A HYBRID MACHINE LEARNING APPROACH TO AUTOMATED SCENARIO GENERATION (ASG) TO SUPPORT ADAPTIVE INSTRUCTION IN VIRTUAL SIMULATIONS AND GAMES

Robert A. Sottilare
US Army Natick Soldier Center
robert.a.sottilare.civ@mail.mil

ABSTRACT

This paper examines machine learning methods to automatically generate a large number of child scenarios from a small number of parent scenarios in support of adaptive instruction conducted in virtual simulations and game-based platforms. Adaptive instructional systems (AISs) include Intelligent Tutoring Systems (ITSs), intelligent mentors, recommender systems, personal assistants, and intelligent instructional media. AISs attempt to tailor instruction for individuals and teams based on their learning needs (e.g., knowledge or skill deficiencies), goals, and preferences. This often requires much more content than current non-adaptive systems which provide one or a very limited set of training scenarios to address a given set of learning objectives. The goal of the research described in this paper is to reduce the authoring burden for developing a large number of unique and relevant training scenarios. The methodology presented also ranks the resulting scenarios with respect to a set of author-specified learning objectives and learner/team competency in the domain of instruction. The unique contributions of this paper are tied to its hybrid machine learning approach, and consideration for both learning objectives and learner/team competency in automatically ranking generated scenarios.

Keywords: adaptive instruction, adaptive instructional systems (AISs), automated scenario generation (ASG), combinatorial optimization search (COS), evolutionary scenario generation (ESG), genetic algorithm (GA), Intelligent Tutoring System (ITS), machine learning, novelty search, ranking algorithm, reinforcement learning

1. INTRODUCTION

Adaptive instructional systems (AISs) are a class of intelligent, machine-based tools that "guide learning experiences by tailoring instruction and recommendations based on the goals, needs, and preferences of each learner [or team] in the context of domain learning objectives" (Sottilare & Brawner, 2018a; Sottilare & Brawner, 2018b; Brawner & Sottilare, 2018). AISs may include technologies like intelligent tutoring systems (ITSs), intelligent mentors, recommender systems, personal assistants for learning, and intelligent instructional media.

In late 2017, the Institute for Electrical and Electronics Engineers (IEEE) Learning Technologies Standards Committee (LTSC) under the auspices of the IEEE Standards Association established an AIS study group to examine opportunities for potential standards to lower the entry and maintenance costs associated with AISs and the AIS study group identified authoring (development) as a major barrier to the adoption of AISs. The AIS authoring process can be divided into two sub-processes: 1) developing or finding appropriate content (often called curation), and 2) sequencing or aligning content with learning objectives (sometimes called building or configuring depending on the authoring tool).

Specifically, the AIS standards study group identified the skill and cost of authoring these systems as very high, usually requiring highly technical individuals with expert programming skills to develop and maintain these systems. Compounding this problem is the fact that AISs often require significantly more content (as much as 2 or 3 times) than non-adaptive systems since each adaptation (tailored instructional sequence) requires new content, and each remediation also requires new content.

We are suggesting that the authoring barrier might be reduced by automating as much of the authoring process as possible. Ideally, we would want to fully automate the entire AIS authoring process, but are taking the approach to solve one problem at a time beginning with the complex problem of automated scenario generation (ASG; Zook, et al, 2012) which can greatly expand the content choices for adaptive instruction offered by authoring tools like the Generalized Intelligent Framework for Tutoring (GIFT; Sottilare, Brawner, Goldberg, & Holden, 2012; Sottilare, Brawner, Sinatra, & Johnston, 2017), the Cognitive Tutor Authoring Tool (CTAT; Aleven, McLaren, Sewall, & Koedinger, 2006), the Authoring Software Platform for Intelligent Resources in Education (ASPIRE; Mitrovic et al, 2006) and other AIS authoring platforms.

The goal of ASG is to create training scenarios for domains that vary in their complexity, definition, and dynamics (Sinatra & Sottilare, 2016), and are ranked by their relationship with specified learning objectives. The basic idea is to automatically create significantly different scenarios where all the variables in the scenario are allowed to vary maximally resulting a large number of training situations available to support tailored instruction. Of course, not all the scenarios created would be relevant, doctrinally correct or even possible in the real-world. A mechanism is needed to rank their relevance or fitness with respect to a set of learning objectives and the competency of the learner in performing the assigned task.

By way of example, we have selected a *room clearing task* under varying conditions to illustrate the functional aspects of ASG and how it might work for military training. Usually, an instructional developer would be responsible to handcraft each scenario using a scenario editor specific to the game/simulation being used and their expertise in the domain of instruction. A military or law enforcement squad or fire team would usually train to master the task of entering and clearing a room of any hostiles. ASG is critical to providing both challenging and doctrinally correct scenarios for adaptive team training.

