Proceedings of the 2017 Winter Simulation Conference
W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, eds.

AGENT-BASED MODEL CONSTRUCTION USING INVERSE REINFORCEMENT LEARNING

Kamwoo Lee
Mark Rucker
William T. Scherer
Peter A. Beling
Matthew S. Gerber
Hyojung Kang

Department of Systems and Information Engineering
University of Virginia
Charlottesville, VA 22904, USA

ABSTRACT

Agent-based modeling (ABM) assumes that behavioral rules affecting an agent's states and actions are known. However, discovering these rules is often challenging and requires deep insight about an agent's behaviors. Inverse reinforcement learning (IRL) can complement ABM by providing a systematic way to find behavioral rules from data. IRL frames learning behavioral rules as a problem of recovering motivations from observed behavior and generating rules consistent with these motivations. In this paper, we propose a method to construct an agent-based model directly from data using IRL. We explain each step of the proposed method and describe challenges that may occur during implementation. Our experimental results show that the proposed method can extract rules and construct an agent-based model with rich but concise behavioral rules for agents while still maintaining aggregate-level properties.

1 INTRODUCTION

Agent-based models use a bottom-up approach to discover complex, aggregate-level properties. These properties emerge from individual agent behaviors and interactions within the environment. This is in contrast to top-down simulation paradigms such as system dynamics and discrete-event simulation. Thus, an inevitable assumption of ABM is that behavioral rules for agents are known to the modeler. Unfortunately, discovering these rules can be challenging, potentially requiring deep insight and domain knowledge.

As we describe in section 2, there have been recent attempts to obtain individual rules for agent-based models directly from data, which are increasingly available in many domains. Inverse reinforcement learning (IRL) (Ng and Russell 2000) is a candidate for such a data driven approach. Instead of directly learning actions for each state, IRL first recovers motivations that explain an observed sequence of actions and then generates behavioral rules from those motivations. In this paper, we propose a method to build an agent-based model with the help of IRL. Through an experiment, we show that the method provides a newly constructed agent-based model with rich but concise behavioral rules (IRL policies) for agents while still maintaining aggregate-level properties. The contribution of this paper is to demonstrate that the combination of individually learned rules and macro-level simulation can provide a test bed for IRL research while creating new options for ABM construction.

In section 2 we review key concepts and recent research in ABM and IRL. In section 3, we present the details of the proposed method. Sections 4 and 5 present an experiment and results using the proposed method on a dynamic segregation phenomenon inspired by Schelling (1971). In section 6 we discuss the challenges of the method, and we conclude the paper by highlighting areas of future research in section 7.

2 LITERATURE REVIEW

2.1 Agent-Based Modeling

ABM is widely used to study complex systems that consist of heterogeneous, autonomous agents who interact with each other and the environment. Applications of ABM include those in social sciences (Axelrod 1997), computational economics (Tesfatsion 2002), and marketing (Negahban and Yilmaz 2014). In ABM each agent takes actions based on its assessment of the environment and its predetermined goals. In this regard, ABM has many benefits over other modeling or simulation techniques (Bonabeau 2002). However, it is often difficult to develop a reliable agent-based model since one needs to develop agents' behavioral rules through a qualitative understanding of the domain and careful calibration of agent and environmental parameters. Moreover, traditional ABM practice relies heavily on expert opinion or qualitative comparisons of behavior to validate a model (Heath et al. 2009).

To overcome these issues, there have been many attempts to incorporate machine learning (ML) techniques and empirical data into ABM methods. Torrens et al. (2011) use locally weighted regression in connection with k-nearest neighbor and k-means clustering algorithms to build a model of walking and movement behavior from real-world movement trajectories. Wunder et al. (2013) present a method for selecting a collection of competing behavioral models by training and validating models on empirical data from a series of public-goods games. Recently, Zhang et al. (2016) offered a framework for data-driven agent-based models where they use logistic regression to train models of heterogeneous agents' behaviors in rooftop solar adoption with key input factors of economic benefit and peer effects.

In contrast with the above research, we propose an approach that (1) recovers the behavioral motivations of agents from their observed behaviors and (2) derives behavioral rules from these motivations without any tuning parameters. This is in contrast to previous ML techniques that have emphasized directly finding behavioral patterns or tuning parameters for a model based on empirical data. Our motivation-based approach allows us to recover not just exhibited behavioral rules but also unobserved rules rooted in the agents' motivations.

2.2 Inverse Reinforcement Learning

Inverse reinforcement learning (IRL) (Russell 1998) is a problem within the reinforcement learning framework based on Markov decision processes (MDPs). Formally, an MDP is represented as a five-tuple (S, A, $P_\cdot(\cdot,\cdot)$, $R(\cdot,\cdot)$, γ) where S is the set states, A is the set of actions, $P_a(s,s')$ is the probability of moving from state s to s' after taking action a, $R(s,a)$ is the scalar reward for taking action a in s, and γ is the reward discount factor. In the IRL problem an MDP is considered fully known except for R, denoted as MDP$\backslash R$. The solution to this problem is the R that explains an observed trajectory of (s,a) pairs. Many machine learning approaches have been proposed to solve this problem: linear optimal reward (Ng and Russell 2000), maximum reward margins (Abbeel and Ng 2004, Ratliff et al. 2006), and maximum reward likelihood (Ramachandran and Amir 2007, Ziebart et al. 2008).

