情報処理学会論文誌 Vol.53 No.1 72–80 (Jan. 2012)

ペアプログラミング学習における状態の推定 ——つまずきの解決の成功と失敗に見られる会話の違い

平井 佑樹[1] 井上 智雄[2,a)]

受付日 2011年4月20日, 採録日 2011年10月3日

概要：プログラミング教育では，プログラム言語の文法やプログラム書法を理解する能力とアルゴリズムを組み立てる能力が要求される．プログラム言語の文法や簡単な例題を理解することができても，実際にプログラムを作成するときにはいくつかのつまずきが発生する．プログラミングを行う方法の 1 つとして，2 人 1 組になって行うペアプログラミングがある．ペアプログラミングによるプログラミングは協調作業であるが，これはプログラミング学習の方法としても用いられている．本研究では，プログラミング学習時のペアプログラミングの成功事例と失敗事例を比較分析した．分析では作業中の会話に着目し，失敗事例の方が発話が長いこと，説明の繰返しが多いこと，一方的な発話が多いことが分かった．この知見は，ペアプログラミングにおいて協調作業がうまく進んでいるかどうかを判断する手がかりを提供し，協調作業の状態推定に有効であると考えられる．

キーワード：ペアプログラミング，プログラミング学習，協調作業，会話

Collaboration Estimation in Pair-programming Learning: Conversation Differences between Success and Failure in Problem Solving

Yuki Hirai[1] Tomoo Inoue[2,a)]

Received: April 20, 2011, Accepted: October 3, 2011

Abstract: 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. When a learner actually makes a program, some problems are caused even if a grammar and an easy example of the program can be understood. Pair-programming is one of the programming techniques in which two programmers work together at one work station. Pair-programming is a collaborative work and is used in programming learning. In this research, success cases and failure cases in pair-programming were compared. From the comparison, it was found that the speech length was long, the number of repeating explanation was high and the number of continuous speech was high in failure cases. The insights provide some clues to identify if collaborative work in pair-programming smoothly progresses and to guess the status of collaborative work.

Keywords: pair programming, programming learning, collaborative work, conversation

1. はじめに

プログラミング教育では，プログラム言語の文法やプログラム書法を理解する能力とアルゴリズムを組み立てる能力が要求される．プログラム言語の文法や簡単な例題を

1 筑波大学大学院図書館情報メディア研究科
Graduate School of Library, Information and Media Studies, University of Tsukuba, Tsukuba, Ibaraki 305–8550, Japan
2 筑波大学図書館情報メディア系
Faculty of Library, Information and Media Science, University of Tsukuba, Tsukuba, Ibaraki 305–8550, Japan
a) inoue@slis.tsukuba.ac.jp

© 2012 Information Processing Society of Japan

理解することができても，実際にプログラムを作成するとなったとき，アルゴリズムを組み立てることができなければ，プログラムを組むことができない [1] 等の問題が生じる．問題の解決を支援する実践も含め，プログラミング教育では，これまでに多種多様の実践が行われており，情報処理学会誌でも 2010 年 10 月号からプログラミング教育に関する連載が行われた [2] 等，その教育方法や学習方法に関して広く注目されている．

プログラミングを行う方法の 1 つとして，プログラミングを 2 人 1 組のペアになって行うペアプログラミングがある．ペアプログラミングは Beck [3] によって提唱されたエクストリーム・プログラミング（XP）と呼ばれるソフトウェア開発手法の 1 つのコンポーネントである．ペアプログラミングでは，1 人が Driver と呼ばれる役割として実際のコードを書き，もう 1 人は Navigator と呼ばれる役割として，Driver をチェックしながらナビゲートし，この役割を随時交代しながら作業を進める．これによって，問題解決に要する時間を短縮させる，つねにコードレビューを行うことができる等の効果が得られる．ペアプログラミングによるプログラム作成は協調作業であり，また，プログラミングに関する専門知識の共有および拡張という点でも有効という指摘があり [4]，知識の獲得，拡張に有効なコラボレーション手法であると考えられる．

このペアプログラミングを用いたプログラミング学習が行われている．特に，プログラミング導入教育におけるペアプログラミングの利用では，個人でプログラミングをするときと比較した場合に，プログラムの質が向上する，課題解決までの時間が短縮する等，その有効性が報告されている [5], [6], [7], [8], [9]．しかしながら，いくつかの実践において一部のペアでは与えられたプログラム作成課題の解決がうまくいっていないという問題は発生している．ペアの組合せ等によってペアプログラミングを行ったときに得られる効果が異なることがあると考えられるが，プログラミング導入教育では，プログラミングの基本事項を学習するために，得られる効果に違いがでてしまうと，後々の学習に悪影響を及ぼす可能性がある．特に，与えられたプログラム作成課題の解決がうまくいかないと，学習に対するモチベーションの低下等が懸念される．そこで，プログラミング導入教育におけるペアプログラミング学習において，プログラム作成課題の解決がうまくいっていないペアを支援することを考えた．

本研究では，プログラミング導入教育においてペアプログラミング学習を実施し，ペアプログラミングの成功事例と失敗事例を比較分析した．分析では解決中に行っているペアの会話に着目し，失敗事例の方が発話が長いこと，説明の繰返しが多いこと，一方的な発話が多いことが分かった．この知見は，ペアプログラミングにおいて協調作業がうまく進んでいるかどうかを判断する手がかりを提供し，協調作業の状態推定に有効であると考えられる．

