PROBLEM FORMULATION ABILITY IS A STUDENT'S PROBLEM IN CS1!

Stanley T. Schuyler, Edinboro University, sschuyler@edinboro.edu
Robert Joseph Skovira, Ph. D., Robert Morris University, skovira@rmu.edu

ABSTRACT

At IACIS 2007 we presented a pilot research project [13] titled *Is the Problematic in CS1 a student's Problem Solving Ability?* The acronym "CS1" refers to first courses in computer programming (and *CS0* to a preparatory course). The symptom of the problem was that 40% of the students enrolled in CS1 courses worldwide withdrew, failed or performed poorly. Several studies suggested that a key factor might be that students lack problem solving ability. The pilot study administered three questionnaires, each with a free writing task, to quantify a student's ability to analyze and identify problems. The results identified one questionnaire that correlated with student's CS1 course grades and accounted for 60% of the variance final exam scores. That result was motivated a more comprehensive study. The questionnaire was revised to focus on assessing *Problem Formulation Ability*. The results in this new study provide strong evidence that *Problem Formulation Ability* affects student performance and that it is one of the components that effectively predicts a student's CS1 course grade. The objective is to find an instrument to effectively place students in either a CS0 or CS1 course.

INTRODUCTION

This paper describes the research and results of investigating Problem Formulation Ability (denoted PFA) and using it to predict the performance of first time students (those with no previous programming experience). A second factor, labeled Learning Technique (LT) [5], was investigated simultaneously with PFA. Both PFA and LT demonstrated effectiveness in predicting first time students' course grades. Neither factor predicted course grades for experienced students (those with previous programming experience). The research on LT is described in [15], which is published with this report for presentation in IACIS 2008.

Problem Formulation Ability

Problem solving involves two sub processes: problem formulation and solution formulation. Table 1 (below) describes five process steps (reprinted from [13]). PFA refers to the first two steps in Table 1, and includes aspects of step 3. Students who know this process should be capable of describing how they perform it. Descriptions showing competency (PFA capability) would contain terms that relate to the first 2 or 3 steps in Table 1. These steps focus on analyzing a situation to define problems or opportunities in it. Terms that describe transition aspects of the 3rd step, steps 4 and 5 are related to solution formulation. Solution formulation is not explored in this study. It is presumed that the solution process is learned in CS1 [8] [9]. However, the learning of the solution process depends on students already knowing the problem formulation process [13].

Table 1: Problem Solving Process Steps Defined

Step	Any Domain Pólya [11] 1957	Any Domain Allen [1] 1995	Programming Domain Lister [8] 2004
1	Understand the Problem: -Know terms -Show what? -Restate -Diagram -Question?	Select the problem or situation or let it select you.	Abstract the problem from the description.
2	Devise a Plan -Imagine -guess, check -List possible -Patterns? -Exceptions? -Model/reason -Use/reuse … -Simplify -Go backwards	Observe, organize, and define the problem or situation.	Decompose the problem into sub-problems.
3	Carry out Plan -Iterate/persist -Trial/error -Change	Learn by questioning everything.	Transform sub-problems into sub-solutions.
4	Look back -Evaluate -Continuous improvement	Visualize solutions; select one, and refine it.	Synthesize sub-solutions to higher levels.
5		Employ the solution and monitor the results.	Evaluate the solution and iterate until verification.

RELATED STUDIES

Research and publication investigating student performance in CS1 began in the 1970's [10]. In

2006 Farrell [3], Lister et al [8], Simon et al [16] and Whittington [20] published on this topic. In 2007 the pilot study cited above was published [13]. Because the literature on CS1 performance is extensive, it cannot be reviewed adequately in this short paper. Some of the previous studies [2][5][8][9][12][16][18][19][20] discuss five significant factor domains: *cognitive*, *aptitude*, *performance*, *personal characteristics*, and *previous experience*. At least 56 different factors have been studied [14].