The next section of this paper explores the scope of the ASG problem space by way of defining terms and describing the process associated with a genetic algorithm (GA) approach.

2. SCOPING THE PROBLEM OF ASG

As part of examining the ASG problem space, we thought it would be useful to provide a few definitions to help shape the scope of our discussion:

- ***Scenario*** - a process in which a learner or learners interact within an environment over a sequence of events which introduce and/or exercise a set of skills defined by a set of learning objectives
- ***Fitness Function*** – criteria used to assess how close a scenario is to achieving a set of defined objectives
- ***Scenario Generator*** - a computational system that solves the problem of producing a set of viable scenarios given knowledge about their attributes and their alignment with the fitness criteria
- ***Initial Population*** – an initial set of scenarios that adequately represent a set of targeted learning objectives and are used to generate future scenarios through some machine learning technique
- ***New Population*** – a resulting set of scenarios automatically generated that are more closely aligned with the fitness criteria than their parents; a scenario's fitness is determined by the weighted linear sum of all evaluation functions

The ASG problem space can be distilled into three distinct challenges: 1) how to insure sufficient variation in the parent population so these traits are passed to subsequent generations; 2) how to promote sufficient variation of complexity and tailoring in the subsequent generations/new populations; and 3) defining the fitness criteria to evolve and rank a population of new scenarios that support specified learning objectives, support goals, preferences, and learning needs of individuals or teams, and are realistic.

Given the definitions and challenges described, we can now concentrate on describing a generalized process for ASG using GAs (Figure 1). We chose to use GAs based on their flexibility in addressing a variety of tasks, their ability to cover the search space, and their ability to address the three challenges we identified. In the next section of this paper, we explore three approaches to developing an ASG capability using GAs.

3. EXAMINING POTENTIAL GA APPROACHES

According to Zook et al (2012), a genetic algorithm usually starts with a population of randomly generated potential scenarios and attempts to modify and/or combine aspects of different scenarios within the population to improve the fitness of the next generation of scenarios according to a given fitness function. A GA is "a search heuristic that is inspired by Charles Darwin's theory of natural evolution" (Mallawaarachchi, 2017) that generates a pool of candidate solutions called a population.

GAs have an advantage over gradient based methods which may trend toward local optima for many complex real-world domains. GAs have the ability to provide a large number of usable (good enough) solutions relatively quickly (Wikipedia, 2018). GAs are also relatively easy to implement and resolve to a solution in most cases. For ASG, this makes them a more attractive choice over other approaches (e.g., deep reinforcement learning, artificial neural networks) which may be difficult to implement.

For our exploration of ASG using GAs, we will use a "*room clearing training task*" as a basis for the examination of three machine learning approaches that exploit genetic algorithms:

- Brute Force Search
- Novelty Search
- Combinatorial Optimization Search

3.1. Brute Force Search Approach

A brute force search (also known as an exhaustive search) solves the generation problem by systematically enumerating *all possible candidates* for the solution and checking whether each candidate satisfies the problem's statement (Wikipedia, Brute-Force Search, 2018). Depending upon the number and type of variables, how we decide to implement the GA for our room clearing task could be complex or very simple. Our example task discussed in Section 4 of this paper is very simple in order to illustrate the principles and process of implementing a GA.

Brute force searches are the easiest to implement, and always find a solution if one exists. However, as the number of candidate solutions grows, the search time also grows rapidly. "Therefore, brute-force search is typically used when the problem size is limited, or when there are problem-specific heuristics that can be used to reduce the set of candidate solutions to a manageable size. The method is also used when the simplicity of implementation is more important than speed" (Wikipedia, Brute-Force Search, 2018).

3.2. Novelty Search

Fitness functions for genetic algorithms are typically goal-focused. The goal in ASG is to understand the alignment of candidate scenarios with specified learning objectives. An exception to this is novelty search. *Novelty search* uses a fitness function to promote behavioral novelty instead of attempting to conduct a search through the use of a static objective or set of objectives (Lehman, 2012). Since the goal of this search is to identify unique candidates, the result is more likely to include candidates outside of what might be normally acquired through a static objective search or in systems where the number of candidates is limited. Conversely for ASG, large numbers of initial candidates (parent scenarios) are likely to yield a more diverse set of child scenarios resulting in some non-viable scenarios.

