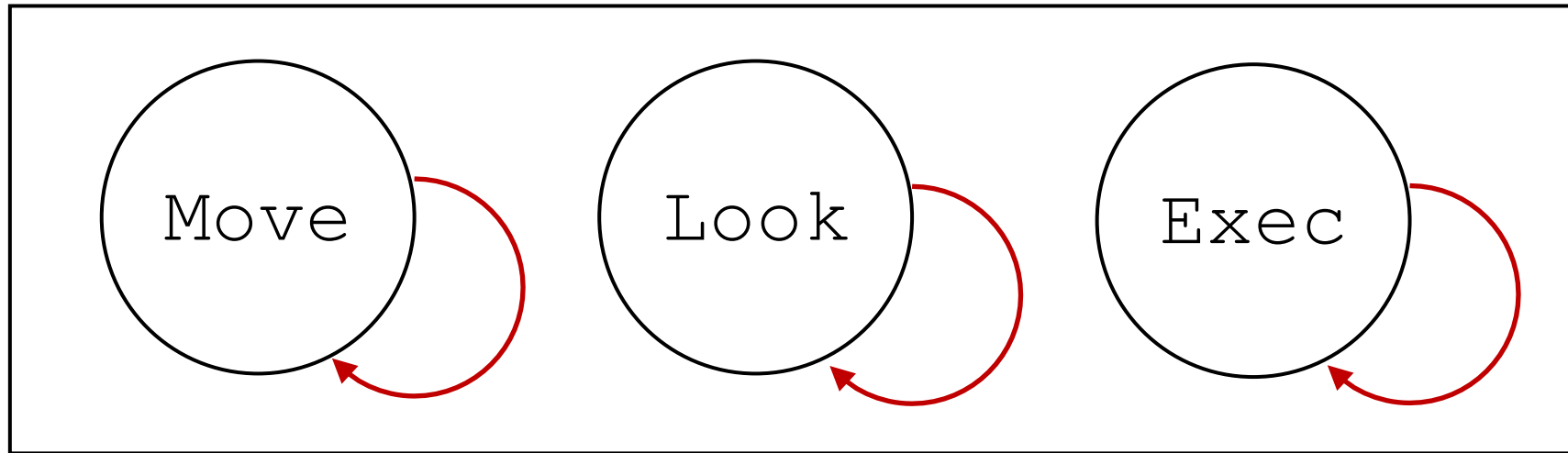
```
(Move) copy bet.txt upload
```

```
(Look) ipconfig >>upload\gimel.txt
```

```
(Look) type dalet.txt >upload\dalet.txt
```

```
(Exec) copy.exe he.txt upload
```

Automated Linkograph: Ontology



(Move) copy alef.txt upload

(Move) copy bet.txt upload

(Look) ipconfig >>upload\gimel.txt

(Look) type dalet.txt >upload\dalet.txt

(Exec) copy.exe he.txt upload

Today we will focus on how to automatically create optimal regexes for the abstraction.

```
Move: (copy) (.) *
```

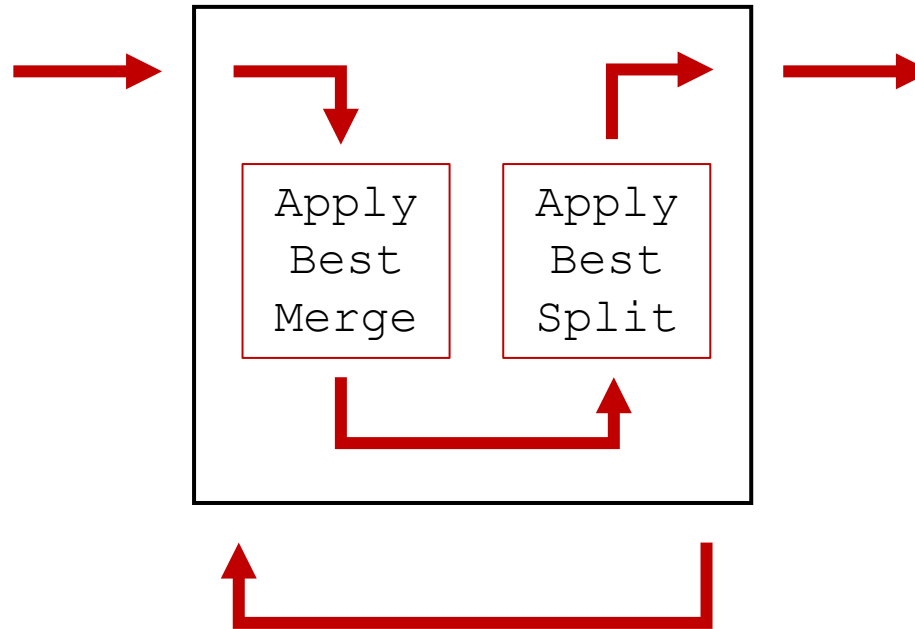
```
Exec: (.) * (\.exe)
```

```
Look: (type) (.) * | (ipconfig) (.) *
```