The methodologies used here included qualitative, quantitative and mixed methods. Since this study leverages linear regression, quantitative research that employed linear regression is featured for review. In particular, six studies are highlighted that investigated 7 of the 56 factors [2][3][6][17][18][19]. Of all factors investigated using regression techniques, the following were reported as significant in explaining the variance in student performance: *Mathematics Aptitude*, 31%-33%[2]; *Spatial Aptitude*, 2.5%-8%[2]; *Previous Experience*, meaning the presence of a home PC, 1%-3% [2], and 6% [18]; *Gender* explained the differences in other factors anywhere from 1% to 5% [2]; *Comfort Level*, similar in meaning to *Self-efficacy* (comfort or self-confidence in a CS1 context), 13%[18], 22%[19]; *Attribution*, meaning ascribing success to self or others, 6%[19]; *Past Math Courses*, 5%[19]; and *Mental Model*, 10% [18]. Depending on the study, multiple factors explained differing total amounts of the variation: 30%[18], 33%[19], and 35% (females) vs. 44% (males)[2].

Programming Aptitude Tests (PATs), and studies on PAT effectiveness, also used quantitative methods [3][17]. The development of the first PAT was done in the 1960's [17]. The study by Farrell reported accounting for 26% of the variance [3]. PAT tests were rejected as indicators of future student performance by IBM in the 1970's [17].

What many of these studies have in common, as pointed out in the pilot study [13], are discussions about "problem solving ability." However, there are no operational definitions or direct measures of it. An unstated assumption seemed to be that if students can solve a given problem, then they possess both problem formulation (steps 1, 2 and part of 3 in Table1) and solution formulation abilities (part of steps 3, all of 4 and 5 in Table 1). Lister [8] challenged this assumption by individually assessing specific skills involved in computer programming: reading, comprehending (demonstrated by tracing code), and solving (demonstrated by completing code). Other studies related problem solving more directly to the language skills involving reading, interpreting, and articulating [2] [16] [9]. All these studies motivated the selection of PFA and LT in this research.

Purpose and Objectives

The purpose of this research is to predict student performance (final course grade), using assessments of Problem Formulation Ability (PFA) and Learning Technique (LT). To this end, the following objectives were established: solidify an operational definition of PFA; automate the method of calculating PFA scores so all narratives are processed identically; verify that narrative content and PFA scores reflect key concepts presented in Table 1; and show that PFA scores are *effective* in predicting course grades of students with no prior programming experience. The determination that a factor is *effective* is that it enters a stepwise linear regression analysis at the p.05 level and accounts for at least 10% of the variance in grades.

METHODOLOGY

PFA Defined

The pilot study yielded a dictionary of terms (words) used by the participants in the free writing task, as well as term frequencies. A matrix of terms and participant's grades was sorted yielding patterns of term usage. By examining synonyms along with the themes contained in the narratives, a key term dictionary was constructed, with each term associated with a meta-tag representing the theme. The dictionary is too large to include in this paper. However, Appendix A contains the list of meta-tags and associated attributes. Weighting factors were derived from observations made from the data in the pilot study as to which meta-tag themes most strongly predicted CS1 performance.

The operational definition of PFA is a score on a scale from 0.0 to 5.0 determined as follows: for each word in a narrative, determine if it matches a word in a keyword dictionary; if a keyword is matched, locate an associated meta-tag for the keyword and increment its use count for the participant; after scanning the narrative, add meta-tag weights for those meta-tags used; the PFA score is the sum of weights divided by a scaling divisor (see Appendix A for weights and the divisor).

How does PFA reflect problem formulation?

