In Multiagent-based Simulation IX, Lecture Notes in Artificial Intelligence 5269, Springer, 2009.

Using simulation to evaluate data-driven agents

Elizabeth Sklar[1,2] and Ilknur Icke[2]

[1] Dept of Computer and Information Science,
Brooklyn College, City University of New York
2900 Bedford Ave, Brooklyn, NY 11210 USA
sklar@sci.brooklyn.cuny.edu
[2] Dept of Computer Science,
The Graduate Center, City University of New York
365 Fifth Avenue, New York, NY 10016 USA
iicke@gc.cuny.edu

Abstract. We use simulation to evaluate agents derived from humans interacting in a structured on-line environment. The data set was gathered from student users of an adaptive educational assessment. These data illustrate human behavior patterns within the environment, and we employed these data to train agents to emulate these patterns. The goal is to provide a technique for deriving a set of agents from such data, where individual agents emulate particular characteristics of separable groups of human users and the set of agents collectively represents the whole. The work presented here focuses on finding separable groups of human users according to their behavior patterns, and agents are trained to embody the group's behavior. The burden of creating a meaningful training set is shared across a number of users instead of relying on a single user to produce enough data to train an agent. This methodology also effectively smooths out spurious behavior patterns found in individual humans and single performances, resulting in an agent that is a reliable representative of the group's collective behavior. Our demonstrated approach takes data from hundreds of students, learns appropriate groupings of these students and produces agents which we evaluate in a simulated environment. We present details and results of these processes.

1 Introduction

Most work that lies at the intersection of education technology and agent-based systems employs agents within intelligent tutoring systems as knowledgeable, automated teachers. Other work has explored the notion of simulated students [1–3] as a means to better understand the processes that underpin human learning by constructing models based on theories from pedagogical and/or cognitive science literature. Some of our earlier work deployed simple agents as learning peers in simple games, where the agent controllers were built from data collected in previously played games [4]. The current work extends these ideas by using data from an on-line educational assessment environment to train agents, and we employ simulation as a means for evaluating the agents we derive.

We are not the first to suggest constructing agents to emulate human behaviors in on-line systems [5–7]. Previous work has shown that training agents to emulate humans produces better results if the training set is a composite of multiple humans grouped according to application-specific metrics [8]. A large data set is generally desirable when training agents, and grouping human data sets with similar characteristics helps smooth out anomalies. In the study presented here, we apply data-mining techniques to interaction logs from a dynamic, on-line educational assessment environment in order to find effective groupings within the human data set. We then use the data groups to train agents that can be deployed in a simulation environment where each agent effectively embodies a *user model* that is characteristic of the humans in its training group. Related work has been reported previously (e.g., [9–11]), but within different user modeling contexts and not with the purpose of subsequently using the models to control agents in a simulation environment.

Our procedure is as follows. The first step is to partition the interaction data set that will serve as the basis for training a suite of distinct, agent-based, probabilistic controllers. This step is crucial to the success of each agent as an emulator of a specific class of human behaviors. If the data used to train the agents is not well defined, then there may be two problems. First, the behavior of individual agents may have too much variation and so an agent would be an unreliable model of the human users in its group. Second, the agents in the suite may not be distinct enough from each other (i.e., their behavior patterns may overlap significantly) and so an agent would be unreliable as a model of only the users in its group. In this paper, we compare different techniques for partitioning the data, using standard clustering methods from the data mining literature. The second step is to produce an agent for each cluster whose actions typify behaviors characteristic of members of the cluster. The final step is to evaluate the results by placing all the agents we generated in a simulated assessment environment and by measuring which agents most closely resemble the human counterparts they are meant to be simulating. Our results demonstrate that one clustering technique clearly produces a better approximation of group behavior patterns than the others, as evaluated through simulation.

