From Miners to Millionaires

Yanan Xiao

Masdar Institute of Science and Technology

Data Mining Course Presentation

1. Introduction
2. Trustrace
3. Empirical Evaluation
4. Results
5. Discussion
6. Future Work

“Theory and Practice were born together.”

Glossaries

- IR: Information Retrieval
- VSM: Vector Space Model
- JSM: Jensen-Shannon Model
- PCA: Principal Component Analysis
- Trustrace: Trust-Based Traceability
- TF/IDF: Text Frequency and Inverse Document Frequency
- MSW: Multiple-Static Weights

Definition of Traceability

Traceability is the only means to ensure that the source code of a system is consistent with its requirements.
And that all and only the specified requirements have been implemented by developers.

Traceability Sad Facts

During software maintenance and evolution, requirement traceability links become obsolete.

Because developers do not/cannot devote effort to updating them.

More sad: Recovering these traceability links later is a daunting and costly task for developers.

State-of-the-Art Technique

The literature has proposed methods, techniques, and tools to recover these traceability links semi-automatically or automatically.

Information Retrieval (IR) techniques can automatically recover traceability links between free-text requirements and source code.

A Mathematician's Approach

A set of requirements:

$$R = \{r_1, \ldots, r_N\} \tag{1}$$

A set of classes:

$$C = \{c_1, \ldots, c_M\} \tag{2}$$

A collection of sets:

$$T = \{T_1, \ldots, T_P\} \tag{3}$$

where each $T_i = T_1, \ldots, T_{N_i}$ is a set of homogeneous pieces of information.

For each set $T_i \in T$, we build a set $R2CT_{i,r_j,t_k}$ for each expert T_i as follows:

$$R2CT_{i,r_j,t_k} = \{(r_j, c_s, \sigma_i^{'}(r_j, t_k))|c_s \in \delta_{T_i}(t_k)\&t_k \in T_i\} \tag{4}$$

And we use the sets $T_i \in T$ to build a set of trustable links T_r:

$$\begin{aligned} T_r = & \{(r_j, c_s, \sigma_i^{'}(r_j, t_k))| \\ & \exists t_k \in T_i : (r_j, c_s) \in \alpha(R2CT_{i,r_j,t_k}) \\ & \&(r_j, c_s) \in \alpha(R2C)\} \end{aligned} \tag{5}$$

In $TC_i(r_j, c_s)$ a new similarity $\sigma_i^*(r_j, c_s)$ computed as:

$$\sigma_i^*(r_j, c_s) = \frac{\sigma_i(r_j, c_s) + \sum_{l \in TC_i(r_j,c_s)} \phi(l)}{1 + |TC_i(r_j, c_s)|} \tag{6}$$

Finally Trumo combine assigned value to each link T_r as follows:

$$\begin{aligned} \psi_{r_j,c_s}(T_r) &= [\textstyle\sum_{i=1}^{P} \lambda_i(r_j, c_s)\sigma_i^*(r_j, c_s)] \\ &+\lambda_{P+1}(r_j c_s)\tfrac{|T_r(r_j,c_s)|}{max_{n,m}|T_r(r_N,c_M)|} \end{aligned} \tag{7}$$

