HKUST SPD - INSTITUTIONAL REPOSITORY

SeqDynamics: Visual Analytics for Evaluating Online Problem-solving Dynamics

Meng Xia, Min Xu, Chuan-en Lin, Ta Ying Cheng, Huamin Qu, Xiaojuan Ma

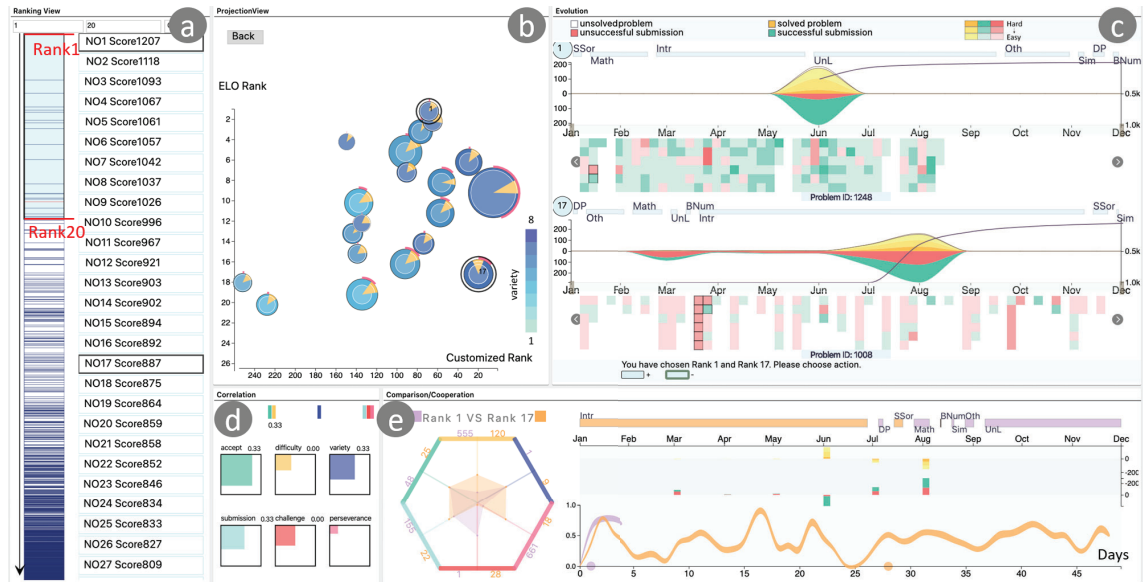The Hong Kong University of Science and Technology, Hong Kong

Figure 1: *SeqDynamics facilitates the evaluation of learners' problem-solving dynamics via levels of analysis: (a) Ranking View displays the overall performance (macro level), (b) Projection View plots a subset of learners with key learning attributes (meso level), (c) Evolution View and (e) Comparison/Cooperation View magnify the details of a problem-solving sequence unfolded over time (micro level). Correlation View (d) displays the correlation of different learning attributes with the performance ranking and enables users to customize the weights.*

Abstract

Problem-solving dynamics refers to the process of solving a series of problems over time, from which a student's cognitive skills and non-cognitive traits and behaviors can be inferred. For example, we can derive a student's learning curve (an indicator of cognitive skill) from the changes in the difficulty level of problems solved, or derive a student's self-regulation patterns (an example of non-cognitive traits and behaviors) based on the problem-solving frequency over time. Few studies provide an integrated overview of both aspects by unfolding the problem-solving process. In this paper, we present a visual analytics system named SeqDynamics that evaluates students' problem-solving dynamics from both cognitive and non-cognitive perspectives. The system visualizes the chronological sequence of learners' problem-solving behavior through a set of novel visual designs and coordinated contextual views, enabling users to compare and evaluate problem-solving dynamics on multiple scales. We present three scenarios to demonstrate the usefulness of SeqDynamics on a real-world dataset which consists of thousands of problem-solving traces. We also conduct five expert interviews to show that SeqDynamics enhances domain experts' understanding of learning behavior sequences and assists them in completing evaluation tasks efficiently.

CCS Concepts

● *Human-centered computing* → *Visual analytics;* ● *Applied computing* → *E-learning;*

1. Introduction

The evaluation of students' cognitive skills and non-cognitive traits and behaviors provide critical implications for designing student recruitment policies and personalizing tutorial instructions [Far03, Ber17, MHB19, JGP*19, SQG*19]. Cognitive skills refer to the core skills which the brain uses to think, read, learn, remember, reason, and pay attention, while noncognitive traits and behaviors refer to specific characteristics such as motivation, conscientiousness, perseverance, self-regulation, and collaboration [Far03]. We can derive people's cognitive skills from their problem-solving outcomes [May92], which is why entrance exams and case tests are widely used in education and job recruitment processes [Ber17]. In contrast, noncognitive traits and behaviors (e.g., self-regulation), deeply valued in recruitment, are difficult to identify by an exam [Far03]. For example, perseverance or self-regulation can only be determined after monitoring students' practices (e.g., frequency, number of attempts, levels of difficulty attempted, etc.) for an extended period. In this paper, we define the process and progress of solving a series of problems over time as problem-solving dynamics, from which students' cognitive skills and noncognitive traits and behaviors can be inferred [PL04].

Existing established educational platforms such as Khan Academy and concurrent online learning platforms including online question pools (e.g., LeetCode [Lee19]) and intelligent tutoring systems (e.g., SimStudent [Sim19]) have collected substantial fine-grained data regarding how students approach and perform in quizzes and assignments [PBH*15], thus providing an opportunity to evaluate learners' cognitive and noncognitive behaviors.

However, evaluating learners' problem-solving dynamics is challenging. First, each learner's problem-solving dynamics is ultimately multi-dimensional time-series data and how to represent and interpret such data with meaningful semantics to reflect both cognitive and noncognitive traits and behaviors is non-trivial. Second, the evaluation process usually involves the comparison among numerous learners' problem-solving sequences. Summarizing students' problem-solving dynamics in different levels of detail to facilitate convenient comparison may be difficult. Third, the absence of a standard definition of "good" problem-solving dynamics hinders the automatic measurement of students' learning behavior [Ber17]. Different instructors perceive the value of various problem-solving behaviors differently. Most studies on problem-solving analysis focus on modeling the process from the cognitive perspective (e.g., Bayesian Knowledge Tracing [CA94] and Deep Knowledge Tracing [PBH*15]). Few studies take the learners' problem choices and attempts along the way into consideration.

In this paper, we introduce *SeqDynamics*, an interactive visual analytics system for instructors to inspect and evaluate learners' problem-solving dynamics comprehensively. *SeqDynamics* automatically summarizes and analyzes problem-solving dynamics from three levels of detail: the learner ranking, the synoptic problem-solving features (the number of problems solved, the percentage of hard problems solved, the variety of problem type, the number of submissions, the time starting to try hard problems, and the percentage of active days), and the changes to these features over time. Multiple views are provided to represent the analysis results for instructors to assess and compare at three scales: ranking

view (the macro level), showing the overall problem-solving performance distribution to facilitate instructors in easily distinguishing the threshold of promising and unpromising candidates; projection view (the meso level), projecting the focused range of learners on 2D coordinates to compare their major problem-solving features and identify the outstanding learners; evolution view and comparison/cooperation view (the micro level), inspecting the detailed problem-solving dynamics, comparing learners with similar synoptic features or grouping learners with complementary features. *SeqDynamics* is equipped with rich and flexible interactions to enable customized and responsive exploration. In particular, we propose two novel visual designs to represent problem-solving dynamics with semantics: a glyph to show the learners' synoptic problem-solving skills from both cognitive and non-cognitive perspectives; a bi-lateral stacked graph with a heatmap-like view to magnify the details of a problem-solving sequence unfolding over time. The main contributions of this work are summarized as follows:

- **Interactive System -** An interactive visual analytics system for instructors to evaluate learners' problem-solving dynamics and select candidates on multiples scales (macro-meso-micro).
- **Visualization Designs -** A set of novel visual designs to represent problem-solving dynamics from both cognitive and non-cognitive facets to facilitate understanding and comparison of problem-solving sequences.
- **Evaluation -** Three scenarios demonstrating the usefulness of *SeqDynamics* on a real-world dataset and five expert interviews showing that *SeqDynamics* enhances their evaluation processes.