Genetic algorithms attempt to satisfy a criteria set by the fitness function so they generally do not identify optimal candidates and with the exception of novelty search do not tend to cover the entire search space. Novelty search in its attempt to identify all unique candidates does tend to cover the more remote areas of the search space often left uncovered by other GA approaches.

3.3. Combinatorial Optimization Approaches

Combinatorial Optimization uses a scenario generation approach to deliver the requisite diversity and quality of scenarios while tailoring the scenarios to a particular learner's needs and abilities. This type of optimization includes an eight step process illustrated in Figure 1 and discussed in detail in Sections 3.1.-3.6 (Shiffman, 2012).

- Step 1: define fitness criteria
- Step 2: create an initial population of N scenarios
- Step 3: assess the fitness of each individual within the population based on the fitness criteria until stop criteria is met, then go to step 8
- Step 4: create a mating pool based on fitness scores and select pairs for reproduction
- Steps 5 & 6: reproduce N times through cross-over and mutation and add each child to the new population
- Step 7: replace the old population with the new one and return to step 3
- Step 8: print results and terminate

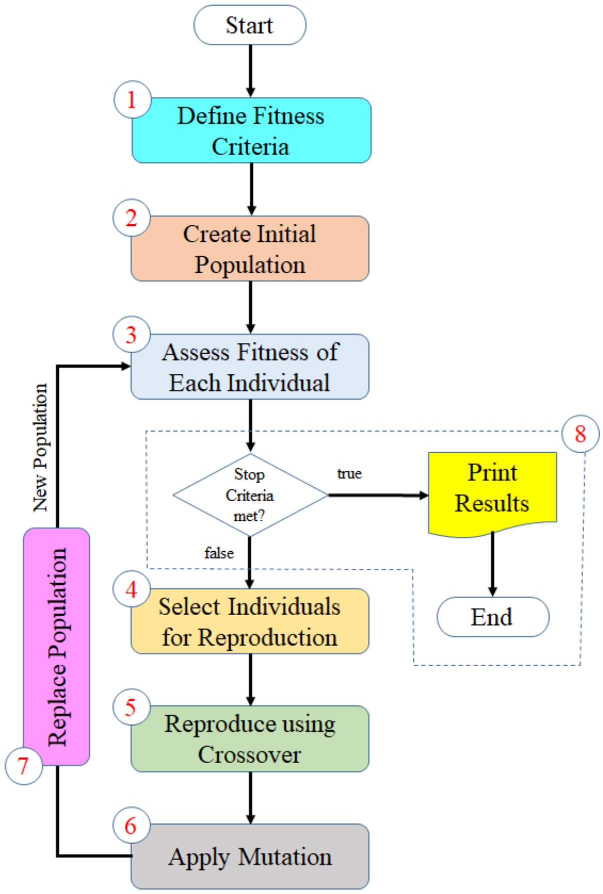

Figure 1. Combinatorial GA Approach

In the next section of this paper, we discuss how we might apply and vary the GA approaches reviewed above to support ASG for our example task, clear rooms training task.

4. COMBINATORIAL OPTIMIZATION WITH NOVELTY SEARCH

In this section, we layout a process for combinatorial optimization with novelty search. The steps discussed below are applicable to a variety of task domains and we provide an example task to illustrate its use. The example task, an understanding of its elements and how these elements relate to a specified fitness function are critical to reusing this process for other domains.

For our example task we have chosen a "*clear rooms training task*" (US Army, 2007) which is usually performed by squads of dismounted soldiers on patrol. Since the room clearing task is a psychomotor task, we refer to the GIFT authoring tools which use a psychomotor task model based on theories advanced by Dave (1970), Simpson (1972), Harrow (1972), and Romiszowski (1999) as adapted by Brown, Bell & Goldberg (2017). The goal is to have the team practice and demonstrate a level of proficiency where automaticity and fluid motion are the norm. In a game-based training environment, the focus is more on the cognitive aspects of the task and mastering the interface, but in a fully immersive virtual environment, the focus is on mastering the physical aspects of the task, interaction with the environment, and interaction (e.g., communication and coordination) with other members of the team.

All military training scenarios describe the task, the conditions under which each task is conducted, and the standards or measures of successful performance. The standards or measures of success for our room clearing task include:

- Enter the room quickly and smoothly
- Clear the doorway immediately
- Remain within arm's reach of another squad member
- Secure room by neutralizing any enemy present
- Maintain sufficient force to defeat any enemy counterattack and continue operations

Note that even an identical environment (e.g., same room layout and same threats at the same locations) could result in a different scenario based on the squad's decisions and performance. The simple decision of entering the room at a different location can impact the sequence of events to follow. If we examine the dynamic elements (e.g., skill events, environment, and constraints) of our room clearing training example, we find that our scenarios can vary by type, sequence, length, and outcome of events, but can also vary in complexity by changes to the environment (e.g., threats or the physical configuration of the building) and the number and type of constraints (e.g., rules of engagement or presence of non-combatants).

