Title	Predictive validity of measures of the pathfinder scaling algorithm on programming performance: Alternative assessment strategy for programming education
Author(s)	Lau, W; Yuen, A
Citation	Journal Of Educational Computing Research, 2009, v. 41 n. 2, p. 227-250
Issued Date	2009
URL	http://hdl.handle.net/10722/125511
Rights	Creative Commons: Attribution 3.0 Hong Kong License

J. EDUCATIONAL COMPUTING RESEARCH, Vol. 41(2) 227-250, 2009

PREDICTIVE VALIDITY OF MEASURES OF THE PATHFINDER SCALING ALGORITHM ON PROGRAMMING PERFORMANCE: ALTERNATIVE ASSESSMENT STRATEGY FOR PROGRAMMING EDUCATION

WILFRED W. F. LAU
ALLAN H. K. YUEN
The University of Hong Kong

ABSTRACT

Recent years have seen a shift in focus from assessment of learning to assessment for learning and the emergence of alternative assessment methods. However, the reliability and validity of these methods as assessment tools are still questionable. In this article, we investigated the predictive validity of measures of the Pathfinder Scaling Algorithm (PSA), a concept mapping assessment utility, using the referent-free and referent-based approaches on programming performance of a group of secondary school students. Results suggest that the predictive validity of both approaches was more or less the same. Among the three similarity measures applied for the referent-based approach, PRX appeared to be the most predictive one whereas PFC and GTD were similar in terms of predictive power. The correlations between the referent-free measure C and the three previously mentioned referent-based measures with the programming performance measures were not as strong as reported in the literature. In the light of these results, we argue that there is a need to reform assessment in programming education.

doi: 10.2190/EC.41.2.e
http://baywood.com

INTRODUCTION

Assessment has long been a central focus in any curriculum framework. It is generally accepted that classroom assessment serves three inter-related purposes: assessment for learning, assessment as learning, and assessment of learning (Earl & Katz, 2006). Assessment for learning aims to explicate students' understanding so that teachers can help students to progress further. It is mostly formative in nature. Assessment as learning emphasizes the metacognitive role of a student to monitor and reflect on his or her learning in relation to assessment. Assessment of learning checks whether students' proficiency is in alignment with the curriculum learning outcomes and it is usually summative in nature. It also plays a predominant role in traditional assessment. Not until recently, there has been a call for a shift from assessment of learning to assessment for learning as a challenge to the dominant role of the former (Birenbaum, Breuer, Cascallar, Dochyd, Dori, Ridgway, et al., 2006).

In response to this change, alternative assessment methods have emerged such as cognitive assessment, performance assessment, and portfolio assessment (Reeves, 2000). Notwithstanding the variety that these assessment methods might take, the primary purposes of these methods are to provide students with authentic feedback to improve learning on one hand and teachers with flexible instructional strategies to enhance pedagogy on the other hand. However, a major concern arises regarding the reliability and validity of these methods. How reliable and valid are they compared with the traditional ones? In terms of validity, this can be accomplished through examining their predictive validity. Predictive validity is "the relation between a predictor or combination of predictors, such as test scores and grades, and an outcome, such as grades in a graduate management program" (Talento-Miller & Rudner, 2008, p. 131) and it is often expressed in terms of correlation coefficient.

Although numerous studies (Acton, Johnson, & Goldsmith, 1994; Goldsmith, Johnson, & Acton, 1991; Gomez, Hadfield, & Housner, 1996; Housner, Gomez, & Griffey, 1993a, 1993b; Johnson, Goldsmith, & Teague, 1994) have been conducted to examine the prediction of similarity measures, which are indices showing the closeness to an expert's mental model, on academic performance, results have tended to be mixed from low to moderate or high association between the variables. Of particular interest in this study is the Pathfinder Scaling Algorithm (PSA) (Schvaneveldt, 1990) since there seems to be substantial research evidence to support the predictive validity of the technique (Goldsmith et al., 1991; Housner et al., 1993a, 1993b; Johnson et al., 1994). The PSA is a psychological scaling technique to assess structural knowledge and hence mental models of individuals. From an assessment perspective, the PSA can also be used as a concept mapping utility for assessing knowledge change and measuring expertise.

Typically, the PSA constructs a network of concepts of the problem domain concerned called the Pathfinder Network (PFNET). Nodes and links represent concepts and relations between concepts respectively in a network (see Figure 1). The construction of a PFNET is based on the graph theory in Mathematics

(Schvaneveldt, Dearholt, & Durso, 1988). To determine whether a link exists between two nodes, the PSA searches through all the possible paths between the two nodes. If the minimum distance between nodes based on all the indirect paths is greater than or equal to the distance of the direct path, then a link is added between the two nodes. Two important parameters, r and q, are used to determine how distance in a network is measured and affects the network density. The distance d_{ij} between nodes N_i and N_j is evaluated as $d_{ij} = \min(W(P_{ij1}), W(P_{ij2}), \ldots, W(P_{ijm}))$ assuming that there are m paths P_{ij} with path weights $W(P_{ij})$ connecting nodes N_i and N_j and $W(P_{ijk}) = \left(\sum_{s=1}^{n_k} w_s^r \right)^{\frac{1}{r}}$ where $w_1, w_2, \ldots, w_{n_k}$ are the weights of the kth path and $k = 1, 2, \ldots, m$. The parameter r defines distance measured in Minkowski metric. When $r = 1$ and $r = 2$, these correspond to the city-block and Euclidean metric respectively. When $r = \infty$, path length equals the maximum distance/weight of the link that forms the path. The parameter q limits the maximum number of links in a path and $1 \leq q \leq n - 1$. It can be shown that when $r = \infty$ and $q = n - 1$ where n is the number of nodes, the network contains the fewest number of links and is the least dense one. In fact, most studies in the literature set the two parameters to these values (Gonzalvo, Canas, & Bajo, 1994; Johnson et al., 1994). This study also used these values for the two parameters.

