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 Scherer
Peter Beling
Matthew S. Gerber
Hyojung Kang

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

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 reframes 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 ABM directly from data using IRL. We explain each step of the proposed method and describe challenges that may occur during implementation. The experimentation results show that the proposed method can extract rules and construct an ABM.

1 INTRODUCTION

Agent-based models (ABMs) use a bottom-up approach to discovering complex, aggregate-level properties. These properties emerge from individual agent behaviors and interactions within the environment. This is in contrast to more classical, top-down simulation paradigms such as system dynamics and discrete-event simulation. An inevitable assumption of ABM, then, 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.

Recently, there have been attempts to obtain these rules directly from data, which is increasingly available in many domains. We consider inverse reinforcement learning (IRL) to be an overlooked candidate for such a data driven approach. Instead of directly learning actions for each state, IRL first recovers motivations to explain an observed sequence of actions and then generates its own sequences of actions from those motivations.

In this paper, we propose a method to build an agent-based simulation model with the help of IRL. Through an experiment, we show that the method provides a newly constructed ABM with rich but concise behavioral rules (or policies, in IRL parlance) for agents while still maintaining aggregate-level properties. The contribution of this paper is to expand the areas of both ABM and IRL by emphasizing their shared strengths.

In section 2 we review key concepts and recent work in ABM and IRL research. In section 3, we present the details of our proposed approach. Sections 4 and 5 present the experiment and result of the proposed method in a phenomenon that we call "desultory segregation" with a synthetic data set. In section

6 we discuss the challenges of the method, and we conclude the paper by highlighting areas of future work in section 7.

2 LITERATURE REVIEW

2.1 Preliminary to ABM

ABM is widely used to study complex systems that consist of heterogeneous and autonomous agents who interact with others and the environment. Each agent takes actions based on their assessment of the environment and on their predetermined goals. In this regard, ABM has many benefits over other modeling or simulation techniques. Bonabeau (2002) summarizes the benefits as follows: (a) ABM captures emergent phenomena that result from the interactions of individuals; (b) ABM provides an intuitive description of a system, hence it makes the model seem closer to reality; and (c) ABM is flexible in terms of modeling and analyzing the system.

However, it is often difficult to develop a reliable ABM since one needs to develop agents' behavioral rules based on a qualitative understanding of domain-specific knowledge and then carefully calibrate individual 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. Since ABMs generally have numerous parameters that characterize agents and the environment in a model, Calvez and Hutzler (2005) propose using genetic algorithms to systematically explore the parameter space of an ABM. Thiele et al. (2014) try several different methods for calibrating model parameters using data. In another vein, Rand (2006) proposes a framework of interaction between ML and ABM where the ML algorithm uses the ABM as an environment while the ABM uses the ML algorithm to update the internal models of the agents. Torrens et al. (2011) uses 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 modeling (DDABM) 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 motivations.

2.2 Preliminary to IRL

Inverse reinforcement learning (IRL) (Russell 1998) has effectively addressed problems as diverse as autonomous helicopter control (Abbeel et al. 2010), correctly identifying a pilot's navigational intent (Yokoyama 2017), and distinguishing cultural preferences in negotiation games (Nouri et al. 2012). These applications share the objective of identifying motivations (the reward in IRL) that drive observed behavior.

IRL is based on the theory of Markov decision processes (MDPs), which naturally capture processes of sequential decision making under uncertainty. Formally, an MDP is represented as a five-tuple (S, A, $P_{\cdot}(\cdot,\cdot)$, $R(\cdot,\cdot)$, γ) where S is the set of all possible states, A is the set of available actions, $P_a(s,s')$ is the probability of moving from state s to s' given action a, $R(s,a)$ is the scalar reward received for taking action a in state s, and γ is the discount factor. In this formulation, an IRL problem is formulated as an MDP without a reward function, often denoted as MDP$\backslash R$, where we discover an R that can motivate the observed sequence of (s,a) pairs. Numerous methods have been proposed to accomplish this task. Within this class of algorithms, Ng and Russell (2000) pose the problem as a linear programming optimization. Abbeel and Ng (2004) as well as Ratliff et al. (2006) pose the problem as a maximum-margin optimization within the feature expectation space. Ramachandran and Amir (2007) as well as Ziebart et al. (2008) solve IRL by finding the R that makes the observed behavior most likely. Qiao and Beling (2011) adopt a Bayesian framework with a Gaussian process model of rewards to achieve results that are relatively insensitive to number of observations.

