IPSJ SIG Technical Report Vol.2013-GN-87 No.14
2013/3/19

Investigating Pair Programming Learning for Seeking Success Factors in Collaboration

JUNSHAN HU[†1] TOMOO INOUE[†2]

Pair programming is a programming technique which is conducted by two programmers work together at one work station. It has been adopted in learning programming. Although it is known to be effective in various aspects, micro observation of learning activity, collaboration, has yet to be conducted in relation to the outcome. In this study, behavior in pair programming learning was investigated in terms of verbal communication and direct programming action, and was compared in relation to the success of problem-solving. Besides the finding that more direct programming actions were taken in successful cases, a couple of successful interaction patterns were found. In the successful cases, the learners took direct programming actions more frequently 1) immediately after the dialogue or 2) immediately before the dialogue. From this, it is suggested that closely-knit dialogue and action can be an indicator of successful problem-solving.

1. Introduction

In the programming education, the ability to understand grammar of a program language and writing of a program and the ability to assemble the algorithm are required. Computer programming is not only the process of developing code, but also it is more like a process of innovation of programmer's ideas. To improve the effectiveness and efficiency of programming, collaborative programming came into being [1].

As one major form of collaborative programming, pair programming was originated in industry as a key component of the eXtreme Programming (XP) development methodology [2]. It is conducted by 2 persons who work on one machine with one set of computer equipments, including one display, one keyboard and one mouse. The programmer who does the keyboard controlling and mouse handling is considered as "Driver"; while another one, who is responsible for observing the code input, giving suggestions, contributing to the programming verbally, is called "Navigator". Pair programming has been accepted in more and more fields because of the higher code quality created and less time spent compared with solo programming [3][4][5][6]. Furthermore, it could improve programmers' programming experience and their cooperative consciousness [6][7]. The programmers' behavior plays a key role in the performance of pair programming [8][9][10][11], the cooperative work between the pair has an immediate influence on the programming result and experience [12][13][14]. However, with a better cooperative work, even the pair programmers would outperform, problem would still be encountered. The problem-solving not going smoothly might lead to the programmers' motivation decreased in the commercial industry, or would result in the students' negative emotions to study.

In this study, pair programming was conducted in an introductory programming course and the pair's behaviors in programming were focused on. We are aiming at analyzing the behavior and the behavior patterns in pair programming, which might be the factors that affect the programmers' performance and the programming result. It was reconfirmed that in pair programming, each utterance lasted for a shorter time in Success case, which had been obtained by previous research. It was found that operation covered more time and each operation lasted for a longer time in Success case. Behavior pattern that "operation after Driver and Navigator's dialogue" and "operation accompanied by Driver & Navigator's dialogue" were set, and it was also found that in Success case, this "operation after Driver and Navigator's dialogue" pattern covered more percentage among total number of operations, and there were more number of operations after dialogue in one minute in Success case; "operation accompanied by dialogue" pattern also covered more percentage among total number of operations, and there were more number of operations accompanied by dialogue in one minute in Success case. The further goal is to learn symptoms to indicate the pair programming status from the analysis. The results and findings are expected to be available to expand the collaborative programming study in Computer Supported Collaborative Learning (CSCL). It is also expected that this study could help to sense the learning status of the pair and intervene in the pair programming learning.

2. Related Works

2.1 Solo vs. Pair Programming

In the previous researches which focused on introductory programming courses, it has been proved that pair programming is more outperformed than solo programming. Pair teams were found to usually develop the program and software with higher quality [3][4][5], but the time spent was shorter than individual programmers [4]. Programmers worked in pair were more self-sufficient, generally performed significantly better on projects and exams [5][7].

These researches have shown the efficiency of pair programming, without mentioning anything about the process and the results of pair programming. Does the pair meet any problems while programming? If so, whether the problem-solving go smoothly or not? The programming process and behavior were not analyzed in these researches. In our study, pair programming is the point we focused because the behavior and cooperative work in it are worth more than that in solo programming. The behavior is analyzed, and the comparison of successful and failed cases is done in this study.

2.2 Behavior Analysis

Behavior in pair programming has been paid increasing attention in more researches. Sfetsos et al. showed that productivity for pairs is positively correlated with communication transactions [8]. Bryant et al. presented that the expertise distribution would influence the pair communication interaction, and noticed that the operation behavior was assisting intra-pair verbal communication [9]. Chong et al. presented that distribution of expertise among a pair had a strong influence on the tenor of pair programming, and keyboard control had a consistent secondary effect on decision making with the pair [10]. As the senior research of this paper, Hirai et al. compared the utterance in Success and Failure cases [11], and the insights, that successful case had longer speech length, more numbers of repeating explanations and more numbers of continuous speeches, would be available to identify the collaborative work and the programming status in pair programming.