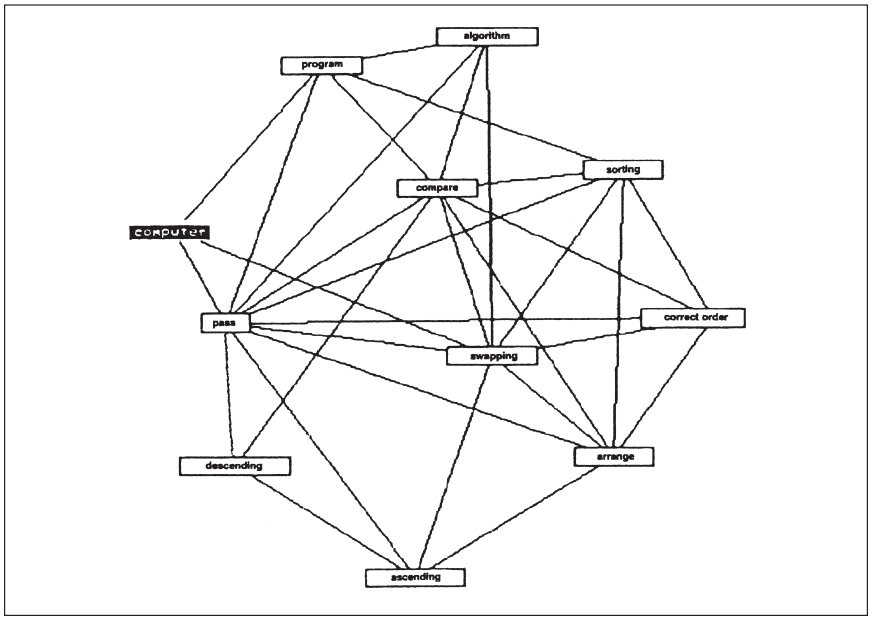


Figure 1. Example of a PFNET.

Details on the properties of PFNET can be found in an article by Dearholt and Schvaneveldt (1990).

In the literature, structural similarity of PFNETs is compared quantitatively through three kinds of measures: PRX, PFC, and GTD.

- PRX is simply correlation on raw proximities.
- PFC is a set-theoretic measure, which calculates the ratio of number of nodes in common to the number of nodes in either networks for each node of the network and averages the ratios for all the nodes to obtain an overall index, i.e. $PFC(A, B) = \frac{1}{n} \sum_{v \in V} \left| \frac{A_v \cap B_v}{A_v \cup B_v} \right|$ where A and B are two undirected label graphs with common node set V and n nodes (Goldsmith & Davenport, 1990, p. 83).
- GTD is a graph-theoretic measure, which is obtained by correlating the distances between the nodes in two networks.

Figure 2 is an example taken from the paper by Goldsmith et al. (1991, p. 90) to illustrate how PFC and GTD are calculated.

Predictive validity of the PSA has been widely reported in diverse learning domains. Goldsmith et al. (1991) obtained predictive validity of the PSA ranging from .61 to .74 using different measures of similarity in a statistics and design course. In a later study of undergraduate students in an introductory psychology course, Johnson et al. (1994) introduced a new set of similarity measures based on dichotomizing the Pathfinder distance with an absolute cutoff of one link distance and the proximity data with 25% cutoff. They found that the new measures were at least as predictive as PRX. Yet subsequent analysis showed that the predictive validity of these measures was reduced when the Mathematics scale of the American College Test was used instead. Housner et al. (1993b) showed that there was a significant increase in correlations between similarity measures (PRX, PFC, and GTD) and course performance variables including midterm examination, final examination, teaching rating, and final grade in a teaching methodology course in physical education. At the end of the semester, the correlations ranged from .58 to .85 as compared with the initial ones from .30 to .49.

Curtis and Davis (2003) demonstrated that after instruction in a managerial accounting course, measures of PFC were positively correlated with examination scores ($r = .5$; $p < .01$) and case analysis performance ($r = .47$; $p < .01$). When regressing case analysis scores on both examination scores and PFC scores, estimated regression coefficients of both predictors were significant and this suggested an incremental validity of PFC; i.e., the extent to which additional predictors help explain the criterion measure that is not explained by other existing ones. In another auditing course, PFC also revealed discriminant validity as it was positively correlated with self-efficacy for auditing tasks ($r = .27$; $p < .06$) but examination scores did not correlate with self-efficacy ($r = .03$; $p > .20$). Schau et al. (2001) reported that the correlations ranged from .33 to .46 between

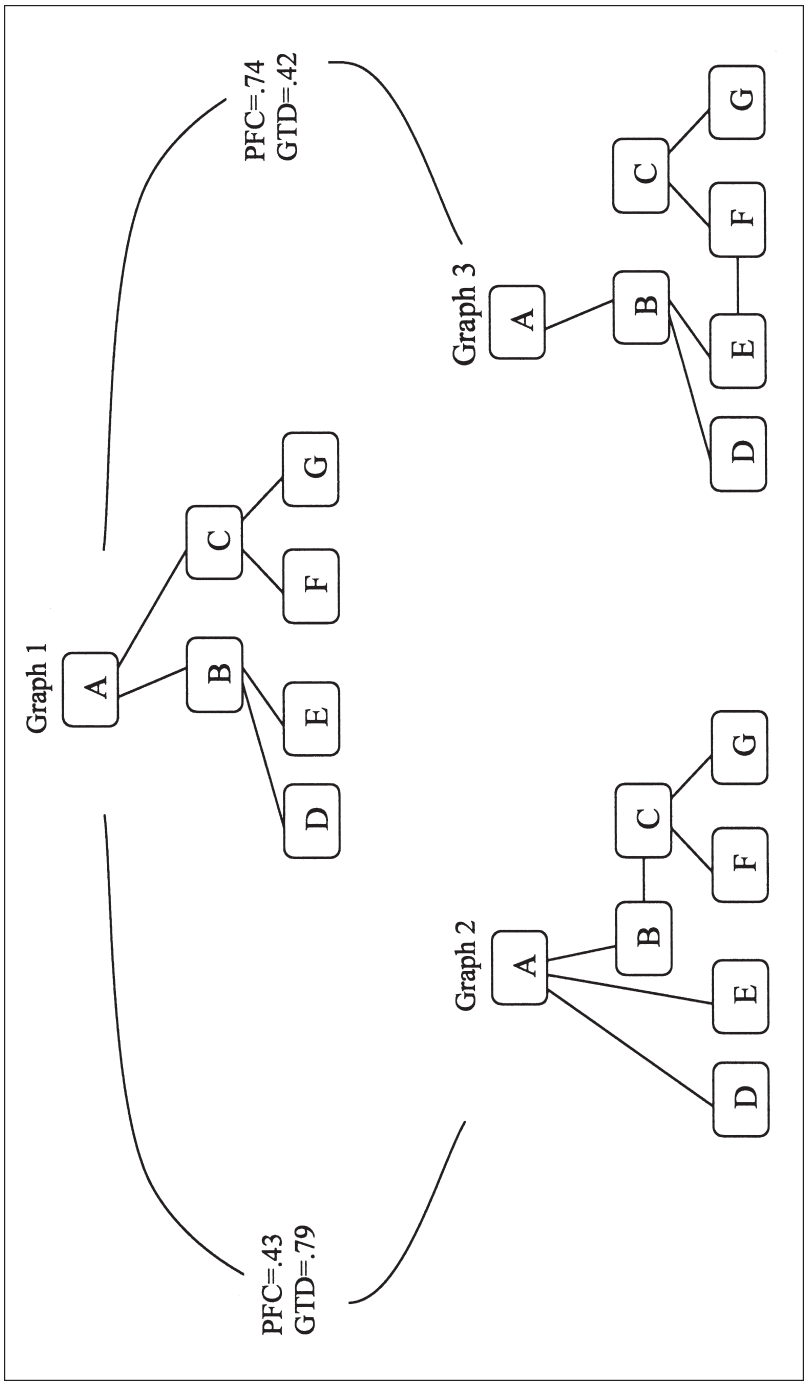


Figure 2. Similarities between Graph 1, Graph 2, and Graph 3 as measured by PFC and GTD (see Tables 1 and 2).