2 Our approach

Figure 1 contains a graph illustrating a generic landscape for an interactive environment. Imagine that each node represents a *state* in the environment, such as a page in a web site, a question in an on-line test or a room in a computer game. The red and green links between nodes indicate paths that users can follow based on actions they take in the interactive environment. For example, clicking on a hyperlink will send a user to another web page, or answering a question correctly will send a user to another question, or earning game points will send a user to another room. All users start their interactions with the system in the same place—at the node on the far left. A sequence of user actions translates

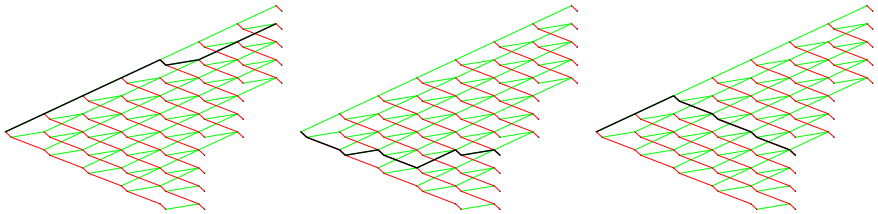


Fig. 1. Sample trajectories.

into a *trajectory* through the landscape. Three sample trajectories are drawn with heavy black lines in the figure.

In fact, the figure is modeled after an educational assessment environment. Each node represents a question in the assessment, and links indicate possible paths from one node to another. All students start at the same node, the leftmost one in the figure. Students who answer a question correctly move along the green link (up and to the right). Students who answer a question incorrectly move along the short red link (down and to the right). Thus Figure 1 illustrates the paths taken by three different students. Student A (left side of the figure) did well, answering most questions correctly. Student B (middle of the figure) did poorly at first, rallied, slumped, rallied, and slumped again to the end of the assessment. Student C (right side of the figure) started off well, but then made a series of mistakes. The differences in performance of these three students is clearly visible. Notice in particular that while students B and C ended up at the same node at the end of the assessment, they took very different paths through the landscape.

Most assessments report results and group students using the final, or "exit", score achieved. In our example, this corresponds to the last node visited, on the right edge of the landscape. However, with the advent of dynamic, on-line testing, a much richer data set is available and so reports can provide more information about student performance than simply an exit score. We believe that in environments like the one that corresponds to our sample data set, a lot can be learned from examining the trajectories taken through the landscape. So, we focus on trajectories and experiment with techniques for grouping students according to similarities in their trajectories. It is important to note that the students do not make directed choices about which paths to take, but rather the system chooses each next node in response to the student's performance up to that point. As mentioned earlier, we wish to create a suite of agents that each mimic certain classes of human behavior. Section 2.1 describes several methods for partitioning the complete data set into clusters, grouping humans with similar behavioral characteristics, as exhibited by following similar trajectories through the landscape. The experiments conducted here are based on a data set of 117 students who had accessed the assessment environment in 2006.

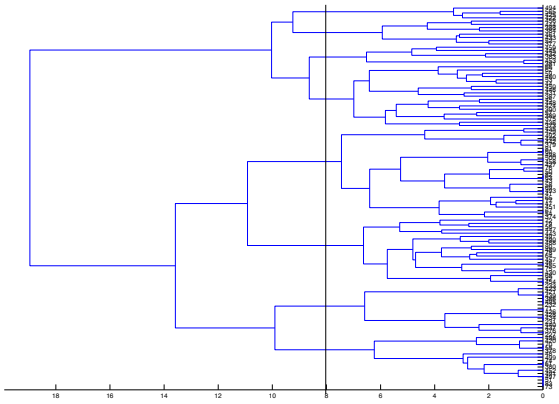


Fig. 2. Hierarchical clustering of student response vectors, using Euclidean distance with 012 coding. The vertical axis identifies individual students. The horizontal axis contains h values.

2.1 Partitioning training data

Data clustering is a well-studied field in the literature, and the particular algorithm chosen for a clustering task varies depending on the characteristics of the data set and the goals of the task. Our aim is to produce a coherent grouping that will serve to train a suite of agents that are distinct from each other. We investigated a variety of techniques and here we compare two types: Euclidean distance based on feature vectors and Hausdorff distance based on geometric similarity.

Euclidean distance. We generated "feature vectors" to encode the student responses in the assessment. For the landscape illustrated in Figure 1, we generated 94-dimensional feature vectors for each student, each dimension representing a student's response to one question. Note that sequences of questions are chosen for students dynamically, based on the their performance and the connections defined in the landscape; so students do not (and can not) visit every node. There are three possible results for any question: *correct*, *incorrect* or *not seen*. We experimented with different ways to encode results including: 0=incorrect, 2=correct, 1=not seen (021 coding), −1=incorrect, 1=correct, 0=not seen (-110 coding), 0 =incorrect, 1=correct, 2=not seen (012 coding), and also a 3-variable coding: seen ($\{1|0\}$), incorrect($\{1|0\}$), correct ($\{1|0\}$). We employed a hierarchical clustering algorithm in Matlab [12] using the Euclidean distance between the feature vectors as the distance measure. Note that according to the metrics described in the remainder of this paper, the feature vector encoding that produced the best results is the 012 coding. Thus, for the sake of brevity, we only present the Euclidean 012 coding results here.