Next, we describe eight essential steps in the GA process described below and shown in the context of Figure 1. We have modified these to fit our example training task and to overcome our defined set of challenges:

- define our fitness criteria
- insure sufficient variation in the parent population
- promote sufficient variation of complexity and tailoring in the child population

For our technical approach to ASG (described in the eight steps below), we have chosen to primarily use a combinatorial optimization approach since it would inefficient to pursue a brute force approach for more complex scenarios than our example task. This will allow us to represent more complex domains in the future with the same GA process. We have also chosen to substitute a Novelty search in Step 2 (create initial population) to provide a more full representation of the search space and provide a higher degree of variability in subsequent generations.

The resulting combinatorial optimization with novelty search process for ASG is based on a Darwinian model of evolution through natural selection and genetic variation:

- Step 1: define fitness criteria
- Step 2: create an initial population of N scenarios has two substeps:
  - 2a. select 3-4 scenarios that vary across the variables selected for the fitness criteria
  - 2b. use Novelty search to expand this population to N unique scenarios using single point crossover and single point mutation
- Step 3: assess the fitness of each individual within the population based on the fitness criteria until stop criteria is met, then go to step 8
- Step 4: create a mating pool based on fitness scores and select pairs for reproduction
- Steps 5 & 6: reproduce N times through cross-over and mutation and add each child to the new population
- Step 7: replace the old population with the new one and return to step 3
- Step 8: print results, output scenario editor file and terminate

4.1. Step 1: Define Fitness Criteria

The first and most important challenge is to define our fitness criteria such that scenarios that more closely align to our learning objectives are ranked higher than those that are more loosely aligned with the learning objectives.

In scoping the ASG problem space, it is necessary to understand the relationship between task learning objectives and attributes of potential solutions in the initial population. For our task, a squad will be trained to clear one or more rooms in a building in a virtual simulation (e.g., Virtual Battle Space). The room clearing task involves many coordinated behaviors, but the learning objectives or standards for the squad can be distilled into five essential assessments defined previously in this section.

The complexity of the task can vary and scenarios that account for varying complexity can be tailored to the competency (experience or prior knowledge) of the team members. Each of these task learning objectives and thereby any associated scenario may be complicated by the:

- Size and shape of the room
- Number of armed enemy forces present
- Number of non-combatants present
- Obstacles at the doorway or in the room

Given this task and varying complexity, we represented any given solution in the population of possible scenarios as a four digit integer where each integer varies from 0 to 9:

- Size and shape of the room - where 0 = simplest room (e.g., a small rectangular room) and 9 =

most complex room (e.g., large room with interior corners and multiple doorways)

- Number of armed enemy forces present – where 0 = no enemy forces present, and 9 = 9 enemy forces present
- Number of non-combatants present – where 0 = no non-combatants present, and 9 = 9 combatants present
- Obstacles at the doorway or in the room – where 0 = no obstacles and 9 = 9 obstacles present

Assuming it is physically possible to fit 9 enemy forces, 9 non-combatants and 9 obstacles in the smallest room, this would mean we can generate up to 10^4 scenarios using a genetic algorithm approach as shown in Figure 2.

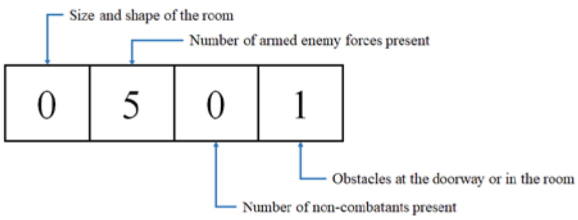

Figure 2. Population Representation Schema

For simplicity, we have elected to measure the complexity of a scenario by summing the four genes or attributes that make up a scenario. For example, the complexity for the chromosome or potential solution shown in Figure 2 would be 6 (0 + 5 + 0 + 1) where 0 would is the lowest complexity and 36 the highest. If we align the complexity of scenario with the domain competency of the team, we would have alignment within Vygotsky's (1987) Zone of Proximal Development (ZPD; Figure 3). This alignment will help maintain engagement and positive affect during the training process.