Table 1. Calculation of PFC between Graph 1 and Graph 2 in Figure 2

	Neighborhood		Intersection		Union		
Node	Graph 1	Graph 2	Set	Size	Set	Size	Quotient
A	{B, C}	{B, D, E}	{B}	1	{B, C, D, E}	4	1 ÷ 4
B	{A, D, E}	{A, C}	{A}	1	{A, C, D, E}	4	1 ÷ 4
C	{A, F, G}	{B, F, G}	{F, G}	2	{A, B, F, G}	4	2 ÷ 4
D	{B}	{A}	∅	0	{A, B}	2	0 ÷ 2
E	{B}	{A}	∅	0	{A, B}	2	0 ÷ 2
F	{C}	{C}	{C}	1	{C}	1	1 ÷ 1
G	{C}	{C}	{C}	1	{C}	1	1 ÷ 1

Sum of quotients = 3.000. FC = 3.000/7 = .43, ∅ = empty set.

Table 2. Graph-Theoretic Distances for Each Pair of Nodes in Graph 1 and Graph 2 in Figure 2

	Node						
Node	A	B	C	D	E	F	G
Graph 1							
A	—	1	1	2	2	2	2
B		—	2	1	1	3	3
C			—	3	3	1	1
D				—	2	4	4
E					—	4	4
F						—	2
G							—
Graph 2							
A	—	1	2	1	1	3	3
B		—	1	2	2	2	2
C			—	3	3	1	1
D				—	2	4	4
E					—	4	4
F						—	2
G							—

postcourse relatedness rating scores of astronomy concepts and multiple-choice examination scores for students as a whole and for students grouped by gender using a measure similar to PRX.

However, there is paucity of related studies conducted in the domain of computer programming and results concerning the predictive ability of various measures in this domain are still inconclusive. Also, previous research has tended to use referent structures (e.g., Nash, Bravaco, & Simonson, 2006; Trumpower & Goldsmith, 2004) to obtain similarity measures for comparing expertise. Davis, Curtis, and Tschetter (2003) argue that the referent-based approach "appears with much greater frequency in structural knowledge research" (p. 203). As such, this study aims to investigate the predictive validity of measures of the PSA using the referent-free and referent-based approaches in the context of learning computer programming where referent-free refers to no reference to expert structure whereas referent-based denotes the use of experts. In particular, the correlations between measures of both approaches (PRX, PFC, and GTD for the referent-based approach and C for the referent-free approach, which is to be explained in the next section) and programming performance measures, which are obtained from a programming performance test, are examined and compared with results from other studies in the literature. Implications of these results for assessment in programming education are discussed.

STRUCTURAL KNOWLEDGE AND CONCEPT MAP

Jonassen (1995) argues that structural knowledge methods can be used to depict mental models of individuals. Structural knowledge here refers to the knowledge of the structure of concepts in a domain and can be manifested visually in a concept map. Among the various available methods for eliciting concept maps, the PSA (Schvaneveldt, 1990) was selected in this study due to the following reasons. First, it gives a quantifiable concept map allowing comparisons with other learners and experts and measurement of change in understanding over time (Reese, 2003). Second, the network structure shows local relations among concepts, which are psychologically meaningful and possesses higher predictive power of free recall performance compared with other multidimensional scaling representations (Cooke, 1992; Cooke & Schvaneveldt, 1988). Finally, it has also been well researched in different domains (Acton et al., 1994; Cooke & Schvaneveldt, 1988; Curtis & Davis, 2003; Gomez et al., 1996; Trumpower & Goldsmith, 2004).

To construct a PFNET, participants are required to provide a rating from 1 to 9 for every possible pair of concepts based on the relatedness of the concepts in the pair concerned (see Appendix A). Ratings are then converted into proximities by subtracting each rating from 10, which are used to construct the network by the PCKNOT software (http://interlinkinc.net/index.html). Also, a weight, which is dependent on the strength of the relation, is given to each link. Theoretically,

n concepts need $\frac{n(n-1)}{2}$ pairwise comparison. However, it should be noted that the evaluation of similarity measures (PRX, PFC, and GTD) depends on the availability of experts in the domain concerned. Acton et al. (1994) remarked that "the experts were not highly similar in an absolute sense" (p. 310) and "the differences that do exist among experts can make an important difference in their utility as a referent structure" (p. 310).

As an alternative to this referent-based approach, Davis et al. (2003) suggest that the coherence measure C provided by the PSA can serve as a referent-free assessment of structural knowledge. Essentially, this measure assesses the internal consistency among judgments of similarity of concept pairs of an individual. The coherence measure of a set of proximity data is evaluated based on the assumption that relatedness between a pair of concepts can be inferred by the relations of the concepts to other concepts in the set. For each pair of concepts, an indirect measure of relatedness is determined by correlating the proximities between the concepts and all the other concepts. Coherence is obtained by correlating the original proximity data with the indirect measures. High correlation signifies high consistence of the original proximities with the relatedness inferred from the indirect relationships of the concepts and vice versa. As such, the predictive validity of both the referent-free measure (C) as well as the referent-based measures (PRX, PFC, and GTD) is compared in this study.

THE PSA AND PROGRAMMING PERFORMANCE

To date, not many studies have ever been done to testify the predictive validity of similarity measures of the PSA on programming performance. Also, most studies were referent-based and PFC was usually chosen as the similarity measure. Kahler (2001) investigated the relationship between structural similarity of students' mental models with the instructor's prototype model and students' project scores over a course in a semester. Three project scores were correlated with PFC and C based on ratings of nine to twenty related programming concepts. Using PFC measure as an index of similarity, it was found that among the three projects, only the correlation between PFC and the project three scores was statistically significant ($r = .501$, $p < .05$). Acton et al. (1994) used a number of referent structures including the instructors, other experts, and averaged top six best students from two basic courses in Pascal programming to obtain measures of PFC and correlated with students' examination performance. The correlations ranged from –.07 to .63. With a few exceptions, the results showed moderate to high values of predictive power of PFC in spite of variability of predictive ability among the experts.

