Camels and humps: a retraction*

Richard Bornat
School of Science and Technology, Middlesex University, London, UK
R.Bornat@mdx.ac.uk

July 24, 2014

Abstract

In 2006 I wrote an intemperate description of the results of an experiment carried out by Saeed Dehnadi. Many of the extravagant claims I made were insupportable, and I retract them. I continue to believe, however, that Dehnadi had uncovered the first evidence of an important phenomenon in programming learners. Later research seems to confirm that belief.

1 Background

University computer-science departments mostly feel that they should teach their students to program. The academic computer-science community as a whole certainly doesn't know how to teach programming well and anecdotally one hears of widespread low achievement, high failure rates and attempts to mask the failure with administrative smokescreens. My own experience is, I believe, typical: in 35 years of trying to teach introductory programmming I never managed to teach more than a minority to program well, and a similarly-sized minority apparently failed to learn very much at all. In trying to overcome the problem it often seems to me that we have tried everything, and that we continue to try old things as if they were new. Although success is often trumpeted it somehow fails to find its way into communal practice, and failure is usually forgotten about (my own failure, for example, to teach programming through proof (Bornat, 1986)).

Widespread low achievement in undergraduate programming studies is well known and well researched (see, for example, (Lister et al., 2004; Tolhurst et al., 2006; Fincher et al., 2005; Simon et al., 2006)). The question of failure rates is less well researched and is controversial (Bennedsen and Caspersen, 2007). There have been all kinds of attempts to find predictors of performance, with little success; Robins et al. (2003) gives a summary.

I believe that the problem is real but that we don't understand its causes. So in the early 2000s, towards the end of my academic career, I encouraged my PhD student Saeed Dehnadi to search for causes in the way that students learn rather than in the way we teach. He responded enthusiastically and uncovered a very peculiar phenomenon: if he asked students struggling in a programming course about the execution of three or four-line computer programs (see appendix A) they would often answer as if they already had a mental model of what the effect of the program would be. Their models weren't always the ones taught in the course, but they were recognisably rational. He found that the same mental models could be seen if he asked novice students, before the course started, the same questions, and this despite the fact that those same students reported no previous contact with programming. He became even more enthusiastic when he discovered that those who

*Originally posted 2014/07/11; modified 2014/07/24 to take account of Lung et al.'s replication attempt, which I had forgotten about until reminded by Cat Ferguson at RetractionWatch, to add an abstract, and to make some minor linguistic alterations.

appeared to use mental model in the pre-course test did much better in the end-of-course exam than those who didn't. The pre-course test phenomenon is reproducible, it turns out, across a wide range of academic institutions and across countries, as is its apparent influence on exam results (Dehnadi, 2009; Dehnadi et al., 2009). We have even seen Dehnadi's models in a cohort of 14-year-old school students (Bornat et al., 2012).

So far so scientific. Dehnadi showed that it is *as if* novices use a mental model of simple program execution to answer his test questions. Apart from those who report some prior contact with programming, we don't know that they had such a model before the test, and there may be clues in the questions that an ingenious novice could use to guess what might be going on. His test is not a very good predictor: most of those who appear to use a model in the pre-course test pass the end-of-course exam but so do many of those who do not. And it doesn't predict the *level* of performance, as we discovered when we tried some deeper statistical analysis (Bornat et al., 2008). But the phenomenon is real and the prediction it makes is reproducible, as Dehnadi showed in a meta-analysis in his thesis (Dehnadi, 2009). That meta-analysis is summarised in (Dehnadi et al., 2009).

But that's not all. It's not enough to summarise the scientific result, because I wrote and web-circulated "The camel has two humps" in 2006. That document was very misleading and, because web documents persist, it continues to mislead to this day. I need to make an explicit retraction of much of what it claimed. Dehnadi didn't discover a programming aptitude test. He didn't find a way of dividing programming sheep from non-programming goats. We hadn't shown that nature trumps nurture. He had, however, found a predictive phenomenon, though he had no explanation of it.

2 How it happened

Though it's embarrassing, I feel it's necessary to explain how and why I came to write "The camel has two humps" and its part-retraction in (Bornat et al., 2008). It's in part a mental health story.

In autumn 2005 I became clinically depressed. My physician put me on the then-standard treatment for depression, an SSRI. But she wasn't aware that for some people an SSRI doesn't gently treat depression, it puts them on the ceiling. I took the SSRI for three months, by which time I was grandiose, extremely self-righteous and very combative – myself turned up to one hundred and eleven. I did a number of very silly things whilst on the SSRI and some more in the immediate aftermath, amongst them writing "The camel has two humps". I'm fairly sure that I believed, at the time, that there were people who couldn't learn to program and that Dehnadi had proved it. The paper doesn't exactly make that claim, but it comes pretty close. Perhaps I wanted to believe it because it would explain why I'd so often failed to teach them. It was an absurd claim because I didn't have the extraordinary evidence needed to support it. I no longer believe it's true.