Hausdorff distance. Since our aim is to group students according to the similarity of trajectories followed on the assessment maps, we decided to employ

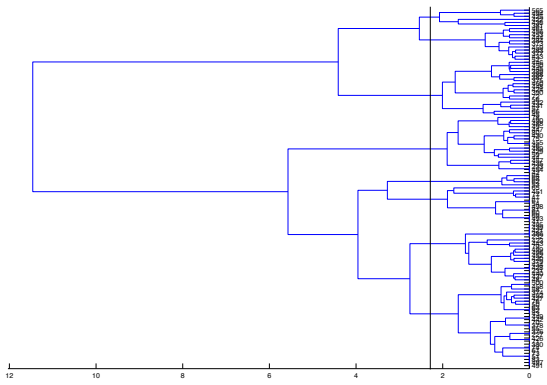


Fig. 3. Hierarchical clustering of student trajectories, using Hausdorff distance. The vertical axis identifies individual students. The horizontal axis contains h values.

clustering techniques on sequential data. Similar work has been reported on classifying Linux users with respect to their experience level based on command logs [13] and on clustering financial time series data [14]. We generated the landscape illustrated in Figure 1 by assigning coordinates to each question in the educational assessment, and we used the Hausdorff Distance to compute the dissimilarity between any two paths. After computing the pairwise distances between each path, we applied a hierarchical clustering algorithm in Matlab.

2.2 Comparing clusterings

We compare the results of different clustering algorithms by measuring the distance between data points within each cluster and the separation between clusters, seeking to minimize *intra*-cluster differences (the level of similarity within a cluster) and maximize *inter*-cluster differences (the amount of dissimilarity from one cluster to another).

One way of comparing clustering results is using a type of figure called a *dendrogram*. A dendrogram consists of brackets that connect objects hierarchically. The height (h) of each bracket indicates the distance between any two objects (or groups of objects) being connected. Figures 2 and 3 show dendrograms for the Euclidean 012 coding and Hausdorff methods, respectively. The Euclidean 012 method found 8 main clusters at level $h = 8$ while the Hausdorff method found 8 main clusters at level $h = 2$. Because lower h values indicate less difference amongst group members, the Hausdorff method has better results.

Another way we compare the clustering results is with metrics that have been demonstrated to compare clusters of trajectories in related work [15]. Two metrics are employed. The first is "shape complexity", or σ, which is computed

Hausdorff method

cluster	size	points	σ	*cov*
1	11	121	2.64025	5687.83
2	6	64	2.42758	3871.61
3	19	203	2.22529	6487.44
4	28	298	4.43650	6841.44
5	4	37	4.89523	6909.63
6	12	108	4.55438	5109.01
7	18	197	4.91468	8704.28
8	19	206	5.40067	8579.10
average			3.93682	6523.79

Euclidean 012 method

cluster	size	points	σ	*cov*
1	6	68	4.87123	8353.60
2	19	204	8.83936	15166.20
3	4	41	2.20674	4062.49
4	8	89	2.14947	7424.48
5	16	175	3.29415	4157.30
6	15	159	1.60082	4469.70
7	21	223	7.63581	12024.50
8	28	275	7.57357	7320.25
average			4.77139	7872.31

Fig. 4. Cluster similarity measures, showing for each cluster: the number of trajectories in the cluster, the total number of points covered by all the trajectories in the cluster, the standard deviation for σ and the standard devation for *cov* (see text for further explanation).