Finally, Trumpower and Goldsmith (2004) investigated the effectiveness of interactive overview on students' learning. Three groups of students learned the

definitions of 12 sorting related programming concepts under three different conditions and then were assessed on definitional knowledge, conceptual knowledge, and procedural transfer knowledge. The expert group viewed the concepts organized according to an expert knowledge network. The random group saw the same network structure as that of the expert group except that the concepts were randomly located. The alphabetical group saw the same 12 concepts organized alphabetically and vertically. While there were no statistically significant differences in performance in the definitional knowledge test among the three groups, results indicated that the expert group showed statistically higher similarity, as measured by PFC, with the expert structure than the other two groups in the conceptual knowledge test and outperformed their counterparts in the procedural transfer knowledge test. This suggested that the expert group, being provided expert training, performed very similar to an expert. Although not explicitly verified, it is likely that similarity measures correlate with test performance scores in their study.

METHOD

Participants

One hundred and thirty-one students from nine secondary schools in Hong Kong took part in this study on a voluntary basis. They were either Secondary 4 (Grade 10) or Secondary 5 (Grade 11) students who opted for the elective module A (algorithm and programming) in the computer and information technology curriculum. The programming language was either Pascal or C. They all learned bubble-sorting algorithm by the time of data collection. The participants were asked to provide some background information including gender, ability group (Band 1, Band 2, or Band 3), and age. Fifty-two females (39.7%) and 79 males (60.3%) participated in this study. The majority of the participants were Band 2 students (49.6%) followed by Band 1 students (41.2%) and Band 3 students (9.2%), in which Band 1 corresponds to the highest ability group whereas Band 3 corresponds to the lowest ability group. As the students had no access to any information about their bands, they were required to self-report their bands based on their previous academic performance. Their ages ranged from 14 to 19. Mean age of the females was 16.42 (SD = 0.11) while mean age of the males was 16.01 (SD = 0.07).

Concept Map

To construct PFNETs, the participants were required to rate from 1 to 9 for every possible pair of concepts based on the relatedness of the concepts in the pair. Ratings were then converted into a proximity matrix, which was used to construct the PFNET. In this study, 11 concepts (computer, program, algorithm,

sorting, arrange, correct order, pass, compare, swapping, ascending, and descending) relevant to sorting were chosen after reviewing three commonly used textbooks in Hong Kong (Chan, 2004; Fung, Lau, & Kai, 2003; Woo, Shiu, & Wang, 2003). For 11 concepts, there were altogether 55 ratings to be done. Although these sorting concepts were found from Hong Kong textbooks, they are in fact generic concepts in learning sorting algorithm that are also found in comparable curricula in other countries such as the Advanced Placement Computer Science A in the United States and the Oxford, Cambridge, and RSA Examinations Advanced General Certificate of Education in Computing in the United Kingdom.

Programming Performance

Based on the taxonomy framework of programming knowledge adopted by Oliver (1993) and Lin (2002), a programming performance test was designed to assess participants' performance in declarative knowledge (DK; three multiple-choice questions), procedural knowledge (PK; four multiple-choice questions), conditional knowledge (CK; four fill-in-the-blank questions), and strategic knowledge (SK; two program writing questions) concerning bubble-sorting algorithm. One mark was awarded to each correct response for the seven multiple-choice questions and the four fill-in-the-blank questions. Each program writing question was scored by considering its syntax (two marks), semantic meaning (two marks), and degree of completion (one mark). The Cronbach's alpha values for DK, PK, CK, and SK were calculated for all the participants (131) and were found to be .54, .71, .78, and .96 respectively. Nunnally (1978) suggests a threshold value of .7 for a scale to be sufficiently reliable, whereas other researchers like Fornell and Larcker (1981) suggest a minimum composite reliability of .60. Thus, the Cronbach's alpha value for DK (.54) in this study is only marginally acceptable and this would certainly be a limitation to the subsequent conclusion. On the other hand, the high Cronbach's alpha value for SK (.96) was probably due to the fact most students could write similar programs for both program writing questions, resulting in high reliability of the scale. However, in view of the reliability coefficient of the whole test (.78), overall speaking, the reliability of the programming performance test meets the recommended standards. Questions of the whole test are listed in Appendix B.

Procedure

Data were collected online through a website. The participants were reminded that their participation was voluntary and data were collected anonymously and used solely for the purpose of research. First, they were asked to fill in some demographic information. Then, they spent another 15 minutes to complete a relatedness-rating task on sorting-related concepts in order to assess their mental models. They also took a 25-minute programming performance test on

bubble-sorting algorithm involving 12 questions given in Appendix B. Throughout the whole process, procedures were developed to detect any missing responses to the questions and prompt the participants to answer these questions again before submission.

The Referent Structure

We used the following criteria in choosing experts for providing referent structures. First, the expert must possess a degree in Computer Science or related discipline and second, the expert must have at least 6 years of either practicing or teaching computer programming. It seems to be justified to set the above criteria to identify experts since a university degree usually takes 4 years to complete and it is commonly agreed that a 10-year period is required to reach the level of an expert (Winslow, 1996). Based on the above criteria, three experts agreed to complete the rating task to provide referent structures. The first expert was a computer officer with 14 years of experience in system development, working in a university research center. He holds a Bachelor's degree in Computer Science, a Master's degree in Management of Information Technology, and another Master's degree in Business Administration. The second expert was an assistant computer officer working in the same research center as the first expert. She had a Bachelor's degree in Computer Science and had 7 years of programming experience. The third expert was a secondary school computer teacher with 10 years of teaching experience. He also holds a Master's degree in Computer Science.

Acton et al. (1994) concluded that individual experts are highly variable in their predictive power and variability can be largely reduced by averaging the ratings of the experts. However, it is possible that there may exist an excellent expert whose model is highly predictive of the performance measures. As such, we examined a number of combinations of referent structures. The similarity measure PFC was selected to correlate with the performance measures since many studies demonstrated its high predictive power among the three measures. Table 3 presents the correlations between the five performance measures (DK, PK, CK, SK, and Total) obtained for the participants with PFC obtained by matching participants' models with that of the first expert (EXP1), the second expert (EXP2), the third (EXP3), the average ratings of the first and second experts (AVE12), the average ratings of the second and third experts (AVE23), the average ratings of the first and third experts (AVE13), and the average ratings of the three experts (AVE123). For instance, in the column "EXP1_PFC", each participant's structure was compared with that of the first expert to obtain a value for PFC, and these values of PFC were then correlated with DK, PK, CK, SK, and the total respectively. Similar procedures were repeated to obtain the correlations for the other combinations of the experts as shown in Table 3.

