Modeling Training Efficiency and Return on Investment for Adaptive Training: GIFT Integration

Gregory A. Goodwin[1], James Niehaus[3]
ARL HRED[1], Charles River Analytics[2]

ABSTRACT

Adaptive training promises more effective training by tailoring content to each individual. Where non-adaptive training may be just right for one segment of the student population, there will be some students that find it too easy while others find it too difficult. Another, often ignored benefit of adaptive training, is improved training efficiency by minimizing the presentation of unnecessary material to learners. One implication of this is that intelligent, adaptive training should require less time to train a population of learners to a given level of proficiency than non-adaptive training. The gains in efficiency should be a function of several factors including learner characteristics (e.g., aptitude, reading ability, prior knowledge), learning methods employed by the adaptive training system, course content (e.g., difficulty and length, adaptability), and test characteristics (e.g., difficulty, number of items). This paper describes work in the second year of a three year effort showing the results of a predictive model for training efficiency based on those factors and how it could be integrated into the Generalized Intelligent Framework for Tutoring (GIFT) architecture. How this model supports return on investment decisions for authors is also discussed.

INTRODUCTION

The Generalized Intelligent Framework for Tutoring (GIFT) is an open-source, modular architecture developed to reduce the cost and skill required for authoring adaptive training and educational systems, to automate instructional delivery and management, and to develop and standardize tools for the evaluation of adaptive training and educational technologies (Sottilare, Brawner, Goldberg, & Holden, 2012a; Sottilare, Goldberg, Brawner, & Holden, 2012b). By separating the components of ITSs, GIFT seeks to reduce development costs by facilitating component reuse.

Meta-analyses and reviews support the claim that intelligent tutoring systems (ITS's) improve learning over typical classroom teaching, reading texts, and/or other traditional learning methods. (Dynarsky et al. 2007; Dodds and Fletcher 2004; Fletcher 2003; Graesser et al. 2012; Steenbergen-Hu and Cooper 2013, 2014; VanLehn 2011). In fact, ITS's have been shown to improve learning to levels comparable to Human tutors (VanLehn et al. 2007; VanLehn 2011; Olney et al. 2012).

As shown in Figure 1, while improved training effectiveness is certainly a benefit of ITS technology, another important benefit is improved training efficiency over one-size-fits-all training. The goal of an ITS is to identify the gaps in knowledge specific to each learner so that training can focus on filling just those gaps. One of the problems of one-size-fits-all training is that to insure all trainees can comprehend the instruction, it must be developed for trainees with the least experience, knowledge, and aptitude. Though less costly to develop, the material is presented a pace that is slow and that includes content not needed for more experienced, higher aptitude trainees. An ITS would be expected to reduce the time needed to deliver training to such trainees.

The reduction in time to train (i.e., improved acquisition rate) is an important metric because reductions in training time represent cost savings. This is especially true for military trainees who are paid a salary.

Reductions in the time needed to train those trainees save salary costs for both trainees and instructors. For large-volume courses, those savings can be substantial.

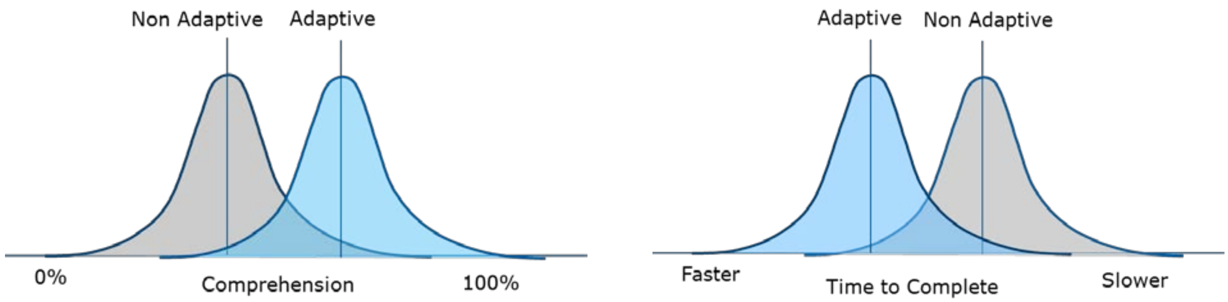

Figure 1: Benefits of adaptive training. On the left, adaptive training can increase the subject comprehension from a fixed time to complete . On the right, adaptive traininc can decrease the time to complete training content with a fixed level of comprehension.

All of this highlights the need for a means to model and predict training efficiency gains (i.e., time saved) by ITSs generally and GIFT specifically. Having the ability to model time saved by the use of adaptive, intelligent training, as compared to existing or non-adaptive training would have benefits throughout the lifecycle of a course. During the design of new training, the training developer could more easily make decisions about the relative costs and benefits of adding adaptive features. For example, adding extensive remedial training for easy-to-understand concepts may benefit such a small percent of the population of learners, that the net reduction in training time would be too small to make those features worth the cost of development.

During training delivery, actual trainee data could be used to verify and/or improve the model. For example, suppose the model assumed that learners with an aptitude above criteria A would have a 95% probability of understanding concept B without needing any remediation. Learner data could then be used to validate or adjust that probability. This improved model could then be used to better determine the true time-savings of the course when delivered by GIFT.

During training evaluation and refinement, the disparity between predicted and observed training outcomes could be used to refine the training. For example, if a segment of training proves to be more difficult than anticipated for a group of learners, it is possible that the training segment should be refined or redeveloped.