2. 関連研究

2.1 プログラミング導入教育におけるペアプログラミング

既存研究では，プログラム作成課題を個人プログラミングとペアプログラミングで行わせたときに，後者の有効性について述べている．たとえば，コードの質が高かった [5], [7]，単位修得率が高かった [5], [6]，中間および最終試験の成績が高かった [6]，課題の提出率があがった [8] という結果が報告されている．また，ペアプログラミング実施前と実施後を比較したときについて，Rountree らはコード理解やコード作成スキルがあがったと報告している [8]．

いずれの研究もペアプログラミングの有効性を述べているが，プログラム作成課題中の過程や，課題解決がうまくいかなかったペアについては言及していない．本研究では，プログラミング導入教育におけるペアプログラミング学習において，ペア同士のやりとりを観察し，課題の解決に失敗したペアについても言及している．

2.2 ペアプログラミングにおけるペア同士のコミュニケーション分析

プログラミング導入教育を対象とした研究ではないが，作業中のペア同士のコミュニケーションを分析している研究がある．Chen らは，作業中のペア同士の発話内容と発話時の状況を書き下し，発話内容から Driver と Navigator には精神的な距離があること，作業中に何らかのコミュニケーション支援が必要であることを示している [10]．Chong らも，作業中のペア同士の発話内容と発話時の状況を書き下し，ペアが所有する専門的な知識が作業中のインタラクションに影響する可能性があること，キーボードの操作権（Keyboard control）がペアの意思決定に影響する可能性があることを示している [11]．Bryant らは，作業中の発話プロトコル分析を行って Driver および Navigator の発話内容の分布を調査し，両者の分布に差はなく，同じレベルで相互作用していることを示している [12]．

文献 [10] については，プログラム作成課題中にうまくいかなかった事例を示しているとも考えることができるが，いずれの研究でも課題解決に成功した事例と失敗した事例を比較することはしていない．本研究では，ペアプログラミングにおけるペア同士のやりとりを観察し，課題解決までのコミュニケーション活動について定量的に分析し，課題解決に成功した事例と失敗した事例を比較して，課題解決がうまくいかない事例の特徴について言及している．

2.3 ペアプログラミングにおける会話の役割

Wray [4] はペアプログラミングにおける会話の役割や会話がもたらす効果を実体験から述べている．文献 [4] によると，ペアプログラミングにおける会話の役割は，ペア間

© 2012 Information Processing Society of Japan

で専門的知識を共有することや，手詰まりになったときに会話をすることによって，それを解決する手がかりを得ることと述べている．また，プログラムについて会話をするプログラマほどより生産的になり，また随時質問を相互に投げかけることは，何よりも生産的になると予測している．

これはつまずきが発生したときに会話をすることによって，つまずきを解決できる可能性があることを示している．本研究では，ペアプログラミングにおける成功事例と失敗事例を比較するが，会話がそれらを比較する指標になると考え，作業中の会話に着目して比較する．

3. ペアプログラミング演習

3.1 演習設定

本研究では，大学の情報系学部 1 年生を対象に開講された「プログラミング I」の講義内においてペアプログラミングを実施した．この講義では C 言語を題材とする教育が行われた．学習目標は次の 3 つである．

- ソフトウェアの記述・構成とプログラミングの仕組みが分かる
- C プログラミングのコンパイルと実行ができる
- C 言語の基礎（データ型，配列，制御構造，関数，文字列，ファイル）が分かる

講義は 2010 年 9 月から 1 回 75 分で計 10 回行われ，ペアプログラミングは講義時間の一部を使って実施した．演習は 6 回行われた．

演習を行うための準備として，演習参加者の C 言語プログラミング経験を問うアンケートを実施した．プログラミング I の学習内容程度を扱うことができると回答した者はプログラミング経験者とした．また，学習内容のすべては扱えないと回答した者はプログラミング初学者とした．初めてペアプログラミングを行う学生がいることを考慮して，本演習の前にペアプログラミングの練習を行った．

本演習では，それまでに学習した内容を題材としたプログラムを作成する課題を与え 30 分程度で完了できるようにした．課題例として**図 1** に演習第 1 回の課題を示す．各回の演習について，実験者から演習参加者に対して次の指示を出した．

- このペアプログラミングでキーボードやマウスを操作できるのはコンピュータの手前に座っている Driver のみです．Navigator はキーボードやマウスに触れないようにして Driver をサポートしてください．触れなければどのようにサポートしてもかまいません．ディスプレイへ指差ししてもかまいません．
- プログラムの実行結果を確認し，課題の答えがでたら課題終了です．課題はできるだけ早く終了させてください．
- Driver，Navigator とも教科書 [13] は閲覧してもかまいません．Web ページの閲覧はしないでください．
- 教員および TA（Teaching Assistant）は，演習中は課題に対する質問は受け付けません．機器の不具合や操作方法に関する質問があるときのみ呼んでください．
- コメント文を使い，人間が見て見やすいものを作るように心がけてください．
- 制限時間は開始の合図から 30 分間です．時間になったら課題が途中の状態でも，できたところまでのコードを提出してください．