Table 3. Predictive Validity of the Similarity Measure PFC of Different Experts

	EXP1_PFC	EXP2_PFC	EXP3_PFC	AVE12_PFC	AVE23_PFC	AVE13_PFC	AVE123_PFC
DK	.05	.18*	–.05	.10	–.04	.05	.10
PK	.22*	.20*	.15	.10	.11	.20*	.17
CK	.12	.26**	.22*	.10	.16	.25**	.20*
SK	–.11	.10	.08	–.01	.10	.11	.08
Total	.06	.24**	.16	.08	.13	.22*	.19*

*$p < .05$. **$p < .01$.

Results showed that experts did vary in terms of their predictive ability. Clearly, the second expert predicted four of the students' performance measures DK, PK, CK, and the total significantly while the others predicted only one to three measures. In the literature, there are studies that utilized a single expert network (Nash et al., 2006) and an averaged expert network (Trumpower & Goldsmith, 2004) as a referent structure. However, the important point is that "the validity of a referent structure is related to its ability to predict exam performance in computer programming courses" (Acton et al., 1994, p. 304). It appears that it is legitimate to select an expert referent based on its predictive ability on programming performance measures. Therefore, the referent structure provided by the second expert was chosen as the expert model for comparison purpose.

RESULTS

Descriptive Statistics

Means and standard deviations of the referent-free measure, referent-based measures, and programming performance measures are shown in Table 4. A glance at Table 4 shows that, in terms of predictive power, the referent-free and referent-based measures are similar. The participants performed better in the multiple-choice questions (DK and PK) as compared with the fill-in-the-blank questions (CK) and the program writing questions (SK).

Predictive Validity of the Referent-Free and Referent-Based Measures

In order to compare the predictive validity of the three referent-based similarity measures PRX, PFC, and GTD and the referent-free measure C, they were correlated with the programming performance measures DK, PK, CK, SK, and the

Table 4. Means and Standard Deviations of the Referent-Free Measure, Referent-Based Similarity Measures, and Programming Performance Measures

	M	*SD*
Referent-Free Measure		
C	0.13	0.29
Referent-Based Similarity Measures		
PRX	0.16	0.24
PFC	0.26	0.12
GTD	0.10	0.21
Programming Performance Measures		
DK	2.18	0.93
PK	2.44	1.42
CK	1.25	1.41
SK	1.82	2.44
Total	7.69	4.37

total. Table 5 presents the results of the correlations. Significant correlations were found between various similarity measures and programming performance measures. It is intriguing to note that each of the measures (PRX, PFC, GTD, and C) predicted four of the five programming performance measures (DK, PK, CK, SK, and total) significantly. It seems that PRX had the largest significant correlations with DK (.19), SK (.26), and the total (.31) among the four measures although the differences were mild.

To control for the effect of individual measure, partial correlations were calculated for the referent-based similarity measures and are shown in Table 6. No significant partial correlation existed between any similarity measures and performance measures when PRX was held constant. Significant partial correlations existed between PRX and SK (.27), GTD and SK (.20), and PRX and the total (.20) when PFC was held constant. When GTD was held constant, partial correlations between PRX and DK (.21) and PFC and DK (.20) were significant. Another pattern is when PRX was held constant, all the previous correlations decreased. When PFC was held constant, all the partial correlations between the other two similarity measures and performance measures decreased except for those between PRX and SK. For the case of GTD, except for those between PRX and DK and PFC and DK, all the other partial correlations decreased. These patterns further suggest that PRX had the highest predictive power among the three measures whereas PFC and GTD were similar in terms of predictive ability.

Table 5. Correlations between the Referent-Free Measure, Referent-Based Similarity Measures, and Programming Performance Measures

	DK	PK	CK	SK	Total
C	.11	.20*	.23**	.22*	.28**
PRX	.19*	.12	.27**	.26**	.31**
PFC	.18*	.20*	.26**	.10	.24**
GTD	.07	.18*	.30**	.20*	.28**

*$p < .05$. **$p < .01$.

Table 6. Partial Correlations between the Referent-Based Similarity Measures and Programming Performance Measures

	DK			PK			CK			SK			Total		
PRX	—	.09	.21*	—	–.04	–.02	—	.12	.07	—	.27**	.16	—	.20*	.16
PFC	.06	—	.20*	.17	—	.11	.09	—	.04	–.13	—	–.09	.02	—	.04
GTD	–.12	–.11	—	.13	.03	—	.15	.17	—	.01	.20*	—	.07	.15	—
Measure held constant	PRX	PFC	GTD	PRX	PFC	GTD	PRX	PFC	GTD	PRX	PFC	GTD	PRX	PFC	GTD

*$p < .05$. **$p < .01$.

DISCUSSION

As the predictive validity of both the referent-free and referent-based measures was more or less the same, it appears that the referent-free approach to concept mapping assessment could possibly be a viable alternative to the widely used referent-based approach. The former approach has its own advantages over the latter one since it can eliminate the disparity between experts (Acton et al., 1994) and is "most appropriate when instructional objectives call for individuals to organize content in a consistent fashion" (Davis et al., 2003, p. 203). Future research in the area of concept mapping assessment might explore the use of this referent-free approach in various settings.

Results also suggest that no matter whether the referent-based or referent-free measures were used, the correlations with the programming performance measures were quite low. Among the three similarity measures, PRX appeared to possess the best predictive ability and this is quite inconsistent with numerous studies in the literature which found the most predictive measure was PFC. For instance, Goldsmith et al. (1991) demonstrated that among the three indicators of similarity, PFC was most predictive of final course points (r = .74). This was followed by GTD (r = .66) and PRX (r = .61). Other studies also support the high predictive power of PFC compared with the other two (Lin & Yu, 2001; Yeh, 2001). However, it is in congruent with the findings by Tu (2001), which showed that PRX was the most predictive of natural science scores of sixth-graders. Whether the differences in terms of predictive power of various measures depend on factors like knowledge domain, grade level, performance measures used, and the number of concepts in the rating task are still unclear and these should be addressed in future studies. On the other hand, such differences in predictive power of various measures might be resolved by considering sex and band in this study.

Although the similarity measures PRX, PFC, and GTD predicted programming performance measures, the predictive power was in fact not as high as the figures reported in the literature (Acton et al., 1994; Kahler, 2001). One possible explanation could be due to the number of concepts used (Goldsmith et al., 1991). In general, predictive power is positively associated with the number of concepts used in the rating task. However, an inspection of the data revealed that there were participants with high scores but low similarities and vice versa. This might imply that they either relied on learning by rote with little conceptual understanding (high score but low similarity) or had conceptual understanding but it was not assessed in this test (low score but high similarity). From the assessment point of view, we argue that traditional assessment method is inadequate to assess student learning and results of this study provide a promising ground for the introduction of concept mapping assessment in programming education.