Regex Optimization Algorithm Overview

Trivial Abstraction

```
Class 1:  
cat  
../Desktop/folder1/attack.exe  
  
Class 2:  
cat  
Desktop/folder2/readme.txt  
  
Class 3:  
cd  
../Desktop/folder1/  
  
Class 4:  
cd  
../Documents/folder3/
```



Final Abstraction

```
Class 1:  
(cat )  
(../)  
(Desktop/)  
(folder1/|folder2)  
(attack|readme)  
(.exe|.txt)  
  
Class 2:  
(cd )  
(../)  
(Desktop/|Documents/)  
(folder1/)
```

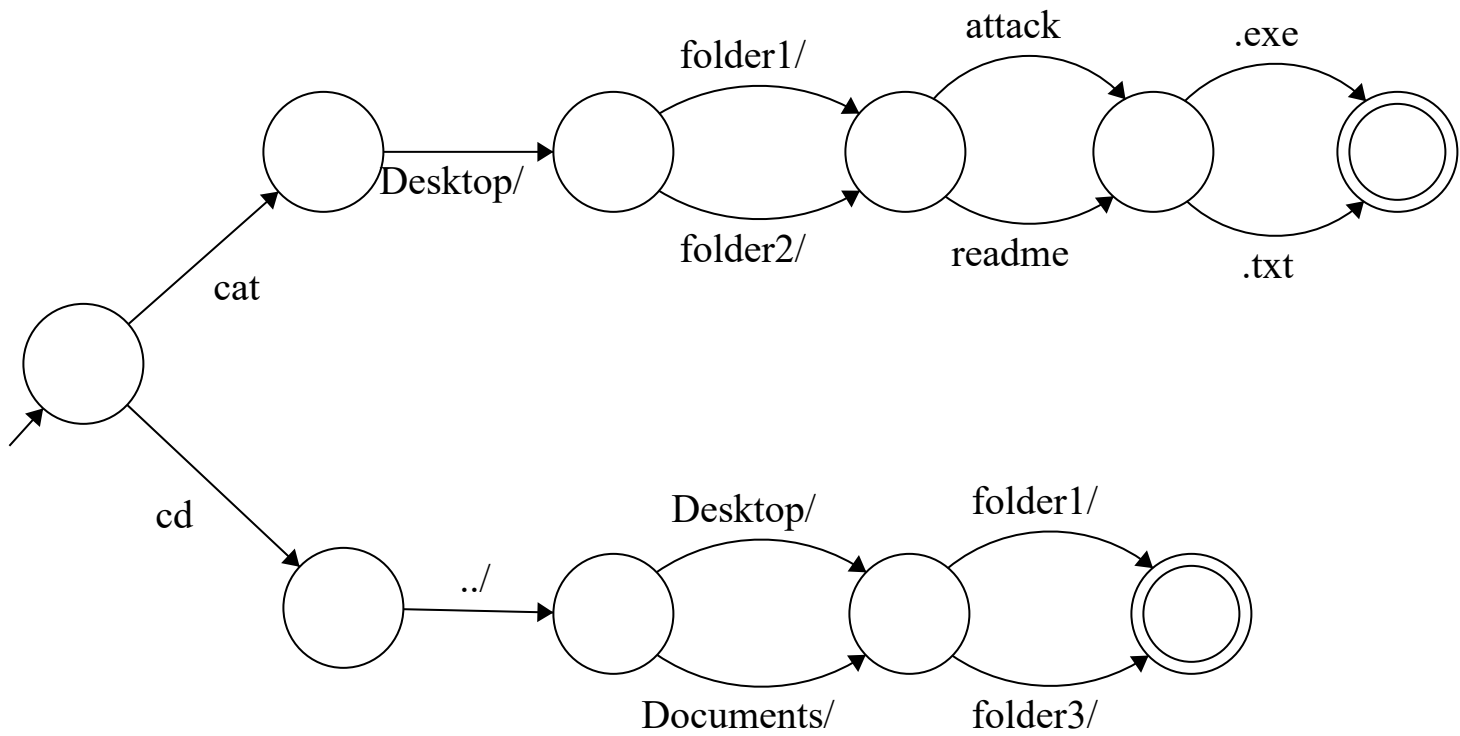
Iterate k times

Regex Format

Disjunction of Conjunctions of Disjunctions

```
(  
  (cat ) (Desktop/) (folder1/|folder2/)  
  (attack|readme) (.exe|.txt)  
)|(  
  (cd ) (../) (Desktop/|Documents/)  
  (folder1/|folder3/)  
)
```

```
(
  (cat ) (Desktop/) (folder1/|folder2/)
  (attack|readme) (.exe|.txt)
) | (
  (cd ) (../) (Desktop/|Documents/)
  (folder1/|folder3/)
)
```



Cold Start

Class 1: `cat ../Desktop/folder1/attack.exe`

Class 2: `cat Desktop/folder2/readme.txt`

Class 3: `cd ../Desktop/folder1/`

Class 4: `cd ../Documents/folder3/`

Merge: Pick Most Frequent Classes

Class 1: `cat ../Desktop/folder1/attack.exe`

Class 2: `cat Desktop/folder2/readme.txt`

Merge: Align Regexes

Class 1: `cat ../Desktop/folder1/attack.exe`

Class 2: `cat Desktop/folder2/readme.txt`

