Journal of Robotics, Networking and Artificial Life, Vol. 3, No. 3 (December 2016), 163-167

Quantitative Evaluation of Flash-based Educational Visualizing Simulator

Kei Takeichi[1], Yoshiro Imai[1], Kazuaki Ando[1], Koji Kagawa[1], Tetsuo Hattori[1*], Yusuke Kawakami[2]

[1] Faculty of Engineering, KagawaUniversity,
2217-20 Hayashi
Takamatsu City, Kagawa 761-0113, Japan
E-mail: {imai, ando, kagawa, hattori [*] }@eng.kagawa-u.ac.jp

[2] DynaxT Co., Ltd.
2271-6 Hayashi
Takamatsu City, Kagawa 761-0113, Japan
E-mail: riverjp2002@gmail.com

Abstract

A Flash-based simulator of CPU scheduling has been developed and utilized for educational visualization in the class of university lecture. We have designed and implemented it with Flash-based scripting language in order to execute it as a stand-alone application as well as in various browsing environment such as Microsoft IE, Chrome and/or FireFox. Based on questionnaire for our simulator, its quantitative evaluation has been carried out by means of statistical analysis. The report describes Flash-based simulator and the results of the quantitative evaluation.

Keywords: Educational visualization, Questionnaire-based Evaluation, Statistical analysis.

1. Introduction

As you know, computer system consists of hardware and software, and now main emphasis of education about computer system may be shifted from hardware to software in almost all educational institute. In higher education, especially, information processing and/or computer science, it is very much important for students to understand structure and behavior of computer system as well as software system. So information processing education trends to focus on software system rather than hardware system. As a matter of course, software system covers many important areas from fundamentals to applications. This time, we focus on Operating System, and particularly CPU scheduling algorithm. It must be cover several kinds of themes that students should understand during their school days. A simple algorithm, namely First Come and First Served is very natural so that it is one the most fundamental strategies to decide its priority for users, clients, processes/tasks and so on.

Priority based algorithm is another candidate to determine the order of execution. It is very significant idea to choose a suitable item around potentially selected targets. It means which is better, or which is optimal of them. Shortest Processing Time First is to be chosen as one of the priority-based algorithms in this study. In other view point, Round Robin is evaluated as a policy of algorithm to realize equality of opportunity around the targets. It is a little complicated but very useful strategy to choose item with equal opportunities.

[*] Dr Hattori has been now a Professor Emeritus of Kagawa University since Apr. of 2015.

Published by Atlantis Press

In order to teach students these above algorithms efficiently, we had better utilize some suitable educational tool to visualize their behavior and results for specific conditions [12]. A visual simulator is one of the useful solutions to provide educational tools for students who want to understand such theme of information processing education in trial-and-error, step-by-step, and demonstrative manners[3]. In this study, we have developed some useful educational tools to demonstrate practical CPU scheduling algorithm and provide visual understanding[45] for students in an effective way.

This paper describes how to design Adobe Flash based simulator for CPU scheduling algorithm, realize it in order to execute on the useful environments, and then perform quantitative and qualitative evaluation through real education, namely in the course of information processing of university. The next section illustrates system configuration of our visual simulator and its behavior on the usually executing environments. The third section demonstrates quantitative evaluation mainly by means of questionnaire for its users in the real class and statistical analysis of its results. The last section concludes our study on development and evaluation of Adobe Flash based visual simulator for CPU scheduling algorithm.

2. Simulator for CPU Scheduling Algorithm

In a lecture of Operating System of our university, most important algorithms of CPU scheduling are FCFS (First-Come First-Served), SPTF (Shortest Processing Time First) and RR (Round Robin), we think. They are very trivial but sometimes very useful in the real operating systems. So we have employed these above algorithms as fundamental procedures to decide CPU scheduling in our visual simulator. The above three procedure based on FCFS, SPTF and RR are as follows;

2.1. First-Come First-Served (FCFS)

FCFS is very much simple, but clearly well-defined strategy to decide next candidate to be performed, just like First-In/First-Out (FIFO queue). Our simulator visualizes below result of FCFS algorithm in Fig.1.

This algorithm is sometimes the target to be compared with other algorithm(s). And moreover, a result applied by this algorithm will be not so worse than other ones derived from complicated algorithm(s).

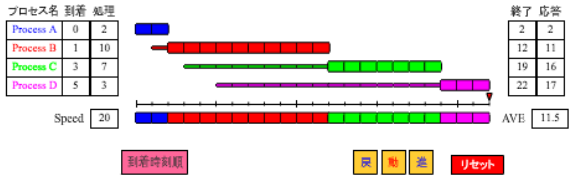

Fig. 1. Visualization of Algorithm: FCFS.