McCracken et al. (2001) contend that "To efficiently teach computer programming skills is difficult. The kinds of assessment that instructors use throughout

their courses must provide appropriate information for understanding students' processes of developing programming skill" (p. 134). It has been advocated that assessment should be more holistic, i.e., paper-and-pencil (traditional) and alternative (authentic). The main rationale for this is that traditional assessment provides little cues as to how students understand or misunderstand in learning since emphasis is usually placed on the outcomes instead of the process. To remedy this problem, Reeves (2000) suggests the use of alternative or authentic assessment. One of these techniques suggested is cognitive assessment which aims to measure students' higher-order thinking skills and it is achieved commonly by externalizing "the relationships they have made among concepts and processes within a domain and to reveal the structure of their knowledge" (p. 107). While there are many traditional assessment strategies, methods of cognitive assessment are still in its stage of infancy.

We suggest the following novel cognitive assessment method. With the aid of the PCKNOT software, this computer-assisted assessment method can "provide timely and specific information on the performance of each student which can be used for diagnosing areas where students have individual difficulties" (He & Tymms, 2005, p. 420). Apart from the traditional assessments such as multiple-choice tests, final papers, and projects, given that the referent-free and referent-based approaches yield similar predictive power of programming performance, the referent-free approach might be adopted to assess students' conceptual understanding of programming knowledge in complement with the traditional ones. Students who receive low coherence scores are considered to be those who are more "at risk." As coherence is a measure of internal consistency among concept relations, low coherence suggests that concepts are not consistently related in the knowledge domain and this points to the existence of misconceptions in learning. To rectify misconceptions, students would be asked to explain the semantic meaning of the relations among the concepts in their knowledge structures. Once misconceptions are uncovered, teachers can teach how to construct a "correct" structure explicitly through identifying correct links, incorrect links, and redundant links.

Regarding assessment and pedagogy in Computer Science education, Nash et al. (2006) argue from a study by Brown and Stanners (1983) that classroom intervention in a form of explicitly teaching conceptual structure accompanied by active engagement of students can bring about expertise in terms of higher similarity to a teacher's knowledge structure and gain in unit quiz grade. There are some other advocates of using concept maps in teaching computer programming (Keppens & Hay, 2008). In sum, this novel way of assessment offers authentic feedback to diagnose student learning, provide practical insights into teaching of computer programming, and achieves the goal of authentic assessment. It also helps students to think more like an expert. Eventually, it is hoped that this practice of assessment can help reform assessment in programming education.

The current study is believed to be a first attempt to compare the predictive validity of the referent-free and referee-based measures of the PSA on programming performance with an aim to inform programming assessment. However, it has several limitations that should be addressed in future research. First, the sample mainly consisted of middle to high ability students, and low ability students were under-represented. Future studies might consider a more balanced sample in terms of students' ability. Second, an even larger random sample size is required in order to generalize any results reliably. Third, it is worthwhile to compare the differences between the concept maps where students draw themselves and the maps that are generated by the PSA. Fourth, the reliability of the scale DK was only marginally acceptable and further refinement of the scale would be required. Finally, we selected to explore sorting algorithm in this study. Other algorithms or programming constructs should be considered to replicate the study and examine any differences in the results.

CONCLUSIONS

This study aims to investigate the predictive validity of measures of the PSA on programming performance. It showed that the correlations between the referent-free and referent-based measures with the programming performance measures were similar but not as high as those reported in the literature. Among the three similarity measures, PRX had the highest predictive power, whereas PFC and GTD had similar predictive ability. Implications of these results for assessment in computer programming are discussed. Although the correlations between the referent-free and referent-based measures with the programming performance measures were not particularly strong, this study represents a first step to advocate the use of concept mapping assessment in a Computer Science education setting. As relevant research becomes more mature, such results might contribute to the assessment reform in programming education.

APPENDIX A

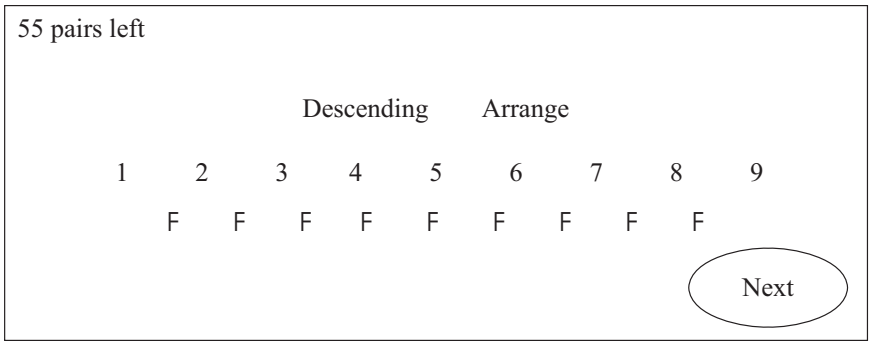
Items for the Assessment of Mental Model

Instructions

1. Eleven concepts related to the bubble-sorting algorithm are selected for this task and they are listed below:

computer	program	algorithm
sorting	arrange	correct order
pass	compare	swapping
ascending	descending	

2. In this task, you are required to judge the relatedness for each pair of concept. The suggested time for this task is 15 minutes, i.e., about 15 seconds for each pair. While you make your decision, you may use different criteria. For instance, the relatedness of each pair could be due to the existence of common properties between the concepts or the concepts under consideration being usually appeared together. Please also give your rating based on your first impression.
3. During the task, a pair of concepts will be shown along with its relatedness score from 1 to 9 in the screen and you need to enter your rating for each pair. The higher the score, the more related the concepts in a pair. In other words, a score of 8 or 9 represents high relatedness while a score of 1 represents low relatedness or even unrelated. As you enter your score, a marker will appear in that score and you may change your score by simply choosing another one. When you finish the rating of a pair, please click the next button and a new pair will be shown until the ratings of the 55 pairs are completed.