as:

$$\sigma = disp/length$$

where *disp* is the displacement or the distance between the first and last points in a trajectory and *length* is the number of points in the trajectory. The second is "divergence", or covariance of the first, middle and last points in the trajectory. Figure 4 compares these values for the Euclidean 012 and Hausdorff methods. The size of each cluster (number of students belonging) is shown as well as the average number of points in the trajectories of all student members. The columns to focus on are the two rightmost, which contain the standard deviation of σ and *cov* for the trajectories that comprise each cluster. The absolute numbers are not important here; what is important is the relationship between the numbers within each column. Smaller numbers indicate tighter coherence amongst cluster members—this is our aim. The average σ for the Hausdorff is 3.95, whereas for Euclidean, the average $\sigma = 4.77$. The average covariance for Hausdorff is 6523.79, whereas for Euclidean, the average $cov = 7872.31$. Using these metrics, the Hausdorff distance clustering technique results in better coherence. We note that a cluster-by-cluster comparison reveals that the Hausdorff coherence is better for the clusters on either end of the spectrum, while the Euclidean is better for those in the middle. This is an interesting result which bears further investigation.

One additional factor to take into account with the clustering methods is the choice of number of clusters. The more clusters there are, the better the coherence amongst cluster members (and the smaller the standard deviation of our two metrics). However, there is a trade-off: perfect coherence can be achieved with clusters of size 1, but being satisified with single-member clusters implies that clustering is not needed at all. Thus, a balance must be achieved between the number of clusters and the coherence. Figure 5 illustrates the change σ and *cov* as the number of clusters decreases. The values in the figure were computed for the Hausdorff methodology (which will be shown later to be the best overall

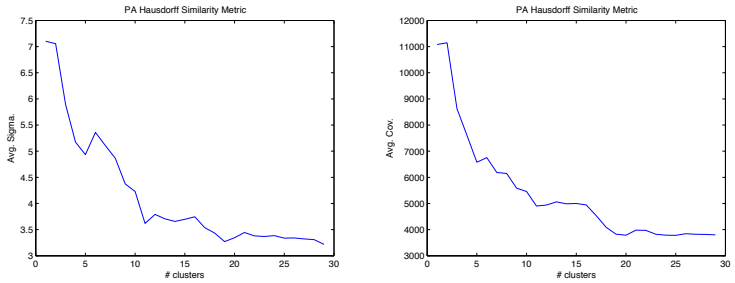


Fig. 5. Change in σ and *div* as number of clusters decreases (Hausdorff clustering algorithm)

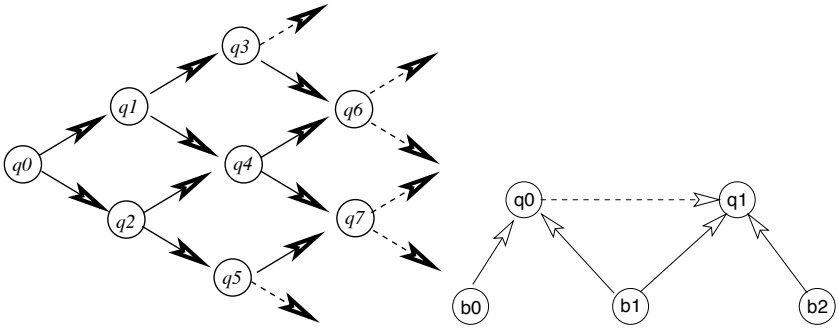
choice of clustering algorithm for our problem). For both metrics, when the number of clusters reaches 5 and again at 10, the angle of decline becomes less sharp. To allow for fair comparison between algorithms, we chose a standard number of clusters for each algorithm: 8.

Recall that our overall goal is to generate clusters that will provide training sets for a suite of agents, where each agent effectively embodies a user model that is characteristic of the humans in its training group. Section 2.3 describes how we trained agents based on the clustered data. Section 2.4 presents evaluation results that demonstrate the effectiveness of the clustering and training procedure.

2.3 Training agents

The next step in our procedure is to create agents whose behavior typifies that of each cluster. We did this for clusters generated by the Hausdorff and all Euclidean coding methods (though we only report here on the best coding, 012). First, we generated a profile for each cluster as follows. For each node in the landscape, we tally the number of students in the cluster who visited that node. We compute statistics based on students' responses to each node as the basis, aggregated for all members of a cluster into an agent training set. For each cluster, we generate one representative agent.