課題番号 1　整数型変数 n と r に $n \geq r > 0$ となる整数値を入力する．このとき，異なる n 個のものから r 個をとって 1 列に並べる順列の数を表わす　$_nPr$　と，異なる n 個のものから r 個をとる組み合わせの数を表わす　$_nCr$　を計算するプログラムを作成せよ．例えば n に 8，r に 3 を入力したとき，

8P3 は 336 で 8C3 は 56 です

などと出力できればよい．$_nPr$ と $_nCr$ はそれぞれ，

$$_n\mathrm{Pr} = \frac{n!}{(n-r)!} \qquad _nCr = \frac{n!}{r!(n-r)!} = \frac{_n\mathrm{Pr}}{r!} \qquad \text{ただし、} n! = \begin{cases} n \times (n-1)! & (n \geq 2) \\ 1 & (n = 1) \end{cases}$$

とする．

図 1　ペアプログラミング演習で用いた課題の例

Fig. 1　An example of the exercises in the pair programming class.

演習参加者は 62 名（学部 1 年生 52 名，2 年生以上 10 名）であった．コミュニケーションのしやすさ，ペアプログラミングのやりやすさ，プログラミング経験差が発生することを考慮して，ペアは実験者が決定し，すべての回を通して同じペアを組ませることはしなかった．なお，参加者には誰がプログラミング経験者，プログラミング初学者なのかということは伝えていない．ペアプログラミングの定義 [3] に従うならば，1 回の演習において，Driver と Navigator の役割を交替させる必要があるが，制限時間が 30 分と短いことから，1 回の演習では，Driver と Navigator のどちらか一方の役割を担当させることとした．演習は全 6 回であるが，すべての参加者について，Driver と Navigator の役割を 3 回ずつ経験できるようにペアを組んだ．ペアの組合せに優劣がでる可能性を考慮して，プログラムの良し悪しや課題完了までの時間等について優劣をつけることはせず，課題解決の途中であってもプログラムを提出した場合に一律の出席点を与えることを参加者に伝えた．演習環境について，演習参加者はプログラムを作成するエディタとして emacs を利用している．

図 2 は演習風景，**図 3** は設置したカメラで撮影したデータの一場面である．撮影は 1 組のペアに対して 3 台のカメラを使用し，演習の妨げにならないようにカメラを設置した．撮影は課題の開始から終了まで行い，ペア同士のやりとりを記録した．

3.2 データ収集

3.1 節における設定のもと，1 回の演習において 4 ペアの撮影を行い，6 回の演習でのべ 24 ペア分の会話を収集した．1 組のペアに対して 3 台のカメラを用いて撮影したが，その 3 台のカメラすべてで映像が鮮明かつ演習開始から演

© 2012 Information Processing Society of Japan

図 2 データ取得用カメラの設置位置
Fig. 2 Setup of the cameras for data collection.

図 3 ペアプログラミング演習で撮影した映像データの一場面
Fig. 3 Scenes from the recorded data of the pair programming class.

習終了まで撮影できている 10 ペアを対象として，つまずきの解決に成功した事例および失敗した事例を抽出した．

本研究におけるつまずきとは，「コンパイル時にエラーが発生した（コンパイルエラー）あるいはコンパイル時にエラーが発生しなくても実行時にペアが意図した出力がなされなかった（実行時エラー）こと」と定義する．つまずきの解決に成功した事例（成功事例）とは，「コンパイルエラーに関するつまずきが発生した後にその後のコンパイルで当該エラーが解決された事例，実行時エラーに関するつまずきが発生した後にその後の実行で当該エラーが解決された事例」と定義する．つまずきの解決に失敗した事例（失敗事例）とは「つまずき発生した後に，演習の制限時間が来たために解決まで至らなかった事例」と定義する．後述の事例では，成功事例，失敗事例にかかわらず事例中に複数回コンパイルエラーや実行時エラーが発生している場合もあり，これらのエラーが発生した時点では解決に失敗していると考えることもできる．しかし，このコンパイルやプログラムを実行した理由は，プログラムを少し修正してエラーが解決したかを確認したり，プログラムの途中に printf 文を挿入して途中の変数値を確認したりするためであり，このようなものは解決に失敗したのではなく，解決の途中段階として解釈した．

この定義に従って成功事例および失敗事例を抽出したところ，10 ペア全体で成功事例は 19 件（コンパイルエラー 12 件，実行時エラー 7 件），失敗事例は 3 件（コンパイルエラー 1 件，実行時エラー 2 件）あった．この 22 件について，分析結果の信頼性を損なわないようにするため，「つまずき発生後，ペア間で会話がないまま Driver が解決した（成功事例 7 件）」，「制限時間を超えてからエラーが発生した（成功事例 4 件）」，「Navigator がキーボード等を操作した（成功事例 1 件）」，「設置したカメラの話等の実験環境に関する話が会話の話題になる場面があった（成功事例 1 件）」事例を分析対象から除外した．以上の結果，成功事例 19 件のうち 6 件，失敗事例 3 件の計 9 件を対象とし，これらについて分析を行った．

表 1 ペアプログラミング演習中に発生したつまずきの事例
Table 1 Success and failure cases observed in the pair programming class.