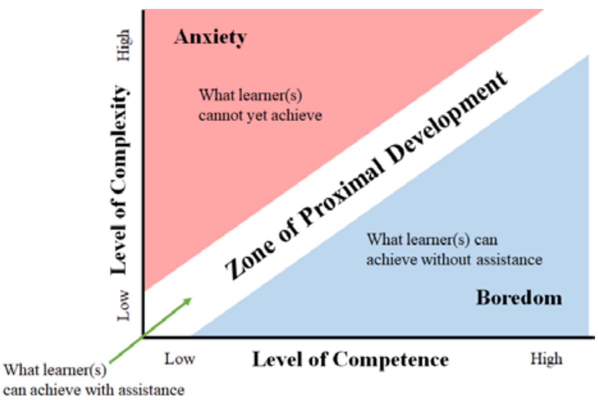

Figure 3. Zone of Proximal Development

We could then use this as a fitness criteria by comparing the complexity of the scenario and domain competency of the team. Defining the team domain competency in four intervals between 0 and 36 provides the following distribution for team competency:

- Expert (28-36)
- High Skills (19-27)
- Moderate Skills (10-18)
- Low Skills (0-9)

We assume that some long term model of the team's domain competence, a pretest, or subject matter assessment would determine where any particular team would fall on this competency scale, but the fitness function for this example task domain would be:

Fitness = domain competency – scenario complexity

For example, a moderately skilled team with a domain competency score of 17 would be sufficiently challenged by a scenario with a complexity of 17 ± σ. Assuming σ = 4, then any scenario in the range of 13-21 would be at an appropriate level of complexity for that particular team. While this example may be overly simplified, it does illustrate the process which could be applied in more complex training domains. If we wished to generate the 10 most appropriate scenarios for this team, we would rank them from lowest difference to highest difference.

4.2. Step 2: Create an Initial Population

The next step in the process is to generate a set of individual scenarios (solutions or chromosomes) which comprise the population. Parameters are represented in the chromosomes as variables are known as genes. Normally, the initial population is generated randomly, but it is critical that sufficient variability is represented in this initial population or the GA search will produce limited results. We have chosen to start with a limited set of four scenarios and then stretch the variability of the population through Novelty search. In this way, we might find sufficient variety in future generations if variability is also represented in the initial population.
For example, an initial population should contain at least one scenario for each of the levels of complexity (easy, moderate, and hard) could be expanded using Novelty search. In our case, we aligned domain complexity intervals with the competency intervals defined in Step 1. Randomly using 0150 (low complexity), 5273 (moderate complexity), 9911 (high complexity), and 4997 (very high complexity) as a seed population for Novelty search will result in several solutions that represent a large portion of the scenario complexity required for future generations. It also allows us to expand our approach to represent other learner/team attributes beyond complexity and competency. The resulting unique set of scenarios will be sufficient to act as an initial population for a combinatorial optimization approach (e.g., crossover and mutation) in subsequent steps discussed below.

4.3. Step 3: Assess the Fitness of Individuals

The fitness function determines the suitability of an individual scenario as a potential solution. The candidate solutions in the population are assessed with respect to the learning objectives which we used to determine the variables in the GA search and matched to scenarios aligning with the competency level of the team.

4.4. Step 4: Select Individuals for Reproduction

In our problem space, ASG, the GA selects the fittest individual scenarios in the current generation to produce offspring for the next generation of the population using a fitness function. In the selection step the goal is to pair the fittest individuals to let them reproduce and pass their genes to the next generation. Some number of pairs (two or more) are selected where the probability of selection of an individual scenario for reproduction is based on its fitness score. Again, assuming a team competency of 17, we selected the top four scenarios in terms of fitness in our first generation resulting from Novelty search for our example task:

- 1772
- 1952
- 1970
- 0773

4.5. Step 5: Reproduce using Crossover

Genetic algorithms are usually used to find solutions meeting the fitness criteria by employing operators like crossover. A single crossover point in the chromosome is chosen and the genes prior to the crossover point are exchanged between the pair of scenarios. For our example task, using the four fittest scenarios defined in Step 4, we might see a pairing between 1772 and 1952 resulting in 1972 and 1752 (Figure 4).

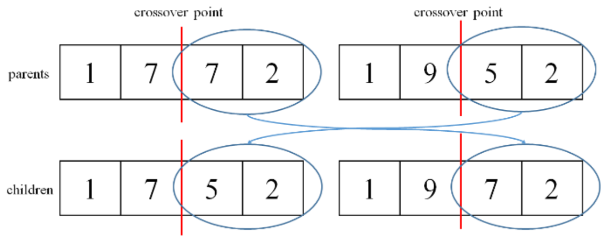

Figure 4. Reproduction Using Crossover (also known as Recombination)

4.6. Step 6: Apply Mutation

In a percentage of the new individuals formed by the crossover reproduction, a random gene is selected for change (single point mutation). Mutation randomly alters a parameter of a randomly chosen event in the scenario and then reevaluates the mutation to determine if its fitness has improved. Mutation is critical step in maintaining diversity in the new population. For our example task, we might see a random mutation that changes 1972 to 8972. While this might seem inefficient since the fitness of the scenario went from 0 to 9, overall mutation infused the new population with greater diversity while the average fitness of the population will continue to optimize.