2. Related Work

This section reviews the prior literature on problem-solving behavior analysis, visual analytics of learning sequences, and time-series visualizations and comparisons.

2.1. Problem-Solving Behavior Analysis

Past analyses of problem-solving behaviors are usually conducted on cognitive skills [LK17]. Bayesian knowledge tracing was first proposed to build procedural models for problem-solving processes [CA94]. It assumed fixed and independent concepts and modeled learners' latent knowledge as binary variables that represent the understanding/non-understanding of a single concept. Owing to the unrealistic assumption that learning concepts are independent of one another, other dynamic probabilistic models have been proposed. For example, Learning Factor Analysis [CKJ06] modeled learner knowledge states via logistic regression to deal with learning concepts at different levels, and Performance Factor Analysis [PJCK09] further considers learners' responses to the learning concepts. However, all these models require accurate concept labeling, which is usually difficult to obtain. There are also other models such as deep knowledge tracing that uses Recurrent Neural Networks to model and predict learner performance in solving problems [PBH*15]. However, it lacks interpretation, which limits instructors to turn it into actionable instructions [LK17]. In addition, the long-term learning attributes related with noncognitive traits and behaviors are not well addressed in the context of online problem-solving behaviors.

2.2. Visual Analytics of Learning Sequences

Visual analytics has been widely and effectively applied in learning sequences analysis as MOOCs (Massive Open Online Courses) become more prevalent. Some of the studies were designed for instructors to better understand students' learning sequences. DropoutSeer [CCZ*16] visualized the clickstream behavior and the assignment performance of different learner groups along the timeline for instructors to better understand the prediction of the dropout in MOOCs learning. ViSeq [CYP*18] visualized students learning sequences of MOOCs learning, such as watching videos, checking problems, and attending forum discussions. They found some general patterns such as students with high scores usually started to review previous lectures two weeks before an exam. MOOCad [MXC*19] focused particularly on the visual analysis of anomalous learning activities. Based on the unsupervised event sequence clustering algorithm, their visual system could assist instructors to find unexpected anomaly learning activities. Other works focus on visualizing the individual learning progress for self-regulation [PÁMMPS18, WLS*14].

However, the data and tasks of those systems are different from our scenarios. We focus on problem-solving dynamics, the students' performance and timestamp for each question along the way. PeerLens [XSW*19] visualized the same type of data for personalized learning path design, but no evaluation tasks (e.g., comparison, ranking) of different problem-solving sequences were supported.

2.3. Time-series Data Visualization and Comparison

Numerous visual techniques on analyzing time-series data have been summarized in the surveys [AMST11, BDA*14]. A straightforward way to visualize time-series events is to place events along a horizontal time axis. This technique has been adopted by many systems such as Lifelines [PMR*96], CloudLines [KBK11] and TimqueSlice [ZCCB13]. Each attribute of the time-series can be further illustrated using a stacked graph layout [HHWN02]. Another way to show a time event sequence is by using a circular graph [FFM12] or a spiral layout to represent periodical temporal data [DH02].

These encoding methods work well for individual-level exploration on presenting the exact sequence of events, however, sometimes it is necessary to show multiple records at the same time for comparison. EventAction [DPSS16] used a calendar view to show time-event sequences and placed them in a ranking list to compare different sequences. MatrixWave [ZLD*15] was designed to compare two event sequences using a matrix view with explicit encoding. We are inspired by these works and propose our novel designs in this paper from the cognitive and noncognitive perspectives.

3. System Overview

Our system was designed to meet the real-world requirements for instructors or recruiters to evaluate learners' problem-solving dynamics (e.g., elite selection). We interviewed two domain experts (E1, E2) in programmer recruiting and two university instructors (I1, I2) to survey their detailed requirements from their practices on recruiting programmers and evaluating students. Each interview
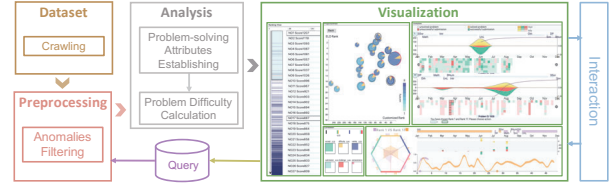

Figure 2: *System overview. It contains four modules: data preprocessing, analysis, visualization, and interaction.*

lasted about one hour. E1 is a coach and recruiter for a competitive programming team in a local university, who has rich domain knowledge in competitive programming. E2, a coach and recruiter, is a founder of an online judge platform with a comprehensive understanding of the dataset adopted by our system context. I1 and I2 are professors who are teaching programming courses. We list the following primary design requirements (R1-R4) derived from the interviews that guided the system designs.

R1: Shows a clear overview of overall students' performance Experts need a comprehensive score and rank of students at the beginning for reference. Based on that, they can have a rough idea of all students' performance and pinpoint a group of students of interest for further evaluation.

R2: Understand problem-solving dynamics from different perspectives over time. The system needs to incorporate useful visual designs to help instructors understand the major problem-solving features in terms of cognitive skills and noncognitive traits. Since some learning characteristics can only be reflected over a relatively long period of time, visualizations should be designed to reveal the students' learning characteristics over time.

R3: Compare/Combine the problem-solving performance on different scales. The system should support the comparison of candidates on different scales. For example, by observing the overall performance, instructors can set some basic rules to filter a group of candidates. According to the students' synoptic problem-solving features (e.g, number of problems solved, the fraction of hard problems, etc.), they can further narrow down the searching range. For candidates that have similar overall performance and major features, instructors need to check detailed problem-solving sequences to better select the best of the best. Experts also hope to see the combination results of learners for grouping potential teams.

R4: Support an interactive and customized exploration of the evaluation. The system enables instructors to customize their own preferences on different learning attributes and filtering rules to generate the exploration results accordingly. In addition, highlighting and updating users' selections simultaneously in multiple coordinated views to achieve consistency are also needed.

Based on these requirements, we have designed *SeqDynamics* to visually represent problem-solving dynamics and yield an improved evaluation of problem-solving behaviors. Fig. 2 illustrates the system architecture, which contains four modules: (1) data collection and preprocessing module collects and preprocesses the raw data to problem-solving sequences indexed by learner ID; (2) analysis module calculates problem-solving attributes and learners'

ranking (R1); (3) visualization module uses multiple coordinated views to support interpretation, comparison , and composition of different problem-solving dynamics in context (R2, R3); and (4) interaction provides responsive feedback to users' interaction and facilitates exploration (R4).

In our application scenario, the system models the context of recruiting and training programmers to prepare for an International Collegiate Programming Contest (ICPC). We make use of the program problem-solving records in a popular online judge [hdu19], with the owner's consent. We focus on the recent problem-solving records of learners who registered after 2017, which consist of 2,538,647 records from 53,535 learners and 5,166 programming questions. Each record includes learner ID, problem ID, judge status, and submission time. To enrich the context information and better evaluate students' learning, we search other online judges and blogs to collect problem types for each problem (e.g., dynamic programming, graph, etc.). During the process, 102 abnormal accounts (too many submissions within one second) are eliminated.