事例	ペア	Driver	Navi.	つまずき
成功 A	ペア A	初学者	経験者	コンパイルエラー 行末にセミコロンをつけ忘れていた．
成功 B	ペア B	初学者	経験者	コンパイルエラー 「enum」という文字列が予約語であった．
成功 C	ペア B	初学者	経験者	コンパイルエラー ソースファイルを上書き保存していなかった．
成功 D	ペア B	初学者	経験者	実行時エラー（セグメンテーションエラー） scanf 文で「scanf(“%d”,n);」と書いていた． &を忘れていたことが原因
成功 E	ペア C	初学者	初学者	コンパイルエラー ブロックの開始と終了が対応できていなかった．（このペアはテキストモードで編集しており，ブロックチェック機能が働いていなかった） プログラム内にスペリングミスがあった．
成功 F	ペア C	初学者	初学者	実行時エラー コンパイルはできるが正しい答えが得られなかった． 関数の返り値が正しく返されていなかった．
失敗 A	ペア D	経験者	経験者	実行時エラー コンパイルはできるが正しい答えが得られなかった． （繰り返し文内において，零で除算する場合があった）
失敗 B	ペア A	初学者	経験者	実行時エラー コンパイルはできるが正しい答えが得られなかった． （グローバル変数宣言により変数値が不正だった）
失敗 C	ペア E	初学者	初学者	コンパイルエラー メインファイルとヘッダファイルがリンクしなかった．

※初学者：プログラミング初学者，経験者：プログラミング経験者　※ペアはすべて 1 年生同士

表 1 に分析対象とする事例を示す．表 1 では成功事例 A〜F の 6 例，失敗事例 A〜C の 3 例をあげている．つまずきは作業中にいくつか発生するため，1 組のペアに対して複数の事例がある場合がある．表 1 にあげられている事例は，最終的に当該ペアが認識したエラーの原因であり，つまずきが発生してからエラーの原因を認識するまでに多少の時間を費やしている．失敗事例 A と B について，つまずきの事例の括弧内は実験者が撮影したビデオ映像を観察することで分かったエラー原因であり，Driver および Navigator はそのエラーに気付いていなかった．

なお，本演習では失敗事例が 3 例しかなかった．これは本演習で演習参加者に与えたプログラム作成課題が，それまでに学習した内容を題材としたもので，30 分程度で完了できるものであることによるためだと考えられる．撮影したほとんどのペアは制限時間内に課題を完了させていた．

4. 発話分析

4.1 データ処理

撮影した映像を用い，ビデオ分析ツール iCorpusStudio [14] を使用して，ペアごとに Driver と Navigator の発

© 2012 Information Processing Society of Japan

表 2 ペアプログラミング演習で収集したコミュニケーションデータ

Table 2 Conversations during working with the exercises.

事例	データ長（秒）	Driver の発話数	Navigator の発話数	発話時間（%）	発話頻度（発話回数/分）	発話長（秒/発話回数）	説明の繰り返し（%）	連続発話率（%）
成功 A	100	10	11	16.2	6.3	1.55	0	47.6
成功 B	151	24	27	31.2	10.1	1.85	0	31.4
成功 C	50	5	6	25.9	6.6	2.35	0	18.2
成功 D	125	14	15	21.3	7.0	1.84	0	17.2
成功 E	121	12	8	14.4	5.0	1.74	5.0	40.0
成功 F	417	35	23	12.6	4.1	1.81	6.9	32.8
失敗 A	624	82	53	26.5	6.5	2.45	5.2	28.9
失敗 B	588	19	61	12.9	4.1	1.89	8.8	62.5
失敗 C	460	63	53	33.8	7.6	2.68	7.8	29.3

話内容を記録した．iCorpusStudio は，同時に複数映像を見ながらタイムラインに対して複数種類のラベリングをし，ラベルのカテゴリごとに CSV 形式のファイルに出力できる．**表 2** に取得したコミュニケーションデータを示す．成功事例のデータ長は表 1 で示した各つまずきが発生した時点からそれが解決されるまでの時間である．失敗事例のデータ長は，つまずきが発生した時点から参加者が解決を止めた時点（制限時間が来たために途中のコードを提出し始めようとした時点）までの時間の長さである．発話数は事例中に発声した回数である．本研究での発話とは，時間の長短，意味の有無によらず，1 回の発声を指す．つまり，「あー」，「うん」等の短い相槌もすべて発話として数えている．説明の繰返し，連続発話率，発話頻度，発話長，発話時間については次節で説明する．

4.2 分析方法

つまずきの解決の成功事例と失敗事例における発話の様子を調べるために，次の点に着目して分析した．

(1) 発話量

本研究では発話量について，発話時間，発話頻度，発話長を分析した．発話時間とはデータ長に対してどの程度発話が生じていたかを割合で示したものである．発話頻度とは 1 分あたりの発話数である．発話長とは 1 回の発話の開始から終了までの平均時間である．表 2 で示されている発話時間，発話頻度，発話長は Driver および Navigator それぞれで算出して平均化し，1 人あたりの値を示している．

(2) 説明の繰返し

一般に，聞き手が誤解しているときに話し手はより強い声の調子や強調したジェスチャで話を繰り返す，あるいは，話し手が曖昧なことをいったときに聞き手が聞き返しや困った表情をすることによって不理解を合図し明確化を求めるような状況がある．このような状況下では，話し手の 1 回の発話で聞き手がその内容を理解できずに話し手が説明を繰り返すことになる．1 回の発話で聞き手が理解できない場合はスムーズなやりとりができていないと考えられる．説明の繰返しとは，このように同じ内容を繰り返して発話することと定義する．

表 3 失敗事例 B で行われた会話の一部

Table 3 Part of the conversation in Failure case B.