An example of such a model was developed by McDonnell Douglas (1977). This model incorporated predictor variables in four broad categories: course content (e.g., difficulty, length of content), instructional design (e.g., instructional strategies/techniques), test characteristics (e.g., difficulty, number of items), and trainee characteristics (e.g., aptitude, motivation). The model predicted about 39% of the variability in trainee's first-attempt lesson time for self-paced computer-based instruction.

To understand how GIFT might begin to model and predict training time for learners, it is necessary to understand how training is adapted by this system. GIFT is a framework that modularizes the common components of intelligent tutoring systems. These components include a learner module, an instructional or tutor module, a domain module, and a user interface. One of the main motivations for creating this framework was to lower the cost and labor needed to create intelligent tutoring systems by facilitating reuse of components and by simplifying the authoring process (Sottilare et al., 2012a).

GIFT adapts training using the learning effects model. At the first point of this model, learner data informs the learner state in the learner module. The learner module receives assessments from both sensors and the domain module. The learner state is used to determine the appropriate instructional strategy by the tutor module. The instructional strategy is then interpreted by the domain module and used to determine the domain specific learning activities needed to instruct the learner in that domain. The responses of the learner to that activity then update the learner module which starts the cycle over again.

Developing a predictive model in GIFT is not a straightforward process given the ways that training is adapted to each individual. We should note that our goal is not to predict the single path that a trainee would be expected to take through a specific course, but rather the probability associated with all possible paths through the training for a given learner. From that we can determine the range and distribution of times that would be expected for that learner to complete the training. Taking this one step further, we could apply this to a population of learners and predict the range and distribution of the time for that population to complete that training.

The development and integration of a probabilistic model for predicting time to train into the GIFT architecture is currently in the second phase of a three phase plan. Goodwin, Kim, and Niehaus (2017) reviews the approach and results of the first phase of this effort, which focused on the design and feasibility of these predictive models of tutor time to complete. In this paper, we describe work being done in the second phase. In the second phase, we are enhancing a predictive model for training efficiency and integrating this model with GIFT architecture, so that GIFT course creators can use these models directly with their GIFT tutors. In the third phase of the work, we will empirically validate the predictive model in GIFT and enhance the models with experimental and collected data.

METHODS

This section (1) reviews our method for modeling adaptive training content and predicting distributions of completion times for both individuals and groups using the GIFT excavator trainer as an example and (2) describes our approach for integrating these models with the GIFT architecture.

Modeling the Content of Adaptive Training

Predicting completion time for a tutor requires a model of the content and how the student can transition between the content. In GIFT, this transition logic is maintained in the Adaptive Course Flow object (formerly known as the Engine for Management of Adaptive Pedgagoy – EMAP, e.g., Sottilare, 2014; Goldberg, 2015). It supports adaptive capabilities for training based on instructional strategies such as the Component Display Theory (CDT, Merrill, 1983). The CDT supports a general framework of skill training that progresses through two types of learning activities, each with two categories: expository (rules and examples) and inquisitory (recall and practice). According to Merrill, learners should progress through these four quadrants in order starting with rules (presentation of general principles), then to examples (presentation of a specific instance), then to recall (declarative knowledge test of the trainee's comprehension), and finally to practice (opportunity for the trainee to perform the skill). By sorting learning activities into these four quadrants, adaptive training systems like GIFT can apply the CDT to any domain as long as content for that domain is so labeled.

To model the content of adaptive training, we use the *Methodology for Annotated Skill Trees* (MAST) (Bauchwitz et al. 2018). In MAST, the "skeleton" of the skill tree breaks down entire procedures into constituent steps, tasks, and subtasks. Annotations are added to the procedure model. Figure 2 shows a portion of a MAST skill tree for an example training GIFT course, the excavator tutor. This skill tree focuses on the information elements that most heavily influence the completion time. On the left, the overall course on Excavator is the root of the tree structure. Its children are the different topics covered by the course, including the Boom Movement topic. This topic features a number of slides with Pictures, Audio, and Text components. Individual trainees may vary in the amount of time they spend examining the Pictures, whether or not they listen completely to the Audio, and the amount of time taken to read the Text. Trainees may also choose to view optional Slides explaining concepts that they may not be familiar with, adding more time. If trainees fail to demonstrate sufficient knowledge in the quiz or fail to complete the simulation tasks appropriately, they are sent back to the beginning of the Boom Movement topic on Slide 1, adding significant time to completion of the course. This model may be expanded to represent a maximum number of failures before the trainee either moves to a different topic or ends the course.