Our contribution is to demonstrate the usefulness of IRL in explaining aggregate behavior within ABMs. Some IRL research has used the reward function to model individual human behavior (Qiao and Beling 2013, Liu et al. 2013, Osogami and Raymond 2013, Yang et al. 2015), but none of these efforts have sought to validate the effectiveness of the recovered reward for recreating aggregate behavior. With only observations of states and actions, all other necessary MDP components (i.e., P and R) can be estimated. If this were all, then we believe IRL would be useful; however, IRL has several characteristics beyond behavioral rule learning that also make it suitable for ABM construction – namely, insight into agents' motivations, a generalization of behavior to unseen states, and coherent chains of actions instead of simple statistical likelihoods.

3 METHODOLOGY

Our method consists of six steps 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.

(a) Collect data of interest
(b) Design an MDP$\backslash R$ model
(c) Cluster agents
(d) Estimate transition probabilities for each cluster
(e) Extract behavioral rules via IRL for each cluster
(f) Construct ABM from extracted rules

An IRL framework decomposes R into weights (w) and reward features (ϕ). To incorporate this framework into the method, we split the design process of MDP$\backslash R = (S,A,P,\gamma)$ into steps 2, 4 and 5. We define states (S), actions (A), and reward features (ϕ) in step 2, estimate transition probabilities (P) in step 4, and recover the reward function (R) in step 5. After that, an ABM is constructed from S and A, and the behavioral rules derived from R. Figure 1 shows the overview of this flow.

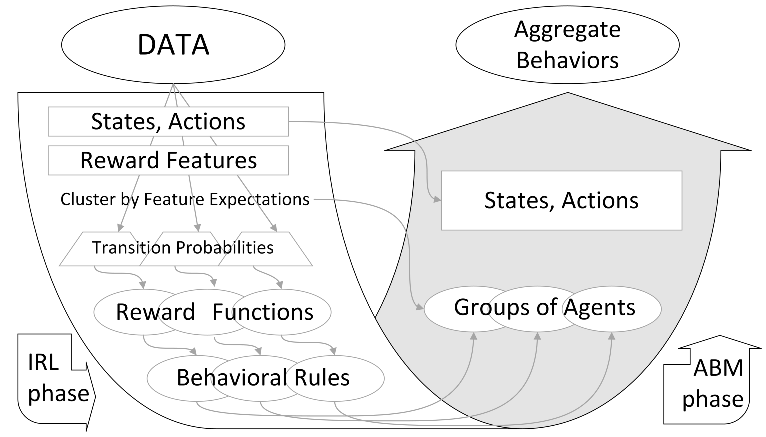


Figure 1: Method diagram illustrating the progression from IRL to ABM

3.1 Collect Data of Interest

The proposed method requires observed sequences of states and actions associated with 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.

3.2 Design an MDP$\backslash R$ model

When designing the MDP$\backslash R$, states and actions should be selected to capture the essence of the underlying environment. They should be as parsimonious as possible while being able to distinguish semantically meaningful changes in the environment and an agent's actions. Reward features (ϕ) are also selected in this step. Formally, the (ϕ) comprise a set of basis functions that map states and actions into agent motivation. More 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 R using this frame.

3.3 Cluster Agents

In step three, the original observations are clustered to decide an appropriate level of heterogeneity of agents in a model while retaining essential diversity. Agents are clustered according to their feature expectation. The feature expectation is the average sum of the (ϕ) defined in step two, which is aggregated through the sequence of visited states. The number of clusters at this step can be determined depending on the desired level of model diversity, ranging from perfectly homogeneous to perfectly heterogeneous.

3.4 Estimate Transition Probabilities for Each Cluster

In step four, the state transition probabilities are estimated for each behavior cluster. Since members of a cluster can be considered to show the same behavioral rules and have the same way to respond to the environment, it is reasonable to assign distinct transition probabilities to each cluster. Sometimes, a transition probability is forced by the environment, such as the change in position of a dropped object due to gravity. However, in many cases, one should estimate it by observation.