Note that in addition to crossover and mutation, other less used operators include addition and deletion (Mitchell, 1996; Zook et al, 2012). Addition inserts a random event into the scenario at a random location and then reevaluates the resulting candidate scenario to determine if its fitness has improved. Deletion removes a random event from the scenario. For simplicity, we elected to apply only crossover and mutation operators for our example task.

4.7. Step 7: Replace Old Population with New Population

In this step, we discard the old population in favor of the new population created using crossover and mutation. It is important for the author to select a large enough number of iterations to allow the average fitness score of each new population to trend toward some optimal value or plateau.

4.8. Step 8: Terminate

In this step, we determine when to terminate the ASG process and output the results. The ASG process may continue until a termination trigger is reached:

- a candidate solution is identified that satisfies some minimum criteria
- a fixed number of generations is reached
- an allocated amount of time has elapsed
- candidate solutions reach a level of fitness where they plateau (no significant change)

The output of the ASG process is two-fold:

- Printed list of scenarios (e.g., 1772) with their associated fitness scores
- Scenario editor input (digital file compatible with common scenario editors for games and immersive virtual environments used for training

5. RESULTS

The pseudo code below represents the resulting combinatorial optimization with novelty search GA used to generate scenarios based on the example training task of room clearing:

1. START: Set competency target = 17 with *fitness = domain competency – scenario complexity* and *fitness goal = 0*
2. Selection: randomly 3 initial scenarios with significantly different complexity scores (results = 0342, 5172, 9145)
3. Novelty Search: expand initial population to 10 unique scenarios using single point crossover and single point mutation (10%) to create 7 additional new scenarios (results = 0342, 5172, 9145, 0372, 5142, 0345, 9142, 5145, 9172, 7372)

4. Compute fitness of each individual scenario (results for generation 0)
    a. 0342 fitness = abs(17-9) = 8
    b. 5172 fitness = abs(17-15) = 2
    c. 9145 fitness = abs(17-19) = 2
    d. 0372 fitness = abs(17-12) = 5
    e. 5142 fitness = abs(17-12) = 5
    f. 0345 fitness = abs(17-12) = 5
    g. 9142 fitness = abs(17-16) = 1
    h. 5145 fitness = abs(17-15) = 2
    i. 9172 fitness = abs(17-19) = 2
    j. 7372 fitness = abs(17-19) = 2
5. REPEAT
    a. Selection for mating pool – based on fitness and stochastic universal sampling (Baker, 1987)
    b. Crossover
    c. Mutation (10%)
    d. Compute fitness
6. UNTIL population has converged
7. END

6. CONCLUSIONS, CHALLENGES AND NEXT STEPS

We presented a process and schema for applying a hybrid (Combinatorial Optimization with Novelty Search) GA approach to the automated authoring of scenarios for games and immersive virtual environments. The process provided wide variability for the resulting scenarios that were aligned to author specified learning objectives and learner/team competency to support adaptive instruction.

The ASG approach in this paper is applicable to a broad number of domains in digital training environments. While we focused this application of GAs to ASG for games and virtual simulations, we also see application of this process to live simulations (e.g., mission rehearsal).

The GA approach to ASG described herein benefits greatly from more specific domain knowledge resulting in better objective values. Of course it takes some time for a person to define schema and to incorporate this specific knowledge in each new domain, so this process is not fully automatic, but can be for the end user once the schema and fitness criteria are defined. A likely next step is to create an author dashboard for unit commanders and subject matter experts to lead them through the process of developing learning objectives and critical variables as input to the ASG process defined herein.

The primary challenges defined were three-fold: 1) generating a large number of feasible scenarios which cover a large portion of the search space for variables like complexity (e.g., easy, moderate, and hard scenarios) and which cover the highest percentage of the learning objectives, 2) ranking scenarios in order of relevance to a set of author-defined learning objectives, and 3) providing output from the ASG process which is compatible with scenario editors for both games (e.g. Virtual Battle Space) and immersive virtual environments used for military training. The first two challenges have been met by the process described in this paper along with alignment between learner/team competency and scenario complexity ala Vygotsky's ZPD. This tailoring will enhance the relevance of the scenario for the learner/team and thereby enhance engagement and learning.