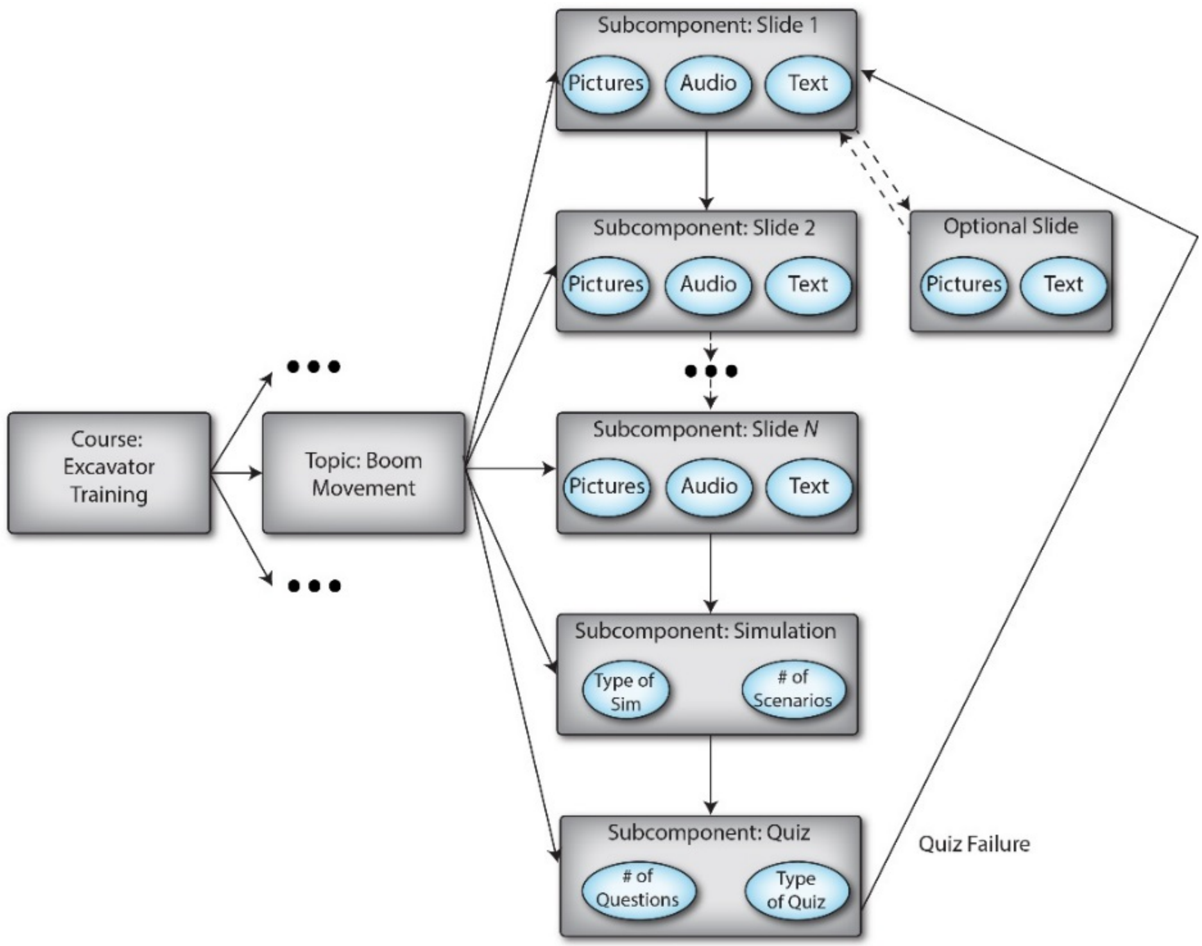

Figure 2: High-level design of a MAST skill tree of a GIFT module with representations of individual instructional elements, branching content, and variables that influence completion times.

After reviewing the Slides, the trainees are asked to practice their skills in Simulation. The MAST model of the simulation can be either a complex procedure describing the steps needed to complete the scenario and optional steps that may or may not contribute to the overall goal. The MAST simulation model may also be simple, representing just the type of simulation and the number of scenarios. To save modeling time and effort, these MAST models are constructed with only the level of detail needed to sufficiently and accurately predict the completion time.

Integrating with GIFT

To effectively predict completion time, we must combine models of students with models of the adaptive training content. We construct probabilistic models of students in the Figaro probabilistic programming language with key variables that influence their completion time of generic content. Figure 3 shows an example of one such set of variables. This student has a fatigue value to represent how tired they are. They have a read speed variable to represent how many words per minute they read under normal conditions. They have an expertise variable that represents how familiar they are with the concepts in the tutor. They have an effort level variable that represents how much effort they are putting into the training. They have an innate comprehension level that represents their general learning aptitude. They also have some status variables that record how many repetitions of different drills and quiz failures they have had during the course of training, for reporting purposes. These and similar parameters can be used to characterize the main student features that influence their completion time of training content. To be used in actual courses, these parameters must be learned and validated with real world data, such as records of previous students attempting a course.

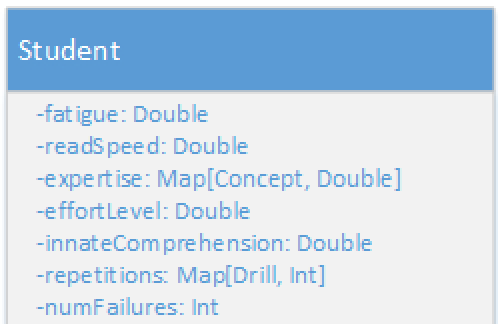

Figure 3: UML for example Student model

Figure 4 shows a model of how a GIFT course can be represented as a set of learning material that the student must read or experience. At the top, the course is composed of multiple concepts. According to Merrill's CDT theory, each concept is taught by presenting a number of rules (on slides), examples (on slides), quizzes to test rules, and exercises to test understanding of the examples. Each slide has a selection of media, which can include text, audio, and video, and is also rated for comprehensibility (e.g., more difficult slides take more time to comprehend). Quizzes are composed of a set of questions, which rules for how many must be answered correctly before the quiz is passed. Exercises are similarly composed of a set of drills with individual difficulties. Representing in the course in this way, along with the control logic that determines which piece of content the student is provided with next, enables the probabilistic modeling of the interaction between anticipated student populations and course content. It also enables the analysis of which content or sections of the course are contributing most to the completion time.