Merge: Atomic Split

```
Class 1: cat |./Desktop/folder1/attack.exe  
Class 2: cat |Desktop/folder2/readme.txt
```

Merge: Add Disjunction

Class 1: cat | .. / Desktop / folder1 / attack . exe
Class 2: cat | Desktop / folder2 / readme . txt

Class 1:

```
(cat ) ( .. / ) ( Desktop / ) ( folder1 / | folder2 )  
( attack | readme ) ( . exe | . txt )
```


Merge: Remove Conjunction

Class 1:

```
(cat ) (../) (Desktop/  
folder1/|folder2|folder3/|folder4/|folder5/  
attack|readme) (.exe|.txt)
```

Merge: Remove Conjunction

Class 1:

```
(cat ) (../) (Desktop/)  
(folder1/|folder2|folder3/|folder4/|folder5/|(.)*  
(attack|readme) (.exe|.txt)
```

Split: Remove Disjunction (Even Split)

Class 1:

```
(cat ) (../) (Desktop/  
(folder1/|folder2|folder3/|folder4/|folder5/|(.)* )  
(attack|readme) (.exe|.txt)
```

Split: Remove Disjunction (Even Split)

Class 1:

```
(cat ) (../) (Desktop/  
folder1/|folder2|folder3)  
attack|readme) (.exe|.txt)
```

Class 2:

```
(cat ) (../) (Desktop/  
folder4/|folder5/|(.)*  
attack|readme) (.exe|.txt)
```

Split: Add Conjunction

Class 1:

```
(cat ) (../) (Desktop/  
folder1/|folder2|folder3)  
attack|readme) (.exe|.txt)
```

Class 2:

```
(cat ) (../) (Desktop/  
folder4/|folder5/|(.)*  
attack|readme) (.exe|.txt)
```

