PatternRank: A Software-Pattern Search System Based on Mutual Reference Importance

Atsuto Kubo
Waseda University
Japan
a.kubo@fuka.info.waseda.ac.jp

Hiroyuki Nakayama
Waseda University
Japan
h-nakayama@fuka.info.waseda.ac.jp

Hironori Washizaki
Waseda University
Japan
washizaki@waseda.jp

Yoshiaki Fukazawa
Waseda University
Japan
fukazawa@waseda.jp

ABSTRACT

There are currently a large number of digitized pattern documents. However, there are no ordering method specialized for patterns, thus, developers have to read and evaluate many pattern documents to find appropriate patterns for developer's facing problems. We focus to popularity of patterns in this paper, because many patterns must refer to core patterns in each domain. Our proposed method calculates an importance value of each pattern document using interpattern references and "in the same pattern catalog"relationships. Since our method is a simple expansion of Google's PageRank method, obtained importance values indicate popularity of each pattern. Resorting search result by using the proposed method helps developers to find a popular patterns from a targeted pattern document set. We built a demo search system using the proposed method and conducted search experiments for 131 pattern documents gathered from the World Wide Web. The experiment results confirmed that popular patterns can be pulled up in the results.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architectures—*Patterns*

General Terms

Design

Keywords

Patterns, Interpattern Relationships, Information Retrieval

1. INTRODUCTION

There are currently a large number of digitized pattern documents[3]. However, there are no ordering method specialized for patterns, thus, developers have to read and evaluate many pattern documents to find appropriate patterns for developer's facing problems. Through answering the question what is an appropriate pattern is difficult and depends on each situation, we focus to popularity of each pattern in this paper, because many patterns must refer to "core" patterns in each domain.

In this paper, we propose a popularity calculation method specialized for pattern documents and demonstrate pattern search using the system with the proposed method.

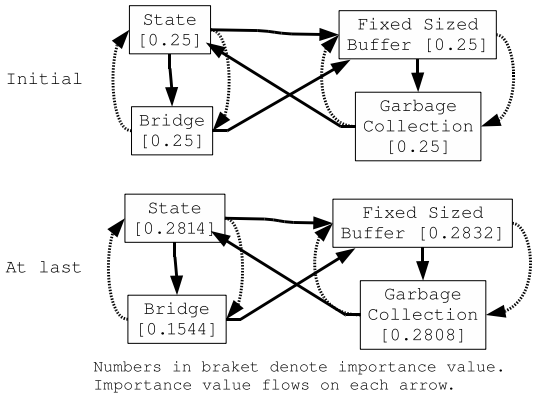

Figure 1: Convergence Calculation

2. PATTERNRANK METHOD

The proposed method named "PatternRank" is a simple expansion of Google's PageRank method[1]. In PageRank, documents are mapped to graph nodes and links between documents are mapped to graph edges. In PatternRank, pattern documents and relationships between pattern patterns play each role above. We deal an occurrence of pattern's name as a relationship between patterns, denoted as a solid arrow in Figure 1. Another point of expansion by PatternRank is introducing "in the same pattern catalog" relationships, denoted as a dotted arrow in Figure 1. Pattern catalogs are specific structure of pattern documents. "in the same pattern catalog" relationships reflect these structure into the proposed method.

Progression of the calculation of our method is illustrated in Figure 1. At first, each pattern node has an equal importance value. New importance value of each node is a sum of incoming edges' value. Importance value of each node is split

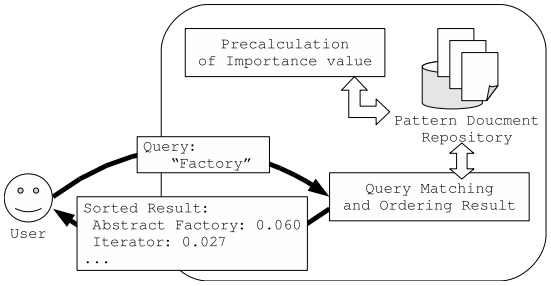

Figure 2: System Overview

up into outgoing reference edges and same-pattern-catalog edges. Amount of same-pattern-catalog edge's importance value is smaller than reference edges and controlled by a parameter named pattern catalog distribution ratio. After several dozen repeats of calculation, importance value of each pattern converges. Obtained value indicates popularity of each pattern. Full explanation of the proposed method is available at the PLoP2008's website (`http://hillside.net/plop/2008/`).