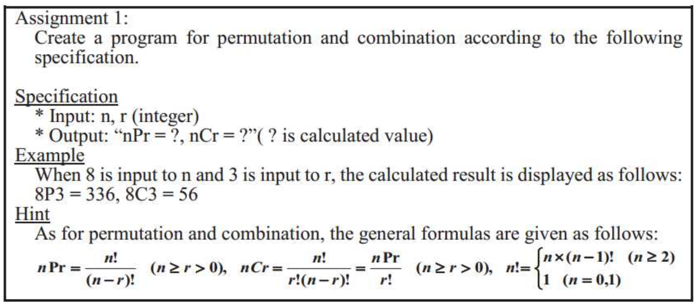
Cooperation plays important role in many group works in different domains. Cooperative behavior was also regarded as key component in pair programming and analyzed in some researches. Lory et al. and Edward et al. presented that the cooperative behavior of pair in a programming course, like think-pair-share, group question and role play, and some other cooperative work, made students work in high efficiency [12], and could increase retention and boost the performance of at-risk students [13]. Duo Wei learnt that cooperative learning method was perceived to be effective in teaching programming classes from students' survey in a pair programming course [14].

With the objective data, in this study we analyzed the operation in pair programming, and compared those in Success and Failure cases; and also paid attention to the behavior pattern related to cooperation in pair programming.

3. Data Collection

Data of pair programming was collected from one introductory programming course named "Programming I" in 2010 and 2011, in which C language was taken as the major teaching content. This course aimed at letting the students understand grammar of a program language and writing of a program, and the algorithm assembling. This course is targeted freshmen in the university's department of information and aimed at letting the students understand what C language is, how to write code in C language, and know the basic knowledge of compiling a program and developing software. Pair programming practice experiment was conducted in this course, and 10 students from 2010 course and 48 students from 2011 participated in the experiment. Each lecture of this course lasts for 75 minutes, and pair programming practice session is regarded as a part of the lecture.

In each pair programming practice session, a program-creation assignment, involving contents hitherto studied, was given to the participants. Here in Figure 1, an example of the assignment is shown.


Assignment 1:
Create a program for permutation and combination according to the following specification.

Specification
* Input: n, r (integer)
* Output: "nPr = ?, nCr = ?"(? is calculated value)
Example
When 8 is input to n and 3 is input to r, the calculated result is displayed as follows:
8P3 = 336, 8C3 = 56
Hint
As for permutation and combination, the general formulas are given as follows:

$$n\mathrm{Pr} = \frac{n!}{(n-r)!} \quad (n \geq r > 0), \quad nCr = \frac{n!}{r!(n-r)!} = \frac{n\mathrm{Pr}}{r!} \quad (n \geq r > 0), \quad n! = \begin{cases} n \times (n-1)! & (n \geq 2) \\ 1 & (n = 0,1) \end{cases}$$

Figure 1 An Example of the Exercise in the Pair Programming Class

Some preparations were done before the data collection, such like the pair combination, the Driver/Navigator role deciding in each pair, the cameras setting up, etc. Figure 2 is a screenshot of session in the "Programming I" course, three cameras were set up in one session; they recorded the pair programming from 3 different angles.

The three cameras were used for collecting pair programming data , the front one is for recording the pair's communication, the desk one is for recording the pair's behavior and activities during pair programming such as typing, using mouse, pointing at the display, referring to the textbook, and some other behaviors; and the other is for recording display. Figure 3 shows the scenes of the three angles taken by the cameras.

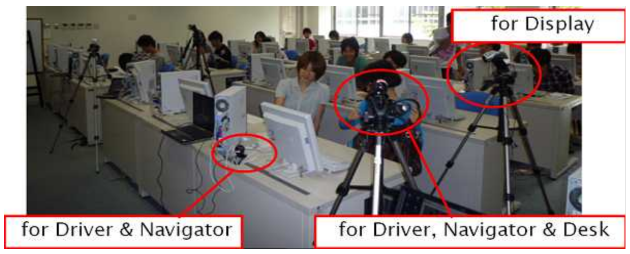


Figure 2 Set Up of the Cameras for Data Collection.

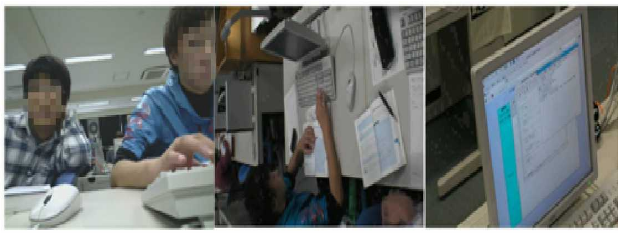


Figure 3 Set Up of the Cameras for Data Collection.