An Architecture's Approach

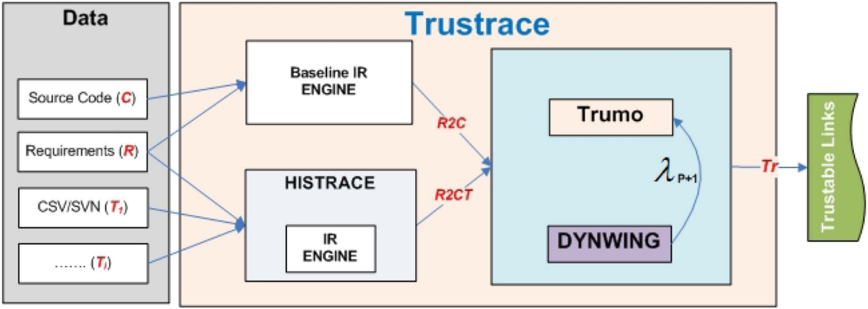

Figure : Trust-based requirement traceability process

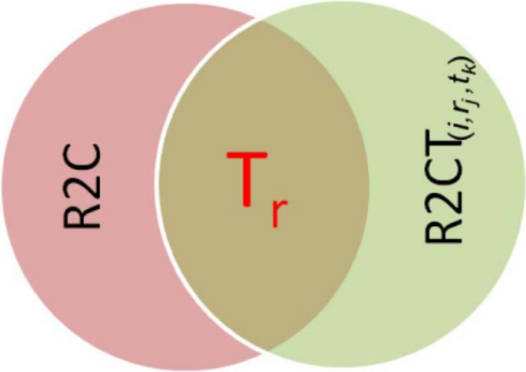

Figure : Overlapping of $R2C$, $R2CT_{i,r_j,t_k}$ and T_r

Trustrace Step-by-Step

- Histrace: Uses requirements' textual descriptions, CVS/SVN commit messages, bug reports and classes to produce experts.
- Trumo: Uses a web model of users' trust to discard and/or rerank the similarity of links in T_r.
- DynWing: Uses Expectation-Maximization approach to choose the right weight per link for different experts.

Histrace

Histrace creates links between the set of requirements, R and the source code, C, using the software repositories.
In the following, T_1 stands for CVS/SVN commit messages, and T_2 for bug reports.

Link each commit message and bug reports

```
((b)[ug]{0,2}\s*[id]{0,3}|id|fix|pr|#)
[\s#=]*[?([0--9]{4,6})]?
```

Tuned to the naming and numbering conventions by the developers of *Rhino*.

In the last step, Histrace removes false-positive links by imposing the following constraint:

Remove false-positive link with regular expression

```
fix(e[ds])?|bugs?|problems?|defects?patch
```

IR Techniques

VSM and JSM are used by researchers. These techniques both essentially use term-by-document matrices.

- Vector Space Model. The well-known TF/IDF measure is chosen. A document is a vector of TF/IDF weights. TF is the local weight whereas IDF a global weight of a term.

$$(TF/IDF)_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} * \log_2 \left(\frac{|D|}{|d : t_i \in d|}\right) \tag{8}$$

- Jensen-Shannon Model. JSM represents each document through a probability distribution, i.e., a normalized term-by-document matrix.

$$p = \frac{n(w,d)}{T_d} \tag{9}$$

Just a fancy way of expressing *word frequency*.

JSM ranks target documents via the “distance” of their probability distribution to that of the source document.

$$JSM(q,d) = H(\frac{p_q + p_d}{2}) - \frac{H(p_q + H(p_d))}{2} \tag{10}$$

$$H(p) = \sum h(p(w)) \tag{11}$$

$$h(x) = -x \log x \tag{12}$$

$H(p)$ is the entropy of the probability distribution p, and p_q and p_d are the probability distribution of the two documents.

Goal

- Quality Focus: The accuracy of Trustrace in terms of precision and recall. Also includes improvement by DynWing in terms of F_1 score.
- Perspective: The perspective of practitioners interested in recovering traceability links with greater precision and recall values.

Research Questions

Accuracy of traceability links:

- **RQ1** recovered by Trustrace compare with JSM and VSM.
- **RQ2** recovered by DynWing compare with PCA.

Analysis Method

- To answer **RQ1**, we perform several experiments with different threshold values on the recovered links to perform statistical tests on the precision and recall values.
- To answer **RQ2**, we use PCA and DynWing to assign weights to the traceability links recovered using Trustrace.

- **RQ1**: Trustrace helps to recover more correct links than IR techniques alone. When two experts are available, Trustrace is always better. In only one case and with just a single expert due to a lack of external source of information, did recall go down.
- **RQ2**: DynWing provides better weights for different experts than a PCA-based weighting technique. However, it is possible that in some cases PCA-based weighting provides the same (but not better) results as DynWing.