APPENDIX B

1. Sorting means
   a. to arrange data in an ascending order
   b. to arrange data in a descending order
   c. to arrange data in a certain specific order
   d. to arrange data in a random order
2. Sorting can be performed on data of type(s)
   a. integer
   b. character
   c. string
   d. all of the above
3. In general, sorting uses programming techniques of
   a. sequence and selection
   b. sequence and iteration
   c. selection and iteration
   d. sequence, selection and iteration

4. What is the output of the following program?

Pascal Version

```
Program sort;
uses wincrt;
var
a: array[1..4] of integer ;
pass, i, temp : integer;

begin
a[1]:=2;
a[2]:=1;
a[3]:=8;
a[4]:=9;
for i:=1 to 3 do
if   a[i] < a[i+1] then
begin
temp:=a[i]; a[i]:=a[i+1]; a[i+1]:=temp
end;
for i:=1 to 4 do write(a[i]);
end.
```

C Version

```
#include <stdio.h>
int main(){
int a[]= {2, 1, 8, 9};
int i, temp;
for (i=0; i<=2; i++){
if (a[i] <[i+1]){
temp=a[i]; a[i]=a[i+1]; a[i+1]=temp;
}}
for (i=0; I<=3; i++)printf("%d",a[i]);
system("PAUSE");
}
```

a. 1289
b. 2819
c. 2891
d. 1298

5. What is the output of the following program?

Pascal Version

```
Program sort;
uses wincrt;
var
a: array[1..4] of integer ;
pass, i, temp : integer;

begin
a[1]:=7;
a[2]:=5;
a[3]:=3;
a[4]:=2;
for i:=1 to 3 do
if   a[i] > a[i+1] then
begin
temp:=a[i]; a[i]:=a[i+1]; a[i+1]:=temp
end;
for i:=1 to 4 do write(a[i]);
end.
```

C Version

```
#include <stdio.h>
int main(){
int a[]= {7, 5, 3, 2};
int i, temp;
for (i=0; i<=2; i++){
if (a[i] <[i+1]){
temp=a[i]; a[i]=a[i+1]; a[i+1]=temp;
}}
for (i=0; i<=3; i++)printf("%d",a[i]);
system("PAUSE");
}
```

a. 5327
b. 2357
c. 5237
d. 3257

6. What is the output of the following program?

Pascal Version	C Version

Program sort;
uses wincrt;
var
a: array[1..4] of integer ;
pass, i, temp : integer;
 	
begin
a[1]:=1;
a[2]:=0;
a[3]:=1;
a[4]:=0;
for pass:=1 to 3 do
for i:=1 to 3 do
if a[i] >= a[i+1]+1 then
begin
temp:=a[i]; a[i]:=a[i+1]; a[i+1]:=temp
end;
for i:=1 to 4 do write(a[i]);
end.
	#include <stdio.h>
	int main(){
	int a[]= {1, 0, 1, 0};
	int pass, i, temp;
	for (pass=1; pass<=3; pass++){
	for (i=0; i<=2; i++){
	if (a[i] >=a[i+1]+1){
	temp=a[i]; a[i]=a[i+1]; a[i+1]=temp;
	}}}
	for (i=0; i<=3; i++)printf("%d",a[i]);
	system("PAUSE");
	}

a. 0101
b. 0110
c. 1100
d. 0011

7. What is the output of the following program?

Pascal Version	C Version

Program sort;
uses wincrt;
var
a: array[1..4] of integer ;
pass, i, temp : integer;
 	
begin
a[1]:=0;
a[2]:=1;
a[3]:=0;
a[4]:=1;
for pass:=1 to 3 do
for i:=1 to 4-pass do
if a[i] +1<= a[i+1] then
begin
temp:=a[i]; a[i]:=a[i+1]; a[i+1]:=temp
end;
for i:=1 to 4 do write(a[i]);
end.
	#include <stdio.h>
	int main(){
	int a[]= {0, 1, 0, 1};
	int pass, i, temp;
	for (pass=1; pass<=3; pass++){
	for (i=0; i<=3-pass; i++){
	if (a[i] <=a[i+1]){
	temp=a[i]; a[i]=a[i+1]; a[i+1]=temp;
	}}}
	for (i=0; i<=3; i++)printf("%d",a[i]);
	system("PAUSE");
	}

a. 1010
b. 1100
c. 1001
d. 0011

8-11. There are 40 students in a class. A computer teacher wants to write a program to arrange the names of students alphabetically. Fill in the blanks for the following program.

Pascal Version

```
Program sort;
uses wincrt;
var
name: array[1..40] of string ;
pass, i : integer;
temp: string;

begin
name[1]:='Au Wing Kay';
name[2]:= 'Chan Tai Man';
name[3]:= 'Au Lai Ling';
. . . . .
name[40]= 'Wong Wing Lun';;
for _____________________________ do
for _____________________________ do
if _____________________ then
begin
_____________________________
end;
for i:=1 to 40 do writeln(name[i]);
end.
```