To date, IRL has been used to address problems as diverse as autonomous helicopter control (Abbeel et al. 2010), airplane pilot objective inference (Yokoyama 2017), measuring cultural preferences in negotiation (Nouri et al. 2012), and predicting human behavior (Qiao and Beling 2013, Liu et al. 2013, Osogami and Raymond 2013, Yang et al. 2015). Our contribution is to demonstrate the usefulness of IRL to not only learn individual agent behavior but also recreate aggregate behavior within agent-based models. We note, however, that IRL is not the only inductive programming technique for learning behavior from demonstrations (LfD) (Argall et al. 2009). Many approaches to this problem seek to map states directly to actions. IRL is distinguished by the presupposition that the reward function will always be the most succinct and transferable description of any behavior, and as such is a better approach to learning behavior than directly mapping states to actions (Abbeel and Ng 2004).

3 METHODOLOGY

Our method begins with observational data for agent trajectories and ends with aggregate behaviors generated by an agent-based model. The method is summarized graphically in Figure 1 below.

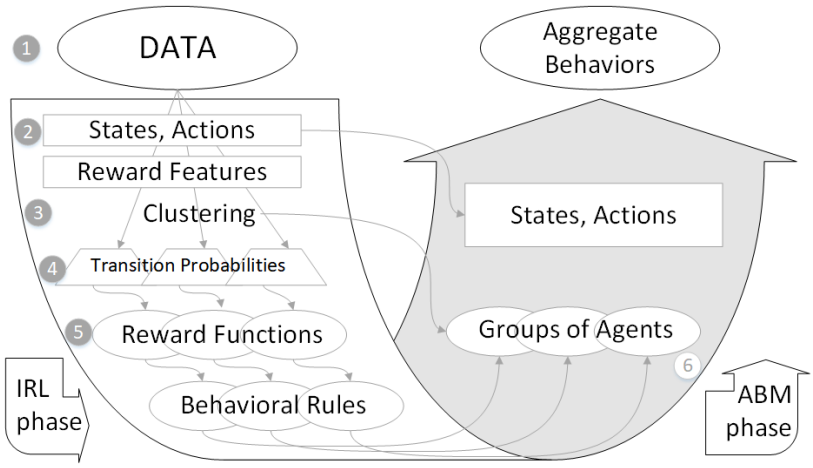


Figure 1: Method diagram illustrating the progression from observational data to aggregate behaviors generated by an agent-based model.

Specifically, the method consists of six steps (numbered 1-6 in Figure 1) and assumes that an agent's states, actions, and transition probabilities can be formalized as an MDP without a reward function (MDP$\backslash R$). Thus, the following steps are chosen to satisfy this assumption while also allowing the interchange of alternative techniques at each step.

1. Collect agent trajectory data
2. Design an MDP$\backslash R$ model
3. Cluster agents
4. Estimate transition probabilities for each cluster
5. Extract behavioral rules via IRL for each cluster
6. Construct agent-based model from extracted rules

3.1 Collect Agent Trajectory Data

The proposed method requires observations of sequential states and actions for a set of agents. Actions are necessary because IRL extracts behavioral rules from changes in the environment as well as agents' decisions. The ideal data for this method are generated sequentially by heterogeneous agents who pursue long-term goals while interacting with other agents and/or the environment. One example of such a data set is the one generated by Huang et al. (2017), where passive smartphone sensors tracked individuals' locations, micro-movements, and communication history for several weeks. The collected observations were shown to be descriptive enough to correctly identify individuals with high social anxiety.

3.2 Design an MDP$\backslash R$ Model

In step two, when designing an MDP$\backslash R$, states and actions should be selected to capture the essence of the underlying environment. Designing the MDP$\backslash R$ determines the complexity of the resulting model. It is desirable that the states and actions should be as parsimonious as possible while being able to distinguish semantically meaningful changes in the environment and an agent's actions. However, if the resulting

agent-based model has complex state-spaces, which can change over the course of a simulation, the state variables that cause the evolution should also incorporated into the design.

Since an IRL framework decomposes the reward function (R) into weights (w) and reward features (ϕ), the reward features (ϕ) should also be selected along with states and actions to formulate an IRL problem. The reward features (ϕ) are a set of basis functions that capture the structure of the reward function (Levine et al. 2010). Intuitively, they serve as a frame of R (a mold of reward in a figurative sense), and IRL later completes the MDP$\backslash R$ by recovering w using this frame. Specifically, if R is assumed to be a linear combination of a set of features, then the R of the MDP can be expressed as:

$$R(s) = w^\top \phi(s), \tag{1}$$

where s is state. Then the expected cumulative reward of a series of states given behavioral rules (π) is

$$\begin{aligned} \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R(s_t)|\pi\Big] &= \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t w^\top \phi(s_t)|\pi\Big] \\ &= w^\top \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t \phi(s_t)|\pi\Big] \\ &= w^\top \mu(\pi), \end{aligned} \tag{2}$$

where $\mu(\pi)$ is called the feature expectation. The above model for $R(s)$ is specific to a particular agent and its behavioral rules π. Thus, each agent has its own $\mu(\pi)$.

3.3 Cluster Agents

In step three, agents are clustered to achieve a desired level of agent diversity, ranging from perfectly homogeneous (one cluster) to perfectly heterogeneous (no clusters), while still retaining essential differences. Agents are clustered according to their feature expectations $\mu(\pi)$. The feature expectation is the cumulative discounted sum of the reward features (ϕ) as defined in Equation 2, which is aggregated through the sequence of states.

3.4 Estimate Transition Probabilities for Each Cluster

In step four, the state transition probabilities are estimated for each agent cluster. Since members of a cluster are motivated to respond to their environments in similar ways, we estimate a separate transition probability distribution for each cluster. Certain transition probabilities are forced by the environment, such as the change in position of a dropped object due to gravity; however, in many cases, it is better to estimate them through observation.

3.5 Extract Behavioral Rules via IRL for Each Cluster

In step five, IRL is applied to the agent trajectories belonging to each cluster, running in the MDP$\backslash R$ framework with the estimated state transition probabilities. Optimal behavior rules are then found to maximize the derived reward over time. The choices of IRL algorithm and behavioral rule extraction procedure depend on the characteristics of the data and assumptions of the model (our experiment in Section 4 will demonstrate one possibility). If the extracted rules appear unacceptably different from the observed behavior, steps two, three and four need to be re-evaluated. There are several potential reasons for excessive deviation. It is possible that an important variable may have been left out in the state, or the reward features may not effectively span the reward space. It is also possible that the behavioral clusters may have too much within-cluster variation, or the estimated transition probabilities may not accurately fit the true transition probabilities in the environment.

3.6 Construct Agent-Based Model from Extracted Behavioral Rules

Lastly, the recovered behavioral rules are used to construct an agent-based model. The transition from IRL to agent-based model has three parts:

1. Define states and actions in the model according to step 2.
2. Assign extracted behavioral rules to agent clusters.
3. In each iteration, agents observe their states and follow the assigned behavioral rules to change their state.

4 EXPERIMENT

To demonstrate the application and performance of the method described above, we (1) generated synthetic data using a known agent-based model of human segregation behavior, (2) constructed a corresponding agent-based model by applying steps 1-6 described in the previous section, and (3) drew quantitative and qualitative comparisons of the known and constructed agent-based models. We provide details of the experiment below.

4.1 Synthetic Data

We generated a data set in which agents exhibit a segregation pattern at the aggregate level. The data represent how the environment and actions that agents take change when they are located in a public space, moving and talking over time. Each agent is distinguished by an immutable, 2-level characteristic that is known to every other agent. This characteristic can be interpreted as ethnicity (Black/White), religion (Hindu/Muslim), or any other 2-level trait. Each agent takes one of four actions at each point in time, which is discretized into ticks:

1. Start a conversation with a nearby agent
2. Continue a conversation for another tick
3. Move a short distance (uniform distribution on 0-0.5 patches) in a random direction
4. Move a long distance (uniform distribution on 3-3.5 patches) in a random direction

A proximity radius (2 patches) is defined to identify nearby agents. Moving a short distance leaves an agent in the proximity radius whereas moving a long distance results in an agent leaving the radius. Agents have one of three behavioral patterns as follows:

1. Unbiased: Speak to anyone within proximity for a random length of time (uniform distribution on 1-10 ticks) and then randomly move a short or long distance.
2. Weak-bias: Speak to agents who share your trait value for five ticks and then move a short distance. Speak to agents who do not share your trait value for two ticks and then move a long distance.
3. Strong-bias: Speak to agents who share your trait value for ten ticks; speak to agents who do not share your trait value for 1 tick; and then always move a short distance.

Using NetLogo, we generated 20 sets of 50 observations for 700 agents according to the above specifications. Each set of data consists of 50 sequential actions with conversation length (0-10), similarity of recent partner (0/1), availability of potential partners nearby (0/1), and familiarity of environment (0/1).

4.2 Designing an MDP$\backslash R$

We constructed an MDP$\backslash R$ with four state variables and four actions as in Figure 2 below.

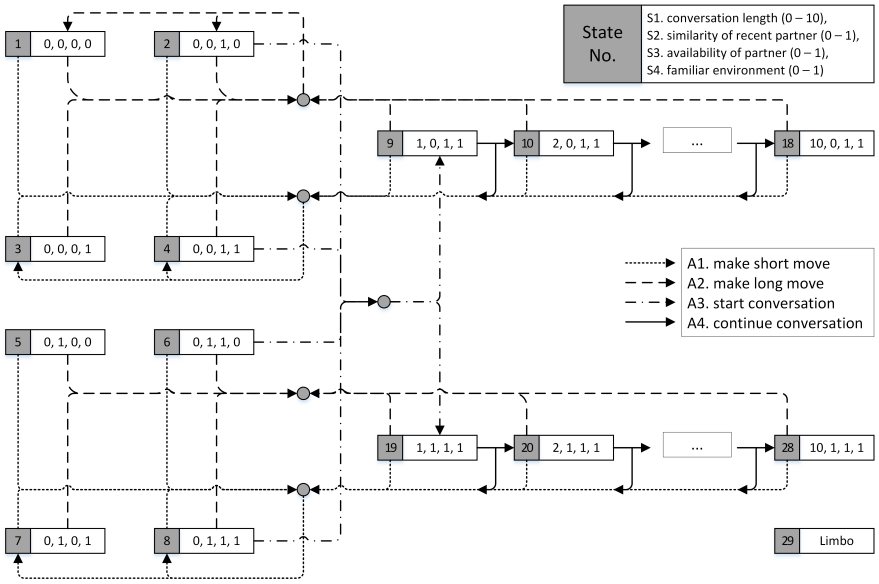


Figure 2: MDP diagram with legal state-action transitions.

Within this formulation, there are four types of illegal action:

1. Continue conversation while not having a conversation
2. Start or continue a conversation when there is no potential partner
3. Start or continue a conversation when it reaches the maximum conversation length
4. Start a conversation when having a conversation

Since IRL is an optimization process that starts with arbitrary behavioral rules, many early motivations inferred by the IRL algorithm target these illegal actions. We introduce an absorbing state (labeled "Limbo" in Figure 2) to deal with these illegal actions. By leading these actions to the absorbing state, we try to prevent the IRL algorithm from inferring the motivation of illegal actions. In this experiment, we set the reward feature space equal to the state space.

4.3 Clustering Agents

We calculated the vector of the expected sum of reward features ($\mu(\pi)$ in Equation 2) for each agent whose behavior was observed. Then we used hierarchical clustering analysis (HCA) with a cosine distance function and average linkage to create a dendrogram as shown below in Figure 3:

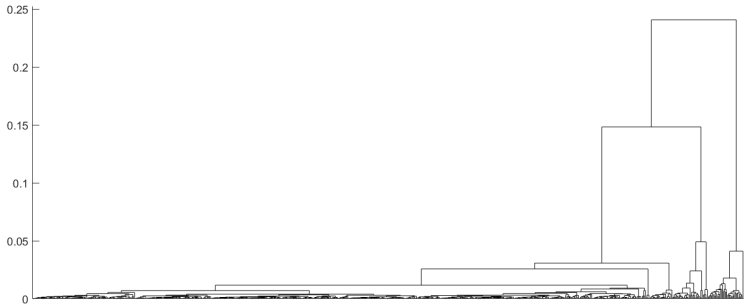


Figure 3: Hierarchical cluster analysis showing the biggest distance between 3 and 4 clusters.

We partitioned the agents by cutting the longest dendrogram legs, which resulted in three clusters. After choosing the number of clusters, we averaged the feature expectations to obtain a single feature expectation for each group. For the rest of the experiment, we treated a cluster as a group of homogeneous agents. For each of these homogeneous groups we determined transition probabilities, weights of rewards, and behavioral rules as described below.

4.4 Determining Transition Probabilities for Each Cluster

We estimated transition probabilities $P_a(s, s')$ for each combination of the 29 possible states (s, s') and four actions (a) of Figure 2. Each probability was estimated by observing the states agents land in after taking an action in an initial state. Using this estimate, not all transition probabilities are known because not all actions are observed for all states in the data set. To overcome this challenge, domain knowledge is used to appropriately copy observed transition probabilities to unobserved state-action pairs (e.g., taking a move action after talking for 10 ticks will be the same as taking a move action after talking for 9 ticks). For any remaining unassigned state-action pairs, the average transition probabilities of overall clusters were used in place of a single cluster.

4.5 Recovering Rewards and Behavioral Rules for Each Cluster via IRL

We used the projection algorithm described by Abbeel and Ng (2004) to obtain behavioral rules for each cluster. The output of this algorithm is not a single reward function, but a set of reward functions that conforms to the observed data. For each reward function, a set of behavioral rules is generated. Then an optimal mixture weight is assigned to each set so that the convex closure of the sets' feature expectations comes as close as possible to the feature expectations of the observed behavior. To achieve deterministic behaviors, we simply pick the behavioral rule set with the largest weight assigned to it. To emulate stochastic behavior, we randomly assign extracted rule sets to agents in equal ratio to the mixture weights. This method results in an agent cluster whose collective feature expectation is acceptably close to the data.

4.6 Constructing the Agent-Based Model

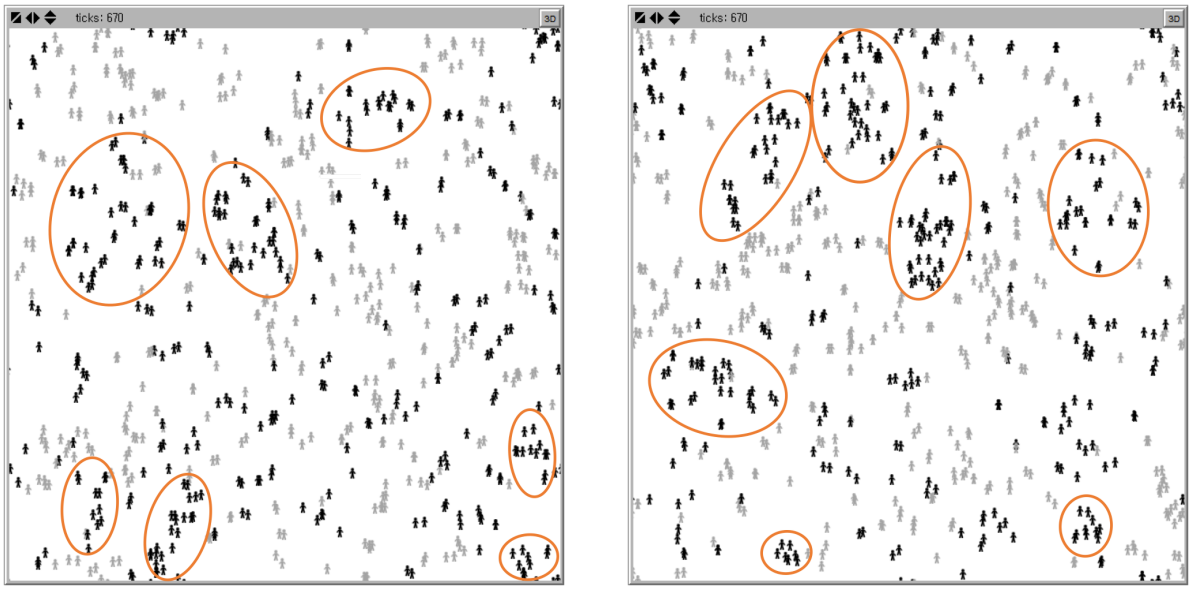
The IRL algorithm generates three sets of behavioral rules, which are used to construct the agent-based model. Using the previously defined states, actions, and features, as well as the behavioral rules recovered for three groups, we constructed the model by initializing states of the model, assigning the behavioral rules to each cluster of agents, and letting the agents observe their states and follow the rules in each iteration.

5 RESULTS

We first checked that the constructed agent-based model indeed represents the behaviors in the data set. Then we further analyzed the model to show that the constructed model can achieve its analytic purpose as an agent-based model. As shown below, the IRL-constructed agent-based model exhibits individual and aggregate behaviors that are similar to those generated by the known (baseline) agent-based model.

5.1 Assessment of the IRL-Constructed Agent-Based Model

We evaluated the IRL-constructed agent-based model by looking at qualitative and quantitative aspects of the model and the data set, focusing primarily on the aggregate-level behavior of agents. The model that generated the synthetic data served as a baseline model for comparison with the IRL-constructed model. Qualitatively, we observed the progress of segregation by visual comparison between the baseline and IRL-constructed models. Although the entire process of segregation cannot be presented here, we depict the similarity of segregation after the same number of ticks in Figure 4 below.



(a) Behavior in baseline model

(b) Behavior in IRL-constructed model

Figure 4: Visual comparison of baseline and IRL-constructed models.

For quantitative assessment, we defined spatial segregation and social segregation as follows:

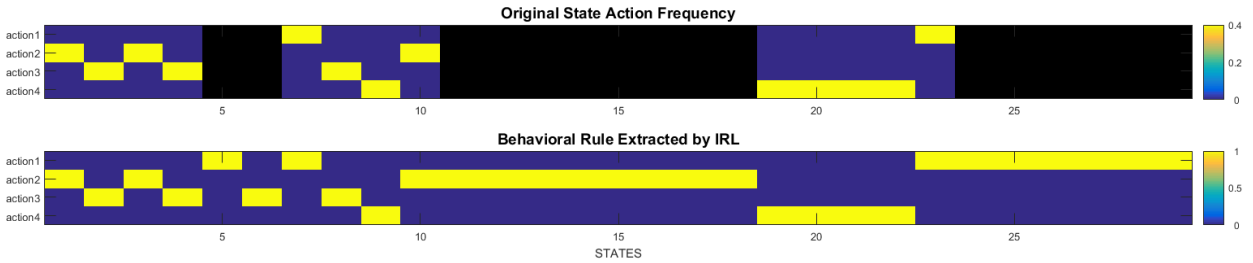
- Spatial segregation: The average percentage of the majority characteristic within a proximity radius
- Social segregation: The average percentage of conversation ticks with the same characteristic partner

The following table shows the mean and variance of these metrics for all time intervals.

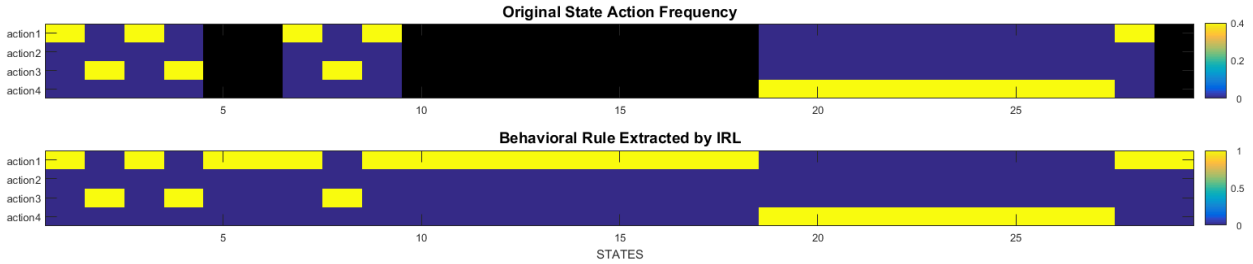
Table 1: Quantitative comparison between the IRL-constructed model (CM) and the baseline model (BM).

		1-300 ticks		**301-600 ticks**		**601-900 ticks**		**901-1200 ticks**	
		CM	**BM**	**CM**	**BM**	**CM**	**BM**	**CM**	**BM**
Spatial segregation	**Mean**	73.61	72.90	76.01	75.39	76.18	77.83	78.09	75.76
	Variance	21.76	21.98	2.81	6.84	3.95	6.05	4.77	3.19
Social segregation	**Mean**	82.04	81.50	85.64	84.97	86.12	85.86	86.33	86.20
	Variance	23.19	22.76	0.05	0.04	0.01	0.06	0.02	0.02

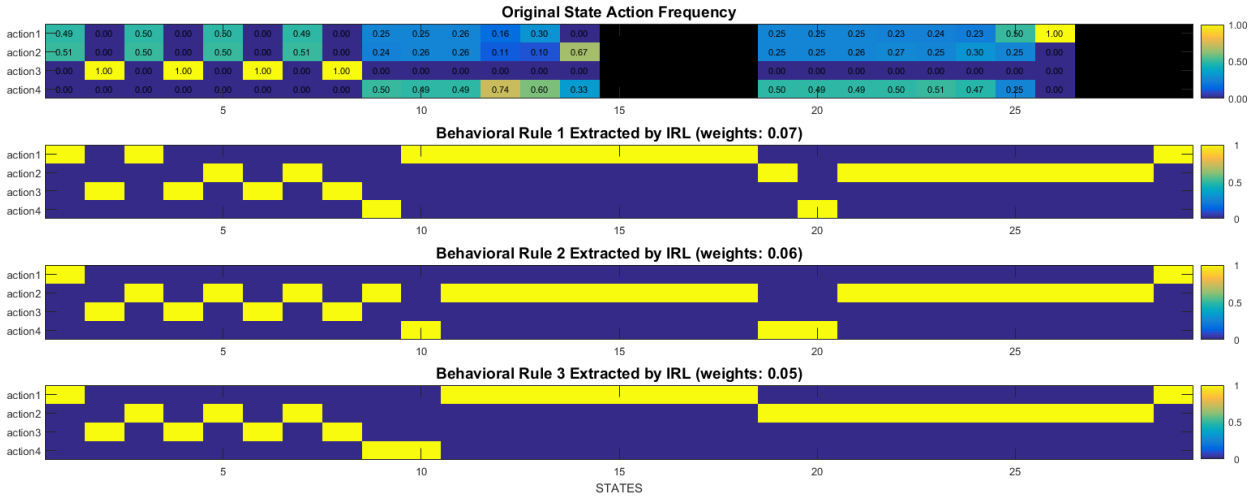
We also used a heat-map visualization to assess the extracted behavioral rules for each agent cluster. Figure 5 shows the deterministic behavioral rules generated for the weakly- and strongly-biased clusters. The black regions represent unobserved state-action pairs. The actions for observed states are correctly extracted, and the actions for unobserved states appear to be reasonable. Consider states S11-18 (conversation length 3-10, different recent partner, no potential partner, familiar environment). For the weakly- and strongly-biased clusters these states are never observed. Recall that these agents always end conversations with trait-different agents after 2 ticks. However, if they were somehow in state S11-18 (happen to have a longer conversation), the learned behavioral rules would end the conversation and move a long distance (weakly-biased agents) or end the conversation and move a short distance (strongly-biased agents).



(a) Weakly-biased agents



(b) Strongly-biased agents



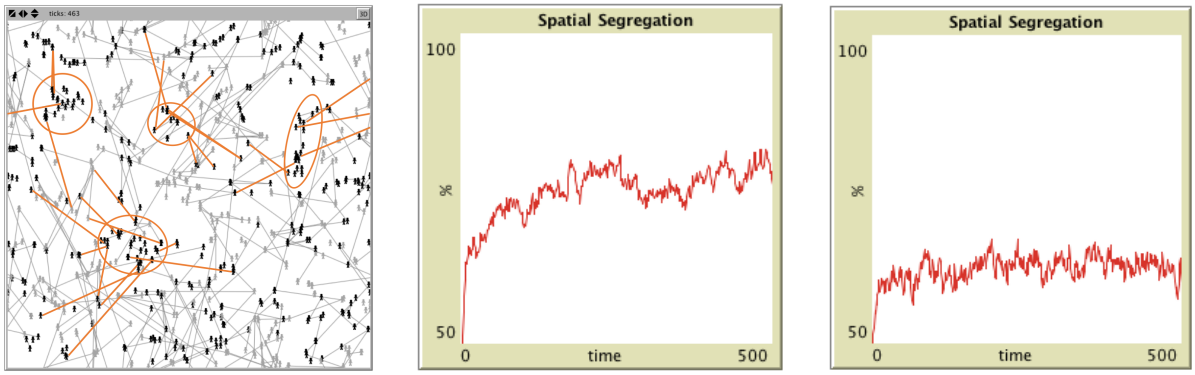
(c) Unbiased agents

Figure 5: Comparison between the original state-action frequency and the extracted behavioral rules. The heat map represents the frequency counts of each action (y-axis) in each state (x-axis) for the original data, and the behavioral rules recovered by the proposed IRL-based method.