4. Problem-Solving Dynamics Analysis

4.1. Establishing Problem-Solving Attributes

We establish problem-solving attributes from both the cognitive and non-cognitive perspectives. Based on previous research [ANH*68] and domain experts' suggestions, we consider three learning attributes to evaluate individual's cognitive skills: the number of problems solved (l_1), the percentage of hard problems solved (l_2), and the diversity of problems solved (l_3). In addition, we consider three learning attributes related to noncognitive traits and behaviors: diligence, willingness to take on challenges, and perseverance. Previous research [Far03] summarized a series of non-cognitive skills from two aspects, conscientious work habits (e.g., perseverance, discipline) and other personality traits (e.g., opening to new experience, aggressive, etc). The roles of those skills in schooling and employment outcomes had been examined. We discussed these attributes one by one with domain experts to check how they are valued and measured in our scenario. We concluded that the total number of submissions (l_4) can reveal whether a student has expended effort on the programming platform, which we recognize as diligence. The time starting to solve hard problems (l_5) shows when a student has adapted to the learning process and is confident enough to try new problems. This indicates one's willingness to take on challenges and is highly emphasized by the experts when selecting the best candidates. l_6, calculated by #(active days)/ #(days after registration), uncovers a student's ability from the time perspective, which we called perseverance, to differentiate good candidates from other students who attempt many problems with enthusiasm only over a short period.

4.2. Problem Difficulty and Learner Ranking

According to domain experts (E1, E2), problem difficulty is a key feature in our context when evaluating and selecting learners, based on which the students' ranking is defined. The current method of determining the rank of a learner is purely based on the number of problems he/she solves [RML08]. However, a student who solves

ten easy questions is different from a student who solves ten difficult ones. Inspired by the ELO ranking algorithm used in areas (e.g., chess, education, work allocation) to define the relative ranking dynamically [GTY, MHB19, Goo17], we modify it to calculate the learner ranking.

The basic idea is to assign ratings based on the expected winning probability of two objects' competing after a match. In our case, we define a *match* as a learner's attempt(s) of solving a question, in which the learner and the problem are treated as opponents. First, before each match, we obtain the expected win probability of an individual p_i and a problem p_p:

$$p_i = \frac{1}{1+a^{(r_i-r_p)/b}} \qquad p_p = \frac{1}{1+a^{(r_p-r_i)/b}}$$

where r_i, r_p are the ratings of a learner and a problem before the match, initialized as 0. The choice of a and b can be chosen freely according to the intended scale of score difference and we follow the original setting as in [Elo78] to use 10, 400 respectively.

We then define the actual competing result of an individual s_i based on the number of submissions he or she has attempted. If attempts $== 0, s_i = 0$; if attempts $<= 2, s_i = 1$; if attempts $> 2, s_i = 0.5$. Accordingly, the score of the problem s_p after is $s_p = 1 - s_i$. A score 1, 0, and 0.5 means victory, loss, and draw, respectively.

After a match, the ratings of the author and the problem are updated through the following:

$$r_i = r_i + k(s_i - p_i) \qquad r_p = r_p + k(s_p - p_p)$$

where k is set to 0.5, a fairly small number that suggests no significant changes in ranking would be made by one learner. The above equations are reiterated based on all the data.

However, the equation above suggests that questions that have never been answered would obtain high ratings. This result is fair when the question is extremely hard and no one has attempted it, yet becomes unreasonable when the question is new and no learner has attempted it. To balance these anomalies, we refine the final updated difficulty scoreD_p that guarantees the score to remain 0 for problems never been attempted: $D_p = 0.5r_p + 0.5D_h$, where r_p is already converged after the iteration and D_h is computed through $D_h = \frac{\sum Interaction\ Score}{Number\ of\ Learners}$.

In the equation, Interaction Score is calculated based on the proficiency and interactions with the questions of the learner. If a more proficient learner cannot solve a problem, this behavior contributes more to the conclusion that the problem is hard than a less proficient learner in the same situation. All the learners are put into a histogram based on the number of problems solved. We then use the distribution to split users into five sectors and give learners a proficiency score a_l from 5 to 1 (5 being the highest). According to domain experts, the difference in the difficulty of a question is very dramatic, and thus we differentiate the problem with different orders of magnitude a_p: a question that has never been solved was given a score 10, solved within two tries was given 1, and solved but with more attempts were given 5. Based on these, then *Interaction Score* $= a_l a_p$. Finally, we have difficulty score D_p for each student and problem.

Validation: Since there is no ground truth for the problem difficulty and the learner ranking, we asked two experienced students

ELO	8	11	14	17	20	37	46	48	70	101	156
Baseline	10	24	50	33	51	30	132	115	175	400	210

Table 1: *The medalists' rankings of ELO algorithm and Baseline using the number of problems finished.*

who solved more than 500 problems to label their presumed difficulty of 100 random questions (easy, medium, and hard). The initial labelling was performed independently, then the two students were asked to sit together and discuss the difficulty. Final labels of the problems were set after the discussion, and we retrieved an overall of 18 questions labelled as "easy", 54 questions as "medium", and 28 as "hard". Our algorithm can differentiate easy problems and hard problems accurately with all 18 questions labelled as "easy" getting a score lower than 0.5 and 83% of 28 questions labelled as "difficult" obtaining a score higher than 0.65. For the 54 problems labelled "medium", 90% got a score between 0.5 and 0.65. The reason that the medium and the high is not perfectly matched may be that some problems are difficult to judge whether the level is medium or hard according to the experts.

As for the learner ranking, we compared our algorithm with the original ranking using the number of problems solved to see how the rankings of medalists in the international coding competitions rank differently in the two schemes. Since our modified ELO aims to highlight the top-performing candidates (a higher chance to win), an algorithm that ranks candidates higher is preferred. We used the data of 2013 in which the username of 11 ACM medalists were known (not listed here for confidentiality). As shown in Tab. 1, for most of the medalists, ELO gave a higher and more focused range of the ranking than the baseline. In summary, the ELO algorithm can give a plausible problem difficulty and learner ranking.

5. Design Tasks

Based on the problem-solving dynamics analysis, we have derived design tasks (T1-T6) to meet the design requirements (R1-R4).

T1 Show the overall problem-solving performance distribution of learners. The visual design should clearly show the overall performance distribution of all the learners calculated by the ELO algorithm to facilitate instructors evaluate the overall performance distribution at a glance and easily distinguish the threshold of promising and unpromising candidates (**R1**).

T2 Interpret the major learning attributes of a student from cognitive and noncognitive aspects intuitively. The visualization should form a clear representation of each student's major problem-solving attributes (e.g. number of problems solved, when starting to try hard problems, etc.). The instructor can then understand a student's talents and efforts (**R2**).

T3 Facilitate the comparison of students based on their major problem-solving attributes. The visualization designs should support the comparison among different students in the major learning attributes, demonstrating their differences and highlight the outstanding students from the group (**R3**).

T4 Analyze time-series problem-solving dynamics of students in detail. To evaluate problem-solving behaviors further, vi-

sualization should support the user to inspect the changes of each problem-solving attribute over time to infer the learning curve as well as the work ethic of each candidate (**R2**). For example, some learners solve problems during a long period regularly while the other solve problems intensively over a short period.

T5 Facilitate the detailed comparison and complementarity of problem-solving dynamics. When evaluating students with similar synoptic learning attributes, instructors need to further compare their problem-solving dynamics in detail to make the final choice. When considering teamwork or group learning, instructors hope to group learners with complementary skills. Thus, the visualization should show the differences or combined results of problem-solving dynamics (**R3**).

T6 Inspect candidates from different perspectives by adjusting the attribute weights. Since different instructors value different learning attributes of students, the visual designs should provide a way to adjust the weights of different learning attributes and update the results accordingly. For example, one emphasizes a creative mindset while others value more about perseverance (**R4**).

6. Visual Design

To address the aforementioned tasks, we present a novel visual analytics system for instructors to evaluate the problem-solving dynamics and select the best candidates from the candidate pool. The visual analysis module of *SeqDynamics* includes: 1) **Ranking View** (Fig. 1a) that displays an overall distribution of the learners' scores and ELO ranking in an ascending order (macro); 2) **Projection View** (Fig. 1b) and **Correlation Panel** (Fig. 1d) that facilitate instructors to customize their own evaluation criteria and compare the synoptic problem-solving features of a subset candidates on a 2D canvas. 3) **Evolution View** (Fig. 1c) that expands the selected learners' problem-solving features over time for detailed inspection (micro); 4) **Comparison/Cooperation View** (Fig. 1e) that facilitates explicit comparisons of two problem-solving sequences and demonstrates the complementarity of two learners. A collection of interactions, such as querying, highlighting, tooltips, and brushing, is also available for users to explore the dataset freely.