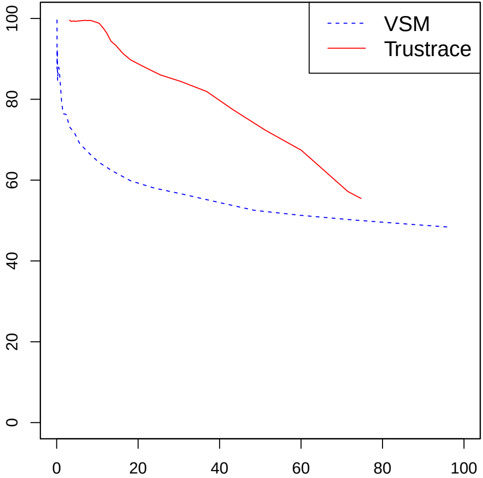

Figure : Precision and recall values of JSM & Trustrace, *Rhino* example

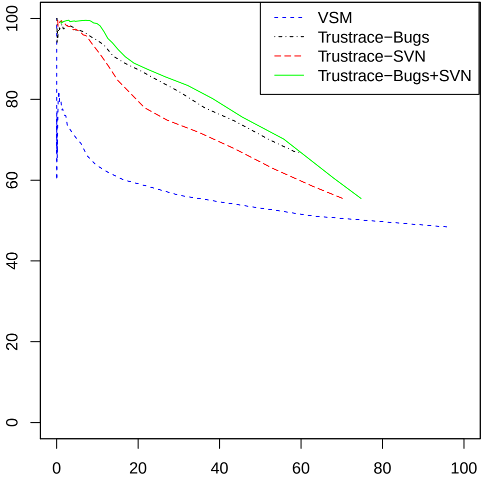

Figure : Precision and recall values of VSM & Trustrace, *Rhino* example

- Data Set Quality Analysis
- DynWing versus MSW versus PCA
- Number of Experts
- Other Observations
- Practical Applicability Trustrace
- Revisiting the Conjectures
- Threats to Validity

Threats to Validity

- **Construct Validity**: Quantify the degree of inaccuracy by validation of the precision and recall using manually built oracles.
- **Internal Validity**: Mitigate this threat by using MSW- and PCA-generated λ values, and by using the same setting for all the experiments.
- **External Validity**: The research approach is applicable to any other systems.
- **Conclusion Validity**: Mitigate this threat by the appropriate nonparametric test Mann-Whitney. And applying the Shapiro-Wilk test to select data.

- *Histrace*: Implement more instances, using emails and forum discussions.
- *Trumo*: Use in other software engineering fields, in particular, test-case prioritization, anti-pattern detection and concept location.
- *Trustrace*: Deploy in a development environment. Perform experiments with real developers.
- *Regular Expression*: Use advanced matching techniques.

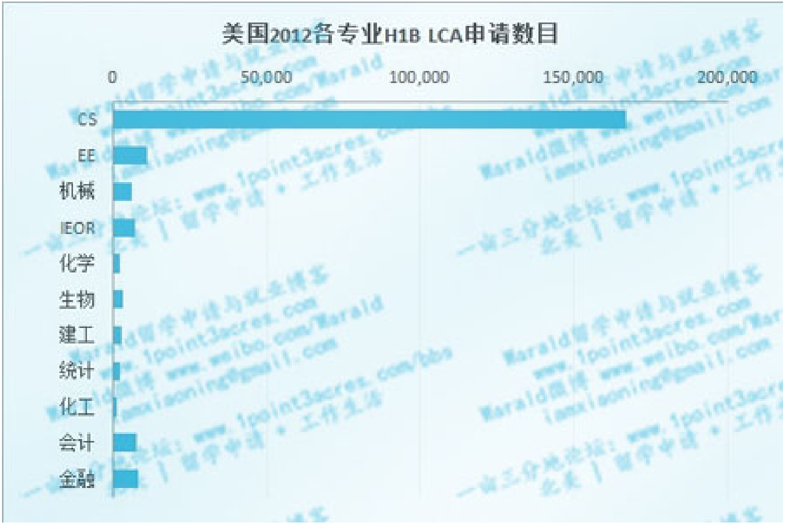

Figure : Year 2012 H1B Applicants

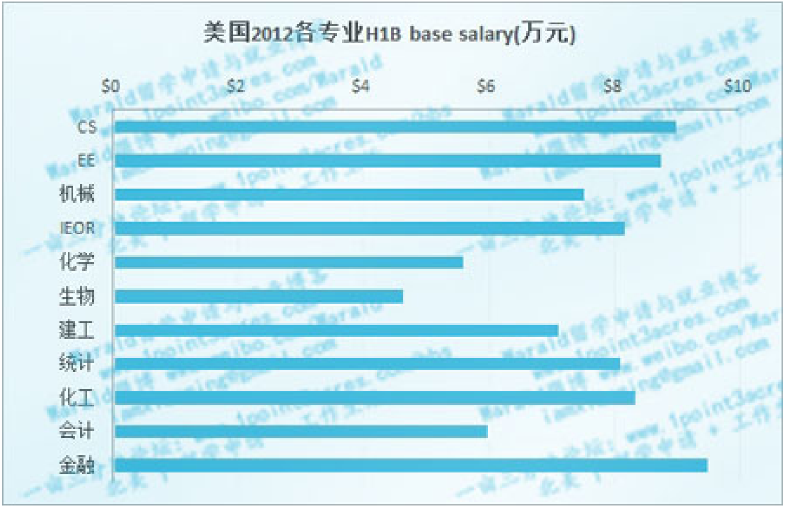

Figure : Year 2012 United States Base Salary, per Profession

Figure : No Question Off Limits!