The theory to support that PFA scores reflect problem formulation ability is rooted in Latent Semantic Analysis (LSA) [7]. The basis for LSA is that documents which share common themes share

key terms reflecting those themes. LSA is essentially a factor analysis of the frequency of words used across documents. In documents which are alike, the factor analysis yields components that indicate some degree of commonality in word use [4] [7]. LSA requires a large number of documents to be effective, on the order of the size of the vocabulary included by the documents [7]. To apply this technique to a few dozens of sampled narratives required that the expressed vocabulary be compressed into 10 to 12 terms (these are the *meta-tags*). The claim is: if the vocabulary were filtered by keyword matching, and keywords were mapped to a smaller number of meta-tags (representative of problem formulation concepts), then a factor analysis on meta-tag frequencies should reveal component clusters reflective of problem formulation steps. If components did not cluster (e.g. one component per meta-tag), or were randomly clustered, then PFA scores would not be indicative of problem formulation ability. For example, keywords describing categorization are mapped to the meta-tag <abstraction>, and keywords related to breaking a problem down into elements are mapped to the meta-tag <decomposition>. These tasks are two aspects of the same competency: *Classifying*. Therefore, these meta-tags should cluster in the same factor component rather that being singular components, or in a cluster with unrelated meta-tags (e.g. <plans>). Whether participant's responses to the questionnaire reflect PFA or not is based on the factor analysis results.

Revised Questionnaire

The questionnaire from the pilot study was revised and contained three parts: 1) a short demographic survey to gather information on major, gender, previous programming courses taken, the number of programming languages studied, and GPA; 2) a rank ordering task in which participants rank ordered nine items related to learning techniques (LT); and 3) a free writing narrative task in which they read a problematic situation description and then described how they would analyze it to identify problems or opportunities. Appendix B presents an abbreviated version of part 3.

Data Collection

Five CS1 course sections were solicited to participate in this research at two western Pennsylvania universities (convenience sample). No compensation or incentive was offered. There were a total of 107 students enrolled in these sections. Of these, 90 volunteered to participate. Participant identification codes (PIDs) were assigned to ensure all responses were anonymous. Participants were administered a questionnaire in the second meeting of each course section. Final exam and course grades were returned at the end of the semester.

Data Analysis

Demographic information was analyzed using Analysis of Variance (ANOVA) to establish whether sampled data was heterogeneous (or not) and formed a legitimate basis for subsequent statistical analysis. Factor analysis was used to evaluate the thematic aspects of meta-tags and PFA. Stepwise linear regression was used to evaluate the effectiveness of PFA for predicting students' course grades.

RESULTS

Demographic Results

Table 2 describes the final disposition of participants in this study. Of 90 participants, 24 withdrew. Of the 66 remaining, 23 indicated they had previous programming experience and 43 indicated they were first time programmers. Of the 43, 5 were identified that gave misleading information (e.g. the content in their narrative reflected prior programming experience), or were inconsistent in participating in the CS1 course (e.g. did not regularly attend or did not do assigned work). The 40% average for poor, failing, or withdrawing students in this sample aligns with percentages worldwide [13].

Table 2: Participant Experience and Completion

	Comp-leted	With-drew	Sub-total	Poor, Fail	With-drew, Poor
No Exper-ience	43	20	63	9	29
	48%	22%	70%	10%	32%
Exper-ienced	23	4	27	3	7
	26%	4%	30%	3%	8%
Totals	66	24	90	12	36
	73%	27%		13%	40%

Summary of ANOVA Results on Demographics

The primary concern with this sample was that the demographics related to gender, major, GPA, and prior programming experience were distributed heterogeneously across course sections. Except for programming experience, all other characteristics were distributed unremarkably. Two course sections had significantly more experienced participants than the other three (p.037, df.4). The impact on analyzing

the dependent variable (course grade) was that there were only a few non-experienced participants from two of the five sections. There were no questions specifically related to course section differences, so the impact on the purpose of this study was minimal.

Meta-tags Thematic Analysis: "What was measured?"

A factor analysis conducted using meta-tag frequency counts from the 66 participants completing their courses is presented in Table 3. The results of this analysis identified 4 components and accounted for 67% of the variance in meta-tag frequencies. These components are assigned the following themes: *Specifying*, *Designing*, *Classifying*, and *Analyzing*. Three of the themes align with problem formulation tasks; however, the second component, *Designing*, is associated with solution formulation because it contained the meta-tags <plans>, <evaluates> and <hypothesizes>.