2.2. Shortest Processing Time First (SPTF)

SPTF is one of the most famous priority-based assigning/allocating algorithms. Whenever every event happens, namely conditions have changed, it must be investigated which candidate has the best priority at that time. So we had better call this algorithm Shortest Remaining Processing Time First (SRPTF), because we must consider not the total processing time but "remaining" processing one in order to decide which candidate has the best priority at that time or later. Our simulator visualizes sample result based on SPTF (SRPTF) algorithm in Fig.2.

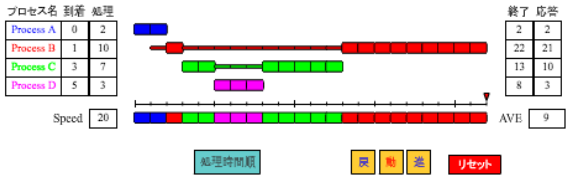

Fig. 2. Visualization of Algorithm: SPTF.

2.3. Round Robin (RR)

RR is a typical non-priority based algorithm in order to provide 'equality of opportunity' which can realize taking turns at it. This algorithm can retrieve candidates which are waiting for service and select/assign one of them who wait for the longest time or longer than others. Our simulator visualizes sample result based on RR algorithm in Fig.3.

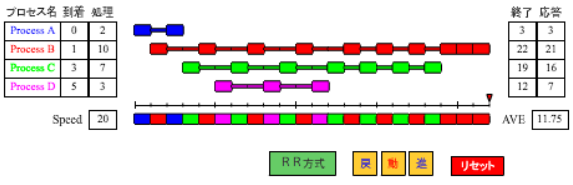

Fig. 3. Visualization of Algorithm: RR.

It is a little complicated for beginners to understand details of RR-based procedure and/or develop a kind of corresponding programs. And we may sometimes meet its results with not suitable performance for specific applications. But 'equality of opportunity' is important for several users to receive their necessary services.

3. Quantitative Evaluation

This section presents quantitative evaluation of our visualizing simulator[6].

3.1. Comparison of Execution Time

As one of the quantitative evaluation for our CPU scheduling simulator, at first, we compare the execution time of simulation of native Flash player with ones on the below three major browsers. The result is summarized in the following Table 1.

Table 1. Comparison of Execution Time(s) between Different Environments.

Host Application	Execution Time
Flash Player (Ver.10) Stand alone	32.25 (sec.)
MS-Internet Explorer 11.0.9600	28.95 (sec.)
Mozilla FireFox 35.0.1	35.19 (sec.)
Google Chrome 40.0.2214.11 m	93.61 (sec.)

3.2. Questionnaire after Using Simulator

As another quantitative evaluation for our simulator, secondly, we have carried out questionnaire in the classroom lecture of Operating System in our university after using our simulator. The questionnaire includes the following six questions.

No.1 of Questionnaire (Q#1)

Is it easy to utilize this simulator? (yes: 2, neutral:1, no:0).

No.2 of Questionnaire (Q#2)

Is it effective to learn CPU scheduling algorithm with this visual simulator? (yes: 2, neutral:1, no:0)

No.3 of Questionnaire (Q#3)

Do you understand CPU scheduling algorithm more suitably with this simulator? (yes: 2, neutral:1, no:0)

No.4 of Questionnaire (Q#4)

Are you interesting in CPU scheduling algorithm by means of this simulator? (yes: 2, neutral:1, no:0)

No.5 of Questionnaire (Q#5)

Are you interesting in other themes of Operating System after usage of this simulator? (yes: 2, neutral:1, no:0)

No.6 of Questionnaire (Q#6)

Do you need to utilize another type of simulator in order to learn Operating System? (yes: 2, neutral:1, no:0)

3.3. Results of Questionnaire

Our questionnaire described before can obtain just 20 answers from students of the class because of carrying out on a voluntary basis, although we used to have the class for Operating System with 40 students or more. The result of such a questionnaire is summarized in Table 2.

Table 2. Results of Questionnaire about our Simulator.