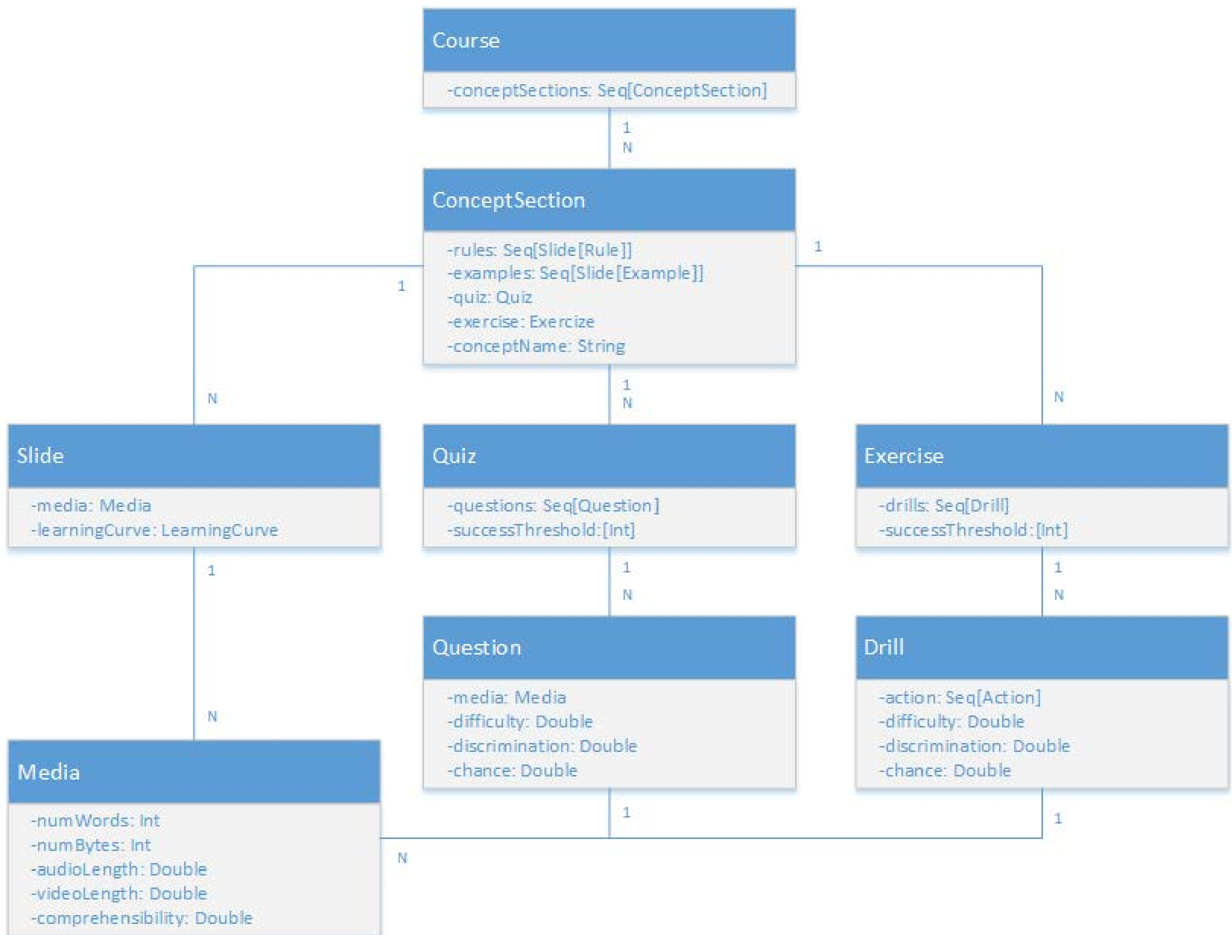

Figure 4: Partial UML diagram for PAST Time model of a Merril instructional theory GIFT Tutor

To effectively use these models, they must be integrated into the authoring cycle of adaptive training. Figure 5 shows a mockup of an interface to enable GIFT adaptive course authors to use these models for predicting completion time and understanding the impact of course design decisions on the ROI of adaptive training. At the top, the user specifies the GIFT tutor of interest, and which student model to use. The student model determines which parameters will be used to represent the student, such as those in Figure 3. The user is also presented with the option of including previous performance data to better tune the models to the population of interest.

To request a prediction, the user specifies a single student or a group of students according to the student model. In the single student case, this can be done by selecting exactly which values are set for each student parameter. In the group of students case, this can be done by specifying joint distributions of these values for the group of interest. Once these parameter values are specified, the models are executed and summary statistics of the prediction are presented, with the option for the user to explore the various components of the prediction.