The third challenge has not been met, but is simply a mechanical translation of the scenario code to a format that can be understood by scenario editors for games and virtual simulations. The delay in meeting this challenge is based on understanding what games/simulations will have the highest use and thereby the most need for the ASG process. We anticipate solving this problem very quickly once a set of target simulation environments is identified.

One next step includes application of this ASG process to a diverse set of task domains for both individual learners and teams. The application of ASG for teams is particularly challenging given the complexity of assessing team learning and performance, and will only strengthen the process over time.

Another next step for GA-based process might also include alignment with other learner/team states that moderate learning. For example, another desired end state for this research is to be able to adapt existing scenarios or select available scenarios based on individual learner emotions (e.g., boredom or anxiety) or behavioral markers (e.g., encouragement) which are antecedents to team states (e.g., team cohesion). The emotional state of the learner(s) might also be a criteria for selecting either a more or less difficult scenario or injecting support feedback (scaffolding) into an existing scenario to make it easier per Vygotsky's (1987) ZPD (Figure 3).

We also anticipate experimenting with local search after the crossover and mutation steps to see if that will yield better solutions or enhance the speed of the ASG process for more complex domains.

Finally, in spite of the complications in using deep reinforcement learning (Rowe, Smith, Pokorny, Mott, & Lester, 2018), we anticipate continuing research in this area with the hope of enhanced results over time. This assumes that we will find a process that will be flexible enough to apply easily in a variety of task domains trained by military organizations.

ACKNOWLEDGMENTS

The research described herein has been sponsored by the U.S. Army. The statements and opinions expressed in this article do not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred.

REFERENCES

Aleven, V., McLaren, B. M., Sewall, J., & Koedinger, K. R. (2006, June). The cognitive tutor authoring tools (CTAT): preliminary evaluation of efficiency gains. In *International Conference on Intelligent Tutoring Systems* (pp. 61-70). Springer, Berlin, Heidelberg.

Baker, James E. (1987). "Reducing Bias and Inefficiency in the Selection Algorithm". *Proceedings of the Second International Conference on Genetic Algorithms and their Application*. Hillsdale, New Jersey: L. Erlbaum Associates: 14–21.

Brawner, K. & Sottilare, R. (2018, June). Proposing Module-level Interoperability for Adaptive Instructional Systems. In the Exploring Opportunities to Standardize Adaptive Instructional Systems (AISs) Workshop of the *18th International Conference of Artificial Intelligence in Education (AIED)*, London, United Kingdom, June 2018.

Brown, D., Bell, B. and Goldberg, B., 2017. Authoring Adaptive Tutors for Simulations in Psychomotor Skills Domains. In Proceedings of *MODSIM World 2017*, Virginia Beach, VA: NTSA.

Dave, R.H. (1970). Psychomotor levels. In R.J. Armstrong (Ed.), *Developing and Writing Behavioral Objectives*. Tucson, Arizona: Educational Innovators Press.

Folsom-Kovarik, J.T. & Brawner, K. (2018, May). Automating Variation in Training Content for Domain-general Pedagogical Tailoring. In Proceedings of the 6th Annual Generalized Intelligent Framework for Tutoring (GIFT) Users Symposium. *US Army Research Laboratory*, Orlando, FL.

Harrow, A. (1972) A Taxonomy of Psychomotor Domain: A Guide for Developing Behavioral Objectives. New York: David McKay.

Lehman, J. (2012). Evolution through the search for novelty. Dissertation: *University of Central Florida*.

Lehman, J., & Stanley, K.O. (2011). Novelty search and the problem with objectives Genetic Programming Theory and Practice IX (pp. 37-56): *Springer*.

Liapis, A., Yannakakis, G. N., & Togelius, J. (2015). Constrained novelty search: A study on game content generation. Evolutionary computation, 23(1), 101-129.

Luo, L., Yin, H., Cai, W., Zhong, J., & Lees, M. (2016). Design and evaluation of a data-driven scenario generation framework for game-based training. *IEEE Transactions on Computational Intelligence and AI in Games*.

Mallawaarachchi, V. (2017). Introduction to Genetic Algorithms – Including Example Code. *Towards Data Science*. Retrieved from: https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3.

Mitchell, Melanie (1996). An Introduction to Genetic Algorithms. Cambridge, MA: *MIT Press*. ISBN 9780585030944.

Mitrovic, A., Suraweera, P., Martin, B., Zakharov, K., Milik, N., & Holland, J. (2006, June). Authoring constraint-based tutors in ASPIRE. In *International Conference on Intelligent Tutoring Systems* (pp. 41-50). Springer, Berlin, Heidelberg.