Table 3: Factor Analysis of automatically identified meta-tags (N=66)

Rotated Component Matrix	Rescaled Component				% Variance
(PFA included to observe alignment)	*Specifying*	*Designing*	*Classifying*	*Analyzing*	
	1	2	3	4	67%
<determines>	.885	.289	.007	-.003	19%
<researches>	.756	-.103	-.017	.506	
PFA	.605	.312	.179	.321	
<communicates>	.399	.281	.144	.101	
<plans>	.109	.948	-.135	.086	18%
<evaluates>	.232	.621	-.007	.201	
<hypothesizes>	.121	.551	-.024	.103	
<abstracts>	.036	-.069	.934	.072	17%
<decomposes>	.113	-.067	.896	.049	
<analyzes>	.164	.196	.398	.858	13%
<questions>	.190	.257	-.110	.537	

Extraction Method: Principal Component Analysis.
Rotation Method: Varimax with Kaiser Normalization.
Rotation converged in 6 iterations.

In order to understand the appearance of the component labeled *Designing*, the experienced (N=23) and inexperienced (N=43) groups were analyzed separately. These results are reported in Table 4A and 4B (below). The experienced group, Table 4A, exhibited 3 components which were aligned with the expected problem formulation themes. The variance explained was 57%, which in part is reflective of the smaller sample size.

Table 4A: Factor Analysis Results of Experienced Group

Rotated Rescaled Component Matrices				
N=23	Experienced Programmers			
	Components (3)			% Var.
Meta-tags	*Specifying*	*Classifying*	*Analyzing*	57%
<<determines>>	.963	-.094	-.085	26%
<<researches>>	.850	-.135	.299	
PFA	.593	-.042	.348	
<<comm..>>	.294	-.025	.082	
<<evaluates>>	.277	-.107	.201	
<<abstracts>>	-.122	.955	.089	18%
<<decomp.>>	.099	.900	-.216	
<<plans>>	.196	-.394	.109	
<<analyzes>>	.508	.078	.758	13%
<<questions>>	.591	-.008	.638	
<<hypoth.>>	.013	-.168	.338	
Rotation converged in 6 iterations.				

Table 4B (below) presents the results for the inexperienced participants. Four components were identified, and the make-up of these components was not well aligned with the problem formulation steps in Table 1. The percent of variance explained is 72%.

Table 4B: Factor Analysis of No-experience Group

Rotated Rescaled Component Matrices					
N=43	No Programming Experience:				
	Components (4)				% Var.
Meta-tags	*Specifying**	*Classifying**	*Designing**	4	72%
<determines>	.876	.093	.009	.132	22%
PFA	.633	.266	.158	.464	
<comm.>	.557	.148	.170	.223	
<hypoth.>	.476	-.058	.382	.059	
<abstracts>	.139	.918	-.071	.059	20%
<decomp.>	.091	.888	-.052	.092	
<analyzes>	-.046	.646	.504	.550	
<plans>	.538	-.157	.789	-.083	18%
<evaluates>	.476	.010	.658	.081	
<questions>	-.041	.025	.654	.026	
<researches>	.410	.108	-.066	.874	12%
Rotation converged in 9 iterations.					

The component *Designing* showed up (absent in the experienced group). The first component, *Specifying*, no longer included the meta-tag <researches>, instead

it included <communication> and the meta-tag <researches> became a singular 4th component. The component related to *Classifying* incorporated the meta-tag <analyzing>, and separated it from <questioning>. We concluded from these results that experienced participants knew what problem formulation was and distinguished it from solution formulation; whereas inexperienced participants could not distinguish between problem formulation and solution formulation processes.