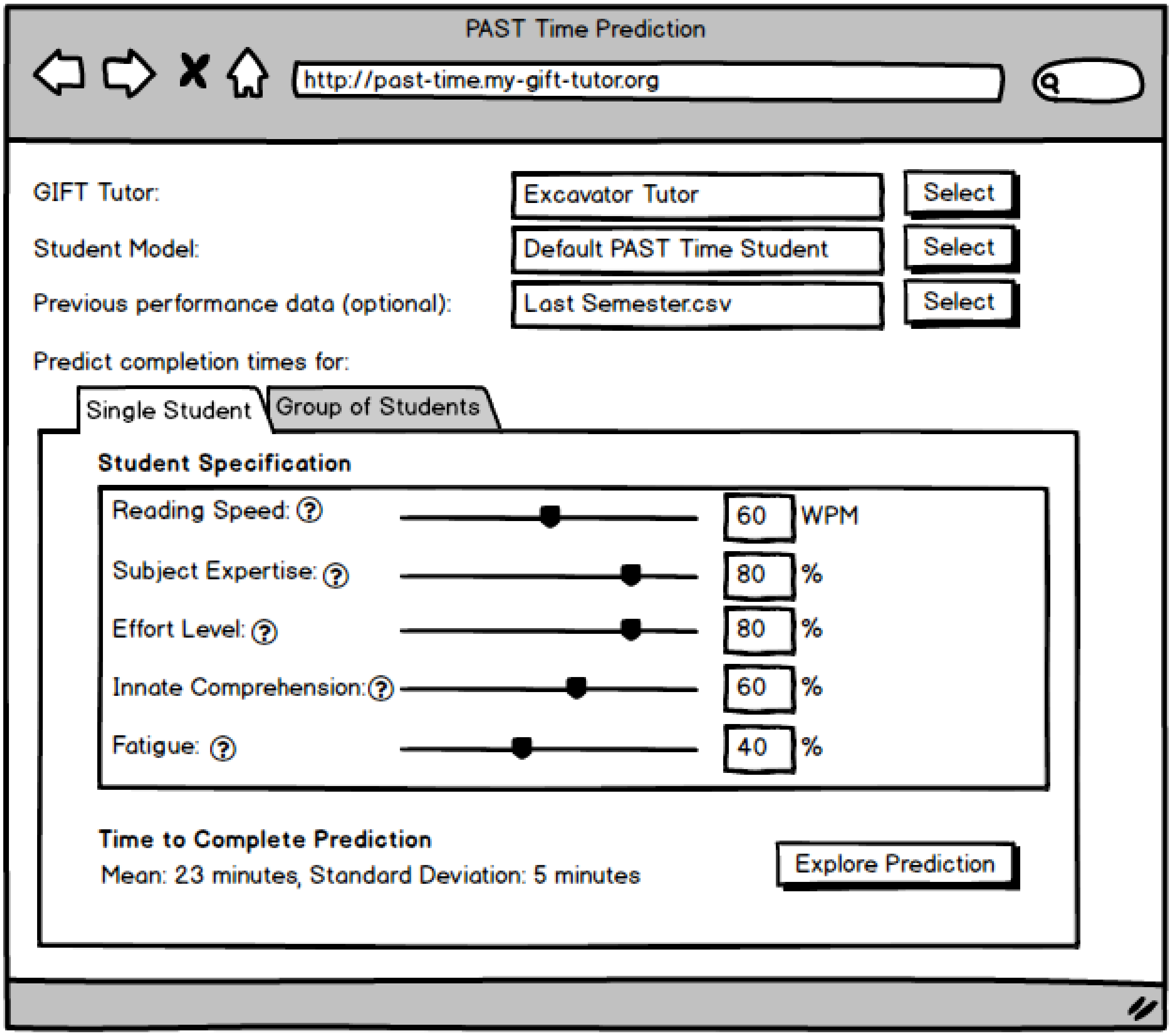

Figure 5: Mockup of PAST Time GIFT integration interface.

RESULTS

Implementing the Adaptive Training Models

This section presents a sample of the implementation of completion time models, and analysis of results of running these models with mock data. The probabilistic model is being implemented using Charles River Analytics' open source probabilistic programming language, Figaro™ (Pfeffer 2012), to construct and learn probabilistic models of the relationships between these factors. The use of Figaro greatly simplifies the authoring of these models which can be complex and require a high degree of experience by users who may not be experts in probabilistic reasoning.

Figure 6 shows an example Figaro function that predicts the completion time for a student reading a slide of information and updates the effect of reading the slide on the individual student. The reading time is computed by summing the media ingestion time of all the media on the slide (e.g., reading some text, looking at pictures, listening to audio, and watching video). The students internal variables are then updated to reflect the effects of reading this slide; their fatigue is increased by a small amount and their expertise in the current concept is increased according to a specified function. The student is updated, the reading time is recorded, and the simulated student is then given the next piece of course content to complete.

```
def readSlide[T <: TrainingElement](slide: Slide[T], concept: Concept, student: Student):
(Student, Element[Double]) =
{
  val readingTime: Element[Double] = mediaIngestionTime(slide.media, student)
  val newFatigue = Math.min(1, Math.max(1.005*student.fatigue, 0.00000001))
  val newExpertise = student.expertise |+| Map(concept -> expertiseIncrease[T](slide, con-
cept, student))
  (student.copy(
    fatigue = newFatigue,
    expertise = newExpertise
  ), readingTime)
}
```

Figure 6: Figaro function that models a student's reading time for a slide

Figure 7 shows the model for a student taking a quiz as part of the adaptive training content. A quiz has multiple questions, which take time to complete. Based on the student's performance and the quiz passing threshold, the student may be sent to remediation for the current concept. In this function, the probabilities of success for each question are determined by the questions difficulty (currently, in classic item response theory) and the student's current aptitude at the concept. The reading time is given by the sum of the reading and thinking times for all the questions. The fatigue is updated by a marginal amount, and the new student is created. The probability of success on this quiz is also returned, so that Figaro can sample across the space of probabilities.

```
def takeQuiz(quiz: Quiz, concept: Concept, student: Student): (Student, Element[Double], El-
ement[Boolean]) = {
  val probs: Seq[Element[Boolean]] = quiz.questions.map(q => probOfSuccess(q, concept, stu-
dent))
  val questions = Container(probs: _*)
  val readingTime = Reduce((x: Double, y: Double) => x+y)(quiz.questions.map{q => mediaIn-
gestionTime(q.media, student)}: _*) // thinking time
  val newFatigue = Math.min(1, Math.max(1.05*student.fatigue, 0.00001))
  (student.copy(fatigue = newFatigue), readingTime, questions.count(x => x).map(_ >=
quiz.successThreshold))
}
```

Figure 7: Figaro function that models a student's completion time of a Quiz

Figaro probabilistic programming is useful in this context for a number of reasons: We can automatically build a model given a specification of the MAST skill tree, the trainee model, and a set of known relationships. Prediction based on the model is already coded in Figaro's inference algorithm, so additional effort is not required to use the model. Figaro supports the creation of dynamic Bayesian networks that model the temporal processes of variables, simulating fatigue and practice effects. We can continuously learn using these models; the probabilistic programs are flexible enough to update relationships between variables based on historical or dynamic data. Figaro's encapsulation mechanism enables easy creation of reusable components. Trainee models and MAST skill trees can be reused for future prediction models. It is embedded in a general purpose language, Scala, which allows the creation of front end graphical interfaces that can edit and invoke the models created in Figaro.

