Worked Out Examples in Computer Science Tutoring

Barbara Di Eugenio[1], Lin Chen[1], Nick Green[1], Davide Fossati[2], and Omar AlZoubi[2]

[1] Computer Science, University of Illinois at Chicago
Chicago, IL, USA
bdieugen/lchen43/ngreen21@uic.edu
[2] Computer Science, Carnegie Mellon University in Qatar
Doha, Qatar
dfossati/oalzoubi@cmu.edu

Abstract. We annotated and analyzed Worked Out Examples (WOEs) in a corpus of tutoring dialogues on Computer Science data structures. We found that some dialogue moves that occur within WOEs, or sequences thereof, correlate with learning. Features of WOEs such as length also correlate with learning for some data structures. These results will be used to augment the tutorial tactics available to iList, an ITS that helps student learn linked lists.

Keywords: Tutoring dialogues, Tutoring strategies, Intelligent tutoring

1 Introduction

Worked out examples (WOEs) demonstrate a step by step solution of a problem for the learner to study. Learning from WOEs has been studied in cognitive research [1, 2], including in the context of Intelligent Tutoring Systems (ITSs) [3, 4]. However, the conditions that trigger WOEs and how tutors structure WOEs have not been extensively investigated. Our domain of interest is introductory data structures in Computer Science (CS). Interestingly, one of the first papers on WOEs [7] also concerns learning in CS, specifically recursion in LISP programming. Within CS, [5, 6] have employed WOEs for classroom instruction.

Our interest in exploring WOEs is two-fold. We believe that in order to deploy WOEs in an ITS, it is essential to uncover the conditions under which WOEs are effective. Additionally, in our previous work, we showed that certain Dialogue Moves (DMs) on the part of the tutor, or sequences thereof, correlate with learning gains [8]. Many of those findings have been implemented in the iList system, that helps students learn linked lists [9, 10]. Still, the tutor interventions we deployed are not conditioned on the larger tutoring strategies the tutor uses. WOEs can provide one type of context to structure those tutor moves.

2 WOEs, their features and learning

Our corpus consists of 54 tutoring sessions with two human tutors on linked lists, stacks, and binary search trees. It had been previously annotated with Student Initiative (SI), and with 5 tutor moves: prompts (PT); positive and negative feedback (PF, NF); Direct Procedural Instruction (DPI) – the tutor provides insight into steps to solve the problem; Direct Declarative Instruction (DDI) – the tutor states facts about the problem [8]. The annotation of WOEs was superimposed on these preexisting annotations. Two coders marked beginning and end of WOEs.[3] We obtained excellent intercoder agreement (κ = .82) on 7 sessions that were double annotated. Each coder then annotated half of the remaining sessions. Fig. 1 shows a WOE excerpt from our corpus starting at TUT2 (it continues beyond TUT6, and it has been modified for space reasons). Fig. 1 also shows the moves each utterance is labelled with.

```
DDI              TUT1   Now a binary search tree must remain ordered.
DPI, WOE-START   TUT2   say we want to insert, um, six.
SI               ST     down there? [pointing to tree drawing]
PF               TUT4   right
SI               ST     five is smaller than six
DDI              TUT5   and the right child of five is null
DPI              TUT6   so we will insert six to its right
```

Fig. 1. A worked out example to insert a node into a binary search tree

Table 1 shows distributional statistics about WOEs, per topic: how many sessions (tutors were free to skip topics), and total number of WOEs; average number of WOEs, average lengths of WOEs in words and in utterances (standard deviations in parenthesis). Tutors use many more WOEs for lists and trees than for stacks; more frequent WOEs for trees are offset by longer WOEs for lists.

Topic	N	Total WOEs	Avg. WOEs	Avg. Words/WOE	Avg. Utts./WOE
Lists	52	180	3.5 (1.4)	498.3 (438)	48.3 (42.7)
Stacks	46	24	0.5 (0.5)	615.5 (115.6)	68.5 (17.1)
Trees	53	454	8.6 (2.7)	212.5 (223)	24.0 (24.5)

Table 1. Worked Out Examples Statistics

As in our previous work, we adopt a multiple regression approach, because it shows how much variation in learning gains is explained by the variation of features in the data. We previously included pre-test score, the length of the

[3] Coders also marked nested WOEs, but since only 21 nested WOEs exist out of 658 total, we will not discuss them further.

tutoring sessions, the DMs we annotated for, and DM *bigrams* and *trigrams*, i.e. DM sequences of length 2 or 3. In our best regression models (R^2=.415 for lists, R^2=.416 for stacks, and R^2=.732 for trees), significant features are pre-test score and trigrams of specific DMs (negative correlations between previous knowledge and learning gains are common: models that only include pre-test score result in R^2=.200 for lists, R^2=.296 for stacks, and an astounding R^2=.676 for trees).