These results demonstrate that participants' words, filtered by keywords and mapped to meta-tags, represented themes related to problem solving processes only if participants systematically expressed problem formulation processes. The less systematically a narrative expressed these concepts, the more randomly meta-tags were associated with components, and the more components there were. To test this claim, all 90 participants were analyzed and 5 components were identified (not shown due to space limitations). The addition of the withdrawers added the fifth component indicating this group had even less alignment of the concepts.

PFA Analysis

Final exam grades and final course grades were both of interest for prediction. Unfortunately it was not possible to administer a common final exam across five course sections at two universities. In addition, post study interviews with CS1 instructors, as well as a review of the final exams, indicated that the exams tested for very different skill sets. Further, each instructor weighted exams differently to assign a final course grade. Therefore, final course grade was used as the dependent criterion variable.

A two-way ANOVA was run to determine whether course sections differed on PFA scores, and whether participants that withdrew differed from those that completed. The results showed that PFA means were significantly different between course sections (p.05, N=90) and between those that completed and those that withdrew (p.004, N=66/24). There was no interaction between course section and completion status.

An examination of programming experience (noted above as significantly different between course sections) explained the differences in PFA means: the mean PFA score for experienced participants was 3.1 and for non-experienced participants 2.43. Figure 1 below illustrates these relationships. Except for Course Section 3, PFA scores were consistently lower for withdrawers as compared to completers. Course Section 3 was an evening class. No demographics were collected to determine if there were more non-traditional students in that class than in the day classes. There was no difference in mean PFA scores as a function of experience within withdrawers (2.45) or within completers (3.1).

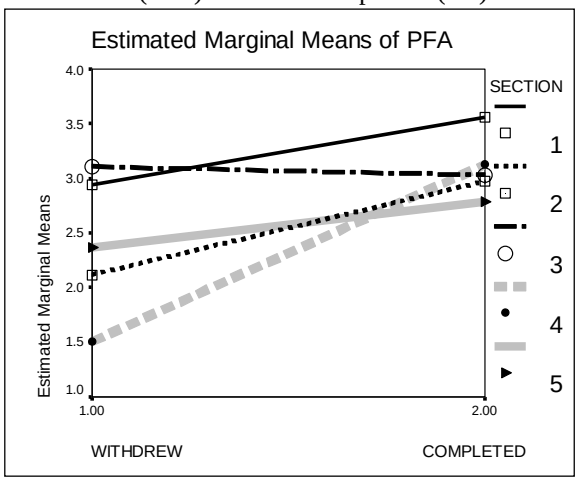

Figure 1: PFA Scores by Course Section

Using PFA to predict course grades

PFA and LTS (see [15] for LT details) variables were submitted as the independent variables to a stepwise linear regression procedure. Table 5 presents the results for 38 of the 43 participants that reported having no previous programming experience (the five determined to be anomalies were excluded).

PFA entered the equation at the p.05 level. PFA accounted for 32% of the variance. A second analysis was run with all 43 participants reported as non-experienced (included the 5 anomalies). The results of this regression showed PFA accounting for 11% of the variance. Figure 2 plots the relationships between actual grades and predicted grades on a 4.0 scale.

An alternative method of determining how well PFA predicted course grades (CSGs) was to see how well the final exam grade, treated as an independent variable, predicted students' course grades. The final exam grade entered the stepwise regression at the p.05 level and accounted for 51% of the variance in CSGs. This result is only 3% more than PFA and LTS jointly account for in participants with no prior programming experience.

CONCLUSIONS

This study demonstrated a method for assessing PFA (Problem Formulation Ability) and using it to effectively predict part of the variance in students' final course grades (for first time programmers in their first programming course). The assessment used

students' narratives produced in response to a free writing analysis task. The narratives were processed using an automated procedure that analyzed narratives uniformly for PFA related keywords. PFA, along with insight into a student's learning technique (LT), together could be an effective method to identify student's that should first attend a *CS0* course to learn how to formulate problems and learn alternative learning techniques. One interpretation of the results is that students that do not ask questions, or do not know how, struggle in a CS1 course.