I also claimed, in an email to PPIG, that Dehnadi had discovered a "100% accurate" aptitude test (that claim is quoted in (Caspersen et al., 2007)). It's notable evidence of my level of derangement: it was a palpably false claim, as Dehnadi's data at the time showed.

Because of some of the other silly things I did, I was suspended from my job at Middlesex. The university was wise enough, once the dust had settled, to recognise that there had been a mental health issue and to take me back, and it was kind enough to support me whilst I recovered from my depression. It was during that later period that Dehnadi and I asked some statistician colleagues if they could help us recover more information from his data. Was there, for example, evidence that his test predicted the performance of novice students: beyond the pass/fail distinction which he had shown he could predict to an extent, could we tell who would do well and who would do better? Were there age or sex differences? (The statisticians asked about those: we weren't looking.) After a lot of work, the answers were, by and large, that we couldn't see any such differences in our data. To find no prediction of performance was disappointing. I was fairly

depressed and I was in charge of the writing, so (Bornat et al., 2008) reads as if the research had been a failure, that nothing had been found. That isn't so: those who appear to use a rational mental model in the pre-course test are more likely to pass the end-of-course exam, but the spread of their marks is not statistically distinguishable from the spread of those who don't appear to use a model in the test.

Neither bombastic elation nor depressive rejection was a correct response. Dehnadi, to his credit, stuck to his guns and did the meta-analysis that showed that he'd discovered a phenomenon and that his test was a worthwhile predictor. He successfully defended his thesis (Dehnadi, 2009) and we published a summary of his results (Dehnadi et al., 2009).

Just to be clear: I do not believe that Dehnadi discovered an aptitude test for programming, as I claimed in 2006. Nor do I believe in programming sheep and non-programming goats. On the other hand, neither do I believe that further investigation showed that he'd found nothing of substance, as I implied in 2008.

3 Other experiments with Dehnadi's test

I'm aware of four published experiments which use Dehnadi's test but were not carried out by him.

Wray (2007) didn't replicate Dehnadi's experiment, but administered the test after the course was completed. It doesn't support the claim implicit in "The camel has two humps" that there are people who cannot pass the test; nearly all of his cohort could deal correctly with the test questions.

Caspersen et al. (2007) gave Dehnadi's test to 142 students before the course began; of those 124 were 'consistent' in Dehnadi's terms and 120 passed the course; of the 18 who were 'inconsistent', 14 passed; and those 14 reported in interviews that in the test they had used a mental model, but had switched models during the test. These results look quite different to those that Dehnadi investigated: there wasn't a conventional written examination; almost all the cohort passed; and only a tiny minority (2 out of 142) may not have used a model in the test. The result wouldn't undermine Dehnadhi's meta-analysis, but it definitely is a statistical outlier, and it fatally undermines any claim that non-programmers are everywhere in the population. In addition to the replication, they ranked test answers by level of 'consistency', and found no evidence of correlation between that ranking and several measures of course performance.

Lung et al. (2008) gave Dehnadi's test to 59 out of 229 novice students at the University of Toronto. Participants were self-selecting, but they compared results between tested and untested students and found no difference. However only 3% of tested students failed the course. When they looked for evidence that the test predicted a final mark above the median in the course examination they didn't find it. They were unable to provide their data to Dehnadi, and it wasn't included in his meta-analysis.

Ford and Venema (2010) administered Dehnadi's test at the end of a programming course, in an attempt to assess not the students but the effectiveness of the course. They found that 50% of their cohort hadn't grasped the notion of assignment, and report that the course was changed as a result.

Caspersen et al. didn't refute Dehnadi's result, and Wray didn't replicate the experiment, but each showed that it isn't the case that programming courses are doomed to fail a large proportion of the intake. Because of a high dropout rate, Lung et al. didn't show that, but like Caspersen et al. they claim to have refuted his result. Those two experiments certainly refuted my claim that Dehnadi had discovered an aptitude test, and they do suggest that even the more cautious claim that 'consistency' in the Dehnadi test affects pass/fail performance in the final examination is affected at least by cultural environment and educational practices.

4 Recent developments

Robins (2010) describes a simulation which produces the "two humps" distribution of marks, given that in a course there are a sequence of topics, each dependent on previous topics. If a student grasps topic 1 then he/she is more likely to grasp topic 2, and so on. Those who don't grasp topic 1 are less likely to grasp topic 2, and therefore less likely still to grasp topic 3, and so on. His notion of learning edge momentum might explain the teaching problems and Dehnadi's results.

Dehnadi, David Barton and I tested a cohort of 14-year-old school students, finding that about 50% of them seemed to use a rational mental model. In (Bornat et al., 2012) we reported this result, linking it to Robins' explanation.