6.1. Ranking View

The Ranking View (Fig. 1a) aims to provide a macro-level view of all learners' performance distributions to facilitate instructors in evaluating the overall performance distribution at a glance and easily distinguish the threshold of promising and unpromising candidates (**T1**). The left side is a vertical distribution graph of the ELO ranking of all learners. Each learner is encoded by a horizontal bar and the y-axis represents the ELO score, descending from the top to the bottom. From the example dataset, the distribution graph starts with the highest-ranking learner (rank 1, score 1207) at the top all the way down to the lowest-ranking learner (rank 1,000, score 162). The vertical gaps between bars imply the absolute score differences between learners. This design effectively encodes both the rankings of learners and their score gaps so users may acknowledge that some learners have similar rankings but large score differences. When users assign a ranking range, a blue mask from the first to the last learner of the queried range is highlighted in the

distribution graph. The right side of the distribution graph is a vertical list of all the learners' ELO scores sorted in ascending order of the rank. Each rectangular box labeled with the learner's ID and an absolute score represents one learner. When a rectangle is selected, the rectangle along with the bar on the distribution graph and the glyph in Projection View corresponding to the learner are also highlighted.

6.2. Projection View and Correlation Panel

The Projection View (Fig. 1b) aims at providing a meso-level view of a subset of queried learners to allow instructors to observe their similarities and differences in problem-solving attributes (**T2, T3**). As summarized in R4, instructors have different preferences on those attributes. Thus we design the projection view with a 2D layout to place queried learners. The y-dimension represents the ranking based on students' performances and the x-dimension indicates the customized ranking based on the weights of cognitive and non-cognitive attributes as tuned by the instructor. In addition, we design a novel glyph to encode each learner's problem-solving attributes and plot them to a 2D canvas for easy comparison. The layout is generated by the adjustable weights of learner attributes in the Correlation Panel.

Correlation Panel. The Correlation Panel (Fig. 1d) has two functions: 1) showing the correlation between the ELO ranking and different problem-solving attributes; 2) providing a way for instructors to customize the weights assigned to different attributes. As shown in Fig. 1, we use six squares to represent six problem-solving attributes calculated in Section 4.1. The percentage of the colored area to the whole area of the square indicates the correlation between ELO ranking and the ranking based on that particular attribute. We use the Spearman correlation since it is a nonparametric measure of correlation [Zar05]. For example, we can see that the ranking based on "accept" (i.e., number of problems solved) has the highest correlation with the ELO ranking. The bars on top of the six squares are used to adjust the weights of different problem-solving attributes to calculate a composite rank, which determines the x position of each learner in the projection view (Fig. 1b). Users may drag the bars to assign different weights to different problem-solving attributes.

Projection Layout. The learners' glyphs are projected onto a two-dimensional space of x and y (Fig. 1b). The y-axis maps the ELO ranking of each learner where a higher position means a higher ranking. The x-axis ranks each learner based on a composite rank of weighted attributes where the further right the higher the composite rank. For example, if the user gives a large weight to the submission attribute and relatively small weights to the other attributes, learners with numerous submissions will generally align toward the right. We implemented the aforementioned two axes for glyph projection instead of performing clustering, since in clustering such as MDS [BG03], the contribution degree of each attribute (dimension) is obscured, making the mapping from the results of MDS too much of a black box to be useful.

Learner Glyph. After the learners are projected onto Projection View, we further facilitate comparison at a glance through intuitive glyphs encoded by six attributes (the number of problems solved,
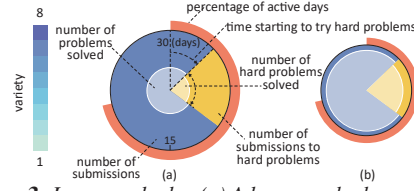

Figure 3: *Learner glyphs: (a) A learner who has many submissions but solves a few problems; (b) A learner who has relatively fewer submissions and solves more problems.*
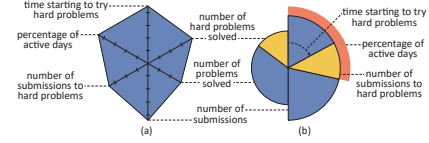

Figure 4: *Two design alternatives for learner glyph.*

the percentage of hard problems solved, the variety of problems solved, the number of submissions, the time to try hard problems, the percentage of active days). We encode learners' glyphs based on the principle of expressiveness and effectiveness. Expressiveness means it can express the major cognitive and noncognitive features; effectiveness represents that the visual patterns of similarities and differences are easily identifiable (**T2**).

Accordingly, our glyph design is constructed from six elements (Fig. 3). (1) The area of the outer circle denotes the absolute number of submissions, where a large circle denotes a larger number of submissions. (2) The area of the inner circle, which is a semi-transparent white overlay, denotes the absolute number of problems solved. The number of problems solved is encoded as a circular subset of the number of submissions because it is mathematically sensible and intuitive for comparison. For example, a learner with a large number of submissions and a small number of accepted questions will have a large outer circle and a small inner circle (Fig. 3a), which can be interpreted as a studious student. In contrast, a learner with a few submissions and a relatively large number of accepted questions will have a small outer circle with an inner circle almost as large as the outer circle (Fig. 3b), which can be interpreted as a potential talent student. (3) The angle of the sector denotes the percentage of submissions to hard problems out of all the submissions, which can reflect the quality of the problems the student solved. (4) The color shading of the outer circle denotes variety. Variety is scaled from 1 to 8, where a value closer to 1 means is shaded a lighter blue, while a value closer to 8 is shaded a darker blue. The values are mapped to the number of problem types that the learner has completed. Therefore, a semi-transparent overlay instead of a solid color fill is used for the encoding of the inner circle so that variety may be encoded across the entire glyph for clarity instead of only in the outer circle, which is inspired by Scheepens *et al.* [SvdWvW14]. (5) The north direction represents the first day that the learner submits a problem and the first side clockwise represents the first day the learner submits a hard problem. The angle between denotes the days taken by the learner to try the first hard problem. (6) The outer radial bar chart drawn from the north represents the percentage of active days of a learner or the degree to which a learner persists in practicing on the platform.
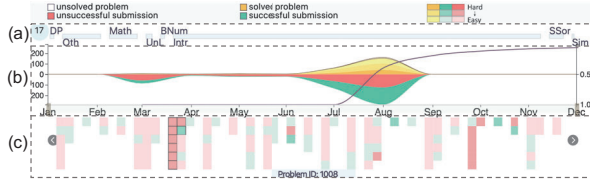
Figure 5: *Evolution view. (a) Problem type bar. (b) Bilateral stacked graph. The upstream has four layers (unsolved, easy, medium, hard problems); the downstream has two layers (unsuccessful and successful submissions). (c) Submission details panel. Each tile represents a submission.*

Glyph Alternatives. The learner glyph is designed through many iterations of feedback and refinement with the four experts, involved in our design process. Over the process, two main alternative designs were seriously considered yet abandoned for various reasons. The first alternative design is a radar chart which encodes the six problem-solving attributes on six different axes (Fig. 4a). However, it is not intuitive enough to compare the number of submissions and the number of problems solved.

In the second alternative design (Fig. 4b), we attempted to enhance the pattern recognition capability and endow meaning to encoding. The encoding is similar to the final design except that the inner and outer circles were each condensed to semi-circles, with the left half representing solved problems and the right half representing submissions. However, this design was also rejected, for two main reasons. First, a comparison of the left and right sides shows both radius and area, which is confusing. Second, the inner sections (# of hard problems solved and # of hard problems) are not aligned in any dimension, which is also hard to compare.

6.3. Evolution View

The Evolution View (Fig. 1c) aims at providing a micro-level view of unfolded problem-solving dynamics of focused learners (**T4**).