Figure 5 also shows the top three among all behavioral rule sets that are used to approximate stochastic behavior for the unbiased cluster. Notice the similar structure between the action frequencies of the observed behavior and the mixed behavioral rules. This demonstrates the need for a more sophisticated algorithm since behavioral rules would need to be mixed in each step to obtain a single stochastic behavior rule set. Ziebart et al. (2008) argue that our IRL algorithm has the fundamental ambiguity of choosing one mixture of behavior rules among many different mixtures that could satisfy feature matching.

5.2 Network Analysis of Segregation

We also performed network analysis on the IRL-constructed agent-based model to analyze relationships between agents. This was done by adding links between frequent conversation partners, from which we observed that links are not stationary nor stay within a boundary of segregation. In other words, agents segregate not because they stay away from other agents and talk to agents within the same group, but because trait-similar agents happen to be positioned such that they collectively form a temporary but recurring boundary. Also, by changing the distances agents move (short-distance and long-distance), we find the length of the short distance is one of the key determinants for this segregation. Figure 6 shows differences in spatial segregation according to the length of agents' short-distance with proximity radius 2.



(a) Links between frequent partners (b) short-distance: 0-0.5 (c) short-distance: 1-1.5

Figure 6: Network structure and spatial segregation. There are as many outside-links as inside-links in a group (left). Comparison of spatial distances according to the length of short-distance (middle and right).

We interpret Figure 6 as indicating that segregation decreases if agents move just slightly farther when they decide to stay in a familiar environment, even though that movement does not take them away from nearby agents.

6 DISCUSSION

In our experiment, we constructed an agent-based model via IRL and compared it with a known (baseline) agent-based model. The experimental results indicate that the proposed method provides a systematic way of finding behavioral rules for an agent-based model. Several challenges remain. One challenge is designing a proper MDP, which requires the selection of states, actions, and rewards that satisfy the Markov property, as well as the estimation of transition probabilities for either unobserved or non-stationary state-action pairs. At some point, the modeler assumes that a series of events meets the Markov property in order for the model to be computationally feasible within the IRL framework. If this assumption is not met, the IRL algorithm may extract an incorrect reward function leading to agent behavior that doesn't match the observational data. A similar problem can also occur when estimating transition probabilities for either unobserved or non-stationary state-action pairs.

The IRL algorithm used for this experiment only works well for deterministic behavior (i.e., when agents always take the same action in the same state). We selected random behavior rule sets for agents within a cluster to approximate randomness, although each agent is operating deterministically. Rather than emulating the randomness with a mixture of deterministic rule sets, more sophisticated IRL algorithms are able to directly learn a single set of stochastic behavior rules, such as Maximum Entropy IRL (Ziebart et al. 2008).

7 CONCLUSION AND FUTURE WORK

We proposed a method for agent-based model construction using IRL, and we constructed an experimental agent-based model from a synthetic data set in order to explore segregation behavior. Considerable work has been done using IRL to predict and model many types of agent behavior, but to our knowledge, IRL has not been applied to the construction of agent-based models. We consider the most important benefit of this approach to be IRL's ability to predict unobserved behavior from an agent's motivation. Along with this, we believe data-driven ABM approaches have the potential to create opportunities for systematic improvement of agent-based model construction. Furthermore, this method creates additional options for finding behavioral rules for agent-based models, as IRL encompasses a wide range of algorithms.

Our next steps include applying this method to model real-world agents with multiple motivations. As part of this work, alternative IRL algorithms will be explored to address limitations with the current method. In addition, more rigorous consideration will be given to the process of designing MDPs from real-world data sets. To this end, two areas of particular interest are how to best estimate transition probabilities and how to best define the IRL reward function structure.

REFERENCES

Abbeel, P., A. Coates, and A. Y. Ng. 2010. "Autonomous Helicopter Aerobatics through Apprenticeship Learning". *The International Journal of Robotics Research* 29 (13): 1608–1639.

Abbeel, P., and A. Y. Ng. 2004. "Apprenticeship Learning via Inverse Reinforcement Learning". In *Proceedings of the 21st International Conference on Machine Learning*, 1. ACM.

Argall, B. D., S. Chernova, M. Veloso, and B. Browning. 2009. "A Survey of Robot Learning from Demonstration". *Robotics and Autonomous Systems* 57 (5): 469–483.

Axelrod, R. 1997. "Advancing the Art of Simulation in the Social Sciences". In *Simulating Social Phenomena*, 21–40. Springer.

Bonabeau, E. 2002. "Agent-Based Modeling: Methods and Techniques for Simulating Human Systems". *Proceedings of the National Academy of Sciences* 99 (suppl 3): 7280–7287.

Heath, B., R. Hill, and F. Ciarallo. 2009. "A Survey of Agent-Based Modeling Practices (January 1998 to July 2008)". *Journal of Artificial Societies and Social Simulation* 12 (4): 9.

Huang, Y., J. Gong, M. Rucker, P. Chow, K. Fua, M. S. Gerber, B. Teachman, and L. E. Barnes. 2017. "Discovery of Behavioral Markers of Social Anxiety from Smartphone Sensor Data". In *Proceedings of the 1st Workshop on Digital Biomarkers*, 9–14. ACM.