3.5 Extract Behavioral Rules via IRL for Each Cluster

In step five, a reward function that fits the observed behaviors is recovered by IRL. Optimal behavior rules are then found which maximize the derived reward over time. The choice of IRL algorithm depends on the characteristics of data and assumptions of the model.

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 the poor performance. 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 could also be 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 ABM from Extracted Behavioral Rules

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

(a) Define states and actions in the model according to step 2.
(b) Assign behavioral rules with equal densities to original clusters.
(c) In each iteration, agents observe their states and follow the behavioral rules to change their state.

4 EXPERIMENT

To demonstrate the application and performance of this method, we construct an example ABM that simulates human segregation behaviors. The rules for each agent are extracted from a synthetic data set by following the steps explained in the previous section.

4.1 Synthetic Data

We generated a data set where people exhibit a segregation pattern at the aggregate level. The data represent how the environment and actions that people take change when they are located in a public space, moving and talking over time. Each person is distinguished by an innate, 2-level characteristic, which is known to every other person. This characteristic can be considered as ethnicity (Black/White), religion (Hindu/Muslim), or any other 2-level trait. Each person can take one of 4 actions at any point in time:

- Start a conversation with a nearby person
- Continue a conversation for another tick
- Move a short distance in a random direction
- Move a long distance in a random direction

A proximity radius is defined to identify nearby people. Moving a short distance leaves a person in the proximity radius whereas moving a long distance results in a person leaving the radius. Time in this data is discretized in ticks, and people have one of three behavioral patterns as follows:

- Unbiased: Speak to anyone within proximity for a random length of time (1-10 ticks) and then randomly move a short or long distance.
- Weak-bias: Speak to people who share your trait value for five ticks and then move a short distance. Speak to people who do not share your trait value for two ticks and then move a long distance.
- Strong-bias: Speak to people who share your trait value for ten ticks; speak to people 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 people according to the above specifications.

4.2 Designing an MDP$\backslash R$

We construct an MDP$\backslash R$ with four state variables and four actions as in Figure 2. Within this formulation, there are 4 types of illegal actions.

(a) Continue conversation while not having a conversation
(b) Start or continue a conversation when there is no potential partner
(c) Start or continue a conversation when it reaches the maximum conversation length
(d) Start a conversation when having a conversation

We introduce a "limbo" state, which is an absorbing state, to deal with these illegal actions. Since the IRL iteration process starts with arbitrary behavioral rules, many early recovered behaviors in the IRL can include the illegal actions. By leading those actions to the absorbing limbo state, we try to prevent the IRL from learning illegal actions. In this experiment, we set the reward feature space equal to the state space.

4.3 Clustering Agents

We calculate the vector of the expected sum of reward features (feature expectations) for each person whose behavior was observed.

Then we use hierarchical clustering analysis (HCA) with a cosine distance function and average linkage to create a dendrogram as shown in Figure 3. Looking at the dendrogram, we choose to split the feature