Bilateral Stacked Graph. The evolution View (Fig. 5b) uses a bilateral stacked graph for users to further inspect time-series information of synoptic features (**T4**), with an accuracy of per day/per month over a total time span of 20 months (dataset length). To echo the glyph design, the upstream area is represented as the inner circle of the learner glyph and the downstream area is represented as the outer circle. The upstream area represents cognitive capability, is mapped to the number of all attempted problems, and segmented by four levels of problem difficulty (from the up to the down): unsolved, easy, medium, and hard. The downstream represents the non-cognitive capabilities, which are mapped to the number of submissions and segmented by two types of submissions: successful and unsuccessful. Green-colored submissions represent the correct answer and red-colored submissions mean the wrong answers. Hovering over each segment highlights it and displays a tooltip of the number of problems or submissions accordingly.

The time-series nature of the bilateral stacked graph enables new levels of insight such as learners increasingly attempting (and correctly answering) a higher proportion of difficult questions and submission frequency peaking during summer vacation or before
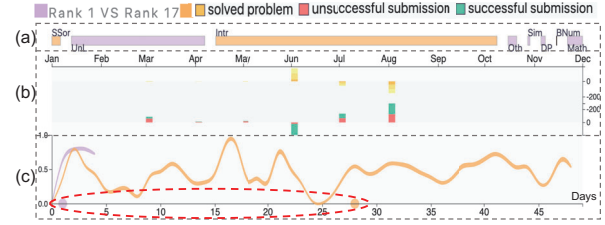

Figure 6: *Comparison/Cooperation view. (a) Problem type bar showing deduction/addition result of problem types. (b) Stacked bar chart demonstrating monthly deduction/addition result of submission and problems tried. (c) Learners' pass rate lines representing each day's pass rate.*

team selection dates. Two design alternatives were considered for the bilateral stacked graph: bar charts and connected scatter plots. Nonetheless, bar charts were abandoned as they do not effectively convey the sequential nature of time-series data and connected scatter plots were also left as they do not effectively carry a sense of aggregation (e.g., a total area consisting of hard, medium, and easily accepted problems). Overlaying the bilateral stacked graph is a line plot of the learner's performance (ranking) over time.

Furthermore, above the bilateral stacked graph is a horizontal bar (Fig. 5a) representing the percentages of each problem type that a learner correctly solves, requested by domain experts. The ordering (left to right) of the problem-type pieces corresponds to the order in which the learner correctly solves each problem type for the first time. Hovering over each piece displays a tooltip of the number of problems correctly solved of its corresponding problem type.

Submission Details Panel. The submission details panel (Fig. 5c) displays the raw submission record for each learner (**T4**). The grid layout is interpreted by column, from left to right, and each column is interpreted by row, from top to bottom. Each tile resembles one submission and correct submissions are encoded with green while incorrect submissions are encoded with red. In addition, submissions to more difficult questions are encoded with a darker shade and vice versa. Hovering over a tile displays a tooltip of the problem ID of the corresponding problem and also highlights all tiles (submissions) that map to the same problem ID. For example, in Fig. 5c, the learners made four submissions to problem 1008 and the first seven submissions are unsuccessful. Users may brush over the bilateral stacked graph to adjust the time range of the submissions detail panel. If the number of submissions exceeds the fixed number of tiles defined by the panel, the next and previous buttons are available to sift through multiple "pages".

6.4. Comparison/Cooperation View.

Comparison/Cooperation View aims at providing explicit and detailed comparison/cooperation results of two learners' problem-solving dynamics as well as the potential cooperation results. On the one hand, when evaluating students with similar synoptic learning attributes, instructors need to further compare their detailed problem-solving dynamics to make their final choice (**T5**). On the
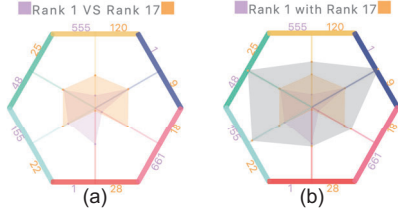
Figure 7: *Comparison/Cooperation view. (a) The comparison result of two learners' hexagons. (b) The cooperation result of two learners' hexagons.*

other hand, apart from individual competition, candidates are required to compete in teams. When considering teamwork or group learning, it is recommended to group learners with complementary skills together(**T5**).

Time-series Comparison/Cooperation. The time-series level comparison consists of three parts: the problem type sequence comparison, the bilateral stacked graph comparison, and the submission detail comparison.

As for the problem type sequence comparison (Fig. 6a), the key information the instructor cares about is which learner has completed more types of problems. Thus, we show the difference between the number of each problem type. The learner ranked 1 is encoded in purple and the one ranked 17 is encoded in orange. The color of each problem type is decided upon by which learner solves more problems on it. The length encodes how many problems one exceeds another. From the red circle in Fig. 6a, we can see 17 does more problems than 1 on "Intr" (Introduction) and "SSor" (Search and Sort). When hovering on each type, the absolute number is shown by the tooltip.

As for the bilateral stacked graph (Fig. 6b), we use explicit encoding instead of juxtaposition or superposition. This is because if we use juxtaposition, it is not easy to compare the height of the stacked graph (the absolute number of solved problems/submissions). The difference in the quantity of each month is explicitly encoded using the stacked bar. The negative part of both upper lateral and lower lateral are encoded in the middle shadow area, the positive part is encoded in the outer sides. Based on this, we can easily judge which learner is more diligent in each period.

The detailed submissions comparison (Fig. 6c) is encoded with a line chart, with the x-axis denoting each day and the y-axis the pass rate. In particular, we use line width to encode the average difficulty of problems tried each day, since a low pass rate may be caused by the problems of high difficulty levels. For example, 17 solved mostly easy questions with a low pass rate around day 25 as shown in (Fig. 6c). Two dots in the x-axis are used to emphasize when learners starts to try hard problems. The cooperation mode has a similar visual encoding by adding values together. In addition, our system supports the combination of more than two sequences.

Key Attributes' Comparison/Cooperation. A conclusive result is provided with the radar chart (Fig. 7), which is widely used in the game to represent a player's skill sets. We use each axis of the hexagon to represent one of the six problem-solving attributes and the value on each axis is the rank of the learner on their problem-

solving attribute, represented by a dot. The six dots of one learner are linked and the area is filled with a transparent color. The two areas of two learners are overlaid (Fig. 7a) in comparison mode. Two or more learners can be added in the cooperation mode (Fig. 7b).

7. Evaluation

We evaluate the usefulness and usability of our system through case studies on a real-world dataset and expert interviews.

7.1. Case Studies

Case 1: Elite Selection. In this scenario, we describe Oliver, the coach for a competitive programming team who wanted to enlist three highly skilled individuals into his team's training camp from a pool of 1000 applicants. In the past, Oliver usually selects people by holding an examination. Nonetheless, he worries that individuals who perform poorly in the examination may still have great potential. Thus, he decides to use *SeqDynamics* to better examine individual students by evaluating their problem-solving history on the online judge.
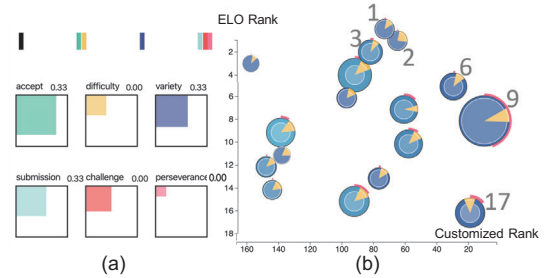

Figure 8: *Enlarged subgraphs of Fig. 1. (a) Correlation Panel. (b) Enlarged section of the projection view in Fig. 1b.*

Oliver first loaded all 1000 applicants into the system and found the ranking distribution sorted by their relative ELO scores (Fig. 1a). He then narrowed down to the top 20 applicants, since the other candidates had relatively lower scores, and then checked the Correlation Panel to see which attributes were positively correlated to the ELO ranking. He discovered that the accept, the submissions, and the variety have larger portions of the inner squares (Fig. 8a). By equally distributing the weights of the three features to one third each, the new customized ranking was generated, based on which he selected candidates with high potential. He then inspected the Projection View (Fig. 1b) for more detailed information and enlarged the top right-hand corner to get Fig. 8b, to see the students who topped both the ELO ranking and his customized ranking.