University students in Mexico appear to perform like undergraduates in the UK (research so far incomplete and unpublished; investigation by Edgar Cambranes-Martinez at the University of Sussex).

There are some interesting papers in PPIG 2014 which take the question further. Raymond Lister and his student Donna Teague (Teague and Lister, 2014; Ahadi et al., 2014; Teague, 2014) have applied neo-Piagetian learning theory to the problem of early learning of programming. They have some very interesting results from testing and interviewing students, and in particular a regression line showing that students who have fallen below in week 3 of the course, according to a test which is partly based on Dehnadi's, have a lower chance of success in the final exam. Their test is partly based on Dehnadi's, but it goes further, and Teague's interviews give much more information than Dehnadi was able to gather.

5 Conclusion

There wasn't and still isn't an aptitude test for programming based on Dehnadi's work. It still appears to be true that novices who answer 'consistently' in the test are more likely to pass a programming course. Current work, by others, begins to suggest reasons for the phenomenon and open future research avenues.

References

Alireza Ahadi, Raymond Lister, and Donna Teague. Falling behind early and staying behind when learning to program. In *PPIG '14: Proceedings of the 25th Annual Workshop of the Psychology of Programming Interest Group*, 2014. URL `http://www.sussex.ac.uk/Users/bend/ppig2014/8ppig2014_submission_15.pdf`. Visited 2014-07-11. 4

Jens Bennedsen and Michael E. Caspersen. Failure rates in introductory programming. *SIGCSE Bull.*, 39: 32–36, June 2007. 1

Richard Bornat. *Programming from First Principles*. Prentice/Hall International, 1986. 1

Richard Bornat, Saeed Dehnadi, and Simon. Mental models, consistency and programming aptitude. In Simon and Margaret Hamilton, editors, *Tenth Australasian Computing Education Conference (ACE 2008)*, volume 78 of *CRPIT*, pages 53–62, Wollongong, NSW, Australia, 2008. ACS. URL `http://crpit.com/confpapers/CRPITV78Bornat.pdf`. 2, 3

Richard Bornat, Saeed Dehnadi, and David Barton. Online testing of mental models, 2012. URL `http://www.ppig.org/papers/24/8.Observing_mental_models-Richard%20Bornat.pdf`. (visited 18/01/2014) PPIG 2012. 2, 4

Michael E. Caspersen, Kasper Dalgaard Larsen, and Jens Bennedsen. Mental models and programming aptitude. In *ITiCSE '07: Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education*, pages 206–210, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-610-3. 2, 3

Saeed Dehnadi. *A Cognitive Study of Learning to Program in Introductory Programming Courses*. PhD thesis, Middlesex University, 2009. 2, 3

Saeed Dehnadi, Richard Bornat, and Ray Adams. Meta-analysis of the effect of consistency on success in early learning of programming. In *PPIG '09: Proceedings of the 21st Annual Workshop of the Psychology of Programming Interest Group*, 2009. URL `http://www.ppig.org/papers/21st-dehnadi.pdf`. 2, 3

Sally Fincher, Raymond Lister, Tony Clear, Anthony Robins, Josh Tenenberg, and Marian Petre. Multi-institutional, multi-national studies in CSEd research: some design considerations and trade-offs. In *ICER '05: Proceedings of the first international workshop on Computing education research*, pages 111–121, 2005. 1

Marilyn Ford and Sven Venema. Assessing the Success of an Introductory Programming Course. *Journal of Information Technology Education*, 9:133–145, 2010. 3

Raymond Lister, Elizabeth S. Adams, Sue Fitzgerald, William Fone, John Hamer, Morten Lindholm, Robert McCartney, Jan Erik Moström, Kate Sanders, Otto Seppälä, Beth Simon, and Lynda Thomas. A multi-national study of reading and tracing skills in novice programmers. *SIGCSE Bull.*, 36:119–150, June 2004. 1

Jonathan Lung, Jorge Aranda, Steve Easterbrook, and Greg Wilson. On the Difficulty of Replicating Human Subjects Studies in Software Engineering. In *ICSE'08. ACM/IEEE 30th International Conference on Software Engineering*, pages 191–200. IEEE, 2008. 3

Anthony Robins. Learning edge momentum: A new account of outcomes. *Computer Science Education*, 20(1):37–71, 2010. doi: 10.1080/08993401003612167. 4

Anthony Robins, Janet Rountree, and Nathan Rountree. Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2):137–172, 2003. 1

Simon, Sally. Fincher, Anthony. Robins, Bob. Baker, Ilona. Box, Quintin. Cutts, Michael. de Raadt, Patricia. Haden, John Hamer, Margaret Hamilton, Raymond Lister, Marian Petre, Ken Sutton, Denise Tolhurst, and Jodi Tutty. Predictors of success in a first programming course. In *Proceedings of the 8th Australasian Conference on Computing Education-Volume 52*, pages 189–196. Australian Computer Society, Inc., 2006. 1