To test the current models, we created a set of mock simulated students to run on a mock tutor. Figure 8 shows the student sampling process, with samples across the space of initial fatigue, innate comprehension, effort level, and reading speed.

```
object StudentGenerator{
  def generateStudents: Seq[Student] = {
    for{
      fatigue <- 0.0 to 0.5 by 0.1
      innateComp <- 0.5 to 1 by 0.1
      effortLvl <- 0.5 to 1 by 0.1
      readSpeed = 300
    }yield{
      Student(fatigue, Constant(readSpeed), Map(), effortLvl, innateComp, Map())
    }}}
```

Figure 8: Mock student generation for system testing

Figure 9 shows the results of this sample set on completion times. Because this is a sample to test model dynamics, the completion time units on the y-axis are arbitrary. In this plot, there is a combined effect of the main variables. In the left, when fatigue is low, low effort and comprehension only make a moderate difference in completion time. On the right, when initial fatigue is high, low effort and comprehension are compounded, as the mock students begin failing quizzes and drills, which causes them to repeat content, which causes them to become more fatigued. At the far right, several samples included one or more students that exceeded our modeling time limit, causing them to be marked as 0 for the purposes of this graph. This effect can be seen in Figure 10 where the relationship between failures and completion time is exponential due to the compounding fatigue factor.