A factor analysis of participants that knew how to express problem formulation (i.e. experienced programmers), yielded three components that related to the first two steps in Table 1. Participants that did not know how to express problem formulation yielded four components that were not clearly aligned with the first two steps in Table 1. The fourth component was associated with solution formulation. PFA ranged from 0.0 to 5.0. The higher the score, the more focused the student was on formulation; the lower the score, the more focused the student was on solution formulation (which presumably they have not yet learned).

REFERENCES

1. Allen, Stephen D. (1995). *Winnie-the-Pooh on Problem Solving*. Dutton Penguin Books USA Inc., New York, NY.
2. Choi-man, Chung. (1988). *Correlates of Problem-solving in Programming*. Chinese University of Hong Kong Educational Journal, 16(2), 185-190.
3. Farrell, Carl. (2006). *Predicting (and Creating) Success in CS1*. In Pratt (Ed.), In Issues in Information Systems, IIS Vol. VII, No. 1, pp. 259-263).
4. Friel. (2008). *Notes on Factor Analysis*. Retrieved February 12, 2008, from Sam Houston State University Web site: www.shsu.edu/~icc_cmf/cj_742/stats9.doc
5. Hughes, Janet., & Peirsi, D. Ramaee. (2006). *ASSISTing CS1 students to learn: Learning approaches and object oriented programming.* In Implagliazzo & Goldweber & Salamoni (Eds.), SIGCSE Bulletin, 38(4), pp. 275-279.
6. Ivins, Jim & Ong, Michele Poy-Suan. (2003). *Psychometric assessment of computing undergraduates*. (Department of Computing, pp. 1-15). Perth, Australia: Curtin University of Technology
7. Landauer, Thomas K., Foltz, Peter W., Laham, Darrell. (1998). *An Introduction to Latent Semantic Analysis.* Discourse Processes, 25, 1-41.
8. Lister, Raymond, Adams, Elizabeth S., Fitzgerald, Sue, Fone, William, Hamer, John, Lindholm, Morten, McCartney, Robert, Mostrom, Jan Erik, Sanders, Kate, Seppala, Otto, Simon, Beth, and Thomas, Lynda. (2004). *A Multinational Study of Reading and Tracing Skills in Novice Programmers*. SIGCSE Bulletin, 36(4), pp. 119-150.
9. Perrenet, Jacob, and Kaasenbrood, Eric. (2006). *Levels of Abstraction in Student's Understanding of the Concept of Algorithm: the Qualitative Perspective*. Implagliazzo, Goldweber and Paola (Eds.), SIGCSE Bulletin, 38(3), pp. 270-274.
10. Petersen, C. G., & Howe, T. G. (1979). *Predicting academic success in introduction to computers*. AEDS, 12, 182-191.
11. Pólya, George. (1957/1945). *How to Solve It*. Doubleday and Co., Inc., Garden City, NY.
12. Rountree, Nathan, Rountree, Janet and Robins. (2004). *Interacting Factors that Predict Success and Failure in a CS1 Course*. SIGCSE Bulletin, 36(4), pp. 101-104.
13. Schuyler, Stanley T., Skovira, Robert J. (2007). *Is the Problematic in CS1 a Student's Problem Solving Ability?* In Issues in Information Systems, IIS Vol. VIII, No 1-2, pp.112-119.
14. Schuyler, Stanley T. (2008). *Using Problem Formulation Ability to predict Student Performance in a First Course in Computer Programming*. Doctoral Dissertation, Robert Morris University.
15. Schuyler, Stanley T. (2008). *The role of Learning Techniques on student performance in CS1 courses*. To be published in the same IIS as this paper for IACIS 2008.
16. Simon and Cutts, Quinton, Haden, Patricia, Sutton, Ken, Box, Ilona, Hamer, John, Lister, Raymond, Tolhurst, Denise, Fincher, Sally, Robins, Anthony, Baker, Bob, deRaadt, Michael, Hamilton, Margaret, Petre, Marian and Tutty, Jodi (2006). *The Ability to Articulate Strategy as a Predictor of Programming Skill*. Eighth Australian Computing Education Conference (ACE2006), 52, pp. 1-8.
17. Tukiainen, Markku., & Monkkonen, Eero. (2002). *Programming aptitude testing as a prediction of learning to program*. PPIG 2002 - 14th Workshop of the Psychology of Programming Interest Group, 14, 45-57.
18. Wiedenbeck, Susan, LaBelle, Deborah and Kain, Vennila, N.R. (2004). *Factors Affecting Course Outcomes In Introductory Programming*. PPIG 2004 – 16th Workshop of the Psychology of Programming Interest Group, Retrieved April 23, 2007 from www.ppig.org.