3. DEMONSTRATION

We built a demo search system using the proposed method illustrated in Figure 2. Importance values of patterns are calculated before user's querying. When a user sends search query to the system, it extracts the pattern documents matching the query from the repository, sorts the documents by importance values of each pattern document, and sends the result to the user.

For the demonstration, 131 pattern documents were gathered from the World Wide Web[2, 4, 5, 6]. A pattern catalog distribution ratio was set into 0.15 in the demonstration.

The top seven results of searching with query "memory" is shown in Table 1. Table 2 shows the result in a same condition except sorting by alphabetical order. The value in the column *Suitable?* becomes "Yes" if each pattern corresponds to problems around memory management, and it is marked by the authors of this paper through discussions on the intent of each pattern. The columns *Rec.* and *Prec.* put the values of recall and precision. Recall is the ratio of two elements, the number of suitable results above each rank and the number of all suitable results. Precision is the ratio of two elements, the number of suitable results above each rank and the number of all results above each rank.

In Table 1, several design patterns related to using memory efficiently, such as `Lazy Evaluation`, `Proxy`, and `Flyweight` patterns appear with high rank. `Lazy Evaluation` pattern achieves reduction of memory use by postponing evaluation when evaluation is needed. `Virtual Proxy` within `Proxy` pattern also achieves reduction of memory use by not creating objects until they are needed. `Flyweight` pattern saves memory by sharing it between objects of the same type that are used often. In the set of pattern marked as suitable, `Flyweight` pattern was lifted up to higher rank by references from many other patterns such as `Composite` and `State` patterns.

4. CONCLUSION

Table 1: Result of Query "memory" ordered by PatternRank

	Pattern	Suitable?	Im.	Rec.	Prec.
1	Flyweight	Yes	0.147	0.11	1.0
2	Lazy Evaluation	Yes	0.145	0.22	1.0
3	High Speed Serial Port	Yes	0.111	0.33	1.0
4	Proxy	Yes	0.111	0.44	1.0
5	Serial Port	Yes	0.111	0.56	1.0
6	Immutable Object	Yes	0.084	0.67	1.0
7	Iterator Interface	Yes	0.078	0.78	1.0

Im. denotes Importance Value, Rec. does Recall, and Prec. does Precision.

Table 2: Result of Query "memory" in Alphabetical Order

	Pattern	Suitable?	Im.	Rec.	Prec.
1	About Inheritance	No	0.017	0.0	0.0
2	Class As Type Code	No	0.017	0.0	0.0
3	Eval To Closure	Yes	0.017	0.11	0.33
4	Flyweight	Yes	0.147	0.22	0.5
5	High Speed Serial Port	Yes	0.111	0.33	0.6
6	Immutable Object	Yes	0.084	0.44	0.67
7	Iterator Interface	Yes	0.078	0.56	0.71

We proposed an importance calculation method specialized for pattern documents. The method enables resorting patterns by the order of popularity. The experiment results confirmed that popular patterns can be pulled up into higher rank in the result of searching. It is expected that resorting search result by using the proposed method helps developers to find a popular patterns from a targeted pattern document set.

5. ACKNOWLEDGMENTS

Our heartfelt appreciation goes to Roberta Coelho, our shepherd. Her comments dramatically improved the theoretical aspect of the proposed method. We also appreciate many suggestions given by Ralph Jonson and Nuno Flores. They genially supported us to improve our method, paper's style and expressions using precious time between workshop sessions. Finally, we would like to thank to all attendees of PLoP, especially Peter Sommerlad and Ralph Johnson, the moderators, led discussion and elicited many useful suggestions for the method.

6. REFERENCES

[1] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *the seventh international conference on World Wide Web*, pages 107–117, 1998.

[2] D. V. Camp. An object-oriented pattern digest. `http://patterndigest.com/`.

[3] S. Henninger and V. Correa. Software pattern communities: Current practices and challenges. In *Proceedings of the 14th Pattern Languages of Programs*, 2007.

[4] V. Huston. Huston design patterns. `http://home.earthlink.net/?huston2/dp/patterns.html`.

[5] E. Inc. Embedded design pattern catalog. `http://www.eventhelix.com/RealtimeMantra/PatternCatalog/`.

[6] S. Walters. Perl design patterns. `http://perldesignpatterns.com/perldesignpatterns.html`.