Web Search Behaviors for Software Development

Keitaro Nakasai, Masateru Tsunoda
Faculty of Science and Engineering
Kindai University
Osaka, Japan
n4keitaro@gmail.com,
tsunoda@info.kindai.ac.jp

Hideaki Hata
Graduate School of Information Science
Nara Institute of Science of Technology
Nara, Japan
hata@is.naist.jp

ABSTRACT
Software developers often use a web search engine to improve work efficiency. However, web search skill (i.e., efficiency to find an appropriate web site) is different for each developer. In this research, we try to clarify better web search behavior. To analyze web search behavior in programming, we made some questions about programming, and subjects solved the questions. The questions are based on Java language. Based on our experiment, to enhance the effectiveness of the web search, we suggest (1) do not read many search result pages without changing the key phrase, (2) read search result pages or the destination web pages linked to the search results carefully, before making new search, (3) Use new keywords which are not used before, when making a new key phrase.

Categories and Subject Descriptors
H.1.2 [**Models and Principles**]: User/Machine Systems—*human factors*

Keywords
Web search behavior, software development; search strategy.

1. INTRODUCTION
On the World Wide Web, there is much useful information for programming. For example, there are many programming languages references and question-and-answer (Q&A) sites of the programming. Software developers often use a web search engine (e.g., google.com) to find the information to improve work efficiency.

Web search skill (i.e., efficiency to find an appropriate web site) is different for each developer. When web search skill is low, work efficiency of programming would be low because one cannot find web sites which contain useful information. To improve work efficiency, we try to clarify how to use a web search engine, by analyzing web search behavior of software developers.

Some studies analyzed web search behavior in programming with limited tasks [2][4][5][6]. For example, Sadowski et al. [5] analyzed developers' behavior to search program codes. However, they do not analyze the search behavior to find useful information for programming in a universal way.

This is extended study of our past study presented in a domestic workshop [3]. We redid all analysis to enhance the reliability of the result. The difference is explained in section 4.

CHASE'16, May 16 2016, Austin, TX, USA
 ISBN 978-1-4503-4155-4/16/05□$15.00
DOI: http://dx.doi.org/10.1145/2897586.2897614

2. WEB SEARCH METRICS
To analyze web search in programming, we made some questions about programming. Subjects solved the questions, and we observed their web search behavior. To clarify the behavior quantitatively, we defined five metrics and measured them in the experiment. We assume that it is important to select search keywords (input to a search engine) and to understand search engine's results (output of a search engine). The metrics are defined based on the assumption. Details of the metrics are explained in the followings.

2.1 RPV: The number of web result pages divided by the number of viewed web pages
We defined a metric which is the number of web result pages divided by the number of viewed webpages. The web result page means that the list of relative web sites made by search engine. The numerator is how many times the page is shown in the task. We used Google as the search engine, and ten web pages' URLs are included in each result page. The denominator is sum of all accessed web pages in the task. For example, if a developer looks at ten web pages and one result page, the metrics value would be 0.1.

Typically, there are two ways to use a search engine. One way is that a developer sees only the web page title and the summary listed in the search result, and he/she judges whether the web page is useful or not. The other way is that a developer does not only see the search results, but he/she accesses and reads the destination web pages linked to the search results. When the value of the metric is high, a developer read search result pages mainly. In contrast, when the value is low, he/she accesses the destination web pages linked to the search results.

2.2 PPR: The number of unique key phrases divided by the number of web result pages
We defined a metric which is the number of unique key phrases divided by the number of web result pages. The key phrase means a set of keywords, which are input to a web search engine by a developer. For example, "class interface" and "null pointer exception" are key phrases. The numerator of the metric is the sum of unique key phrases in a task. For example, if used key phrases are "class interface," "class interface" and "null pointer exception," the value is two. The denominator is defined in the above subsection.

When the value is high, a developer changes key phrase frequency. On the contrary, when the value is low, he/she reads search result pages carefully, and visits web sites shown in the page. Note that when a developer uses "go back" button, the value became low, since it increases the denominator.