Student	Q#1	Q#2	Q#3	Q#4	Q#5	Q#6
S01	2	2	2	1	1	2
S02	2	2	1	1	1	2
S03	1	1	0	2	1	2
S04	2	2	2	1	2	2
S05	2	2	1	1	1	1
S06	2	2	2	1	1	2
S07	2	2	1	1	1	2
S08	2	2	2	1	1	2
S09	2	2	2	1	1	1
S10	2	2	2	1	1	2
S11	2	2	1	1	1	2
S12	2	2	2	1	1	1
S13	1	2	1	1	1	2
S14	2	1	2	1	1	2
S15	2	2	2	2	1	2
S16	2	2	1	1	1	2
S17	2	2	1	2	1	2
S18	2	2	2	2	2	2
S19	2	2	2	1	1	2
S20	1	2	2	2	2	2

Q#1 has 17 numbers of answer "yes" per 20 students (i.e. 85%), Q#2 has 18 numbers of answer "yes" per 20 students (i.e. 90%) and Q#3 has 12 numbers of answer "yes" per 20 students (i.e. 60%). From the questionnaire, many students do feel easy to utilize our CPU scheduling simulator and consider to be effective for learning CPU scheduling algorithm by means of using our simulator. And majority, namely six out of ten, of replying students understand CPU scheduling algorithm more suitably with this simulator. At the same time, however, Q#4 has just 5 numbers of answer "yes" per 20 students (i.e. 25%) and Q#5 has only 3 numbers of answer "yes" per 20 students (i.e. 15%).

4. Statistical Analysis

In order to perform test of independence among Q#1, Q#2 and Q#3, we will demonstrate to calculate "χ^2 test of goodness-of-fit" for relation between results from Q#1 and Q#2 as well as one for Q#1 and Q#3, respectively. Relation between results from Q#1 and Q#2 is expressed in the left-hand of Table 3, while relation for Q#1 and Q#3 is done in the right-hand. The former has 2 x 2 table-items and the latter has 2 x 3 ones.

Table 3. Relation between Results from Q#1 and Q#2 (left-hand) & from Q#1 and Q#3 (right-hand).

		Q#2			Q#3		
		yes	neutral	no	yes	neutral	no
Q#1	yes	16	1		11	6	0
	neutral	2	1		1	1	1
	no						

Based on Table 3, a two-way contingency table for Q#1 and Q#2 can be introduced, which is shown in Table 4, while another two-way contingency table for Q#1 and Q#3 can be also done, which is shown in Table 5.

Table 4. Two-way Contingency Table for Q#1 and Q#2.

		Q#2		
		yes	neutral	SUM_R
Q#1	yes	16(18*17/20)	1(2*17/20)	17
	neutral	2(18*3/20)	1(2*3/20)	3
SUM_C		18	2	20

The χ^2 (goodness-of-fit statistic) for Table 4 can be calculated in the following expression of Eq. (1).

$$\chi^2 = \sum_{i=1}^{2}\sum_{j=1}^{2}\{(y_{ij} - \frac{sum_R * sum_C}{Total})^2 / \frac{sum_R * sum_C}{Total}\}$$
$$= \{16 - (18*17/20)\}^2 / (18*17/20) + \cdots$$
$$+ \{1 - (2*3/20)\}^2 / (2*3/20) = 2.135 \quad (1)$$

As described in Eq. (1), degree of freedom for Table 4 is $\nu = (2-1) \times (2-1) = 1$. So we can have χ^2 $\alpha=0.05(\nu=1) = 3.8415$ from the χ^2 distribution table. "Statistical independence" between results from Q#1 and Q#2 can be confirmed so that almost users of our simulator not only consider it to be easy to utilize but also recognize effectiveness to learn CPU scheduling algorithm with it respectively and independently.

Table 5. Two-way Contingency Table for Q#1 and Q#3.

		Q#3			SUM_R
		yes	neutral	no	
Q#1	yes	11(12*17/20)	6(7*17/20)	0(1*17/20)	17
	neutral	1(12*3/20)	1(7*3/20)	1(1*3/20)	3
SUM_C		12	7	1	20

Just like the same way, the χ^2 (goodness-of-fit statistic) for Table 4 can be calculated in the following expression of Eq. (2).

$$\chi^2 = \sum_{i=1}^{2}\sum_{j=1}^{3}\{(y_{ij} - \frac{sum_R * sum_C}{Total})^2 / \frac{sum_R * sum_C}{Total}\}$$
$$= \{11 - (12*17/20)\}^2 / (12*17/20) + \cdots$$
$$+ \{1 - (1*3/20)\}^2 / (1*3/20) = 0.1365 \quad (2)$$

As described in Eq. (2), degree of freedom for Table 5 is $\nu = (2-1) \times (3-1) = 2$. So we can have χ^2 $\alpha=0.05(\nu=2) = 5.9915$ from the χ^2 distribution table. "Statistical independence" between results from Q#1 and Q#3 can be also confirmed so that users of our simulator not only consider it to be easy to utilize but also understand CPU scheduling algorithm more suitably with our simulator respectively and independently. Major part of our simulator's users recognize that they understand CPU scheduling algorithm more suitably with the simulator whether or not such simulator is considered to be easy to utilize.