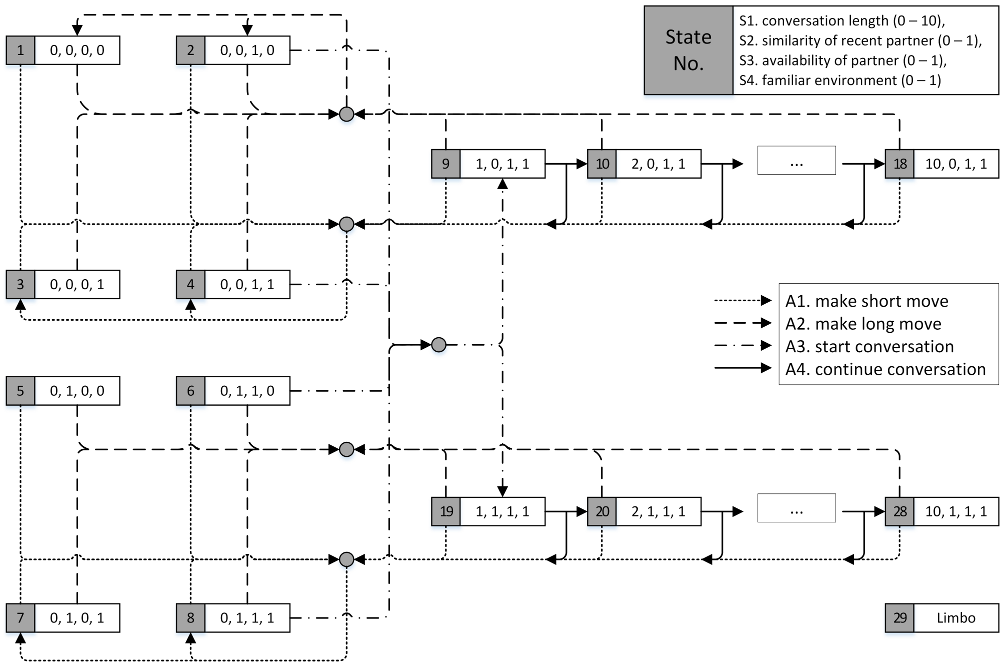


Figure 2: MDP diagram with legal state-action transitions

expectations (observed people) where the difference is most significant, which result in three groups. After choosing the number of clusters, we average the feature expectations to obtain a single feature expectation for each group. For the rest of the experiment, we treat a cluster as a group of homogeneous people. We determine transition probabilities, weights of rewards, and behavioral rules for each cluster as described below.

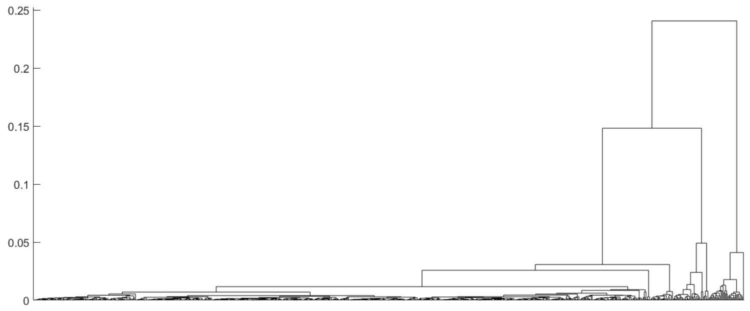


Figure 3: Hierarchical Cluster Analysis showing the biggest distance between 3 and 4 clusters

4.4 Determining Transition Probabilities for Each Cluster

We determine the transition probabilities, $P_a(s,s')$, for each combination of the 29 possible states (s, s') and 4 actions (a) of Figure 2. We estimate each probability by observing which state people land in after taking an action in a 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 are used in place of a single cluster.

4.5 Recovering Rewards and Behavioral Rules For Each Cluster via IRL

We use 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 functions, 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 becomes 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 ABM

The IRL algorithm generates 3 sets of behavioral rules which are used to construct the ABM. Using the previously defined states, actions, and features, as well as the behavioral rules recovered for 3 groups, we construct the model by initializing states of the model, assigning the behavioral rules to each group of agents, and letting the agents observe their states and follow the rules in each iteration.

5 RESULTS

The constructed model shows very similar individual and aggregate behaviors to the observed behaviors. We first check that the constructed ABM indeed represents the behaviors in the data set. Then we further analyze the model to show that the constructed model can achieve its analytic purpose as an ABM.

5.1 Assessment of the model

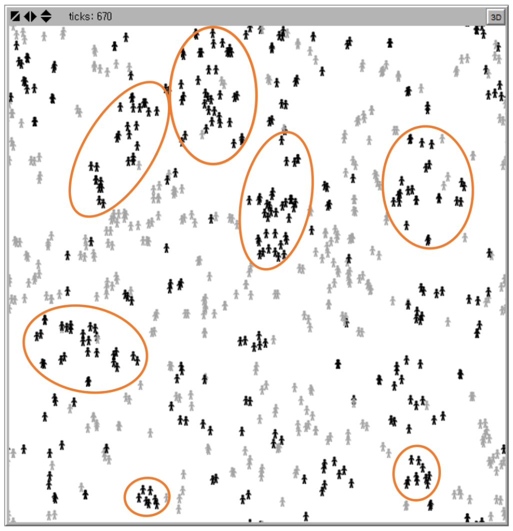
In order to evaluate the constructed ABM, we look at both qualitative and quantitative aspects of the model and the data set. We consider that the most important aspect is the aggregated behavior of people. So, the model that generated the synthetic data doubles as a baseline model here. Qualitatively, we observe the progress of segregation by visual comparison between the baseline and the newly constructed model. Although the entire process of segregation cannot be presented here, we can show the similarity of segregation after same number of ticks (Figure 4).

For the purpose of quantitative assessment, we define spatial segregation and social segregation as follows:

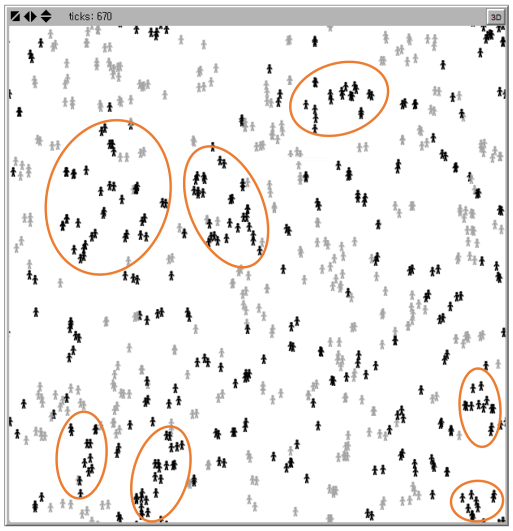
- Spatial Segregation: On average, the percentage of the majority characteristic within a proximity radius
- Social Segregation: On average, percentage of conversation ticks with the same characteristic partner

Table 1 shows that both mean and variance of these metrics for all time intervals are close enough to be identical considering the randomness of the simulation.

Further, we use a heat map visualization to assess the result of extracted behavioral rules for each cluster. Figure 5 shows that IRL generates reasonable deterministic behavioral rules 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 are believable. 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: remember these individuals always end conversations with a different character after 2 ticks. However, if they were somehow in state S11-18



(a) Extracted Behavior by IRL



(b) Behavior in Baseline Model

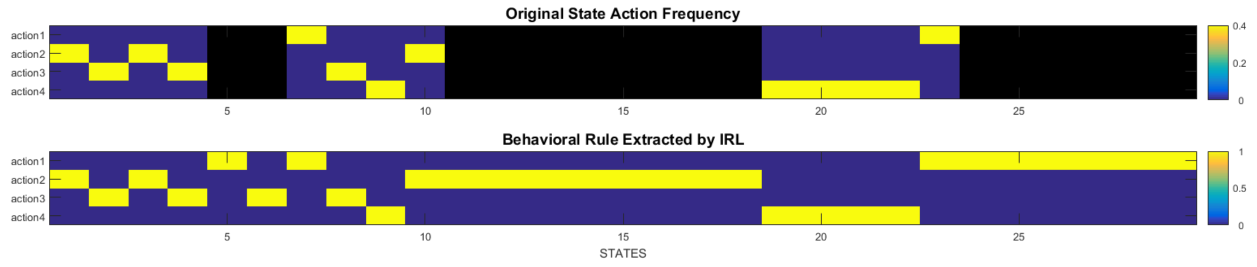
Figure 4: Visual comparison between constructed model and baseline model

Table 1: Quantitative comparison between constructed model (CM) and baseline model (BM)