Each node represents a question in the assessment, and each question is designed to elicit information about students' abilities. The assessment is a multiple-choice test, and each possible incorrect answer is associated with one or more *bugs* that (likely) exist in a students' knowledge if s/he chose the corresponding incorrect answer. There is an overall mapping from each type of bug evaluated in the assessment to each node in the landscape. For example, take the simple landscape illustrated in Figure 6a. All students start at the node labeled $\mathsf{q0} \in Q$ in the figure. There are one or more bugs, each of which we will refer to as $\mathsf{b}_j \in B$, that each node (question q_i) is assessing. Essentially a table with $|Q|$ rows by $|B|$ columns is engineered when the assessment is designed, assigning a



(a) example landscape (b) example influence diagram

Fig. 6. Agent training structures.

Boolean value to each cell in the table indicating which errors are revealed by each question. We use this table to pose two types of questions:

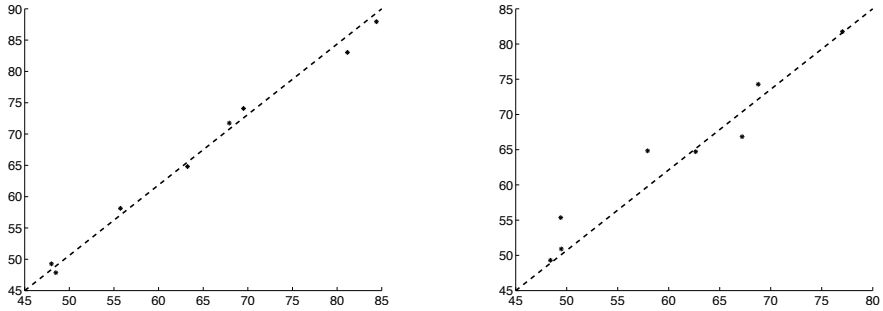
1. *a modeling question*—what is the probability that a student possesses bug b_j, given that they answered question q_i incorrectly, i.e., what is $Pr(b_j|q_i)$?
2. *a prediction question*—what is the probability that a student will answer question q_i incorrectly, given that they possess bug b_j, i.e., what is $Pr(q_i|b_j)$?

Note that there is not a one-to-one correspondence between bugs and questions. Thus we rephrase our two questions:

1. *modeling*—what is the probability that a student possesses the bugs in set B, given that they answered the questions in set Q incorrectly?
2. *prediction*—what is the probability that a student will answer the questions in set Q incorrectly, given that they possess the bugs in set B?

The influence diagram [16, 17] shown in figure 6b provides a graphical illustration of this situation. There are two types of variables represented: bugs $\{b_0, b_1, b_2\}$ and questions $\{q_0, q_1\}$. Question q_0 is designed to assess whether a student possesses the bugs in set $\mathrm{B}' = \{b_0, b_1\}$; question q_1 is designed to assess whether a student possesses the bugs in set $\mathrm{B}'' = \{b_1, b_2\}$.

We use the interaction data set described earlier along with the influence diagram associated with the landscape contained in Figure 1 to compute probability tables, one per cluster. We tally the number of students within the cluster who visited each node and the percentage of them who answered the question incorrectly, indicating particular bugs. Thus, for each cluster, we have a table that indicates how likely it is that a member of that cluster possesses each bug. This is essentially a user model which becomes the heart of the control function for each "cluster agent."



(a) Hausdorff (b) Euclidean (012 coding)

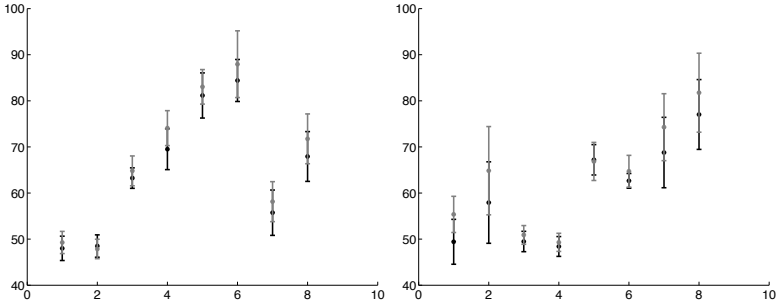
Fig. 7. Correlation between agents and humans. The x-axis represents σ values for the human trajectories within each cluster and the y-axis represents σ values for the trajectories of the corresponding agents.