Split: Add Conjunction

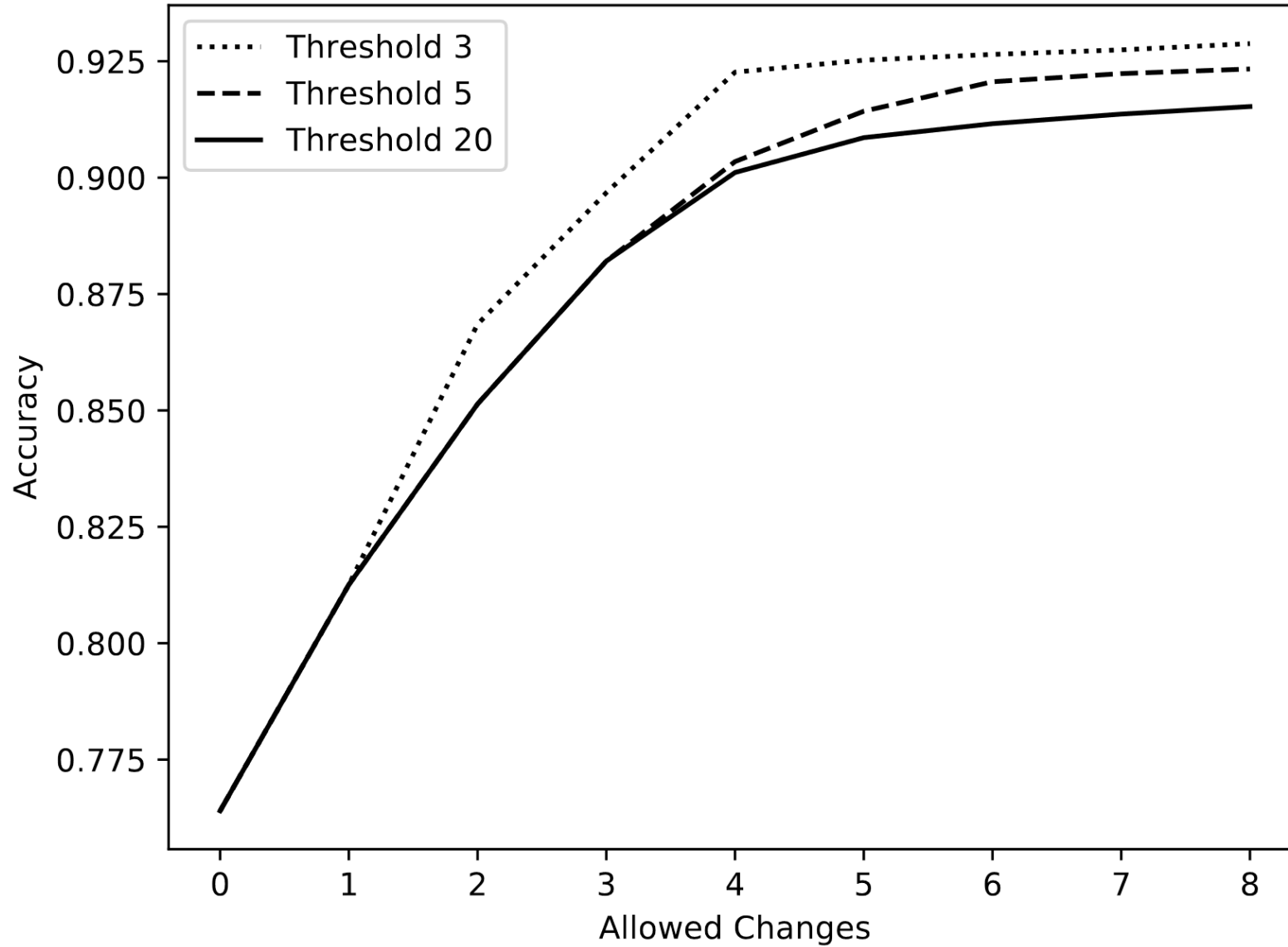
Class 1:

```
(cat ) (../) (Desktop/  
folder1/|folder2|folder3)  
attack|readme) (.exe|.txt)
```

Class 2:

```
(cat ) (../) (Desktop/  
folder4/|folder5/  
attack|readme) (.exe|.txt)
```

Cold Abstraction Average Accuracy vs Allowed Changes:



Reverse Cold Abstraction Average Accuracy vs Allowed Changes:

