# An intruder detection approach based on infrequent rating pattern mining

**J.M. Luna, A. Ramírez, J.R. Romero and S. Ventura**

*Dept. of Computer Science and Numerical Analysis*
*University of Cordoba, Spain*

**Computational Intelligence in Knowledge Discovery @ 10th Intl. Conference on ISDA**
*November 29 – December 1, 2010 Cairo, Egypt*

# Contents

# Introduction

- Collaborative recommender systems (CRSs) have become a routine activity:
  - Predictions usually based on similarity between neighbours
  - A potential source of frauds and deception
    - Malicious parties want to promote/demote their items of interest
    - Injection of fake user profiles to distort recommendations
- Only one application domain of intrusion detection problem
- Datamining widely used to explore useful knowledge from larga datasets

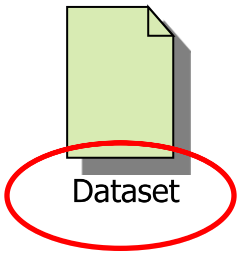
# Introduction

- Association Rule Mining (ARM)
  - A very well-known method for discovering interesting patterns and close relations between items

$$A \rightarrow C,\ A \cap C = \varnothing$$

- Rare Association Rule Mining (RARM)
  - Searches for non-frequent, unusual or exceptional association rules by mining rare itemsets
  - Non-ordinary items could help to discover potential intruders throughout the dataset maintained by the rating system
- Exhaustive search of the rule space would be non-scalable and potentially endless (e.g. Apriori-Inverse, ARIMA, etc)

# The suspicious attacker detection process

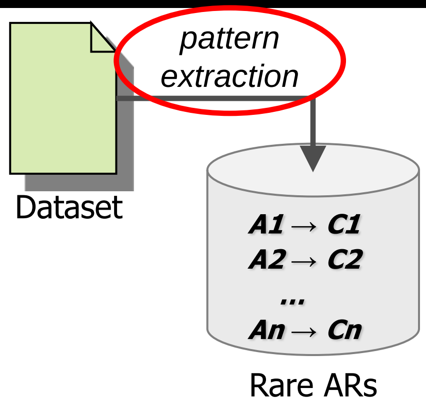



- A rating dataset contains uncorrupted user preferences per item
- Each item (e.g. movies) have a numerical rating
- In the example(*), items are rating from 1 to 5

| | Item1 | Item2 | Item3 | Item4 | Item5 | Item6 |
|---|---|---|---|---|---|---|
| User1 | 5 | 2 | 3 | 3 | | |
| User2 | 2 | | 4 | | 4 | 1 |
| User3 | 3 | 1 | 3 | | 1 | 2 |
| User4 | 4 | 2 | 3 | 1 | | 1 |
| User5 | 3 | 3 | 2 | 1 | 3 | 1 |
| User6 | | 3 | | 1 | 2 | |
| User7 | 4 | 3 | | 3 | 3 | 2 |
| User8 | | 5 | | 1 | 5 | 1 |

(*) B. Mobasher *et al.* "Toward Trustworthy Recommender System: An Analysis of Attack Models and Algorithm Robustness" *ACM Trans. Internet Technology*, 7(4)-23, 2007.

# The suspicious attacker detection process

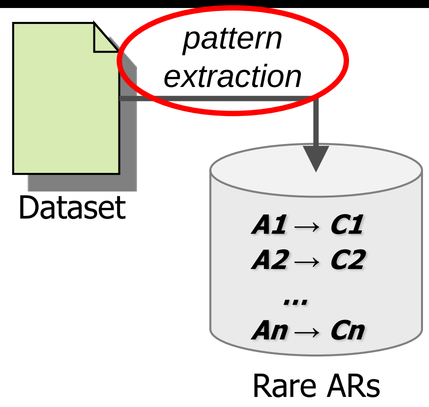


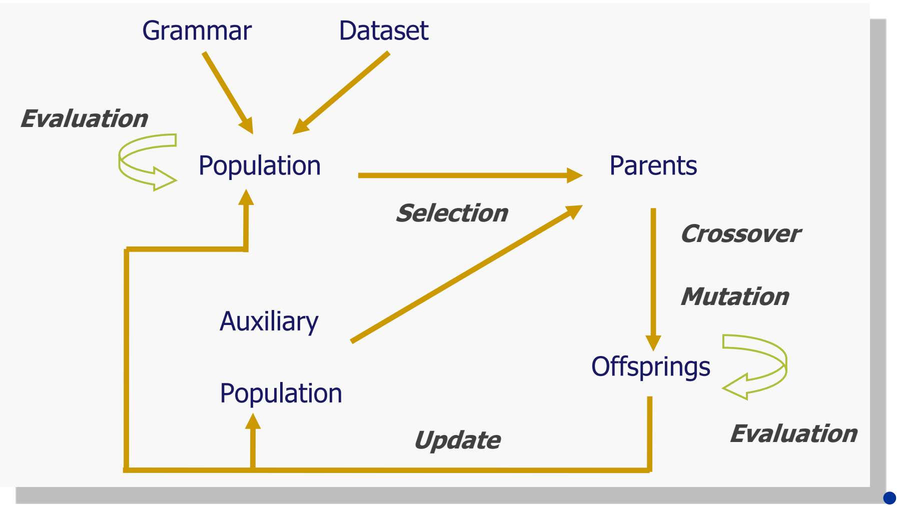

$G = (\Sigma_N, \Sigma_T, P, \text{Rule})$ with:
$\Sigma_N$ = {Rule, Antecedent, Consequent, Comparison, Comparator, Attribute Comparison }
$\Sigma_T$ = { "AND", "<=", "<", ">=", ">", "name", "value" }
$P$ = {Rule = Antecedent, Consequent ;
Antecedent = Comparison | "AND", Comparison, Antecedent ;
Consequent = Comparison ;
Comparison = Comparator, Attribute Comparison ;
Comparator = "<=" | "<" | ">=" | ">" ;
Attribute Comparison = "name", "value" ;}

- For the extraction of frequent ARs we proposed an evolutionary approach: **G3PARM**(*)
  - High efficiency and low memory requirements
  - Different types of attributes
  - Based on a context-free grammar
  - Each individual is a derivation tree that represents a rule
- Extension of the G3PARM for RARM
  - Extraction of rare association rules
  - Post-processing step that simplifies rules with redundant attributes

(*) J. M. Luna, J. R. Romero y S. Ventura. G3PARM: A Grammar Guided Genetic Programming Algorithm for Mining Association Rules. *IEEE World Congress on Computational Intelligence (WCCI 2010)* Barcelona, Spain, 2010

# The suspicious attacker detection process




- It searches for the minimum support for each rule by maximizing the fitness function:
  - Support of the rule

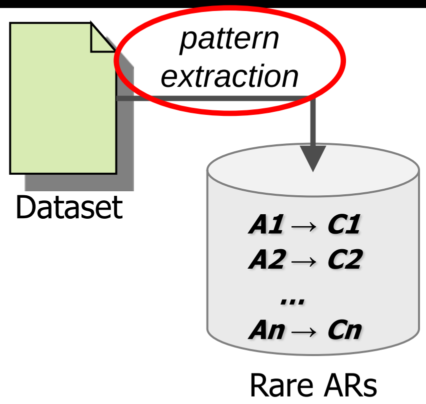
$$supp(A \rightarrow C) = \frac{|\{A \cup C \subseteq T, T \in D\}|}{|D|}$$

  - A support threshold (minimum support)

$$fitness(A \rightarrow C) = \begin{cases} \frac{1}{Supp(A \rightarrow C) - Thr} & if\ Supp(A \rightarrow C) > Thr \\ 0 & otherwise \end{cases}$$

- Confidence and support of the rule are used to update the auxiliary population

# The suspicious attacker detection process

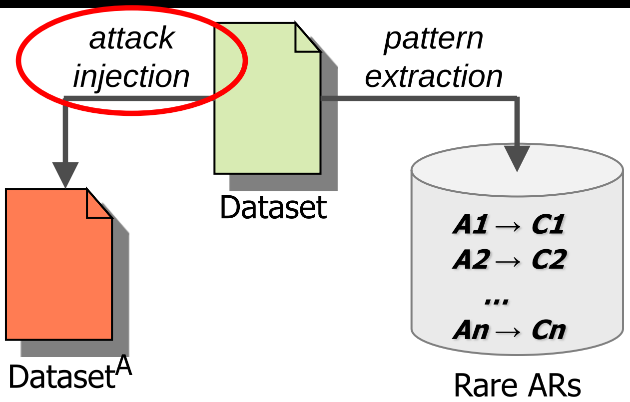



*Two rare rules are mined*

$Item1 < 3$ AND $Item2 > 3 \rightarrow Item6 \leq 4$
$Item3 \geq 4$ AND $Item5 > 3 \rightarrow Item4 < 2$

| | Item1 | Item2 | Item3 | Item4 | Item5 | Item6 |
|---|---|---|---|---|---|---|
| User1 | 5 | 2 | 3 | 3 | | |
| User2 | 2 | | 4 | | 4 | 1 |
| User3 | 3 | 1 | 3 | | 1 | 2 |
| User4 | 4 | 2 | 3 | 1 | | 1 |
| User5 | 3 | 3 | 2 | 1 | 3 | 1 |
| User6 | | 3 | | 1 | 2 | |
| User7 | 4 | 3 | | 3 | 3 | 2 |
| User8 | | 5 | | 1 | 5 | 1 |

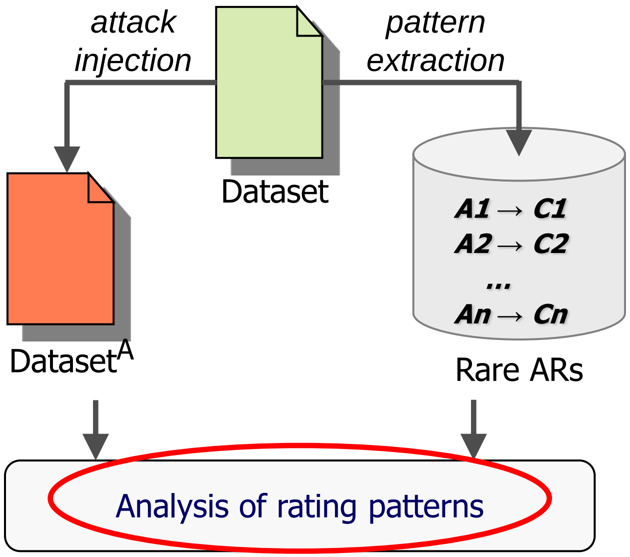
# The suspicious attacker detection process




With the elapse of time, some fraud profiles are injected in the dataset *(three attacks on Item6)*

| | Item1 | Item2 | Item3 | Item4 | Item5 | Item6 |
|---|---|---|---|---|---|---|
| User1 | 5 | 2 | 3 | 3 | | |
| User2 | 2 | | 4 | | 4 | 1 |
| User3 | 3 | 1 | 3 | | 1 | 2 |
| User4 | 4 | 2 | 3 | 1 | | 1 |
| User5 | 3 | 3 | 2 | 1 | 3 | 1 |
| User6 | | 3 | | 1 | 2 | |
| User7 | 4 | 3 | | 3 | 3 | 2 |
| User8 | | 5 | | 1 | 5 | 1 |
| Attacker1 | 5 | | 3 | | 2 | 5 |
| Attacker2 | 5 | 1 | 4 | | 2 | 5 |
| Attacker3 | 5 | 2 | 2 | 2 | | 5 |

# The suspicious attacker detection process




- For each rare rule mined:
  - Calculate the relative support of each attribute in a rule in the *original dataset*
  - Again, the relative support is computed using the *suspicious ratings*
  - $|\Delta s| = |s_1 - s_0|$ is obtained
  - The probability that an item is attacked:

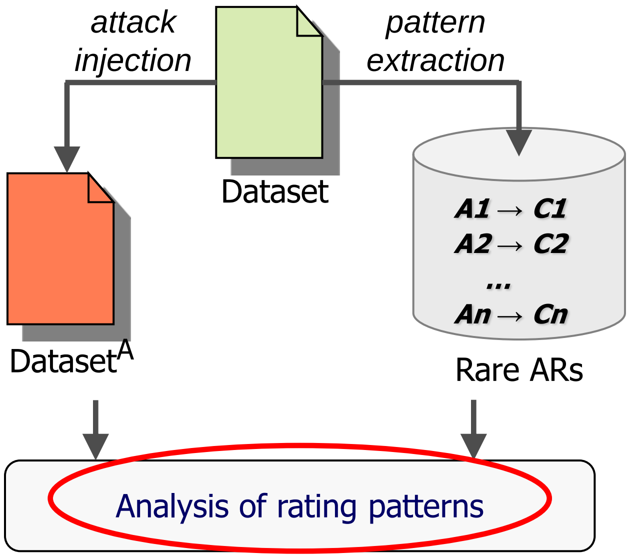
$$P_{attack} = \frac{|\Delta s|}{max(|s_0 - L_m|, |s_1 - L_M|)}$$