While programming together, the pairs are required to follow the instructions:

- The time limit for the assignment is 30 minutes.
- As soon as the pair combination was decided, the roles of driver and navigator could not be exchanged during pair programming practice session.
- Driver is the only one who can operate the keyboard and

mouse. The navigator could only observe and support the work of the driver without touching the mouse or keyboard.

- The assignment should be finished as soon as possible. It ends when the program is executed and a correct answer to the assignment is obtained.
- Driver and navigator could search in the textbook but not be allowed to use the Internet.
- The teacher or the teaching assistants are only available for equipment consulting. They do not accept any questions concerning the assignment while pair programming practice.
- The pair could add pertinent comment to make the program easy to understand as they like.

4. Data Processing

During pair programming, pairs would encounter different programming problems while programming and then solve them successfully, or not. We consider each problem-solving as one case, in one pair's practice, they would have none, one or more cases. Every case gets successful or failed result at last. In this case, we have exact definition for these as follows:

- A "Case" should be the problem solving process, beginning from a problem encountered and end with it being solved (problem be solved successfully) or time up (problem-solving be failed).
- A problem could be a compilation error that occurs when learners compile their program, or a runtime error that occurs including whose result does not meet the students' expectation.
- "Success" is that problem being solved by the pair within the given limited 30 minutes.
- "Failure" is that problem not being solved in the end.

We recorded each pair from different angles by using three cameras, so actually we have three videos for one pair: front video, desk video and display video. In this study we use ELAN (EDUICO Linguistic Annotator) [15][16], a tool for the creation of annotations on video and audio resources, to synchronize the three videos into one integrated video, and then to tag and annotate the behaviors in the integrated one. Figure 4 is the screenshot of the video tagging and annotation with ELAN. The videos are shown on the top of the ELAN interface, and at the bottom the tiers and annotations could be added. In Figure 3, the three videos had been one integrated video, and tags and annotations were done in this integrated video and then saved.

We annotated programmers' behavior by adding the tiers of utterance and operation. In this study utterance and computer operation are the behaviors we concerned with because they are the major behaviors in pair programming. Communication in pair programming has been regarded as the key behavior and analyzed in many previous works, and programmers operate the computer to finish the program-creation assignment. Some other behaviors, like referring to the textbook, or pointing at the display, are supposed to be analyzed in our future behavior analysis in pair programming. In this study, an utterance is the identifier of the programmer's speaking something, no matter whether he/she is talking to his/her partner or to himself/herself. It could be a sentence or just meaningless word as "Ah!", "Eh……", "Mm……", and some other mood words. Operation includes the keyboard controlling and mouse handling. Generally keyboard controlling was treated as operation because a programmer mainly uses keyboard to input code or do some other operation. As a matter of fact, mouse handling should also be regarded as computer operation because selecting, copying and pasting are all basically done by mouse.

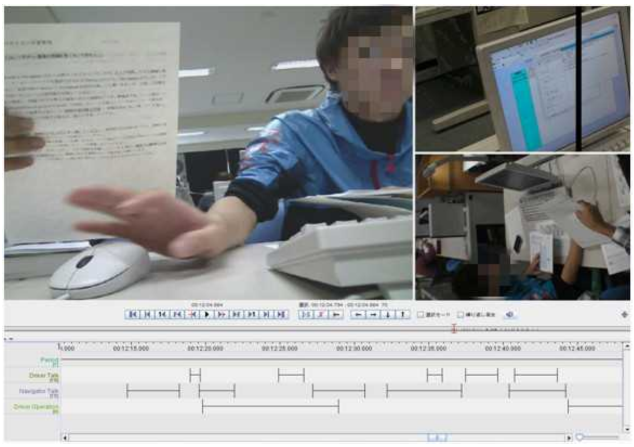


Figure 4　Screenshot of ELAN Annotation Interface

5. Parameter

5.1 Utterance data

Utterance was analyzed from the "Utterance Ratio", "Utterance Frequency" and "Average Utterance Length".

Utterance ratio is about that "what percentage of the entire case is programmer's utterance time". Utterance length is how much time the driver and navigator were talking; and data length is the length of problem-solving period. We could get the result of each programmer's utterance ratio with the formula and the result was shown in percentage.

Utterance ratio = (Utterance length / Data length)/2

Utterance frequency is the identifier of showing "how many utterance numbers there are in one minute". "Minute" is used as the time unit, so the data length should be converted to minute for analysis. To calculate each programmer's utterance frequency, the following formula was used, and then we got the result of one programmer's utterance frequency.

Utterance frequency = (Utterance numbers / Data length)/2

Average utterance length is the identifier of showing that "how much time (in second) each utterance lasts". Utterance length is how much time the two programmers were talking; and utterance numbers is the total number of utterance spoke by both driver and navigator.