発話番号	話者	発話開始	発話終了	発話内容
14	N	1345.0	1346.2	while 文を
15	D	1346.3	1346.6	うん
16	N	1348.4	1350.0	main の外に出そう
17	D	1351.0	1352.1	main の
18	N	1356.1	1357.1	上のほうでいいや
19	D	1357.7	1358.3	うん
20	N	1360.0	1361.7	うん，そこまでの間
21	D	1362.3	1363.5	うん
22	N	1364.2	1366.7	while 文で，切り取りで
23	N	1369.7	1370.4	でー
24	N	1371.4	1372.6	えーっと
25	N	1373.4	1374.6	while 文（Driver が操作）
26	N	1378.0	1380.8	あっ，while 文じゃない，あ，ごめん，元に戻して
27	N	1384.9	1387.0	ごめん，おかしくなった
28	N	1387.5	1391.8	この関数は外に出してもいいから（Driver が操作）
29	N	1393.0	1396.4	あー，もう以下全部，中括弧閉じるまで（Driver が操作）
30	N	1407.9	1413.5	でも出したから，これで宣言，漏れたらあぶない
31	N	1416.0	1420.1	jyun（変数名）は大丈夫なんだよ，ans も大丈夫，n は
32	N	1422.0	1423.2	ああ，n は
33	N	1431.7	1433.6	関数の外で宣言していいんだっけ
34	N	1437.9	1440.9	グローバルなんちゃらってある
35	N	1442.3	1443.1	索引で
36	N	1446.0	1450.5	グローバル，グローバル変数
37	N	1456.0	1458.0	あっ，ブロックの外いけるや

※話者：D は Driver，N は Navigator，発話開始と発話終了は発話開始・終了時間（単位：秒）

分析では，次の 3 つの状況が起きた回数を事例ごとに数えることとした．

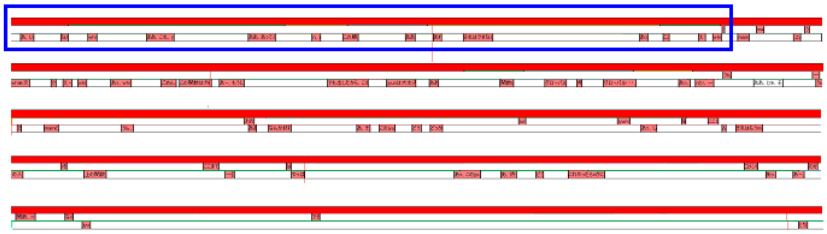
- 話し手が発話した後に，聞き手が誤解していると話し手が感じ取り，話し手が説明を繰り返した．
- 話し手が発話した後に，聞き手が何も反応しない（発話しない，何も操作しない）ときに，話し手が説明を繰り返した．
- 話し手が発話した後に，聞き手が聞き返しをして，話し手が説明を繰り返した．

説明の繰返しの例として，**表 3** に失敗事例 B で行われた会話の一部を示す．ここでは while 文を別の行に移動させようとしている．この会話の発話番号 28 において Navigator が「この関数は外に出してもいいから」と指示を出し，Driver が操作をするが，それを見た Navigator は Driver が誤解していると気付き「あー，もう以下全部，中括弧閉じるまで」と Navigator がやりたいことについて繰り返し説明している．これは話し手が発話した後に，聞き手が誤解していると話し手が感じ取り，話し手が説明を繰り返したという状況にあたる．このような状況が起きた回数を数えている．回数を数えたあと，発話数に対する説明の繰返しの比率を算出した．

(3) ペアの両者が交互に発話できているか

話し手と聞き手が交互に発話を行うことで，会話はスムーズに行われていると考えることができる．たとえば，話し手の説明に対して聞き手が相槌等の応答をすることにより，話し手は伝えたいことが聞き手に伝わったことを確認でき，聞き手は自身が話し手の発話内容を理解したと話し手に伝えることができる．交互に発話できていないと，

© 2012 Information Processing Society of Japan



※1 つのタイムラインは 3 行で 1 セット．上から 2 段目は Driver の発話内容と発話時間，上から 3 段目は Navigator の発話内容と発話時間が記録されている

図 4 失敗事例 B で見られた会話
Fig. 4 Conversational sequence in Failure case B.

一方的に話し手から聞き手に発話することになり，話し手は聞き手が理解できているか分からず，聞き手も話し手の発話内容を理解できていない可能性が高く，スムーズなやりとりができていないと考えられる．

分析では，各事例に対して Driver あるいは Navigator が連続的に発話した回数を数え，その割合を算出することとした．これを算出することにより，両者がどのくらい連続して発話している，つまりどのくらい交互に発話できていないかが分かる．

交互に発話できていないことの例として，**図 4** に失敗事例 B のタイムラインを示す．ここでは図 4 上部の四角で囲まれた部分において，Navigator が 14 回，Driver が 1 回発話している．ここで Navigator は最初の発話に続いて 13 回連続して Navigator が発話していることが分かる．このようにして Driver あるいは Navigator の連続発話回数を数え，次の式で連続発話率を計算した．

$$\text{連続発話率} = \frac{\text{連続発話回数}}{\text{発話数} - 1} \times 100(\%)$$

分母が（発話数 − 1）である理由について，前述の例を使うと，発話数 14 に対して 13 回連続で発話していることを 100%連続発話したとするためである．

5. 分析結果

本章では 4 章で示したデータおよび分析方法に従って，成功事例と失敗事例を比較した結果を示す．統計処理は SPSS Statistic 17.0 を用いて行い，後述の Mann-Whitney の U 検定は両側検定を行った．