2.3 WPP: The number of keywords divided by the number of unique key phrases

We defined a metric which is the number of keywords divided by the number of unique key phrases. The numerator is the number of keywords in all key phrases. For example, when a key phrase is "class interface," the number of keywords is two. That is, the metric is the average of keywords in a key phrase.

The Google search guideline suggests it is better to use few keywords in a key phrase. We analyzed whether the guideline is effective or not in programming. When the value is high, a developer uses many keywords in a key phrase. In contrast, when the value is low, he/she uses a few keywords.

2.4 RPT: The number of web result pages divided by task time

We defined a metric which is the number of web result pages divided by task time. Task time is time to solve a question given to subjects. For example, when a subject sees 10 web result pages and the task time is 10 minutes, the value is 1.0.

When the value is high, a developer inputs key phrases many times to find information, but he/she does not access the destination web pages linked to the search results. On the contrary, when the value is low, a developer read the result page or the destination pages carefully.

2.5 WPK: The number of unique keywords divided by the number of keywords

We defined a metric which is the number of unique keywords divided by the number of keywords. The number of unique keywords is sum of unique keywords in key phrases in a task. For instance, when a developer uses "class instance," "class abstract," and "class instance," unique keywords are "class," "instance," and "abstract." In this case, the numerator is three, and the denominator is six.

When the value is high, to make a new key phrase, a developer does not use same keywords used in past. In contrast, when the value is high, he/she uses the same keyword to make new key phrases.

3. EXPERIMENT

3.1 Overview

We made some questions about programming, and subjects solved the questions. To analyze their web search behavior, we made a system for the questions. It checks subject's answer, and when it is correct, the system shows a next question. After the tasks are finished, we classified results into two groups based on whether their answer is correct or not. In addition, we classified them based on the time to finish the task.

The number of subjects was ten. Nine of them were undergraduate students, and the rest is a graduate student. Their major is computer science. In the experiment, we used a windows laptop computer. Subjects used same web browser (Google Chrome), same search engine (Google) and same IDE (Eclipse). Before the experiment, the histories of the web browser were deleted.

In the analysis, we did not care time spent to write java programs, since the questions about programming can be solved easily when subjects find appropriate libraries.

To clarify the purpose of the study, we set two research questions as follows:

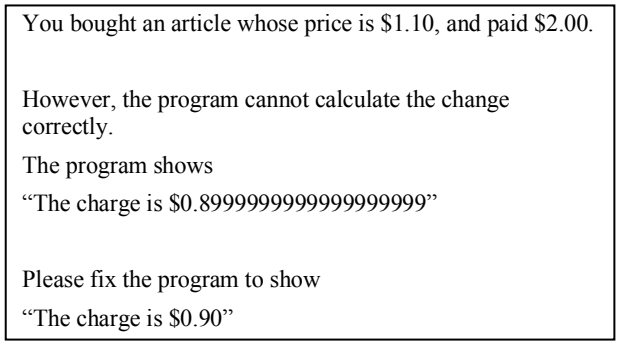
RQ1: Can we identify effective web search behavior using the defined metrics?

RQ2: If RQ1 is "yes," which metrics is useful to identify the effective web search behavior?

3.2 Questions about Programming

The questions about programming in the experiment are based on Java language, since all subjects know Java language. In the question, a source code is given, and subjects modify it. The subjects modified the code on Eclipse using our experimental system.

- Question one: In a given source code, floating-point operation raise an error. In the question, we asked how to fix the problem using a library of Java [1]. To solve it, subjects have to find the appropriate library.
- Question two: A given source code uses Array List as the data structure to store data. Subjects should modify the program to use Associative Array. The question does not indicate to use Associative Array directly. But Associative Array should be used to fulfill the required function described in the question.
- Question three: In a given source code, it gets data from a web site. It shows character codes, although an expected result is html document format. To solve it, subjects have to



(a) Question about programming

```
double answer = 2.00 – 1.10:

System.out.println("The charge is $" + answer );
```

(b) Given source code

```
BigDecimal answer = new BigDecimal("2.00").
                        subtract(new BigDecimal("1.10"));
System.out.println("The charge is $" + answer );
```

(c) Correct answer

Figure 1. Source code and the correct answer of the question one

find an I/O library of Java.