Levine, S., Z. Popovic, and V. Koltun. 2010. "Feature Construction for Inverse Reinforcement Learning". In *Advances in Neural Information Processing Systems*, 1342–1350.

Liu, S., M. Araujo, E. Brunskill, R. Rossetti, J. Barros, and R. Krishnan. 2013. "Understanding Sequential Decisions via Inverse Reinforcement Learning". In *Proceedings of the 2013 IEEE 14th International Conference on Mobile Data Management*, Volume 1, 177–186. IEEE.

Negahban, A., and L. Yilmaz. 2014. "Agent-Based Simulation Applications in Marketing Research: An Integrated Review". *Journal of Simulation* 8 (2): 129–142.

Ng, A. Y., and S. J. Russell. 2000. "Algorithms for Inverse Reinforcement Learning". In *Proceedings of the 17th International Conference on Machine Learning*, 663–670.

Nouri, E., K. Georgila, and D. R. Traum. 2012. "A Cultural Decision-Making Model for Negotiation based on Inverse Reinforcement Learning". In *Proceedings of the Cognitive Science Society*.

Osogami, T., and R. Raymond. 2013. "Map Matching with Inverse Reinforcement Learning". In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, 2547–2553. AAAI.

Qiao, Q., and P. A. Beling. 2013. "Recognition of Agents Based on Observation of Their Sequential Behavior". In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 33–48.

Ramachandran, D., and E. Amir. 2007. "Bayesian Inverse Reinforcement Learning". In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2586–2591.

Ratliff, N. D., J. A. Bagnell, and M. A. Zinkevich. 2006. "Maximum Margin Planning". In *Proceedings of the 23rd international conference on Machine learning*, 729–736. ACM.

Russell, S. 1998. "Learning Agents for Uncertain Environments". In *Proceedings of the 11th annual Conference on Computational Learning Theory*, 101–103. ACM.

Schelling, T. C. 1971. "Dynamic Models of Segregation". *Journal of Mathematical Sociology* 1 (2): 143–186.

Tesfatsion, L. 2002. "Agent-Based Computational Economics: Growing Economies from the Bottom Up". *Artificial life* 8 (1): 55–82.

Torrens, P., X. Li, and W. A. Griffin. 2011. "Building Agent-Based Walking Models by Machine-Learning on Diverse Databases of Space-Time Trajectory Samples". *Transactions in GIS* 15 (s1): 67–94.

Wunder, M., S. Suri, and D. J. Watts. 2013. "Empirical Agent Based Models of Cooperation in Public Goods Games". In *Proceedings of the 14th ACM conference on electronic commerce*, 891–908. ACM.

Yang, S. Y., Q. Qiao, P. A. Beling, W. T. Scherer, and A. A. Kirilenko. 2015. "Gaussian Process-Based Algorithmic Trading Strategy Identification". *Quantitative Finance* 15 (10): 1683–1703.

Yokoyama, N. 2017. "Intent Inference of Aircraft via Inverse Optimal Control Including Second-order Optimality Condition". In *AIAA Guidance, Navigation, and Control Conference*, 1254.

Zhang, H., Y. Vorobeychik, J. Letchford, and K. Lakkaraju. 2016. "Data-driven Agent-based Modeling, With Application to Rooftop Solar Adoption". *Autonomous Agents and Multi-Agent Systems* 30 (6): 1023–1049.

Ziebart, B. D., A. L. Maas, J. A. Bagnell, and A. K. Dey. 2008. "Maximum Entropy Inverse Reinforcement Learning". In *Proceedings of The Twenty-third AAAI Conference on Artificial Intelligence*, 1433–1438.

AUTHOR BIOGRAPHIES

Kamwoo Lee is a Ph.D. student in Systems and Information Engineering at the University of Virginia. He received a B.S. in Computer Science and Engineering from Seoul National University in 2003. His research focuses on implementing machine learning and simulation techniques for public policy applications. His email address is kl9ch@virginia.edu.

Mark Rucker is an M.S. student in Systems and Information Engineering at the University of Virginia. He received a B.S. in Computer Science from Harding University in 2008 and spent eight years working in the technology industry. His research focuses on inverse reinforcement learning. His email address is mr2an@virginia.edu.

William T. Scherer is a Professor of Systems and Information Engineering at the University of Virginia. His interests include systems engineering methodology and intelligent decision support systems, with applications to financial engineering and intelligent transportation systems. His email address is wts@virginia.edu.

Peter A. Beling is a Professor of Systems and Information Engineering at the University of Virginia. His research interests include decision making in complex systems and Bayesian scoring models, with applications in credit decision-making and analysis of high-frequency trading. His email address is beling@virginia.edu.

Matthew S. Gerber is an Assistant Professor of Systems and Information Engineering at the University of Virginia. His research interests are in sequential decision making and reinforcement learning, with applications to cyber-human systems. His email address is msg8u@virginia.edu.

Hyojung Kang is a Research Assistant Professor in the Department of Systems and Information Engineering at the University of Virginia. Her research focuses on operations research and applied statistics, with applications to model-based, implementable solutions for complex systems. Her email address is hkang@virginia.edu.