Average utterance length = Utterance length / Utterance numbers

5.2 Operation data

We analyzed the operation behavior in pair programming from three views, “Operation ratio”, “Operation frequency”, and “Average Operation Length”.

Operation ratio is about that “what percentage of the entire case is the Driver’s operation time”. Operation length is that how many seconds all the operations lasted in one case, and data length is the length of problem-solving period.

Operation ratio = Operation length / Data length

Operation frequency is the identifier of showing that “how many operation numbers there are in one minute”. Operation numbers is the total number of operation done by the Driver in one case, and data length is the length of problem-solving period, here it was converted to minute for analysis.

Operation frequency = Operation numbers / Data length(min)

Average operation length is the identifier of showing that “how much time (in second) each operation lasts”. Operation length is that how many seconds all the operations lasted in one case, and operation numbers is the total number of operation done by the Driver in one case.

Average operation length = Operation length / Operation numbers

5.3 Data of Operation after Driver and Navigator’s Dialogue

We supposed there would be behavior pattern related to the pair’s cooperation in pair programming, which would lead to successful problem-solving. Correlation between utterance and operation was supposed to be one factor that showing the cooperation of the pair. Here we set the pattern defined as “operation after Driver and Navigator’s dialogue” pattern as Figure 5, as one cooperative pattern of correlation between utterance and operation. If the last two utterances before Driver’s operation are the turn-taking utterances spoke by both Driver and navigator, it would be regarded as match with our definition of “operation after (Driver and Navigator’s) dialogue”. This kind of dialogue must be at least one pair of turn-taking utterances.

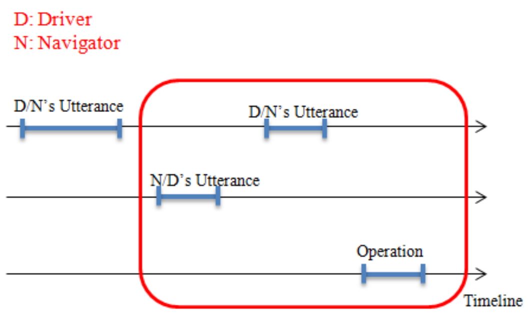


Figure 5 Operation after Driver and Navigator’s Dialogue

We analyzed the operation after Driver and Navigator’s dialogue by analyzing “Ratio of operation after dialogue”, and “Frequency of operation after dialogue”.

Ratio of operation after Driver and Navigator’s dialogue represents that “what percentage of the operation numbers is the ‘operation after dialogue’.” Number of operation after dialogue is that how many times the operation after dialogue appeared in one case, and operation numbers is the total number of operation done by the Driver in one case.

Ratio of operation after Driver and Navigator’s dialogue = Number of operation after dialogue / Operation numbers

Frequency of operation after dialogue is the identifier of showing that “how many numbers of operations after dialogue there are in one minute”. The data length here is also converted to minute for analysis. Number of operation after dialogue is that how many times the operation after dialogue appeared in one case, and data length is the length of problem-solving period.

Frequency of operation after Driver and Navigator’s dialogue = Number of operation after dialogue / Data length(min)

5.4 Data of Operation accompanied by Driver and Navigator’s Dialogue

Another behavior pattern was named “Operation Accompanied by Driver & Navigator’s Dialogue” and defined as that shown in Figure 6. This pattern is about that after the operation being finished, Driver and Navigator began the turn-taking utterance, which was regarded as “Dialogue” here.

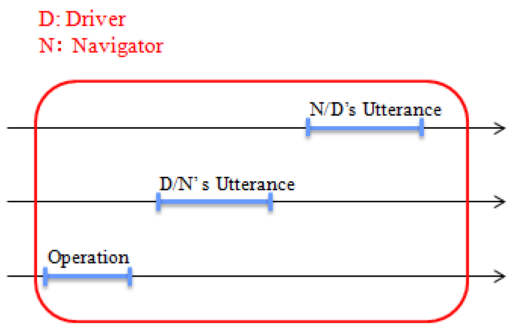


Figure 6 Operation Accompanied by Driver & Navigator’s Dialogue

We analyzed the operation accompanied by Driver and Navigator's dialogue by analyzing "Ratio of operation accompanied by dialogue", and "Frequency of operation accompanied by dialogue".

Ratio of operation accompanied by a dialogue represents that "what percentage of the operation numbers is the 'operation accompanied by a dialogue'." Number of operation accompanied by a dialogue is that how many times the operation accompanied by a dialogue appeared in one case, and operation numbers is the total number of operation done by the Driver in one case.