- Question four: In a given source code, it does not show appropriate error message. To solve the question, subjects need to understand the exception hierarchy of Java.

In the questions, subjects solve it easily, if they find appropriate information on the web. The difficulty of the question one and two is relatively low, and the question three and four is relatively high. We do not instruct how to search needed information. Figure 1 shows the source code given to subjects and the correct answer of the code.

In the experiment, we did not limit time for subjects to solve the questions. However, subjects can pass each question if the time is more than 20 minutes. In this case, we regarded the answer as incorrect. In the experiment, subjects cannot see other question. For example, when solving question two, he/she cannot see the question one. Since in our preliminary experiment, some subjects go back to previous questions.

4. RESULTS

We asked subjects whether they already know what library is needed to solve the questions or not before web search. We eliminated their data from the analysis, because we analyze the web search behavior, when developer does not have enough knowledge about the problem. For example, if a subject knows a library which is needed to solve question one, we removed the data of the subject before calculating metrics on question one. The elimination is the major and important difference from our past study presented in a domestic workshop [3]. It enhances the reliability of the result.

We divided the results into two groups based on two criteria.

- When the answer was correct, we classified it to "Correct group." When it was not, we classified it to "Incorrect group."
- When the task time was shorter than the median in each question, we classified it to "Short group." When it was not, we classified it to "Long group." Note that the short group does not include subjects whose answer was incorrect.

Table 1 shows results of questions on each subject. On the table, "R" indicates correct answer, "W" indicates incorrect answer, and "K" indicates that the subject have the knowledge (i.e., needed library) of the question. In Table 2, the task time of each subject on each question are shown. The unit of the time is minutes.

Table 3 shows the average and the median of five metrics, classifying subjects. The column "all" is without classification. We named the five metrics as RPV to WPK. The average and median of the metrics are smaller than 1.0, except for RPT. So, if the difference of the metrics between two groups is about 0.1, it is not ignorable, since there is almost 10% difference.

Result of RPV: The average of the correct group and the short group, and the median of the short group were smaller than the

Table 1. Results of questions on each subject

	A	B	C	D	E	F	G	H	I	J
Question 1	R	R	W	R	W	W	R	K	R	R
Question 2	R	R	R	K	R	W	W	K	R	R
Question 3	W	R	W	K	R	W	R	K	R	W
Question 4	W	R	R	K	K	W	R	R	W	W

Table 2. Task time of each subject on each question

	A	B	C	D	E	F	G	H	I	J
Question 1	19.13	28.08	41.12	10.70	41.57	30.73	32.22	6.42	13.85	30.70
Question 2	26.72	46.43	25.97	4.58	18.83	29.60	62.37	5.18	18.85	18.53
Question 3	26.18	67.18	20.82	12.22	8.27	31.00	23.73	10.55	30.85	96.68
Question 4	29.48	18.78	20.60	2.55	3.32	21.97	12.55	14.33	59.68	64.63

Table 3. Web search behavior metrics

		All	Correct	Incorrect	Short	Long
RPV: The number of web result pages divided by the number of viewed webpages	**Average**	0.56	0.54	0.57	0.53	0.57
	Median	0.57	0.58	0.56	0.57	0.58
PPR: The number of unique key phrases divided by the number of web result pages	**Average**	0.44	0.46	0.41	0.46	0.43
	Median	0.40	0.45	0.37	0.45	0.40
WPP: The number of keywords divided by the number of unique key phrases	**Average**	2.53	2.54	2.52	2.62	2.46
	Median	2.50	2.73	2.38	2.82	2.38
RPT: The number of web result pages divided by task time	**Average**	0.78	0.75	0.82	0.62	0.90
	Median	0.68	0.60	0.69	0.58	0.69
WPK: The number of unique keywords divided by the number of keywords	**Average**	0.54	0.56	0.50	0.60	0.49
	Median	0.48	0.48	0.44	0.56	0.44

other groups. However, the median of the correct group was larger than the incorrect group. That is, the result was inconsistent. Also, the difference was not large. So, we did not conclude that the metric relates to the effectiveness of web search.