5.1 発話量

(1) 発話時間

図 5 は両事例全体でどの程度発話が生じていたかを割合で示したものである．以下エラーバーは標準偏差を表す．成功事例の平均発話率は 20.4%，失敗事例では 24.4%であった．Mann-Whitney の U 検定で両事例の発話時間を比較したところ，有意水準 5%のもとで有意な差があるとは認められなかった（$U = 6$，$p = 0.439$）．よって，成功，失敗によって発話時間に差があるとはいえない．

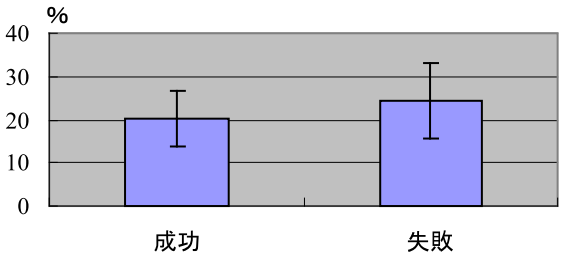


図 5 1 人あたりの発話時間
Fig. 5 Speech time.

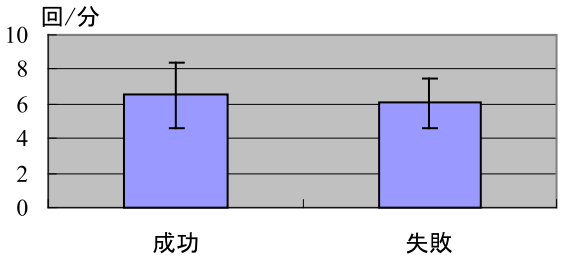


図 6 1 人あたりの発話頻度
Fig. 6 Speech frequency.

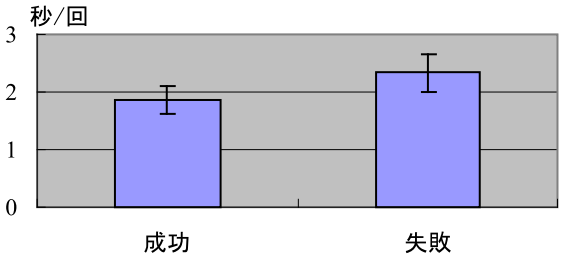


図 7 1 人あたりの発話長
Fig. 7 Speech length.

(2) 発話頻度

図 6 は両事例における発話頻度を示している．成功事例の平均発話頻度は 6.51 回，失敗事例では 6.06 回であった．Mann-Whitney の U 検定で両事例の発話頻度を比較したところ，有意水準 5%のもとで有意な差があるとは認められなかった（$U = 8.5$，$p = 0.897$）．よって，成功，失敗によって発話頻度に差があるとはいえない．

(3) 発話長

図 7 は両事例における発話長を示している．成功事例

© 2012 Information Processing Society of Japan

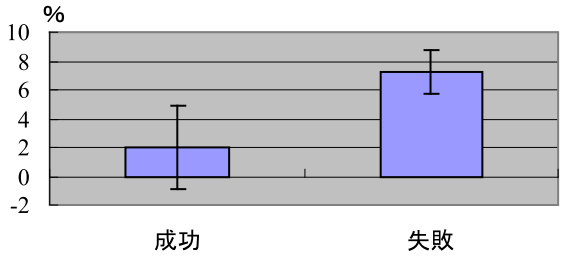


図 8 説明を繰り返した回数（発話数に対する比率）
Fig. 8 Rates of repeated explanation.

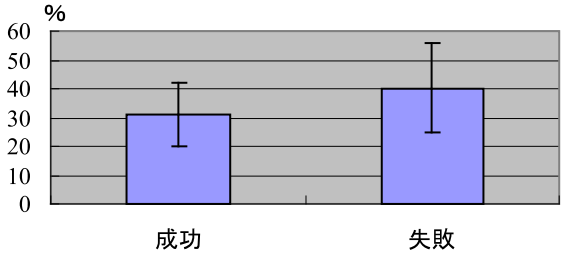


図 9 連続発話率
Fig. 9 Rates of consecutive speech.

の平均発話長は 1.85 秒，失敗事例では 2.34 秒であった．Mann-Whitney の U 検定で両事例の発話長を比較したところ，有意水準 5%のもとで有意な差が認められた（$U = 1$, $p = 0.039$）．これにより失敗事例の発話長が成功事例よりも長いことが分かった．

5.2 説明の繰返し

図 8 は両事例における説明の繰返し比率を示している．成功事例の平均比率は 1.98%，失敗事例は 7.26%であった．Mann-Whitney の U 検定で両事例の説明の繰返し比率を比較したところ，有意水準 5%のもとで有意な差が認められた（$U = 1$，$p = 0.031$）．失敗事例の説明の繰返し比率は成功事例よりが大きいことが分かった．