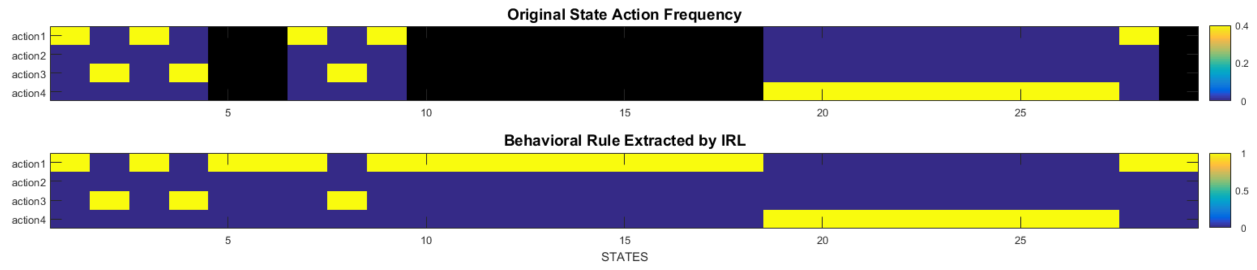
		1-300 ticks		**301-600 ticks**		**601-900 ticks**		**901-1200 ticks**	
		CM	**BM**	**CM**	**BM**	**CM**	**BM**	**CM**	**BM**
Spatial Segregation	**mean**	73.61	72.90	76.01	75.39	76.18	77.83	78.09	75.76
	variance	21.76	21.98	2.81	6.84	3.95	6.05	4.77	3.19
Social Segregation	**mean**	82.04	81.50	85.64	84.97	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

(happen to have a longer conversation), the learned behavioral rules would end the conversation and move a long distance (weakly-biased people) or end the conversation and move a short distance (strongly-biased people).

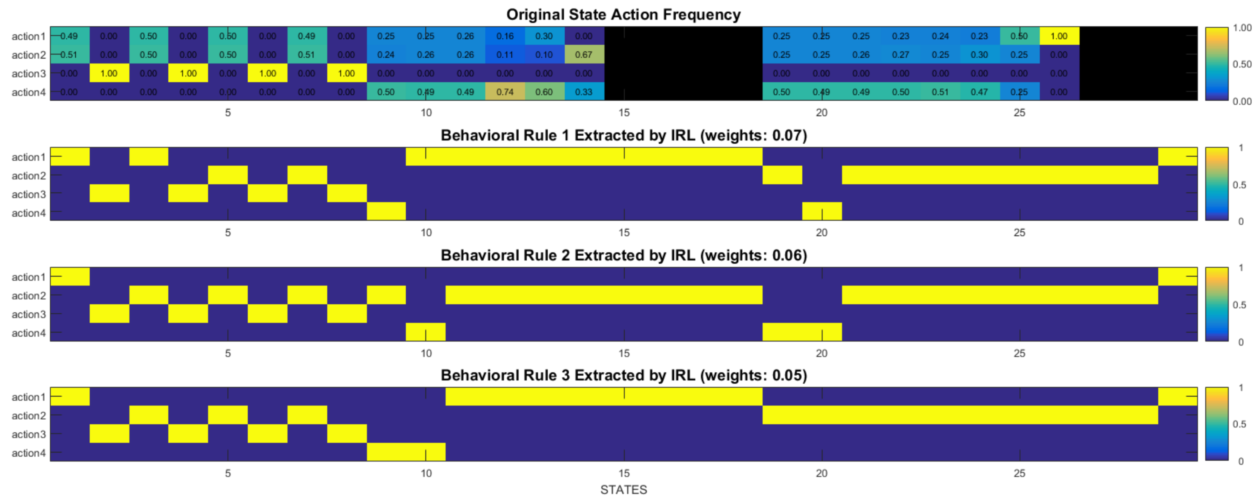
Figure 5 shows the top 3 among all behavioral rule sets that are used to approximate stochastic behavior for the unbiased cluster. Notice the similar structure between the action frequencies of the observed behavior and the mixed behavioral rules. This clearly demonstrates the need for of a more sophisticated algorithm since behavior rules would need to be mixed in each step to obtain a single stochastic behavior rule set. Ziebart et al. (2008) argues that the IRL algorithm we use has the fundamental ambiguity of choosing one mixture of behavior rules among many different mixtures which could satisfy feature matching.