Ratio of operation accompanied by Driver and Navigator's dialogue = Number of operation accompanied by a dialogue / Operation numbers

Frequency of operation accompanied by a dialogue is the identifier of showing that "how many numbers of operations accompanied by a dialogue there are in one minute". The data length here is also converted to minute for analysis. Number of operation accompanied by dialogue is that how many times the operation accompanied by dialogue appeared in one case, and data length is the length of problem-solving period.

Ratio of operation accompanied by Driver and Navigator's dialogue = Number of operation accompanied by dialogue / Operation numbers

6. Results

With the parameters, we analyzed utterance, operation, and the two patterns, "operation after Driver and Navigator's Dialogue" and "operation accompanied by Driver and Navigator's Dialogue" in each case. Table 1 show the collected data of Utterance, Table 2 includes the data of Operation and Table 3 is about the data of Operation after Driver and Navigator's dialogue, Table 4 is about the data of Operation accompanied by Driver and Navigator's dialogue. Forty-five cases were analyzed, and the mean value of "Success" and "Failure" were shown in these tables.

Table 1 Utterance Data

Case type	Number of cases	Mean utterance ratio (%)	Mean utterance frequency (num/min)	Mean utterance length (sec/num)
Success	29	17.6	5.89	1.82
Failure	16	19.2	4.91	2.52

Table 2 Operation Data

Case type	Number of cases	Mean operation ratio (%)	Mean operation frequency (num/min)	Mean operation length (sec/num)
Success	29	34.0	2.96	7.80
Failure	16	22.1	2.83	4.99

Table 3 Data of Operation after Driver & Navigator's Dialogue

Case type	Number of cases	Mean ratio of operation after dialogue (%)	Mean frequency of operation after dialogue (num/min)
Success	29	60.9	1.77
Failure	16	22.8	0.62

Table 4 Data of Operation Accompanied by Driver & Navigator's Dialogue

Case type	Case amount	Mean ratio of operation accompanied by a dialogue (%)	Mean frequency of operation accompanied by a dialogue (num/min)
Success	29	42.9	1.28
Failure	16	25.9	0.66

6.1 Result of Utterance Analysis

In Table 1, the mean utterance ratio of Success is 17.6%, while that of Failure is 19.2%. With Mann-Whitney U test, $p > 0.1$ ($p = 0.53$), there is no significant difference between Success and Failure cases. And as for the mean utterance frequency of Success and Failure, for Success it was 5.89 numbers in one minute, while for Failure it was 4.91 numbers in one minute. With Mann-Whitney U test, $p > 0.1$ ($p = 0.22$), there is no significant difference between Success and Failure cases.

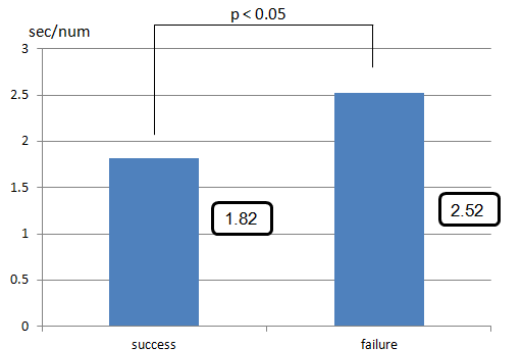


Figure 7 Average Utterance Length- How much time (in second) each utterance lasts.

From Figure 7, average utterance length of Success is 1.82 seconds, while for Failure it is 2.52 seconds. With Mann-Whitney U test, $p < 0.05$ ($p = 0.008$), the difference between Success and Failure is marginally significant. Success case has shorter average utterance length than Failure case. As a result, each utterance lasts for shorter time in Success case. And this significant result was already obtained by the senior research, which was about the 2010 pair programming analysis.

6.2 Result of Operation Analysis

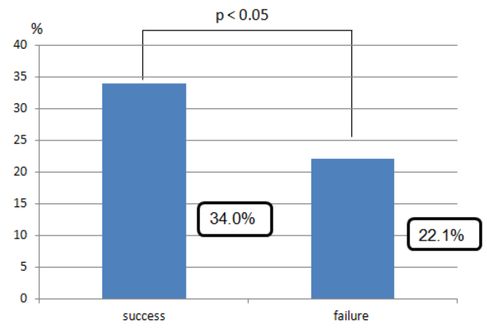


Figure 8 Operation Ratio- What percentage of the entire case is the Driver's operation time.

Figure 8 shows the mean operation ratio of Success cases is 34.0%, while of Failure cases it is 22.1%. With U test, $p < 0.05$ ($p = 0.02$), the difference between Success and Failure is marginally significant. Success case had higher operation ratio than Failure case. That is, operation covers more time in Success case.