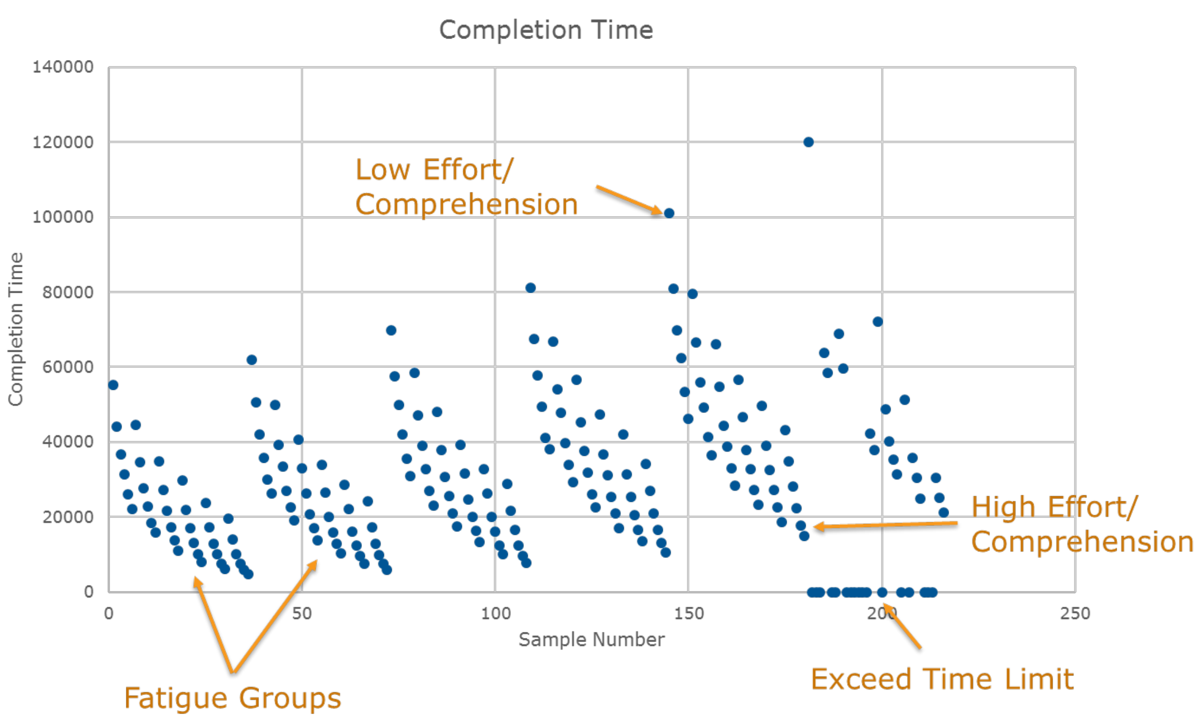

Figure 9: Completion time of mock students, sampled across the space

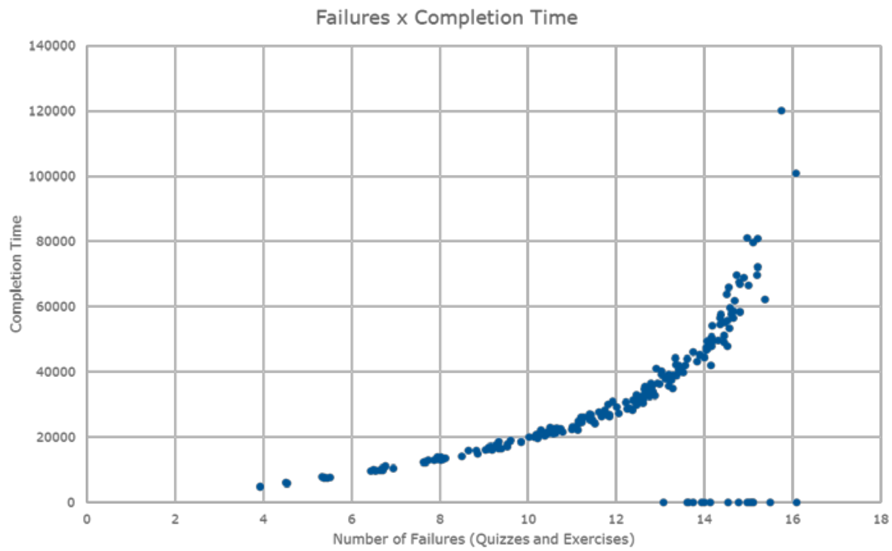

Figure 10: Relationship between failures on quizzes and drills and completion time of mock students

Models like these enable adaptive training course authors to quickly explore what-if scenarios with ranges of students and different configurations of adaptive content. The third phase of our effort will focus on learning actual student models from empirical data, enabling us to calibrate these models to actual performance, realistic completion times, and identify which observable and latent variables are most valuable in this prediction.

DISCUSSION

We believe that including a capability to predict training time for trainees in GIFT has several significant advantages for accelerated learning. First, it facilitates return on investment (ROI) calculations by enabling the author to determine training time reductions resulting from the addition of adaptive features. Second, it provides a means for GIFT to monitor student progress against an expected timeline. Students who take much longer to complete training than expected may not be fully engaged in the training or may be having difficulty with the material. These are conditions that might prompt a response by GIFT. Finally, it can play a role in quality control of GIFT courses. For example, if segments of a course take much longer than expected across multiple trainees, GIFT could flag those sections for review by the course author to insure that the material is presented clearly.

Determining the ROI for training is not always easy. As Fletchter and Chatham (2010) put it, how does one determine the benefit of a pound of training? In some cases it may be fairly straight forward. For example, one might measure the increase in revenue produced by the introduction of new training for a sales staff. While this may work for commercial businesses, the military is not a profit making organization, therefore one must look at other factors like cost avoidance to get a measure of ROI.

Determining this can be quite difficult as one rarely has before and after data on the operational impact of training. In rare cases it can be found. For example, Fletcher and Chatham (2010) examined the benefits of Top Gun training given to pilots during the Vietnam war. Because of this training, kill ratios of Navy pilots improved from 2.4 enemy kills per loss up to 12.5 enemy kills per loss. The authors determined that the training had reduced U.S. losses by about 10-12 aircraft during the war When they looked at the cost of procuring and employing that many aircraft during the war, they calculated that the training had saved the Navy about $132 million dollars for an ROI of about 2.5.

Determining the ROI for adaptive vs. non-adaptive training in terms of cost avoidance measures in an operational context would be very difficult. Adaptive training is still relatively new and opportunities to do side-by-side comparisons with traditional non-adaptive training are virtually non-existent. Rather than trying to quantify an impact in the operational environment however, we can look at the impact in a training environment. Specifically, one of the key advantages of adaptive training would be to reduce the overall time needed to deliver the training to a population of trainees.

A challenge for authors of adaptive training is determining how *adaptive* the training should be. While adding adaptive features can potentially save training time, it also increases the cost of development. How does one determine, when the training is adaptive enough? Using an ROI metric can help to answer this question. On one hand is the cost of adding the adaptive feature. On the other hand is the value of the time saved by that adaptive feature. The value of that time could be calculated by looking at the total salary paid to the trainees over that time (e.g., 1,000 trainees/year x .5h/trainee x $35/h = $17,500/year). So, as long as the cost of adding the adaptive feature was less than value of the time saved, there would be a positive ROI and therefore justification for adding that particular adaptive feature.

As can be seen, our model supports this strategy for the design and development of adaptive training in GIFT by helping to predict the effect of adaptive features on the training time for a known population of learners.

There are several challenges we may face as we develop this model. First, the initial MAST skill tree may not contain sufficient variables to predict adaptive training completion times. Our initial literature review and analysis have identified a potential set of most influential variables, but these variables may not be reflective of the completion time upon closer inspection. We will mitigate the identified risk by widening the scope of task models to incorporate more predictive variables if necessary.

Second, while the model predictions may be highly accurate, there is a risk that the system will be too difficult or time consuming to use for some or all of the target populations of instructional designers, course managers, and instructional staff. We mitigate this risk by conducting a requirements analysis early in the effort to closely examine the needs of these user groups and design our system and interfaces to best meet those needs. We will apply human factors and user-centered design and understand the challenges of and methods for developing highly useful and usable decision-aiding tools for practitioners.

Third, while this approach combines state of the art probabilistic approaches and identifies key variables from the literature and past experience, there is a potential that the initial predictions will not sufficiently account for the variability of trainee completion times. We plan to mitigate this risk by incorporating historical data early and adjusting the analysis techniques to capture the maximum amount of variability from data that can be reasonably collected in the field.

When complete, this will be the first system to predict the completion times of GIFT and to enable effective assessments of the ROI that is useful for key design and implementation decisions of an adaptive training system. It includes an innovative application of the procedure skill modelin the MAST skill tree to flexibly represent the adaptive training content for analysis. It is the first application using a probabilistic programming language (i.e., Figaro) to predict completion times for adaptive training technologies, including both unobserved latent variables and temporal factors, such as trainee fatigue, boredom, or flow.