Oliver noticed the upmost three candidates, 1, 2, and 3, who were the top scorers calculated by ELO ranking and the rightmost three candidates, 6, 9, 17, who ranked top in the customized ranking (T3).

From the first glance at the six candidate glyphs (Fig. 8b), 9 were much bigger (both inner and outer circle) than the others, indicating 9 had tried and solved more questions. Therefore 9 is selected. Then Oliver discovered 3 were more brightly colored and there was a relatively smaller yellow sector, implying that 3 had attempted less

problem types and overall simpler questions compared to the rest of the candidates, and therefore could be eliminated (T2, T3).
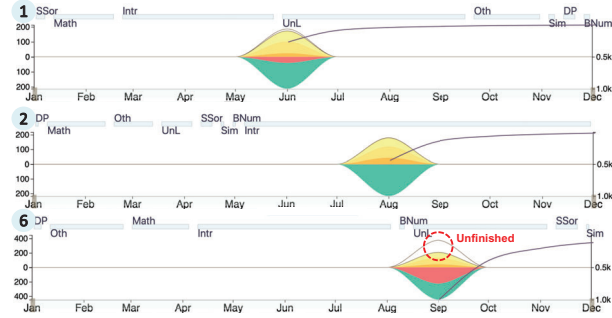

Figure 9: *The bilateral stacked graphs of learners ranked 1, 2, 6.*

Afterwards, Oliver inserted 1, 2, 6, 17 into the Evolution View (Fig. 9) to facilitate greater in-depth understanding and comparison. This view specifically allowed him to vividly visualize each candidate's progressive performance and behaviors (T4). He found that 1, 2, 6 were fast learners and performed outstandingly within a short span. However, among them, 6 had an excessive amount of unfinished questions and was thus eliminated. By inspecting the downstream of the bilateral stacked graphs of 1 and 2, he found that 2 had no red branch while 1 had a distinct red branch, implying that 2 solved problems with a comparatively higher passing rate. Thus, Oliver selected 2 into the team.

With 2 and 9 chosen, Oliver needed to compare 1 and 17 to determine the allocation of the final position in his team of 3. Through the Comparison View of 1 and 17 (Fig. 5a), 1 tried more types of problems than 17. Also, he discovered that 17 started to attempt difficult problems 28 days later than 1 when aligning by start time as seen by the dots near the bottom of Fig. 5c. He then concluded that 17 was less unwilling to step out of his comfort zone and hence selected 1 as the final candidate to enlist. Overall, Oliver successfully selected three candidates, 1, 2, 9, through comprehensive user-driven evaluations.

Case 2: Personal Training. In this case, the student ranked 20 asked his programming coach, Mary, to design a training plan for him so that he could perform well in the upcoming competitive programming contest in three-months time.

At first glance at the Projections View (Fig. 10a), Mary observed that 20 did not excel in terms of glyph size, implying the need for high-intensity programming practice. For a better perspective of 20's learning attributes for customized training, she elected him into the Evolution View for further examination.

From the problem bar in Fig. 10b, Mary identified that 20 tackled more problems on "intro" and "math", but never tried "sorts", "sim-ulation", and "bigNumber". Thus, she concluded that 20 lacked di-versity in problem types tackled. Moreover, she also discovered that 20 had some unfinished questions as shown in Fig. 10b. Through the darker tone of the submissions detail panel (Fig. 10c), she con-cluded that 20 had a relatively high ranking because the problems he finished were of high difficulty levels. She also found that when-ever 20 could not solve a problem in one try, he would often return
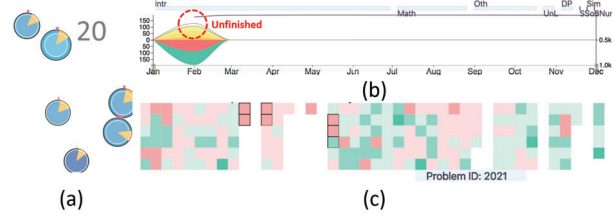

Figure 10: *Information on the learner ranked 20. (a) Glyphs surrounding 20. (b) 20's bilateral stacked graph. (c) 20's detailed submission tiles.*

to the question after one or two days to resubmit a new attempt. This was shown by the highlighted tiles representing the same question being submitted on different days (Fig. 10c). Through the coordination of *SeqDynamics*, Mary believed that 20 is talented and has the potential to succeed in competitive programming. She then encouraged him to attempt a greater number of questions of a wider of variety as he lacked practice in a few problem types. If a question was unsolvable at that time, she advised him to record the question and continue making different attempts until the problem is solved. Mary also created a high-intensity problem set to push 20 to increase his frequency of problem-solving. Overall, she designed a well-rounded training plan based on the personality of 20 which was clearly presented via the interaction of multiple views.

Case 3: Team Formation. In this example, the candidate who ranked 5 approached John, his programming teacher, for help on finding a teammate for the programming competition training camp. With many years of experience, John believed that a great team should consist of teammates with similar working routines while complementing the deficiencies of one another. He turned to *SeqDynamics* to make this job easier and more exhaustive.

From the glyph of 5 (Fig. 12a), John saw that 5 was a risk-taker who began attempting difficult questions early on and had perseverance when tackling challenging problems (T2). Thus, he increased the weights of challenge and perseverance for Projection View and discovered that 8, 9, 10, and 11 clustered near 5. Since they all had similar habits to 5, John began to consider the level of programming ability among the four other candidates. He used the Cooperation View (Fig. 12b) to simulate their cooperation outcome. Consequently, 8 and 9 were excluded from consideration because they both tried relatively few difficult problems, which can be identified from the short slabs in their cooperation hexagons.

With 10 and 11 remaining, John looked further into the problem variety (Fig. 11) and found that both 10 and 5 had not solved a single question involving Dynamic Programming or Big Number, which was shown on the cooperation hexagon by the areas corresponding to the problem type being much smaller than the other areas (Fig. 12b). However, when John added 11 and 5, a nearly perfect hexagon was formed in the cooperation hexagon (Fig. 12b) and they had attempted all 8 types of problems of similar difficulties. Thus, John recommended 11 as a suitable teammate (T6).

7.2. Expert Interview

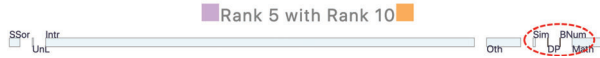Our system was introduced to five domain experts from three dif-

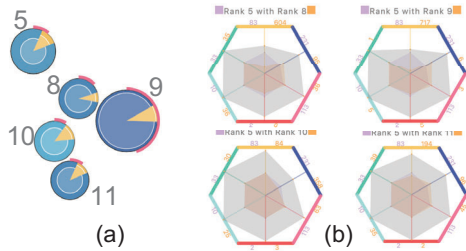**Figure 11:** *The addition result of problems types for 5 and 10.*

Figure 12: *Case 3: Team Formation. (a) Candidates that are similar to 5. (b) The cooperation hexagons of 5 vs. 8, 9, 10, and 11.*

ferent universities (who are not the authors of this paper), including three coaches of competitive programming teams (E1, E2, E3) and two instructors (C1, C2) teaching programming courses. E1 and E2 have been working with us throughout the design process while the other three have just explored our system for the first time. We conducted semi-structured interview with each expert to evaluate four aspects of the proposed system: system usability, system effectiveness, visual designs, and interactions.