C Version

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(){
char name[40][25]={"Au Wing Kay",
"Chan Tai Man", "Au Lai Ling",...,
"Wong Wing Lun"};
char temp[25];
int pass, i;
for _____________________________ {
for _____________________________ {
if _____________________________ {
_____________________________  }}}
for (i=0; i<=3; i++)printf("%s/n",name[i]);
system("PAUSE");
}
```

12. In a certain competition, only students who were born on or before 1989/1/1 are eligible to participate. You are required to write a program to sort students' records according to their date of birth from the eldest to the youngest. You may use "StudName" for the variable of student names and "DOB" for the variable of date of birth of students. This question will be assessed based on its syntax, semantic, and degree of completion. Try your best to write the program and don't leave it blank.

13. In a certain year, a company wants to find out the top 10 performing branches and reward them for their performance. You are required to write a program to sort sales records according to their sales volume from the greatest to the smallest. You may use "BranchName" for the variable of branch names and "SalesVol" for the variable of sales volume. This question will be assessed based on its syntax, semantic, and degree of completion. Try your best to write the program and don't leave it blank.

REFERENCES

Acton, W. H., Johnson, P. J., & Goldsmith, T. E. (1994). Structural knowledge assessment: Comparison of referent structures. *Journal of Educational Psychology, 86*(2), 303-311.

Birenbaum, M., Breuer, K., Cascallar, E., Dochyd, F., Dori, Y., Ridgway, J., et al. (2006). A learning integrated assessment system. *Educational Research Review, 1*(1), 61-67.

Brown, L. T., & Stanners, R. F. (1983). The assessment and modification of concept interrelationships. *Journal of Experimental Education, 52*, 11-21.

Chan, R. W. N. (2004). *Computer and information technology C programming*. Hong Kong: Radian Publishing Co.

Cooke, N. J. (1992). Predicting judgment time from measures of psychological proximity. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 18*(3), 640-653.

Cooke, N. J., & Schvaneveldt, R. W. (1988). Effects of computer programming experience on network representations of abstract programming concepts. *International Journal of Man-Machine Studies, 29*, 407-427.

Curtis, M. B., & Davis, M. A. (2003). Assessing knowledge structure in accounting education: An application of Pathfinder associative networks. *Journal of Accounting Education, 21*, 185-195.

Davis, M. A., Curtis, M. B., & Tschetter, J. D. (2003). Evaluating cognitive training outcomes: Validity and utility of structural knowledge assessment. *Journal of Business and Psychology, 18*(2), 191-206.

Dearholt, D. W., & Schvaneveldt, R. W. (1990). Properties of Pathfinder networks. In R. W. Schvaneveldt (Ed.), *Pathfinder associative networks: Studies in knowledge organisation* (pp. 1-30). Norwood, NJ: Ablex.

Earl, L., & Katz, S. (2006). Rethinking classroom assessment with purpose in mind: Assessment for learning, assessment as learning and assessment of learning. [Electronic Version]. Retrieved March 27, 2008 from www.wncp.ca/assessment/rethink.pdf

Fornell, C., & Larcker, D. F. (1981). Evaluating structural equation models with unobservable variables and measurement error. *Journal of Marketing Research, 18*, 39-50.

Fung, J., Lau, A., & Kai, S. (2003). *Certificate computer and information technology elective module (A): Algorithm and programming*. Hong Kong: Longman Hong Kong Education.

Goldsmith, T. E., & Davenport, D. M. (1990). Assessing structural similarity of graphs. In R. W. Schvaneveldt (Ed.), *Pathfinder associative networks: Studies in knowledge organisation* (pp. 75-87). Norwood, NJ: Ablex.

Goldsmith, T. E., Johnson, P. J., & Acton, W. H. (1991). Assessing structural knowledge. *Journal of Educational Psychology, 83*(1), 88-96.

Gomez, R. L., Hadfield, O. D., & Housner, L. D. (1996). Conceptual maps and simulated teaching episodes as indicators of competence in teaching elementary mathematics. *Journal of Educational Psychology, 88*(3), 572-585.

Gonzalvo, P., Canas, J. J., & Bajo, M. T. (1994). Structural representations in knowledge acquisition. *Journal of Educational Psychology, 86*(4), 601-616.

He, Q., & Tymms, P. (2005). A computer-assisted test design and diagnosis system for use by classroom teachers. *Journal of Computer Assisted Learning 21*(6), 419-429.

Housner, L. D., Gomez, R. L., & Griffey, D. C. (1993a). A Pathfinder analysis of pedagogical knowledge structures: A follow-up investigation. *Research Quarterly for Exercise and Sport, 64*(3), 291-299.

Housner, L. D., Gomez, R. L., & Griffey, D. C. (1993b). Pedagogical knowledge structures in prospective teachers: Relationships to performance in a teaching methodology course. *Research Quarterly for Exercise and Sport, 64*(2), 167-177.

Johnson, P., Goldsmith, T., & Teague, K. (1994). Locus of the predictive advantage in Pathfinder-based representations of classroom knowledge. *Journal of Educational Psychology, 86*(4), 617-626.

Jonassen, D. H. (1995). *Operationalizing mental models: Strategies for assessing mental models to support meaningful learning and design-supportive learning environments.* Paper presented at the Computer Support for Collaborative Learning (CSCL) Conference, Mahwah, NJ.

Kahler, S. (2001). *A comparison of knowledge acquisition methods for the elicitation of procedural mental models.* Unpublished PhD dissertation, North Carolina State University.

Keppens, J., & Hay, D. (2008). Concept map assessment for teaching computer programming. *Computer Science Education, 18*(1), 31-42.

Lin, H. F., & Yu, M. N. (2001). An evaluative research study on learning the mathematics concept of algebra by high school students—Using the quadratic equation as an example [in Chinese]. *Journal of Education and Psychology, 24*(2), 303-326.

Lin, Y. S. (2002). *The effects of self-explanation on learning programming IF statement* [in Chinese]. Unpublished Master's thesis, National Taiwan Normal University, Taipei, Taiwan.

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B.-D., et al. (2001). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. *SIGCSE Bulletin, 33*(4), 125-180.

Nash, J. G., Bravaco, R. J., & Simonson, S. (2006). Assessing knowledge change in Computer Science. *Computer Science Education, 16*(1), 37-51.

Nunnally, J. C. (1978). *Psychometric theory* (2nd ed.). New York: McGraw-Hill.

Oliver, R. (1993). Measuring hierarchical levels of programming knowledge. *Journal of Educational Computing Research, 9*(3), 299-312.

Reese, D. D. (2003). *PFNET translation a tool for concept map quantification.* Paper presented at the Annual Conference of the Association for Educational Communications and Technology, Anaheim, CA.

Reeves, T. C. (2000). Alternative assessment approaches for online learning environments in higher education. *Journal of Educational Computing Research, 23*(1), 101-111.

Schau, C., Mattern, N., Zeilik, M., Teague, K. W., & Weber, R. J. (2001). Select-and-fill-in concept map scores as a measure of students' connected understanding of science. *Educational and Psychological Measurement, 61*(1), 136-158.

Schvaneveldt, R. W. (1990). *Pathfinder associative networks: Studies in knowledge organization*. Norwood, NJ: Ablex Publishing.

Schvaneveldt, R. W., Dearholt, D. W., & Durso, F. T. (1988). Graph theoretic foundations of Pathfinder networks. *Computers and Mathematics with Applications, 15*(4), 337-345.

Talento-Miller, E., & Rudner, L. M. (2008). The validity of Graduate Management Admission Test scores: A summary of studies conducted from 1997 to 2004. *Educational and Psychological Measurement, 68*(1), 129-138.

Trumpower, D. L., & Goldsmith, T. E. (2004). Structural enhancement of learning. *Contemporary Educational Psychology, 29*, 426-446.

Tu, C. Y. (2001). The application of the Pathfinder network on assessment: The example of learning about the concepts of astronomy for sixth-grade students [in Chinese]. *Journal of Education and Psychology, 24*(2), 367-391.

Winslow, L. E. (1996). Programming pedagogy—A psychological overview. *SIGCSE Bulletin 28*(3), 17-22.

Woo, M. H. C., Shiu, Y. C., & Wang, B. S. K. (2003). *Computer and information technology for HKCEE module A1: Algorithm and programming using C*. Hong Kong: Digital Vision Educational Publishing Company.

Yeh, C. H. (2001). An analysis of knowledge structures by the Pathfinder method—Freshmen academic achievement in a psychology class [in Chinese]. *Journal of Education and Psychology, 24*(2), 421-450.

Direct reprint requests to:

Wilfred W. F. Lau
Faculty of Education
The University of Hong Kong
Hong Kong
e-mail: wilfredlau@graduate.hku.hk