Result of PPR: The average and the median of the correct group and the short group were larger than the other groups. Also, the difference was about 0.05. So, the metric is regarded to relate to the effectiveness of web search to some extent.

When the value of the metric is large, developers change the search key phrase frequently, without seeing many web result pages. That is, it is not very effective that reading many web result pages without changing the key phrase.

Result of WPP: The average and the median of the correct group and the short group were larger than the other groups. However, the difference of the average was small, and the difference of the median was almost 0.4. So, we did not conclude that the metric relates to the effectiveness of web search. Although Google suggests it is better to use few keywords in a key phrase, it is not confirmed by our experiment.

Result of RPT: The average and the median of the correct group and the short group were smaller than the other groups. The difference was about 0.1. So, when the value of the metric is small, the web search behavior is effective.

When the value is small, a developer does not use the search engine (or input key phrases) frequently, to solve the task. So, we suggest reading web result pages or the destination web pages linked to the search results carefully.

Result of WPK: The average and the median of the correct group and the short group were larger than the other groups. The difference was about 0.05. So, when the value is high, web search behavior is effective.

When the value of the metric is high, a developer does not use same keywords used in past key phrases, when he/she makes a new key phrase. That is, it is better to use new keywords which are not used before.

Based on the results, we answered "yes" to RQ1, and answered "Metrics PPR, WPP, RPT and WPK" to RQ2.

5. CONCLUSION

We analyzed the web search behavior in programming. To analyze it, we made some questions about programming, and subjects solved the questions. Based on the experimental results, we suggest effective web search as follows:

- Do not read many web result pages without changing the key phrase.
- Read search result pages or the destination web pages linked to the search results carefully, before making new search.
- Use new keywords which are not used before, when making a new key phrase.

Although in the actual software development, developers use a search engine for various reasons, we believe the suggestion will be effective especially when developers search program libraries. The limitations of our study are that the number of subjects was relatively small, and the subjects were students, and not real programmer. To enhance the reliability of the study, they should be improved. As future work, we make more metrics to analyze web search behavior in programming, and make more questions to clarify other characteristics of web search behaviors.

6. ACKNOWLEDGMENT

This study has been supported by JSPS KAKENHI Grant Number 26540029 and 25330090, and has been conducted as a part of JSPS Program for Advancing Strategic International Networks to Accelerate the Circulation of Talented Researchers.

7. REFERENCES

[1] Bloch, J. and Gafter, N. 2005. *Java Puzzlers: Traps, Pitfalls, and Corner Cases*, Addison-Wesley Professional.

[2] Bajracharya, S. and Lopes, C. 2012. Analyzing and mining a code search engine usage log, *Empirical Software Engineering*, 17, 4, 424-466.

[3] Nakasai, K. and Tsunoda, M. 2015. Analysis of web search action in software development, Fundamentals of software engineering (FOSE) (in Japanese).

[4] Gallardo-Valencia, R. and Sim, S. 2011. Information used and perceived usefulness in evaluating web source code search results, *CHI Extended Abstracts*, 2323-2328.

[5] Sadowski, C. Stolee, K. and Elbaum, S. 2015. How developers search for code: a case study, In *Proc. of Joint Meeting on Foundations of Software Engineering*, 191-201.

[6] Sim, S. Umarji, M., Ratanotayanon, S. and Lopes, C. 2011. How Well Do Search Engines Support Code Retrieval on the Web? *ACM Transactions on software Engineering and Methodology*, 21, 1, 4.