Romiszowski, A. (1999). The development of physical skills: Instruction in the psychomotor domain. In Instructional-design theories and models: a new paradigm of instructional theory (Vol. 2). Mahwah, NJ: Erlbaum.

Rowe, J., Smith, A., Pokorny, R., Mott, B., and Lester, J., (2018, May). Toward Automated Scenario Generation with Deep Reinforcement Learning in GIFT. In Proceedings of the 6th Annual Generalized Intelligent Framework for Tutoring (GIFT) Users Symposium. *US Army Research Laboratory*, Orlando, FL.

Shiffman, D. (2012). The Nature of Code: Simulating Natural Systems with Processing. Daniel Shiffman.

Simpson E.J. (1972). The Classification of Educational Objectives in the Psychomotor Domain. Washington, DC: Gryphon House.

Sinatra, A.M. & Sottilare, R. (2016). Chapter 14 – Exploring the Diversity of Domain Modeling for Training and Educational Applications. In R. Sottilare, A. Graesser, X. Hu, A. Olney, B. Nye & A. Sinatra (Eds.) Design Recommendations for Intelligent Tutoring Systems: Volume 4 – Domain Modeling. *US Army Research Laboratory*, Orlando, Florida. ISBN: 978-0-9893923-9-6 (digital version).

Sottilare, R. A., Brawner, K. W., Goldberg, B. S., & Holden, H. K. (2012). The generalized intelligent framework for tutoring (GIFT). Orlando, FL: US Army Research Laboratory–Human Research & Engineering Directorate (ARL-HRED).

Sottilare, R. A., Brawner, K. W., Sinatra, A. M., & Johnston, J. H. (2017). An updated concept for a Generalized Intelligent Framework for Tutoring (GIFT). GIFTtutoring. org.

Sottilare, R. & Brawner, K. (2018a, March). Exploring Standardization Opportunities by Examining Interaction between Common Adaptive Instructional System Components. In Proceedings of the *First Adaptive Instructional Systems (AIS) Standards Workshop*, Orlando, Florida.

Sottilare, R. & Brawner, K. (2018b, June). Component Interaction within the Generalized Intelligent Framework for Tutoring (GIFT) as a Model for Adaptive Instructional System Standards. In the Adaptive Instructional System (AIS) Standards Workshop of the *14th International Conference of Intelligent Tutoring Systems (ITS)*, Montreal, Quebec, Canada, June 2018.

Sottilare, R.A. & Ososky, S. (2017, July). Defining Complexity in the Authoring Process for Adaptive Instruction. In *Foundations of Augmented Cognition* (pp. 237-249). Springer International Publishing.

US Army (2007). Field Manual No. 3-21.8 - The Infantry Rifle Platoon and Squad. Department of the Army Washington, DC, 28 March 2007

Vygotsky, L. (1987). Zone of proximal development. *Mind in society: The development of higher psychological processes*, 5291, 157.

Wikipedia. (2018). Brute-Force Search. Retrieved from: https://en.wikipedia.org/wiki/Brute-force_search

Zook, A., Lee-Urban, S., Riedl, M.O., Holden, H.K., Sottilare, R.A., & Brawner, K.W. (2012, May 29-June 1). Automated scenario generation: Toward Tailored and Optimized Military Training in Virtual Environments. Paper presented at the *International conference on the Foundations of Digital Games*, Raleigh, NC.

AUTHOR BIOGRAPHY

Dr. Robert A. Sottilare is the Adaptive Training Research Lead at the US Army Natick Soldier Center where the focus of his research is automated authoring, instructional management, and analysis tools and methods for intelligent tutoring systems (ITSs). He is a co-creator of the Generalized Intelligent Framework for Tutoring (GIFT), an open source, AI-based adaptive instructional architecture. He is also the lead editor for the Design Recommendations for Intelligent Tutoring Systems book series and founder/chair of the GIFT Users Symposia. Dr. Sottilare is a member of the IEEE and AI in Education Society. He leads an IEEE working group for adaptive instructional system (AIS) standards. He is widely published and is a frequent speaker at the Defense & Homeland Security Simulation (DHSS) Workshop, the Florida AI Research Society, Augmented Cognition, and AI in Education conferences. Dr. Sottilare is also a faculty scholar and adjunct professor at the University of Central Florida where he teaches a graduate level course in ITS design. He is also a frequent lecturer at the United States Military Academy (USMA) where he teaches a senior level colloquium on ITS design. He is the recipient of two lifetime achievement awards in Modeling & Simulation: US Army RDECOM (2012; inaugural recipient) and National Training & Simulation Association (2015; Governor's Award – highest level).