2.4 Evaluating agents

Finally, we evaluate the efficacy of our methodologies by simulating an assessment using each agent generated. We performed 999 evaluation trials for each agent—since the agents are controlled probabilistically, they will not behave exactly the same way in each simulation run. We evaluate our efforts in four ways. First, we wish to determine how well each agent fits the cluster profile from which it was modeled. The correlation between the behaviors of the trained agents and the groups of humans the agents are emulating is illustrated. Second, we highlight the separation between agents (i.e., distinctiveness of behavior patterns). Third, we show that the method of training agents as emulators, rather than taking the raw human data as the basis of user models, provides more coherent results. Fourth, we provide visualizations that compare compounded agent trajectories with humans in corresponding clusters.

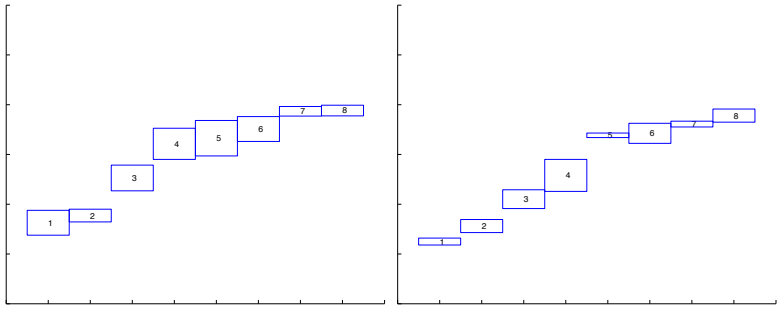
Correlation between agents and humans. We use the shape complexity (σ) metric described earlier to compare the relationship between trajectories generated by agents with those generated by humans. Figure 7 plots the average σ for each cluster based on the trajectories exhibited by humans (x-axis) against the corresponding value for trajectories generated by the agent representing each cluster in the 999 evaluation runs. The correlation with the Hausdorff technique is quite high.

Separation between agent behaviors. We also want to compare the separation between agents, recalling our goal to produce a suite of agents, each of which represents different behavior patterns. Figure 8 shows the mean and standard deviation of σ for each of the clusters computed for each clustering technique. The black bars represent the clusters based on human trajectories; the grey bars represent the trajectories generated by the agents in the 999 evalu-



(a) Hausdorff (b) Euclidean (012 coding)

Fig. 8. Separation between agents, using σ.



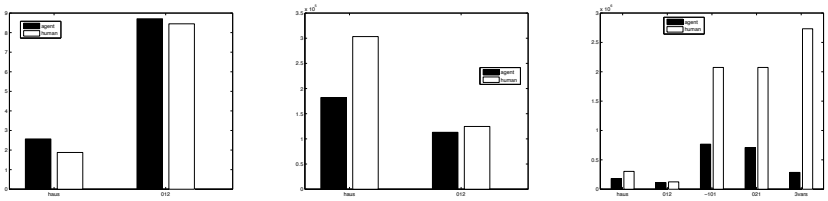
5 overlaps, 35% 4 overlaps, 15%

(a) Hausdorff (b) Euclidean (012 coding)

Fig. 9. Separation between agents, using y values.

ation runs. Once again, the Hausdorff produces superior results because greater distinction between each cluster can be seen in the lefthand plot.

We also examine a different mode of comparison in order to illustrate more clearly the distinction between cluster-based behaviors. Instead of looking at σ, we examine the mean and standard deviation of y values for the trajectories within each cluster. The landscape is shaped such that all trajectories extend horizontally along the same range of x values, but do not cover the same extent vertically. We compare the vertical extent of each cluster by plotting a box outlining one standard deviation around the mean y value. These plots are shown in Figure 8 for both Hausdorff and Euclidean 012 methods. According to this comparison, the Euclidean 012 method appears to perform better than the Hausdorff because it has fewer boxes that overlap and the total amount of overlapping area is less. However, in the Euclidean 012 plot, agent 5's extent is completely subsumed by that of agent 6, which is a worse result. Another disadvantage of this plotting method is that it does not represent the complete picture because



comparison using σ comparisons using y values

(a-b) Hausdorff vs Euclidean 012 (c) All methods

Fig. 10. Coherence with agents, as compared to humans. Plot (a) uses σ for comparison. Plots (b) and (c) use the average variance of y-values for comparison.

the distribution of y values, weighted by frequency of occurrence, is not normal and more closely matches a Poisson distribution. Thus, we are exploring other methods of comparing separation between agent behaviors.