| | Item1 | Item2 | Item3 | Item4 | Item5 | Item6 |
|---|---|---|---|---|---|---|
| $s_0$ | 0.16 | 0.14 | 0.20 | 0.66 | 0.33 | 1.00 |
| $s_1$ | 0.10 | 0.10 | 0.25 | 0.57 | 0.25 | 0.66 |
| $\lvert\Delta s\rvert$ | 0.06 | 0.04 | 0.05 | 0.09 | 0.08 | 0.33 |
| $L_m$ | 0.11 | 0.10 | 0.12 | 0.44 | 0.22 | 0.66 |
| $L_M$ | 0.44 | 0.40 | 0.50 | 0.77 | 0.55 | 1.00 |
| $max(..)$ | 0.28 | 0.26 | 0.30 | 0.22 | 0.33 | 0.33 |
| $P_{attack}$ | 0.23 | 0.15 | 0.16 | 0.40 | 0.24 | 1.00 |

**$L_m$**: relative support obtained by dividing the absolute support by the number of instances (in $Dataset_A$)

**$L_M$**: relative support obtained if all the instances of the attack injection are satisfied by the attribute of the rule

# The suspicious attacker detection process




- $P_{attack}$ indicates whether an item is being potentially attacked
  - If $P_{attack}$ is greater than a threshold, a potential attack is considered
- The highest value does not always imply an attack
  - Item4 has a $P_{attack}$ of 40%
- An **influence measure** is required to analyze how effective the attack is.

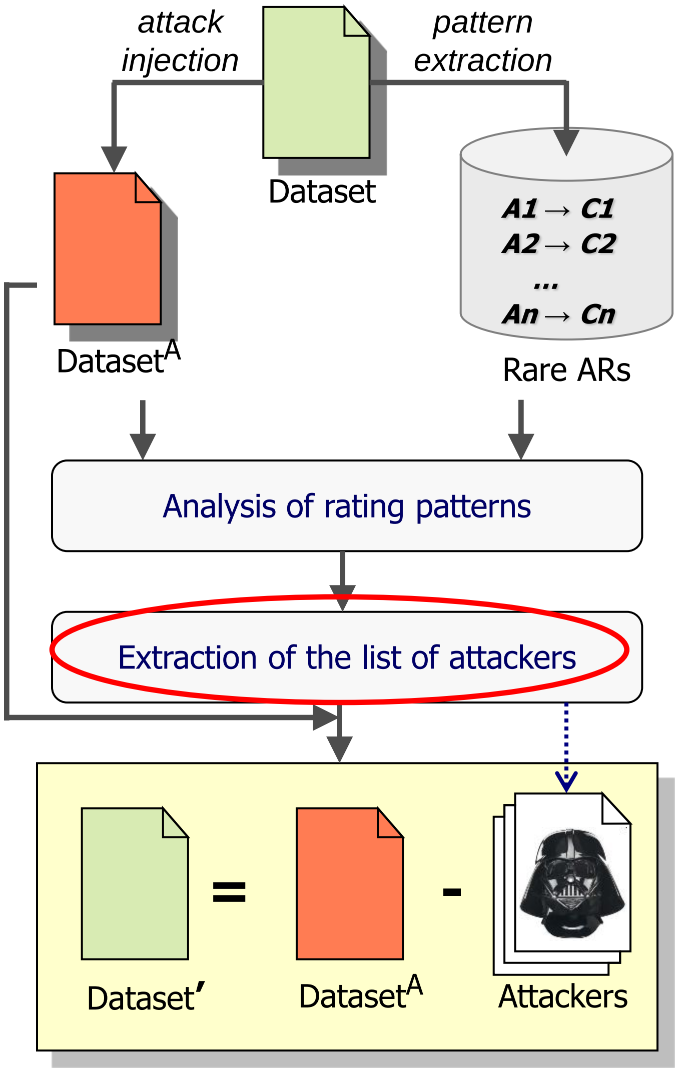
| | Item1 | Item2 | Item3 | Item4 | Item5 | Item6 |
|---|---|---|---|---|---|---|
| $P_{attack}$ | 0.23 | 0.15 | 0.16 | 0.40 | 0.24 | 1.00 |

| Item1 | Item2 | Item3 | Item4 | Item5 | Item6 |
|---|---|---|---|---|---|
| 0.029 | -0.010 | 0.000 | 0.005 | 0.005 | 0.300 |

$$Infl = \frac{\Delta s \Delta \bar{r}}{r_{max} - r_{min}}$$

**Δr** is the increment of the average score for an item before and after the potential attack

# The suspicious attacker detection process




- A preliminary list of attackers can be experimentally built by analyzing $P_{attack}$ and *Infl*
  - We need to study which *new* profiles satisfy the item (e.g. *Item6*)
- These profiles (`Attackers`) are removed from the dataset (`Dataset`$_A$)
- A new iteration would start with the elapse of time using `Dataset'`