Procedure. Each interview had three sections and lasted for at least 60 minutes in total. First, we introduced the purpose of our project, the dataset, and the original programming platform. We explained the workflow and visual encoding of *SeqDynamics* through an example of elite selection as described in Section 7.1 (20 mins). Second, we invited experts to explore the system and use the original online learning platform as the baseline to complete three tasks following a think-aloud protocol: (1) select the top three candidates who can represent the university for a programming competition (10 mins); (2) make a training plan for a given student (10 mins) and: (3) find a teammate for a given student (10 mins). The reason that we assign a limited time for each task is to test the effectiveness of the system. Finally, we collected their feedback on usability, visual design, and interactions, and solicited suggestions for potential improvements (10 mins). We summarized our observations together with experts' feedback from the following aspects.

System Usability Overall, all five experts commented that *SeqDynamics* was useful and easy to use after carefully exploring its detailed features when completing the tasks. For example, in Task 1 (elite analysis), E1, E2, and E3 said that they usually held an exam to select the candidates, sometimes taking the number of problems solved in the online programming system into consideration if such data were available. With our system, they could evaluate students from a more dynamic and comprehensive view of detailed learning behaviors, compared with selecting merely through an exam. During the task, they often set their own criteria and selected the candidates. E1 put more weights on the time to try the hard problems and the percentage of hard problems. *"These can reflect a student is creative and I would not value the number of submissions"*(E1). He explained that in the real competition, wrong submissions are punished. *"The customized function and the ranking in both dimensions (Projection View) are really useful for me to narrow down the*

range". E2 and E3 found the evolution view depicted a candidate in an accurate and multi-faceted way, which helped make the training plan (Task 2). *"From the submission details (Evolution View), I found the student's problem-solving ability is improving since he needs fewer submissions for the easy questions gradually (dark to light color). I would advise him to try harder problems on the similar question type"*(E3). As for Task 3, E3 said that in team competition, each student can focus on certain types of question and students have expertise on different types of questions are suggested to be grouped. C1 and C2 echoed this point. C1 first screen candidates using certain criteria (e.g., the number of problems solved and the time to try hard problems) and then he focused on whether the two candidates compensate each other on the question types in the Comparison View. Also, C2 used the glyphs to find a teammate who was good at solving hard problems but not necessarily the easy ones for a student who was good at solving easy questions. They also pointed out some other issues related to our system. For example, C1 mentioned that he wanted to customize his own evaluation metric based on the time students spent on each problem. E3 said that, *"It is difficult to guarantee the authenticity of the data, since some students may copy and paste the answer from elsewhere."* We summarize and discuss these points in Discussion.

System Effectiveness Since the evaluation criteria for candidates are different among the experts, we assume the system is effective as long as experts can finish experimental tasks (T1-T3) within the given time and give their justification of their answers [XSW*19]. During the interviews, all the experts completed the three tasks within the given time and they provided the rationales behind their answers, e.g., about selecting specific candidates or making a particular training plan. For example, C2 gave concrete training plan for a candidate, *" reduce the practice of questions type 'Introduction' and put more emphasis on other types of questions, since from the question type bar, this student has already solved many 'introduction' of various difficulty levels"*. E3 mentioned that some tasks may be difficult to achieve without this system, e.g., quickly forming a team with complementary capabilities.

Visual Designs After the experts completed the tasks, we collected their feedback on the visual designs of the system. Four experts (E1,E2,E3,C1) felt that the visual designs was intuitive and easy to understand, except for C2 who said it took a while for her to fully grasp different percentages (e.g., hard problems, active days) of the glyph. According to our observation, when they conducted the tasks, the experts rarely encountered difficulty in interpreting the visual designs or the color encoding. They were good at making use of the visual cues to support their decisions. For example, the inner circle and outer circle of the glyph were used by all five experts to judge whether a student has the talent to be the candidate for competitions. Fewer submissions and more problems solved meant higher proficiency. *"The encoding (glyph) is very intuitive and I can tell a learner's talent and assiduousness at a glance"*(E1). E2 mentioned that he took the dynamic change of the submission number evolving with time and the transparency of the submission tails in the Evaluation view into consideration when making his decisions. As for the visual designs of the comparison/cooperation view, *"The hexagon can clearly show the strength and weakness of two candidates"*(C2). In addition, E1 stressed that the number of glyphs plotted on the 2D canvas was limited. C2 sug-

gested making a video or info-tips for explaining the glyph, which would facilitate the understanding more quickly.

Interactions The experts appreciated the rich interactions supported by the system. E2 pointed out that filtering in the Ranking View and brushing in the Projection View helped him quickly narrow the scope of inspection down to a dozen students of interest. C1 mentioned that the tooltips and legends of the system were helpful to her operations.

8. Discussion

System Limitations (1) Scalability. The intuitive glyphs can efficiently help tasks in comparison with a group of candidates. Though users can use tools like Line-up [GLG*13] and compare candidates as the item with multi-dimensional attributes, experts appreciated the glyphs as they can verify the weights of different attributes immediately and help them identify the major features of the students quickly, which also facilitated them to narrow down the scope of their search. However, when more glyphs are plotted to the 2D canvas, visual clutters can become a problem. We currently take the following strategies to mitigate this problem. First, the filter function in the ranking view allows users to preselect a limited number of candidates according to their ELO ranking. Second, after plotting the limited number of glyphs on the 2D canvas, the brushing interaction enables users to further brush an area of interest and then rescale a subset of glyphs to the whole 2D canvas to reduce glyph overlaps. Besides, large glyphs are forced displayed behind the smaller ones to avoid full occlusion. Non-linear axes and hierarchical glyphs can be considered as alternative means to reduce visual clutter in the future. (2) Degree of flexibility. The weight setting function in *SeqDynamics* was appreciated by all the experts. When more data becomes available in the future, we need to improve the flexibility by providing a convenient interface for users to personalize the evaluation metrics of interests based on the raw data. For example, C1 mentioned that the time spent on problems is another metric that can be also taken into consideration, though we have no such data currently. (3) Data issue. We may need to integrate more data from different platforms and design more fine-grained methods to detect possible cheating behavior in the system.

Generalization Our framework and visual design could easily be extended to analyzing series of activities, especially in areas that involve competitions (e.g., e-sports, sports, board games, card games, etc). In these areas, coaches always need to analyze and select potential candidate team members according to their practice and competition history on both cognitive skills and noncognitive traits and behaviors. To be more specific, as for the data, the students' submission sequence data (i.e., time, student ID, problem ID, and results) can be replaced by players' practice sequence data (i.e., time, player ID, match ID, and results) and the embedded ELO ranking algorithm can calculate the ranking of competitions featuring two opposing teams/players. The six attributes can be replaced by the number of wins, the number of hard-mode games, the different conditions of the games, the number of games, the time to play the hard mode, and the percentage of active days. Some detailed visual designs might need to be adapted to the new scenario. In addition, the level-of-detail visual analysis from different per-

spectives and tasks such as elite selection, planning of training, and team composition can be generalized to other application scenarios.

9. Conclusion and Future Work

In this paper, we propose *SeqDynamics*, a visual analytics system that assists instructors in exploring and evaluating learners' problem-solving dynamics from both the cognitive and noncognitive perspectives. The system facilitates the evaluation at multiple scales with multiple views: 1) the Ranking View displays an overall distribution of the learner ranking; 2) the Projection View with Correlation Panel enables users to select a subset of candidates and compare them using glyphs; 3) the Evolution View expands the problem-solving details over time, and: 4) the Comparison/Cooperation View shows explicit comparison between two problem-solving sequences and demonstrates the complementariness of selected learners for a team. It also provides a rich set of interactions for evaluating problem-solving dynamics interactively. In the future, we plan to explore more ways to reduce the glyph overlap, extend our system to domains that involve competitions (e.g., e-sports, sports), and enhance it with more tasks (e.g., performance prediction).