In other words, both of results from Q#1 and Q#2 are not only very good scores, namely 85% of the former's answers are "yes" and 90% of the latter's answers are "yes", but also the two scores are statistically independent each other, namely there is no reason that one scores can become good because another scores are good. And major part (i.e. 60%) of users, whose answers from Q#3 are "yes", understand CPU scheduling algorithm more suitably by means of our simulator independently from its operability.

5. Conclusions

This paper has described an Adobe-Flash based educational visualizing simulator for students to learn CPU scheduling algorithm graphically and practically. Our Flash-based simulator can execute on the major Web browsers such as Microsoft InternetExplorer, Mozilla FireFox and Google Chrome and provide efficient explanation for students of lecture "Operating System".

As quantitative evaluation for our simulator, we have carried out a questionnaire for the students using the simulator and have applied statistical analysis for the results of the questionnaire as shown in the followings.

- 85% of answers from Q#1:"Is it easy to utilize the simulator?" is "yes", and 90% of answers from Q#2:"Is it effective to learn CPU scheduling algorithm with the simulator?" is "yes". Therefore, it is confirmed that our visual simulator for CPU scheduling algorithm can provide easy utilization and effective learning to its users.
- Major part, namely 60%, of learners can understand CPU scheduling algorithm more suitably with the simulator, although only 25% of users are interesting in CPU scheduling algorithm by means of the simulator.
- According to statistical analysis, the results from Q#1, Q#2 and Q#3 are statistically independent from one another, and it is confirmed that each answer did not have an effect on other answering behaviors statistically.

Then, we can consider that easy utilization, effective learning and CPU scheduling algorithm understanding are statistically independent from one another, and they are good attributes for our visual simulator for CPU scheduling algorithm.

As a further study, we would also like to evaluate each student's understanding situation in the learning process more precisely, utilizing some estimation and evaluation methods adopted in time series analysis and Kansei Engineering as described in Ref. 7-10.

References

1. H. Sarjoughian, Yu Chen and K. Burger, A component-based visual simulator for MIPS32 processors, in *Proc. 38th Annual Conf. Frontiers in Education (FIE 2008)*, pp. F3B-9 – F3B-14 (October 2008).
2. M.T. Kabir,M.T. Bari and A.L. Haque, ViSiMIPS: Visual simulator of MIPS32 pipelined processor, in *Proc. 6th Int. Conf. Computer Science & Education (ICCSE2011)*, pp. 788 – 793 (August 2011).
3. Y. Imai, S. Tomita, H. Niimi and T. Kitamura, Web-Based Computer Visual Simulator. in *Technology Enhanced Learning* (IFIP International Federation for Information Processing), Volume 171, pp 111–120 (Summer 2005).
4. K.C. Lee, J. Lee, Programming physics softwares in Flash, *Computer Physics Communications*, Volume 177, Issues 1-2, pp 195–198 (July 2007).
5. P. Stodulka, P. Privitzer, J. Kofranek, M. Tribula and O. Vacek, Development of WEB accessible medical educational simulators in Proc. *6th EUROSIM Congress on Modelling and Simulation*, 6 pages (September 2007).
6. Y. Imai and K. Takeichi, Development and Evaluation of Adobe Flash based CPU Scheduling Simulator Executable on Major Multiple Web Browsers in *Proc. 2015 IEEE Int. Conf. Intelligent Networking and Collaborative Systems (INCoS-2015)*, pp.149–155 (September 2015).
7. Yoshihide Koyama, Tetsuo Hattori, Hiromichi Kawano, Model Introduced SPRT for Structural Change Detection of Time Series (I) -- Formulation --, *Journal of Robotics, Networking and Artificial Life* 1(1) (2014) ISSN:2352-6386, pp. 54-59.
8. Yoshihide Koyama, Tetsuo Hattori, Katsunori Takeda, Hiromichi Kawano, Model Introduced SPRT for Structural Change Detection of Time Series (II) , *Journal of Robotics, Networking and Artificial Life* 1(3) (2014) ISSN:2352-6386, pp. 237-243.
9. Yusuke Kawakami, Tetsuo Hattori, Hiromichi Kawano, Tetsuya Izumi, Experimental Investigation of Feature Quantity in Sound Signal and Feeling Impression Using PCA, *Journal of Robotics, Networking and Artificial Life* 1(4), (2015), ISSN 2352-6386, pp. 303-311.
10. Yusuke Kawakami, Tetsuo Hattori, Yoshiro Imai, Yo Horikawa, Kazuaki Ando, R.P.C. Janaka Rajapakse, Automated Processing of Multiple-Brightness Peak Histogram Image Using Curvature and Variance Estimation, *Journal of Robotics, Networking and Artificial Life* 3(1) (2016), ISSN 2352-6386, pp. 55-60.