# Experimentation and Results

## *Experiment setup*

- As a dataset, we used the online Jester Online Joke Recommender System
  - 4.1 million continuous ratings
  - 100 jokes (i.e. items)
  - 73,421 users (i.e. profiles)
  - Ratings ϵ [-10, 10]

| **Parameter** | **Value** |
|---|---|
| Population size | 50 individuals |
| Number of generations | 100 |
| $P_{crossover}$ | 0.9 |
| $P_{mutation}$ | 0.2 |
| Maximum number of derivations (CFG) | 24 |
| Auxiliary population size | 30 individuals |
| Confidence threshold | 0.9 |
| Minimum suppport threshold | 0.0005 |
| Maximum suppport threshold | 0.125 |
| $r_{min}$ | -10 |
| $r_{max}$ | 10 |

- The algorithm configuration
  - Five different executions with five different seeds (150 rules at most)

# Experimentation and Results

## *Running the process*

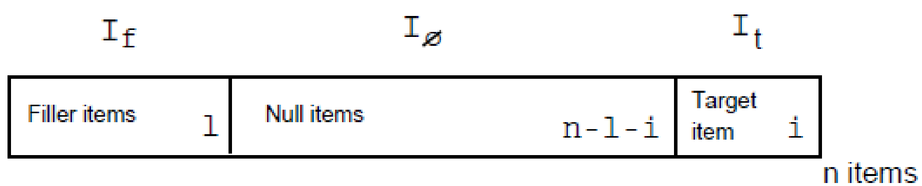
$$joke78 > -2 \quad AND \quad joke57 \geq -6 \quad AND \quad joke57 \geq -8 \quad AND \quad joke67 < 2 \rightarrow joke54 \geq -6$$

*Example of rare rule extracted*

$$joke57 \geq -6 \quad AND \quad joke67 < 2 \quad AND \quad joke78 > -2 \rightarrow joke54 \geq -6$$

*Same rule after normalization*

- We simulated a push attack (promotion of items) based on average ratings




**Average Attack:**

- ($I_t$) *Target item*
- ($I_f$) *Filler items*
- ($I_ø$) *Null items*

- Different attacks were injected:
  - 5 different **l** values (*fillers*): 20, 30, 40, 50, 60
  - 3 different items promoted with low ratings (joke58, joke74, joke79)
  - Each injection is about 10% the dataset size
  - $P_{attack}$ >= 0.8 and $P_{attack}$ >= 0.5 *(too low, just for comparison)*

# Experimentation and Results

*Pursuing the attackers*

- When $P_{attack} = 0.5$:
  - New rules containing the target item are found
  - … but more filler items could also be considered as injected
  - The influence measure reveals the real target item

- As an example:
  - An item ($\mathbf{I^s}$) was marked as suspicious: $P_{attack} = 0.5$
  - However, the target item ($\mathbf{I^t}$) obtained $P_{attack} = 0.2$
  - After measuring the influence:
    - **Infl($\mathbf{I^s}$) = -0.006**
    - **Infl($\mathbf{I^t}$) = 0.022** *(the biggest value in the dataset)*

| Attacked item | $l$ = 20 | $l$ = 50 | $l$ = 60 |
|---|---|---|---|
| Joke58 | 0 | 44 | 44 |
| Joke74 | 0 | 37 | 37 |
| Joke79 | 0 | 39 | 39 |

*Rules containing … >= 0.5*

# Concluding Remarks

- Concluding Remarks
  - An evolutionary proposal for the detection of malicious profile injections in user-based CRSs
  - A variation of the G3PARM algorithm for RARM
  - Introduction of measures for the analysis of rating patterns

- Future Work
  - Validate with different types of attacks
  - Reaction in non-simulated environments

# An intruder detection approach based on infrequent rating pattern mining

**José-María Luna, Aurora Ramírez, *José-Raúl Romero* and Sebastián Ventura**

*Dept. of Computer Science and Numerical Analysis*

*University of Córdoba, Spain*

# Thank you!