10. Acknowledgements

References

[AMST11] AIGNER W., MIKSCH S., SCHUMANN H., TOMINSKI C.: *Visualization of time-oriented data*. Springer Science & Business Media, 2011. 3

[ANH*68] AUSUBEL D. P., NOVAK J. D., HANESIAN H., ET AL.: Educational psychology: A cognitive view. 4

[BDA*14] BACH B., DRAGICEVIC P., ARCHAMBAULT D., HURTER C., CARPENDALE S.: A review of temporal data visualizations based on space-time cube operations. In *Eurographics conference on visualization* (2014). 3

[Ber17] BERGNER Y.: Measurement and its uses in learning analytics. *Handbook of Learning Analytics* (2017), 35. 2

[BG03] BORG I., GROENEN P.: Modern multidimensional scaling: Theory and applications. *Journal of Educational Measurement 40*, 3 (2003), 277–280. 6

[CA94] CORBETT A. T., ANDERSON J. R.: Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction 4*, 4 (1994), 253–278. 2

[CCZ*16] CHEN Y., CHEN Q., ZHAO M., BOYER S., VEERAMACHANENI K., QU H.: Dropoutseer: Visualizing learning patterns in massive open online courses for dropout reasoning and prediction. In *Visual Analytics Science and Technology (VAST), 2016 IEEE Conference on* (2016), IEEE, pp. 111–120. 3

[CKJ06] CEN H., KOEDINGER K., JUNKER B.: Learning factors analysis–a general method for cognitive model evaluation and improvement. In *International Conference on Intelligent Tutoring Systems* (2006), Springer, pp. 164–175. 2

[CYP*18] CHEN Q., YUE X., PLANTAZ X., CHEN Y., SHI C., PONG T.-C., QU H.: Viseq: Visual analytics of learning sequence in massive open online courses. *IEEE transactions on visualization and computer graphics* (2018), 1–1. 3

[DH02] DRAGICEVIC P., HUOT S.: Spiraclock: a continuous and non-intrusive display for upcoming events. In *CHI'02 extended abstracts on Human factors in computer systems* (2002), ACM Press, pp. 604–605. 3

[DPSS16] DU F., PLAISANT C., SPRING N., SHNEIDERMAN B.: Eventaction: Visual analytics for temporal event sequence recommendation. In *In Proceedings of Visual Analytics Science and Technology (VAST)* (2016), IEEE, pp. 61–70. 3

[Elo78] ELO A. E.: *The rating of chessplayers, past and present.* Arco Pub., 1978. 4

[Far03] FARKAS G.: Cognitive skills and noncognitive traits and behaviors in stratification processes. *Annual review of sociology 29*, 1 (2003), 541–562. 2, 4

[FFM12] FISCHER F., FUCHS J., MANSMANN F.: Clockmap: Enhancing circular treemaps with temporal glyphs for time-series data. *Proc. EuroVis Short Papers, Eurographics* (2012), 97–101. 3

[GLG*13] GRATZL S., LEX A., GEHLENBORG N., PFISTER H., STREIT M.: Lineup: Visual analysis of multi-attribute rankings. *IEEE transactions on visualization and computer graphics 19*, 12 (2013), 2277–2286. 11

[Goo17] GOODSPEED R.: Research note: An evaluation of the elo algorithm for pairwise visual assessment surveys. *Landscape and Urban Planning 157* (2017), 131–137. 4

[GTY] GEORGE TRIMPONIAS X. M., YANG Q.: Rating worker skills and task strains in collaborative crowd computing: A competitive perspective. In *The Web Conference 2019*, p. 10.1145/3308558.3313569. 4

[hdu19] Hangzhou dianzi online judge. http://acm.hdu.edu.cn/, 2019. Accessed: 2019-3-31. 4

[HHWN02] HAVRE S., HETZLER E., WHITNEY P., NOWELL L.: Themeriver: Visualizing thematic changes in large document collections. *IEEE transactions on visualization and computer graphics 8*, 1 (2002), 9–20. 3

[JGP*19] JOVANOVIĆ J., GAŠEVIĆ D., PARDO A., DAWSON S., WHITELOCK-WAINWRIGHT A.: Introducing meaning to clicks: Towards traced-measures of self-efficacy and cognitive load. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge* (2019), ACM, pp. 511–520. 2

[KBK11] KRSTAJIC M., BERTINI E., KEIM D.: Cloudlines: Compact display of event episodes in multiple time-series. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2432–2439. 3

[Lee19] Leetcode. https://leetcode.com/, 2019. Accessed: 2019-3-18. 2

[LK17] LIU R., KOEDINGER K.: Going beyond better data prediction to create explanatory models of educational data. *The Handbook of Learning Analytics* (2017), 69–76. 2

[May92] MAYER R. E.: *Thinking, problem solving, cognition. A series of books in psychology.* New York: WH Freeman/Times Books/Henry Holt & Co, 1992. 2

[MHB19] MOLENAAR I., HORVERS A., BAKER R. S.: Towards hybrid human-system regulation: Understanding children'srl support needs in blended classrooms. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge* (2019), ACM, pp. 471–480. 2, 4

[MXC*19] MU X., XU K., CHEN Q., DU F., WANG Y., QU H.: Moocad: Visual analysis of anomalous learning activities in massive open online courses. 3

[PÁMMPS18] PÉREZ-ÁLVAREZ R., MALDONADO-MAHAUAD J., PÉREZ-SANAGUSTÍN M.: Design of a tool to support self-regulated learning strategies in moocs. *J. UCS 24*, 8 (2018), 1090–1109. 3

[PBH*15] PIECH C., BASSEN J., HUANG J., GANGULI S., SAHAMI M., GUIBAS L. J., SOHL-DICKSTEIN J.: Deep knowledge tracing. In *Advances in Neural Information Processing Systems* (2015), pp. 505–513. 2

[PJCK09] PAVLIK JR P. I., CEN H., KOEDINGER K. R.: Performance factors analysis–a new alternative to knowledge tracing. *Online Submission* (2009). 2

[PL04] PARK J., LEE L.: Analysing cognitive or non-cognitive factors involved in the process of physics problem-solving in an everyday context. *International Journal of Science Education 26*, 13 (2004), 1577–1595. 2

[PMR*96] PLAISANT C., MILASH B., ROSE A., WIDOFF S., SHNEIDERMAN B.: Lifelines: visualizing personal histories. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (1996), ACM, pp. 221–227. 3

[RML08] REVILLA M. A., MANZOOR S., LIU R.: Competitive learning in informatics: The uva online judge experience. *Olympiads in Informatics 2*, 10 (2008), 131–148. 4

[Sim19] Simstudent. https://hcii.cmu.edu/research/simstudent, 2019. Accessed: 2019-3-18. 2

[SQG*19] STONE C., QUIRK A., GARDENER M., HUTT S., DUCKWORTH A. L., D'MELLO S. K.: Language as thought: Using natural language processing to model noncognitive traits that predict college success. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge* (2019), ACM, pp. 320–329. 2

[SvdWvW14] SCHEEPENS R., VAN DE WETERING H., VAN WIJK J. J.: Non-overlapping aggregated multivariate glyphs for moving objects. In *2014 IEEE Pacific Visualization Symposium* (2014), IEEE, pp. 17–24. 6

[WLS*14] WÄSCHLE K., LACHNER A., STUCKE B., REY S., FRÖMMEL C., NÜCKLES M.: Effects of visual feedback on medical students' procrastination within web-based planning and reflection protocols. *Computers in Human Behavior 41* (2014), 120–136. 3

[XSW*19] XIA M., SUN M., WEI H., CHEN Q., WANG Y., SHI L., QU H., MA X.: Peerlens: Peer-inspired interactive learning path planning in online question pool. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (2019), ACM, p. 634. 3, 10

[Zar05] ZAR J. H.: Spearman rank correlation. *Encyclopedia of Biostatistics 7* (2005). 6

[ZCCB13] ZHAO J., COLLINS C., CHEVALIER F., BALAKRISHNAN R.: Interactive exploration of implicit and explicit relations in faceted datasets. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 2080–2089. 3

[ZLD*15] ZHAO J., LIU Z., DONTCHEVA M., HERTZMANN A., WILSON A.: Matrixwave: Visual comparison of event sequence data. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (2015), ACM, pp. 259–268. 3