In Table 2, the mean operation frequency of Success cases is 2.96 numbers in one minute, and of Failure cases it is 2.83 numbers in one minute. It could be seen easily that the results are similar. With Mann-Whitney U test, $p > 0.1$ ($p = 0.61$), there is no significant difference of operation frequency between Success and Failure cases. So as the result of the test, we cannot say that Success case is with lower operation frequency.

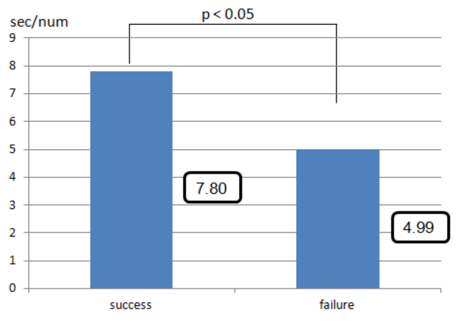


Figure 9 Average Operation Length- How much time (second) each operation lasts.

Shown as Figure 9, for Success cases, each operation lasts for 7.80 seconds averagely, while for Failure each utterance lasts for 4.99 seconds. With Mann-Whitney U test, $p < 0.05$ ($p = 0.03$), the difference of average operation length between Success and Failure is marginally significant. Success case has longer average operation length than Failure case. As a result, each operation lasts for a longer time in Success case.

6.3 Result of Operation after Driver and Navigator's Dialogue Analysis

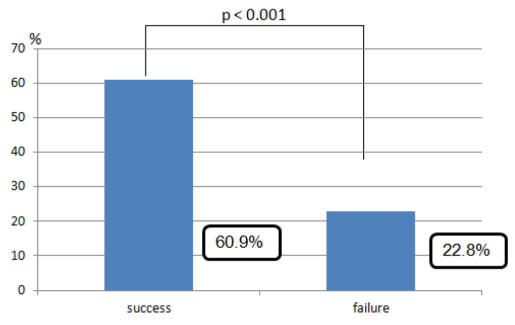


Figure 10 Ratio of operation after Driver and Navigator's dialogue- What percentage of the operation numbers is the "operation after dialogue".

From Figure 10, the mean ratio of operation after (Driver and Navigator's) dialogue of Success cases is 60.9%, while of Failure cases it is 22.8%. With Mann-Whitney U test, $p < 0.001$ ($p = 5.30584e-07$), the difference between Success and Failure is highly significant. We can get the result that Success case had higher ratio of operation after dialogue than Failure case. That is, operation after dialogue covers more percentage among the total operation numbers in Success case.

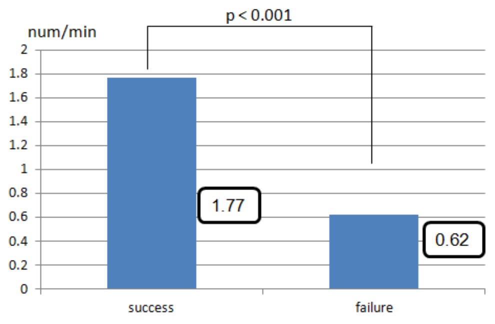


Figure 11 Frequency of operation after Driver and Navigator's dialogue- How many numbers of operations after dialogue there are in one minute.

The mean frequency of operation after dialogue is shown in Figure 11. Of Success cases it is 1.77 numbers in one minute, and of Failure cases it is 0.62 numbers in one minute. With U test, $p < 0.001$ ($p = 1.317632e-06$), the difference of frequency of operation after dialogue between Success and Failure is highly significant. As the result shown, Success case had higher frequency of operation after dialogue than Failure case. That is, there are more number of operations after Driver and Navigator's dialogue in one minute in Success case.

6.4 Result of Operation accompanied by Driver and Navigator's Dialogue Analysis

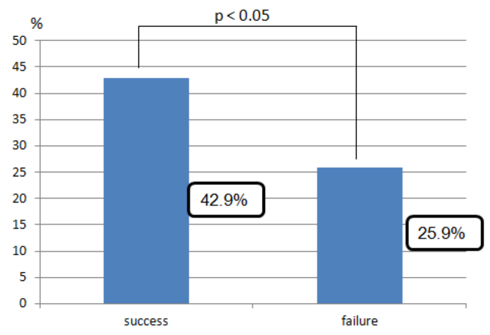


Figure 12 Ratio of operation accompanied by Driver and Navigator's dialogue- What percentage of the operation numbers is the "operation accompanied by dialogue".

From Figure 12, the mean ratio of operation accompanied by (Driver and Navigator's) dialogue of Success cases is 42.9%, while of Failure cases it is 25.9%. With Mann-Whitney U test, p < 0.05 (p = 0.02), the difference between Success and Failure is marginally significant. We can get the result that Success case had higher ratio of operation accompanied by dialogue than Failure case. That is, operation accompanied by dialogue covers more percentage among the total operation numbers in Success case.