Donna Teague. Neo-Piagetian Theory and the Novice Programmer. In *PPIG '14*, 2014. URL `http://www.sussex.ac.uk/Users/bend/ppig2014/23ppig2014_submission_21.pdf`. Visited 2014-07-11. 4

Donna Teague and Raymond Lister. Blinded by their Plight: Tracing and the Preoperational Programmer. In *PPIG '14: Proceedings of the 25th Annual Workshop of the Psychology of Programming Interest Group*, 2014. URL `http://www.sussex.ac.uk/Users/bend/ppig2014/6ppig2014_submission_8.pdf`. Visited 2014-07-11. 4

D. Tolhurst, B. Baker, J. Hamer, I. Box, R. Lister, Q. Cutts, M. Petre, M. De Raadt, A. Robins, S. Fincher, et al. Do map drawing styles of novice programmers predict success in programming? A multi-national, multi-institutional study. In *Proceedings of the 8th Australasian Conference on Computing Education-Volume 52*, pages 213–222. Australian Computer Society, Inc., 2006. 1

Stuart Wray. SQ Minus EQ can Predict Programming Aptitude. In *Proceedings of the PPIG 19th Annual Workshop*, 2007. URL `http://www.ppig.org/papers/19th-Wray.pdf`. Visited 2012-09-30. 3

This questionnaire measures the way you think. There are no right or wrong answers.

First we collect some information about you.

After that the questions are designed to be partly familiar, partly unfamiliar. We want to see how you deal with that situation. Please answer carefully.

If you change your mind about an answer, you can go back and alter it, any time until you finish the survey.

Figure 1: test introduction

1. Read the following statements and tick the box next to the correct answer in the next column.	**The new values of a and b**	**Use this column for your rough notes please**

`int a=10;`
`int b=20;`
 	
`a=b;`
	☐ a=20 b=0
	☐ a=20 b=20
	☐ a=0 b=10
	☐ a=10 b=10
	☐ a=30 b=20
	☐ a=30 b=0
	☐ a=10 b=30
	☐ a=0 b=30
	☐ a=10 b=20
	☐ a=20 b=10
	
	Any other values for a and b
	a= b=
	a= b=
	a= b=	

Figure 2: a single-assignment question

A A brief description of the test

Dehnadi's test asks participants to predict the effect of programs with two, three or four assignment statements. It opens with a deliberately uninformative description (figure 1). The questions themselves, for example figure 2 and figure 3, give little more information, though there is a clue ('what are the new values') that they are asking about changes, a clue in the way the program is written that things might happen top to bottom, a clue in the declarations that something might happen right-to-left, and maybe a clue about sequence in the use of semicolon.

Dehnadi defined eleven assignment models that he expected to recognise in test answers. Eight of them were the possible combinations of three binary attributes: right-to-left/left-to-right; move/copy; add/overwrite. Another was swap, still another a reading of the equality sign as equality rather than assignment. The last was "nothing happens", which was remarkably rare.

Novices will answer the test: there are rarely more than a few blank test scripts (school students may perhaps be more rebellious and in our single experiment so far devised some new ways to spoil their papers). It's easy to recognise models in the single-assignment questions – most models correspond to a particular tick box – but in multiple-assignment questions answers are ambiguous. Dehnadi looked for the model which explained most answers.

Dehnadi described those who appeared to use the same model in eight or more of his twelve questions *consistent*, and those who did not *inconsistent*. Those labels should properly be applied to the answers, not the people: applying them to novices makes it seem that something psychometric has been discovered,

7. Read the following statements and tick the box next to the correct answer in the next column.	**The new values of a, b and c**	**Use this column for your rough notes please**

`int a=5;`
`int b=3;`
`int c=7;`
 	
`a=c;`
`b=a;`
`c=b;`
	☐ a=0 b=0 c=7
	☐ a=7 b=7 c=7
	☐ a=3 b=5 c=0
	☐ a=3 b=5 c=5
	☐ a=12 b=15 c=22
	☐ a=0 b=0 c=15
	☐ a=8 b=15 c=12
	☐ a=3 b=12 c=0
	☐ a=5 b=3 c=7
	☐ a=5 b=5 c=5
	☐ a=3 b=3 c=3
	☐ a=3 b=5 c=7
	☐ a=7 b=5 c=3
	☐ a=3 b=7 c=5
	☐ a=12 b=8 c=10
	☐ a=8 b=10 c=12
	
	Any other values for a, b and c
	a= b= c=
	a= b= c=
	a= b= c=
	a= b= c=	

Figure 3: a three-assignment question

which is not the case.

We have recently automated the test (correcting some errors in the original) so that it can be taken online. Information is available from the author.