5.3 ペアの両者が交互に発話できているか

図 9 は両事例における連続発話率を示している．成功事例の平均連続発話率は 31.2%，失敗事例では 40.2%であった．Mann-Whitney の U 検定で両事例の連続発話率を比較したところ，有意水準 5%のもとで有意な差があるとは認められなかった（$U = 8$，$p = 0.796$）．よって，成功，失敗によって連続発話率に差があるとはいえない．

6. 考察

本研究では，ペアプログラミング学習におけるペア同士のコミュニケーションデータから，Driver と Navigator の会話を定量的に分析することで，作業中に発生するつまずきの解決に成功した事例と失敗した事例の違いを示している．本章では，5 章で得られた結果について検討を行い，協調作業支援に与える示唆を論じる．

6.1 作業中のペア同士のやりとり

説明を繰り返した回数，発話長について有意差が認められ，失敗事例のほうがそれらが大きいことが認められた．連続発話率について有意差は認められなかったものの，失敗事例のほうが成功事例に比べて約 9%連続発話率が高いことが分かった．

このことから，つまずきを解決できないことが発話長等に現れていることが分かった．表 3 の事例ではそれが顕著に現れている．発話番号 14 から 22 までは Driver と Navigator がほぼ交互に発話をしており，両者のやりとりはできていると考えられる．しかし，発話番号 22 から発話番号 37 まで Navigator が連続して発話しており，かつ 1 回の発話長が 2 秒以上ある発話数は 10 と多いと考えられる．4.2 節で説明したように説明の繰返しも行われている．この間 Driver は Navigator の指示どおりにコンピュータを操作していた．しかし，発話番号 29 に対して Driver がコードを修正した後，しばらく Driver は発話を行わず，じっと画面を見ているだけであった．発話番号 29 までは，Driver は Navigator が伝えたいことを理解していたと考えられるが，その後 Navigator が連続して発話長の長い発話をしてしまったために，Driver が混乱をおこしてしまい，Navigator に対して何もいえなくなったと考えられる．そこにグローバル変数という新たな話題が提示されるが，Driver はそれまでの Navigator からの発話内容を消化できずに新たな話題に移行してしまい，さらに混乱が起きてしまったと考えられる．この後，演習の制限時間までグローバル変数に対応しているが，Driver は Navigator がいったとおりに操作することに終始しており，つまずきの解決には至らなかった．

このように発話長の長さ，説明の繰返し，連続発話をきっかけとして，Driver と Navigator のやりとりがうまくいかず，聞き手が話し手の内容をきちんと理解できなかった，あるいは混乱してしまった可能性がある．その結果，失敗事例ではつまずきの解決まで至らなかったのかもしれない．しかし，表 3 の例では，これらの 3 要素が連鎖的に発現してしまったために，両者のやりとりがうまくいかなかった可能性がある．それぞれが単独で発現した場合にどのようなやりとりが発生するかについては，今後データ数を増やす等をして検討していく必要がある．

一般に，個人の能力や授業内容の理解度には差があると考えられ，たとえばプログラミング経験を指標として課題の成功と失敗を判断しようとしても，それが難しいであろうことは，表 1 の事例においてプログラミング経験によらず成功や失敗がみられることから分かる．本研究では，成功事例と失敗事例を比較する指標として，ペアの会話に着目したところ，成功と失敗の差が会話に表れていることが分かった．

© 2012 Information Processing Society of Japan

6.2 本研究の限界

本演習では，講義の一部の時間を使ってペアプログラミング演習を行ったために，演習時間が毎回 30 分となった．Beck らが提唱したペアプログラミングの定義に従うならば Driver と Navigator の役割を交換する必要があったが，時間の関係で各回の演習において，どちらか一方の役割を担当するということにした．当然ながら，演習中に役割を交換することで，会話内容に変化が見られ，つまずきの解決がさらにスムーズに進む可能性も考えられる．しかし，この制約は演習参加者に等しく与えられており，本研究ではその制約の中で，つまずきの解決に成功した事例と失敗した事例を比較している．

本演習で出したプログラム作成課題は，それまで学習した内容で，30 分程度で完了できるものであることから，難易度は低めに設定されていた．これは課題を完了できないと，学習に対するモチベーションが下がってしまうことを懸念したためであるが，そのために，ほとんどのペアが課題を完了させており，失敗事例が少なかった．課題の難易度を上げることで，本演習よりもつまずきが多く発生すると考えられ，その結果多くの失敗事例データを得ることができると考えられる．これについては，今後さらに検討していく必要がある．

6.3 協調作業の状態推定

本研究の結果から，たとえば，ペアプログラミング学習において，ある話し手の発話長が平均よりも長く，話し手が一方的に発話していると，ペアプログラミングがうまくいっていないのではないかと推定することができるであろう．さらに，ペアプログラミングは協調作業であることから，より一般に協調作業の状態推定にも有効である可能性はある．

本研究における演習では，同学年同士のペアかつ講義開始が 9 月であったことから，ペア同士のコミュニケーションは比較的容易に行うことができたと考えられる．つまり，会話のやりとりがうまくいかず，聞き手が話し手の発話内容が理解できなくても容易に聞き返すことができた，あるいは，話し手が容易に聞き手に何らかのサポートをすることができただろう．このために失敗事例が少なかったのかもしれない．しかし，仮に同学年同士ではない，あるいはペアが初対面であった場合には容易にコミュニケーションをとることはできない．本研究ではペアプログラミングにおいて，会話に関するいくつかの指標が作業の状態と関係があることが分かったが，これらの指標がより一般的な協調作業と関係があるということが分かれば，会話を作業状態の指標として使うことができるであろう．