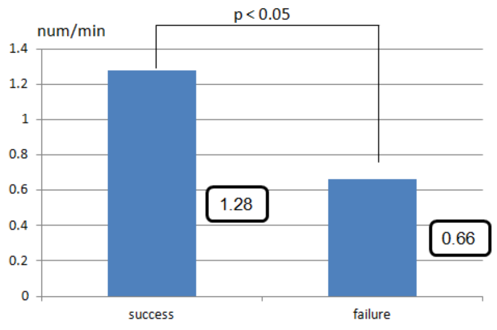


Figure 13 Frequency of operation accompanied by Driver and Navigator's dialogue- How many numbers of operations accompanied by dialogue there are in one minute.

The mean frequency of operation after dialogue is shown in Figure 13. Of Success cases it is 1.28 numbers in one minute, and of Failure cases it is 0.66 numbers in one minute. With U test, p < 0.001 (p = 0.03), the difference of frequency of operation after dialogue between Success and Failure is significant. As the result shown, Success case had higher frequency of operation accompanied by dialogue than Failure case. That is, there are more number of operations accompanied by Driver and Navigator's dialogue in one minute in Success case.

7. Discussion

The Utterance analysis results presented that Success case has shorter average utterance length than Failure case. In Success case, students' each utterance lasted for shorter time. As to the utterance ratio and utterance frequency, no significant differences were found between Success and Failure cases. This utterance analysis had already been done by the Hirai's research [11]; in this study we reconfirmed this by analyzing more pair programming data. In Hirai's research, the rates of repeated explanation and rates of consecutive speech were also analyzed. Students failed in problem-solving explained more to each other and had more repeated sentences because of the bad understanding of each other. In our study, we just paid attention to the utterance behavior simply, the point we would like to focus on more was the behavior pattern, that is, the correlation of behaviors.

In Success case, the operation ratio was higher, and the average operation length was longer. Operation covered more time and each operation lasted for a longer time in Success case. It is not surprise to get the result that Success had more operation time and longer average length than Failure. According to our observation, students failed in problem-solving usually had more other behavior such as searching in the textbook or writing on the paper because they need to search for ideas and solutions to the problem. And students in Success case, generally they solved the problem smoothly with the knowledge they had acquired, so the time for searching for solutions had been saved; they typed the code fluently, almost without stop, which resulted in more operation time and longer average operation length in Success.

Success case had higher ratio and frequency of operation after dialogue than Failure case. From our observation of the data, this dialogue was mainly the opinion exchange between driver and navigator, which should be one kind of cooperative work between the pair. As presented in previous researches, cooperation was found as one factor what would influence the efficiency in many domains, including the programming field. Students in programming course performed in high efficiency because of the cooperative activities, their retention and performance were increased and boosted. In this study, it was found that operation after dialogue covered more percentage among the total operation numbers, and there were more number of operations after Driver and Navigator's dialogue in one minute in Success case. Dialogue between the pair showed the knowledge and opinion exchange and cooperation in pair programming. With this, decision in higher quality which agreed by both was made and then operated by the driver. As Chong said, their pair programming partner could give suggestions, but fundamentally, the driver, that is, the developer at the keyboard decided which suggestion to follow [10]. If the driver did not agree with the suggestion, he would not type the code, or would begin another dialogue about the suggestion.

In Success case, operation accompanied by dialogue covered more percentage among the total operation numbers, and there were more number of operations accompanied by dialogue in one minute. When observing and analyzing the data, we noticed that students in Success cases liked to ask their partner about the operation just been done, or preferred to explain why this be operated. There was more favorable interaction and good understanding between the pair in Success case.

We analyzed two behavior patterns in our study, "Operation after Driver & Navigator's Dialogue" and "Operation accompanied by Driver & Navigator's Dialogue". There are absolutely two different patterns; however, overlap appeared between these two patterns, as shown in Figure 16.

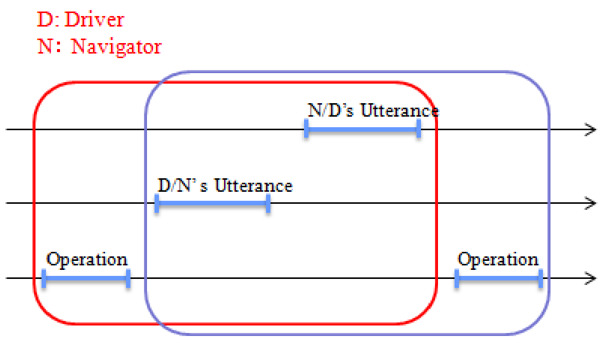


Figure 16 Overlap of "Operation after Driver & Navigator's Dialogue" and "Operation accompanied by Driver & Navigator's Dialogue"

This kind of overlap happened because that behavior was the only element considered when we did the analysis. With behavior we cannot judge that if the dialogue was related to the operation had been done or the operation to be done. To accurate the behavior pattern analysis, in the next stage, we would take the dialogue content into our analysis. What the pair programmers talked would be transcribed, and we will judge that if the dialogue belongs to the operation before or the operation next.