19. Wilson, Brenda Cantwell, Shrock, Sharon. (2001). *Contributing to Success in an Introductory Computer Science Course: A Study of Twelve Factors*. Technical Symposium on Computer Science Education. Proceedings of the 32nd SIGCSE technical symposium on Computer Science Education, pp. 184 - 188
20. Whittington, Kieth J. (2006). *Increasing Student Retention and Satisfaction in IT Introductory Programming Courses using Active Learning.* In June 25-28 (Ed.), Proceedings of the 2006 Informing Science and IT Education Joint Conference (pp. 307-312). Salford, UK: Informing Science Institute.

Table 5: Stepwise Linear Regression to predict final course grade

Descriptive Statistics				Dependent Variable: Course Grade (CSG) Independent variables: PFA and LTS
	Mean	Std. Deviation	N	
CSG	2.874	1.1037	38	
PFA	3.148	.8904	38	
LTS	.1608	.2452	38	

Model Summary					Change Statistics				
Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	R Square Change	F Change	df1	df2	Sig. F Change
1-PFA	.585	.343	.324	.9073	.343	18.757	1	36	.000
2-LTS	.716	.513	.485	.7918	.171	12.271	1	35	.001
Predictors: (Constant), PFA, LTS									

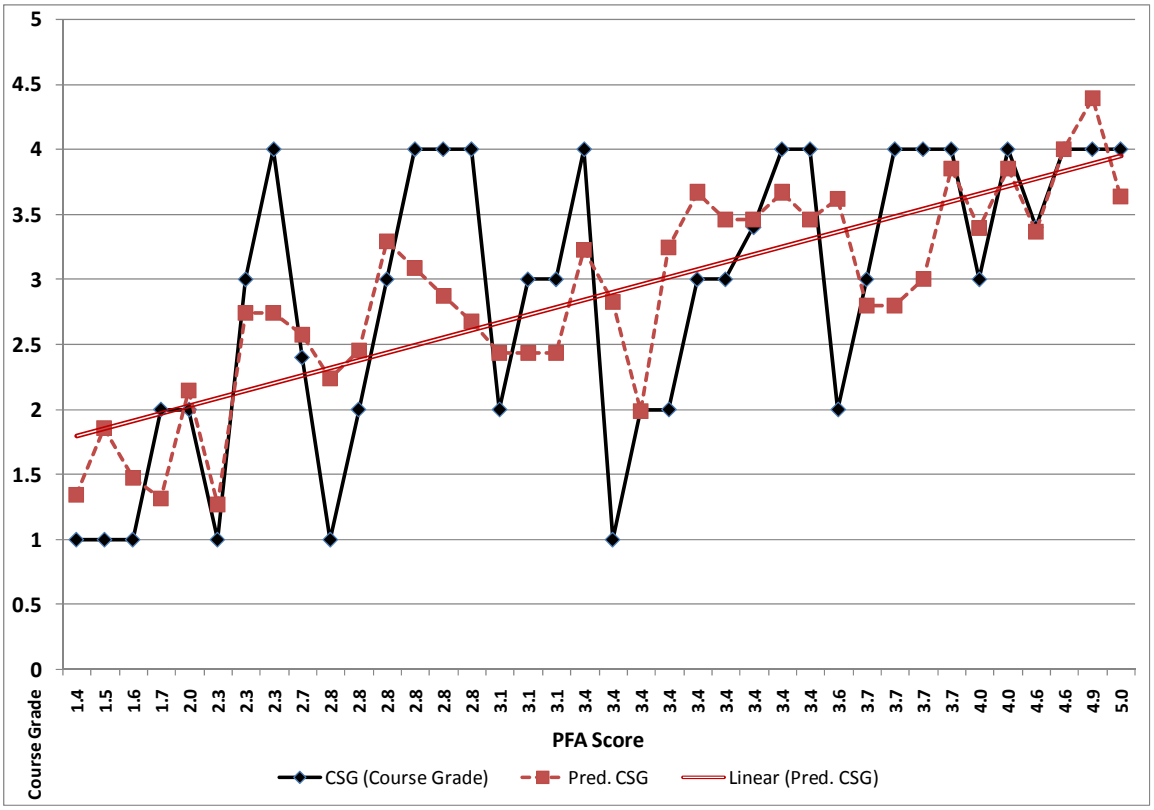

Figure 2: Course Grades Compared to Predicted Course Grades.

Appendix A: Problem Formulation Associated Meta-tags - adapted from [14]

Wt.	Meta-tag	Intended Semantics
2.90	<analyzes>	Expression contains evidence that the participant knows to analyze given information to understand the meaning of words and concepts describing the situation, and/or the situation itself.
1.00	<questions>	Expression contains questions posed to clarify or seek additional information related to who, what, where, when, and how questions.
4.00	<researches>	Expression contains evidence that the participant knows to gather additional information from other sources via observation, surveys, interviews, etc.
1.00	<abstracts>	Expression contains evidence that the participant knows to name, classify, categorize, organize or sort, associate, relate, find patterns, model, and summarize information provided or gathered.
1.00	<determines>	Expression contains evidence that the participant knows to identify solution requirements, users needs and scenarios, external interfaces, and the scope (boundaries) of the problem-solution situation.