このような可能性は，今後データ数を増やす等をして，さらに検証をしていく必要があるが，本研究の結果は協調作業をどのように支援していけばよいかを検討する 1 つの手がかりとなるであろう．

7. おわりに

本研究では，プログラミング導入教育で実施したペアプログラミング学習において，作業中に発生したつまずきの解決に成功した事例と失敗した事例をペア同士の会話の観点から定量的に分析比較した．

その結果，つまずきの解決に失敗した事例について，成功した事例と比較したときに，(1) 1 回の発話長が長い，(2) 説明を繰り返す回数が多い，(3) 一方的に発話する回数が多いことが分かった．

今回得られた知見に基づいて，より一般的な協調作業の状態を推定することが可能になれば，これまで以上にペアプログラミング学習や協調作業の効果が得られるようになるであろう．

謝辞 本研究のデータ収集に協力いただいた鄭乃文氏に感謝いたします．本研究の一部は，科学研究費補助金特別研究員奨励費（23・2956），筑波大学大学院図書館情報メディア研究科プロジェクト研究費および科学研究費補助金基盤研究（C）22500104 による．

参考文献

[1] 新開純子，宮地 功：プロセスの重視と評価活動を取り入れた C プログラミング教育の効果，教育システム情報学会誌，Vol.26, No.1, pp.16–25 (2009).
[2] 大西建輔：プログラミング，何をどう教えているか，連載開始にあたって，情報処理，Vol.51, No.10, p.1341 (2010).
[3] Beck, K.: *Extreme Programming Explained: Embrace Change, Reading, PA*, Addison-Wesley (1999).
[4] Wray, S.: How Pair Programming Really Works, *IEEE Software*, Jan./Feb. 2010, pp.50–55 (2010).
[5] McDowell, C., Werner, L., Bullock, H. and Fernald, J.: The Effects of Pair-Programming on Performance in an Introductory Programming Course, *Proc. ACM SIGCSE*, pp.38–42 (2002).
[6] Nagappan, N., Williams, L., Ferzli, M., Wieve, E., Yang, K., Miller, C. and Balik, S.: Improving the CS1 Experience with Pair Programming, *Proc. ACM SIGCSE*, pp.359–362 (2003).
[7] Hanks, B., McDewell, C., Draper, D. and Krnjajic, M.: Program Quality with Pair Programming in CS1, *Proc. Innovation and Technology in Computer Science Education* (*ITiCSE*), pp.176–180 (2004).
[8] Rountree, J., Rountree, N., Robins, A. and Hannah, R.: Observations of Student Competency in a CS1 Course, *Proc. Australasian Computing Education Conference*, pp.145–149 (2005).
[9] Urness, T.: Assessment Using Peer Evaluations, Random Pair Assignment, and Collaborative Programming in CS1, *Proc. Consortium for Computing Sciences in Colleges*, pp.87–93 (2009).
[10] Chen, W. and Nordbo, M.: Understanding Pair-Programming from a Socio-cultural Perspective, *Proc. Computer Supported Collaborative Learning* (*CSCL*), pp.138–140 (2007).
[11] Chong, J. and Hurlbutt, T.: The Social Dynamics of Pair

© 2012 Information Processing Society of Japan

Programming, *Proc. 29th International Conference on Software Engineering* (*ICSE*), pp.354–363 (2007).
[12] Bryant, S., Romero, P. and Boulay, B.: Pair Programming and the Mysterious Role of the Navigator, *International Journal of Human-Computing Study*, Vol.66, No.7, pp.519–529 (2008).
[13] 高橋麻奈：やさしい C 第 3 版，ソフトバンク (2007).
[14] iCorpusStudio, available from ⟨http://www.ii.ist.i.kyoto-u.ac.jp/iCorpusStudio/⟩ (accessed 2011-04-07).

平井 佑樹 （学生会員）

2007 年東京学芸大学教育学部情報教育課程情報教育専攻卒業，2009 年同大学大学院教育学研究科総合教育開発専攻修了，修士（教育学）．現在，筑波大学大学院図書館情報メディア研究科博士後期課程在学中．2011 年より日本学術振興会特別研究員（DC2）．協調学習支援，ヒューマンインタラクションに関する研究に従事．本会コンピュータと教育（CE）研究会およびグループウェアとネットワークサービス（GN）研究会各会員．人工知能学会会員．

井上 智雄 （正会員）

筑波大学図書館情報メディア系准教授．1998 年慶應義塾大学大学院理工学研究科計測工学専攻博士課程修了，博士（工学）．ヒューマンインタラクション，CSCW，協調活動支援システムの研究に従事．本会論文賞，同山下記念研究賞，ほか受賞．電子情報通信学会ヒューマンコミュニケーション基礎研究会，同食メディア研究会，日本 VR 学会サイバースペースと仮想都市研究会，同香りと生体情報研究会，IEEE TC CSCWD，IEEE TC HCI 各運営委員．『Communication and Collaboration Support Systems』（IOS Press）等執筆．ヒューマンインタフェース学会，電子情報通信学会，ACM，IEEE ほか各会員．

© 2012 Information Processing Society of Japan