REFERENCES

Army Research Laboratory. (2015). GIFT 2015-1, Generalized Intelligent Framework for Tutoring Release Page. Retrieved from http://www.gifttutoring.org

Bauchwitz B, Niehaus J, Weyhrauch P. Modeling and Comparing Nurse and Physician Trauma Assessment Skills. Military Medicine, Volume 183, Issue 1, 2018, Pages 47–54,

Dodds P, Fletcher JD. Opportunities for new “smart” learning environments enabled by next generation web capabilities. Journal of Educational Multimedia and Hypermedia. 2004;13:391–404.

Dynarsky M, Agodini R, Heaviside S, Novak T, Carey N, Camuzano L, Sussex W. Effectiveness of reading and mathematics software products: findings from the first student cohort; March Report to Congress 2007. [accessed 2015 May 15]. http://ies.ed.gov/ncee/pdf/20074005.pdf.

Fletcher JD. Evidence for learning from technology-assisted instruction. In: O’Neil HF, Perez R, editors. Technology applications in education: a learning view. Mahwah (NJ): Erlbaum; 2003. p. 79–99.

Fletcher, J. D., and R. E. Chatham. 2010. “Measuring Return on Investment in Military Training and Human Performance.” In Human Performance Enhancements in High-Risk Environments, edited by J. Cohn and P. O’Connor, 106–28. Santa Barbara, CA:

Praeger/ABC-CLIO.

Goodwin GA, Kim JW, Niehaus J. Modeling Training Efficiency and Return on Investment for Adaptive Training. InProceedings of the 5th Annual Generalized Intelligent Framework for Tutoring (GIFT) Users Symposium (GIFTSym5) 2017 Jul 17 (p. 229). Robert Sottilare.

Goldberg, B., & Hoffman, M. (2015). Adaptive course flow and sequencing through the engine for management of adaptive pedagogy (EMAP). In *Proceedings of the AIED Workshop on Developing a Generalized Intelligent Framework for Tutoring (GIFT): Informing Design through a Community of Practice* (pp. 46-53). Madrid, Spain.

Graesser AC, Conley M, Olney A. Intelligent tutoring systems. In: Harris KR, Graham S, Urdan T, editors. APA Educational Psychology Handbook: vol. 3. Applications to Learning and Teaching. Washington (DC): American Psychological Association; 2012. p. 451–473.

McDonnell Douglas Corporation. A survey and analysis of military computer-based training systems: (A two part study). Vol II: A descriptive and predictive model for evaluating instructional systems; 1977. Defense Advanced Research Projects Agency. [accessed 2015 October] http://www.dtic.mil/dtic/tr/fulltext/u2/a043358.pdf.

Merrill, M. D. (1983). Component display theory. In C. M. Reigeluth (Ed.), *Instructional-design theories and models: An overview of their current status* (pp. 282-333). Hillsdale, NJ: Lawrence Erlbaum Associates.

Olney AM, Person NK, Graesser AC. Guru: designing a conversational expert intelligent tutoring system. In: Boonthum-Denecke C, McCarthy P, Lamkin T, editors. Cross-disciplinary advances in applied natural language processing: issues and approaches. Hershey (PA): Information Science Publishing; 2012. p. 156–171.

Pfeffer, A. (2012). Creating and manipulating probabilistic programs with Figaro. Workshop on Statistical Relational Artificial Intelligence (StarAI).

Sottilare R, Brawner KW, Goldberg BS, Holden HK. The generalized intelligent framework for tutoring (GIFT). Orlando (FL): Army Research Laboratory (US); Human Research and Engineering Directorate (HRED); 2012a [accessed 2015 May]. https://gifttutoring.org/attachments/152 /GIFTdescription_0.pdf.

Sottilare R, Goldberg BS, Brawner KW, Holden HK. A modular framework to support the authoring and assessment of adaptive computer-based tutoring systems (CBTS). In: Proceedings of the Interservice/Industry Training Simulation and Education Conference; 2012 Dec 3–6; Orlando, FL. Arlington (VA): National Defense Industrial Association; 2012b.

Sottilare, R. A. (2014). Using learner data to influence performance during adaptive tutoring experiences. In *Proceedings of International Conference on Augmented Cognition--HCII2014* (pp. 265-275). Crete, Greece:

Springer.Steenbergen-Hu S, Cooper H. A meta-analysis of the effectiveness of intelligent tutoring systems on K-12 students' mathematical learning. Journal of Educational Psychology. 2013;105(4):970–987.

Steenbergen-Hu S, Cooper H. A meta-analysis of the effectiveness of intelligent tutoring systems on college students' academic learning. Journal of Educational Psychology. 2014;106:331–347.

VanLehn K, Graesser AC, Jackson GT, Jordan P, Olney A, Rosé CP. When are tutorial dialogues more effective than reading? Cognitive Science. 2007;31(1):3–62.

VanLehn K. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. Educational Psychologist. 2011;46 (4):197–221.

ABOUT THE AUTHORS

Dr. Gregory Goodwin *is the acting Branch Chief of the Creative and Effective Technologies Branch at the Army Research Laboratory – Human Research and Engineering Directorate in Orlando, FL. For the last decade, he has worked for the Army researching ways to improve training methods and technologies. He holds a Ph.D. in Psychology from Binghamton University and an M.A. in Psychology from Wake Forest University.*

Dr. James Niehaus *is a senior scientist at Charles River Analytics. Dr. Niehaus's areas of expertise include artificial intelligence, training systems, and health technology. Dr. Niehaus has a B.S. in computer science from the College of Charleston and a Ph.D. in computer science from North Carolina State University.*