3.20	<hypothesizes>	Expression contains evidence that the participant knows to imagine, guess, anticipate, and/or list alternatives.
2.10	<decomposes>	Expression contains evidence that the participant knows to break down the problem (opportunity) or solution requirements into component parts.
1.00	<plans>	Expression contains evidence that the participant recognizes a sequencing of future tasks for solution development such as algorithm design and implementation.
1.00	<evaluates>	Expression contains evidence that the participant recognizes that information, requirements, alternatives, decomposition, plans and development require verification by review and/or testing.
1.00	<communicates>	Expression contains evidence that the participant knows that solutions need documentation to transfer information to parties involved in the solution development process.
-1.50	<detractor>	Expression contains counter-evidence that the participant does not know how to perform one or more aspects of analyzing the situation.
3.54	Divisor	The sum of the weights divided by 5 to yield a PFA between 0 and 5

Appendix B: Abbreviated Description of Part 3 of the PFA/LT Questionnaire [14]

You will read a situation description described in the box below and describe how you would analyze the situation.

NOTE: You are not to do the analysis: *you are to* **describe what you would do and what you would ask** if and when you were analyzing it (e.g. a plan of action, not conclusions or results).

Situation:

*You may, someday in the future (**but not now**), be asked to design and possibly build applications (computer programs) that relate to the results of your analysis.*

Situation Description:

Scientists, engineers, students, instructors, market analysts and others have data in the form of categories. Each category contains various sized lists of alphabetic and numeric data entries. These data need to be analyzed quantitatively by calculating totals, averages, and other statistical calculations. The results of the calculations are used to summarize characteristics of the categories, make comparisons, draw conclusions, and communicate discoveries to other interested parties.

Describe Your Approach to Analyzing this situation:

Reminder: You are not to do the analysis: you are to describe what you would do and what you would ask to analyze it.

1.