For future direction, new behavior analysis is going to be considered. In this study we analyzed the pattern "Operation after Driver & Navigator's Dialogue", next we would like to concern about other patterns which might be the identifier of showing pair programmers' interaction and cooperation. We also plan to conduct the control experiment of pair programming to see whether the cooperative work would really affect the programming result and now what element should be controlled is considered now.

8. Conclusion

In the programming education, the ability to understand grammar of a program language and writing of a program and the ability to assemble the algorithm are required. As one of the programming learning methods, pair programming was originated in industry as a key component of the eXtreme Programming (XP) development methodology. It improves software quality, and reduces the cost of software development.

In this study, pair programming practice sessions were conducted in a course named "Programming I", and pair programming practice sessions were recorded and observed, and the problem-solving periods were obtained and then analyzed as Success or Failure cases. We reconfirmed that Success case had shorter average utterance length, which has also already obtained by previous analysis of pair programming [11]. It was found that Success case had higher operation ratio, and longer average operation length than Failure case. It is also presented that Success case had higher ratio and frequency of operation after dialogue than Failure case, and higher ratio and frequency of operation accompanied by dialogue. More symptoms should be obtained and what would make pair programming learning and cooperative work more effective are expected to be learnt. And one control experiment is about to be conducted to see the cooperative pattern's impact on pair programming in the future.

Reference

1) L. L. Constantine, Constantine on Peopleware. Englewood Cliffs, NJ: Yourdon Press, 1995.
2) Beck, K. (1999). Extreme Programming Explained: Embrace Change, Reading, PA: Addison-Wesley.
3) Nosek, J. T. (1998) The Case for Collaborative Programming. Communications of the ACM, 41 (3), 105-108.
4) Williams, L., Kessler, R., Cunningham, W., Jeffries, R. (2000). Strengthening the Case for Pair programming. IEEE software, 17 (4), 19-25.
5) McDowell, C., Werner, L., Bullock, H., Fernald J. (2002). The Effects of Pair programming on Performance in an Introductory Programming Course, Proc. ACM SIGCSE, ACM Press, 38-42.
6) Muller, M. M. (2003). Are Reviews an Alternatvie to Pair Programming? Seventh International Conference on Empirical Assessment in Software Engineering, UK.
7) Nagappan, N., Williams, L., Ferzli, M., Wieve, E., Yang, K., Miller, C., and Balik, S. (2003). Improving the CS1 Experience with Pair programming, Proc. ACM SIGCSE, ACM Press, 359-362.
8) Sfetsos, P., Stamelos, I., Angelis, L., Deligiannis, I. S. (2006). Investigating the Impact of Personality Types on Communication and Collaboration-Viability in Pair Programming, in XP/Agile 7, 43-52.
9) Bryant, S., Romeo, P., Boulay, B. (2006). Pair Programming and the e-appropriation of Individual Tools for Collaborative Software Development, Proc. ACM SIGGROUP, 55-70.
10) Chong, J., and Hurlbutt, T. (2007). The Social Dynamics of Pair programming, Proc. International Conference on Software Engineering (ICSE), IEEE Press, 354-363.
11) Hirai, Y., Inoue, T. (2012). Collaboration Estimation in Pair Programming Learning: Conversation Differences between Success and Failure in Problem Solving, Infomration Processing Society of Japan Journal, Vol 53, 72-80.
12) Lori, P., Mike, J. (2001). Making Parallel Programming Accessible to Inexperienced Programmers through Cooperative Learning. SIGCSE 2001, 224-228.
13) Edward, F. G., Katherine, D., Keith, J. W., John, H. (2006). Panel: Cooperative Learning – Beyond Pair Programming and Team Projects. SIGCSE 2006, 458-459.
14) Duo, Wei. (2012). An Evaluation of a Cooperative Learning Method in Programming and Problem Solving I. Consortium for Computing Science in Colleges, 69-77.
15) Wittenburg, P., Brugman, H., Russel, A., Klassmann, A., Sloetjes, H. (2006). ELAN: a Professional Framework for Multimodality Research. Proceedings of LREC 2006, Fifth International Conference on Language Resources and Evaluation.
16) ELAN (EUDICO Linguistic Annotator), Available from http://tla.mpi.nl/tools/tla-tools/elan/