Coherence within agent behaviors. Figure 10 illustrates the coherence amongst the behaviors of an agent versus humans within a single cluster. Because the agents behave probabilistically, based on the influence diagram and table explained in Section 2.3, a single agent will act (slightly) differently each time it executes in the simulated assessment. The variance of y values is examined for each agent. The smaller the variance of y values, the more coherence there is in the agent's behavior. The lefthand plot (a) compares the Hausdorff and Euclidean 012 methods. The Euclidean 012 method produces better coherence. The plot also compares coherence in the agent's behavior to the coherence across the set of trajectories belonging to the humans in the cluster. In both cases (Euclidean 012 and Hausdorff), there is better coherence in the agent behaviors. This is what we have been striving for. The improvement in coherence is even more marked in the righthand plot (b), which compares the Hausdorff and all four Euclidean coding methods. Clearly the Euclidean 012 method produces superior results.

Visual comparison. Finally, a sampling of trajectories for each cluster and corresponding agent are shown in Figure 11, using the Hausdorff method, and in Figure 12, using the Euclidean 012 method. For each row in the figures, the first (leftmost) plot shows the trajectories (blue lines) over 999 test runs of the agent. The remaining plots in the row show a representative sample of human student trajectories (black lines) for each cluster. The point is that the blue lines represent a composite set of black lines within the same cluster.

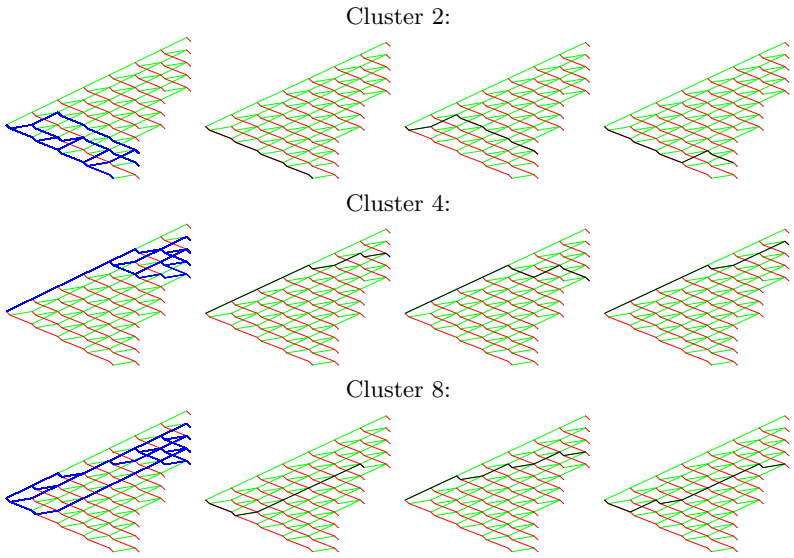


Fig. 11. Comparing agent and human trajectories, clustered using Hausdorff method.

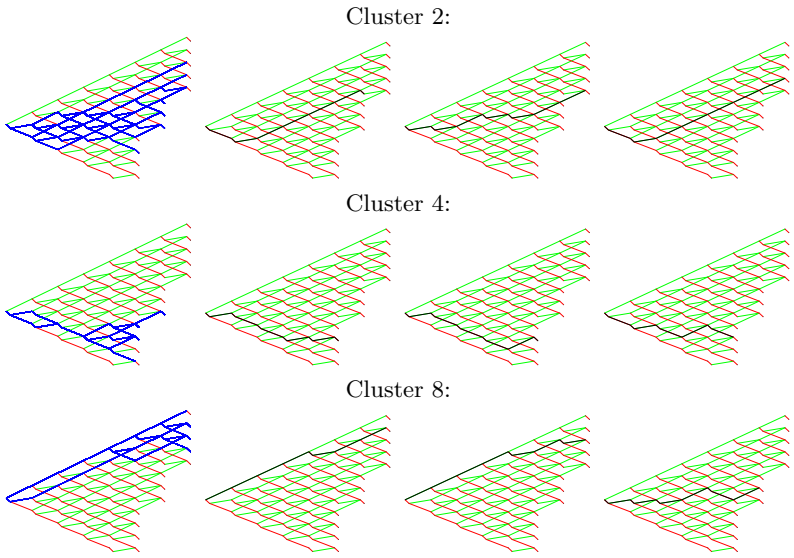


Fig. 12. Comparing agent and human trajectories, clustered using Euclidean 012 method.