We now add WOEs and their features to the regression. Simply adding the number of WOEs per session does not correlate with learning gains, other than for stacks; however, this correlation is negative. Next, we explore models where we differentiate between DMs within and outside of WOEs. We ran every regression model that results from the systematic combination of pre-score, length of dialogue, number of WOEs, length of WOEs in words and utterances, and then, for each DM, how many occur outside, and how many inside, a WOE. As a result, we obtain better regression models, but only for lists and stacks (see Table 2). Even if some correlations are only marginally significant, together they throw further light on WOEs. For trees, the best previous model includes pre-test and the DM trigram [PF,SI,DDI]. Using **only** the occurrences of this trigram of DMs within WOEs (as in Fig. 1), we obtain a slightly improved $R^2 = .737$.

Topic	Predictor	β	R^2	P
Lists	Pre-test	-0.442	.485	$<.01$
	WOE_Prompt	$-.0006$		$= 0.073$
	WOE_#Utterances	.002		$= 0.092$
	PF	.005		$= 0.099$
Stacks	Pre-test	$-.37$	.606	$<.005$
	WOE_PF	0.077		$< .005$
	WOE_Prompt	$-.021$		$<.05$
Trees	Pre-test	$-.736$	.737	$<.0001$
	WOE_[PF,SI,DDI]	.037		$< .005$

Table 2. The most explanatory models include WOE features

From the models shown in Table 2, we can confirm that WOEs can be a successful tutorial strategy, but we need to look "under the hood". First, effective features of WOEs depend on the specific topic; e.g., longer WOEs are effective only for lists. Positive feedback (PF) within and outside WOEs is important: PFs within WOEs marginally correlate with learning gains for stacks, and robustly correlate with learning as part of the sequence [PF,SI,DDI] for trees; PFs outside of WOEs correlate with learning gains for lists (this confirms our previous results on positive feedback). Surprisingly, for lists and stacks, prompts within WOEs are **negatively** correlated with learning gains. This seems to suggest that during WOEs, where the tutor is demonstrating a solution, students should not be invited to participate in problem solving, which is otherwise well known as conducive to learning. It turns out that, on average, more prompts occur in WOEs for stacks (11.1), than for lists (7.7), than for trees (3.3). This may in part be due to the respective difficulty of these data structures, with stacks being easiest, next lists, and then trees. This may also explain the negative correlation between number of WOEs and learning gains, for stacks.

3 Future work

Our findings open various lines of inquiry for future work, such as, what the role of prompts within WOEs is. We also intend to analyze the internal structure of WOEs, and what may trigger a WOE. A preliminary analysis shows that DDIs are the most frequent DM that immediately precedes the start of a WOE (see TUT1 in Fig. 1) with 435 occurrences out of 658 (66%); in 113 cases (17%) the preceding DM is a DPI. This seems to suggest that most of the time the tutor sets the stage for a WOE with a DDI. We will integrate our findings within the probabilistic model that iList uses to generate its next move. This model is based on the "promise" of the current and previous student steps [10].

Acknowledgments. This work is supported by award NPRP 5-939-1-155 from the Qatar National Research Fund.

References

1. Sweller, J.: The worked example effect and human cognition. Learning and Instruction **16**(2) (2006) 165–169
2. Atkinson, R.K., Derry, S.J., Renkl, A., Wortham, D.: Learning from examples: Instructional principles from the worked examples research. Review of Educational Research **70**(2) (2000) 181–214
3. Renkl, A., Atkinson, R., Maier, U., Staley, R.: From example study to problem solving: Smooth transitions help learning. Journal of Experimental Education **70** (2002) 293–315
4. Ringenberg, M., VanLehn, K.: Scaffolding problem solving with annotated, worked-out examples to promote deep learning. In: ITS 2006, the 8th International Conference on Intelligent Tutoring Systems. (2006) 625–634
5. Moura, I.C.: Worked-out examples in a computer science introductory module. In: Proceedings of the World Congress on Engineering, Vol II. (2012)
6. Luukkainen, M., Vihavainen, A., Vikberg, T.: A software craftsman's approach to data structures. In: Proceedings of the 43rd ACM Technical Symposium on Computer Science Education. SIGCSE '12 (2012) 439–444
7. Pirolli, P., Anderson, J.R.: The role of learning from examples in the acquisition of recursive programming skills. Canadian Journal of Psychology **39**(2) (1985)
8. Chen, L., Di Eugenio, B., Fossati, D., Ohlsson, S., Cosejo, D.: Exploring Effective Dialogue Act Sequences in One-on-one Computer Science Tutoring Dialogues. In: BEA6, The 6th Workshop on Innovative Use of NLP for Building Educational Applications. (2011)
9. Fossati, D., Di Eugenio, B., Brown, C., Ohlsson, S., Cosejo, D., Chen, L.: Supporting Computer Science curriculum: Exploring and learning linked lists with iList. IEEE Transactions on Learning Technologies **2**(2) (2009) 107–120
10. Fossati, D., Di Eugenio, B., Ohlsson, S., Brown, C., Chen, L.: Generating proactive feedback to help students stay on track. In: ITS 2010, 10th International Conference on Intelligent Tutoring Systems. (2010)