3 Conclusion

We have described a methodology for generating agent-based simulations of human behavior in a structured interactive environment. We employed interaction data from an on-line educational assessment environment and created clusters of students with similar behaviors, and then trained agents whose actions typify cluster members. We explored two methods of clustering, one based on a feature-vector comprised of right/wrong answer choices made by each student and employing a Euclidean distance metric to determine groupings. The second is a graphical approach, based on examining the paths students take through the underlying landscape of the assessment and employing a Hausdorff distance metric to determine groupings. From these, we generated a profile for each cluster based on bugs in student knowledge exhibited by the assessment responses. We used these profiles to train agents to emulate cluster members, and finally, we evaluated the efficicacy of these methods by comparing the trajectories produced by agents acting in a simulated assessment to those of humans produced in the real assessment. Our results show that when we use shape complexity (σ) as the basis for comparison, the Hausdorff method is superior to the Euclidean methodologies. Interestingly, attempts to make comparisons based on other metrics, such as the variation in y values aligns the Hausdorff and Euclidean 012 methods more closely. Still, however, the Euclidean 012 method always produces better results than the other Euclidean encodings explored.

Our current work involves extending these methods to other types of data, both from within the education sector and outside it. The type of generic landscape illustrated in Figure 1 can be used to represent the underlying structure of a wide range of interactive environments. Being able to generate agents that emulate human behavior in such environments has broad application and can be used not only for evaluating clustering techniques, as illustrated here, but also for producing controllers for agents that might be deployed as actors within such interactive environments.

Acknowledgments

This work was partially supported by the National Science Foundation under NSF IIP #06-37713 and by the US Department of Education under #ED-07-R-0006.

References

1. VanLehn, K., Ohlsson, S., Nason, R.: Applications of simulated students: An exploration. Journal of Artificial Intelligence in Education **5**(2) (1996) 135–175
2. Sklar, E., Davies, M.: Multiagent simulation of learning environments. In: Fourth International Conference on Autonomous Agents and Multi Ag ent Systems (AAMAS). (2005)

3. Spoelstra, M., Sklar, E.: Using simulation to model and understand group learning. Agent Based Systems for Human Learning, International Transactions on Systems Science and Applications **4**(1) (2008)
4. Sklar, E.: CEL: A Framework for Enabling an Internet Learning Community. PhD thesis, Department of Computer Science, Brandeis University (2000)
5. Cypher, A.: Eager: Programming repetitive tasks by example. In: Proceedings of CHI'91. (1991)
6. Maes, P.: Agents that reduce work and information overload. Communications of the ACM **37**(7) (1994) 31–40,146
7. Balabanović, M.: Learning to Surf: Multiagent Systems for Adaptive Web Page Recomendation. PhD thesis, Stanford University (1998)
8. Sklar, E., Blair, A.D., Pollack, J.B.: Chapter 8: Training Intelligent Agents Using Human Data Collected on the Internet. In: Agent Engineering. World Scientific, Singapore (2001) 201–226
9. Hofmann, K.: Subsymbolic user modeling in adaptive hypermedia. In: The 12th International Conference on Artificial Intelligence in Education, Young Researcher Track Proceedings. (2005) 63–68
10. Mavrikis, M.: Logging, replaying and analysing students' interactions in a web-based ILE to improve student modeling. In: The 12th International Conference on Artificial Intelligence in Education, Young Researcher Track. (2005) 101–106
11. Merceron, A., Yacef, K.: Educational data mining: a case study. In: The 12th International Conference on Artificial Intelligence in Education. (2005)
12. `http://www.mathworks.com/products/matlab/`
13. Jacobs, N., Blockeel, H.: User modeling with sequential data. In: Proceedings of the 10th International Conference on HCI. (2003) 557–561
14. Basalto, N., Bellotti, R., De Carlo, F., Facchi, P., Pascazio, S.: Hausdorff clustering of financial time series. Physica A **379** (2007) 635–644
15. Zhang, Z., Huang, K., Tan, T.: Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. In: ICPR (3). (2006) 1135–1138
16. Pearl, J.: Probabilistic reasoning in intelligent systems: Networks of plausible inference. Morgan Kaufmann, San Mateo, CA (1988)
17. Russell, S., Norvig, P.: Artificial intelligence: A modern approach. 2nd edn. Prentice Hall (2002)