(a) Weakly-biased people

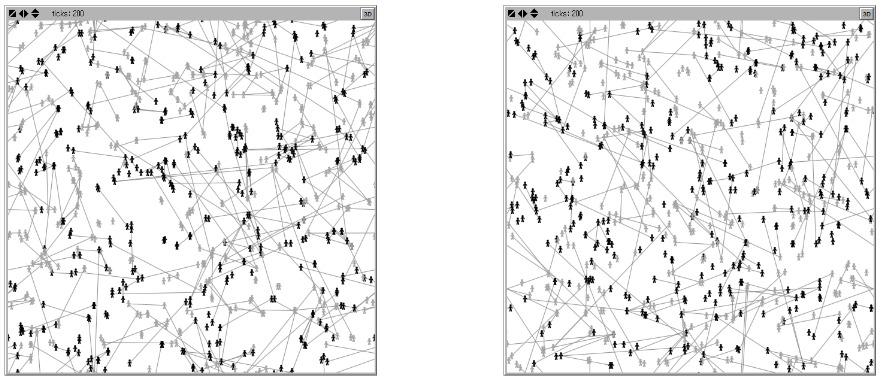


(b) Strongly-biased people



(c) Unbiased people

Figure 5: Comparison between the original state-action frequency and the extracted behavioral rules



(a) short-distance: 0-0.5 (b) short-distance: 1-1.5

Figure 6: Clear segregation (left) vs. Subtle segregation (right)

5.2 Further Analysis: Desultory Segregation

To the constructed ABM we also add a network to analyze relationships between agents and then change a set of parameters to conduct a hypothetical experiment. By adding links between frequent conversation partners, we can see the links are not stationary nor stay within a boundary of segregation. In other words, people develop a group not because they stay away from the other groups and talk to the people within the same group, but because people of the same characteristic happen to be positioned such that they collectively form a temporary but recurring boundary, which is what we call "desultory segregation". Also, by changing the distances people move (short-distance and long-distance), we find the length of the short distance as one of the key determinants for this desultory segregation. Figure 6 shows a tipping point of the different aggregated behaviors according to the length of people's short-distance when the proximity radius is 2. We can interpret this as meaning that the desultory segregation disappears if people move just slightly further when they decide to stay in a familiar environment even though that movement does not make them leave nearby people.

6 DISCUSSION

In the experiment, we tested the method by constructing an ABM via IRL. The experiment showed that the proposed method provides a systematic way of finding behavioral rules for an ABM. There may be some challenges that affect the performance of the method. In this discussion, we briefly examine these.

One challenge is designing a proper MDP, which requires the selection of states, actions, and rewards that satisfy the Markov property, and the determination 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 can introduce subtle logic errors. A similar but different problem is determining 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. Finally, while it might not need to be explicitly stated, at this time, we see no way to extend this method to theoretical thought experiments where data is not available up front. Furthermore, even when data is available, without enough data or with low-quality data the results should be considered circumspect.

7 CONCLUSION AND FUTURE WORK

We proposed a method for ABM construction using IRL, and we constructed an experimental ABM to explore desultory segregation behavior within a synthetic data set. 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 ABMs. 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 approaches to ABM have the potential to create opportunities for systematic improvement of ABM construction. Furthermore, this method creates additional options for finding behavioral rules for ABM, 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 considerations 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 learn 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 twenty-first international conference on Machine learning*, 1. ACM.

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.

Calvez, B., and G. Hutzler. 2005. "Automatic tuning of agent-based models using genetic algorithms". In *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, 41–57. Springer.

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.

Liu, S., M. Araujo, E. Brunskill, R. Rossetti, J. Barros, and R. Krishnan. 2013. "Understanding sequential decisions via inverse reinforcement learning". In *Mobile Data Management (MDM), 2013 IEEE 14th International Conference on*, Volume 1, 177–186. IEEE.

Ng, A. Y., and S. J. Russell. 2000. "Algorithms for inverse reinforcement learning". *Icml*.

Nouri, E., K. Georgila, and D. R. Traum. 2012. "A Cultural Decision-Making Model for Negotiation based on Inverse Reinforcement Learning.". In *CogSci*.

Osogami, T., and R. Raymond. 2013. "Map Matching with Inverse Reinforcement Learning.". In *IJCAI*.

Qiao, Q., and P. A. Beling. 2011. "Inverse reinforcement learning with Gaussian process". In *American Control Conference (ACC), 2011*, 113–118. IEEE.

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. Springer.

Ramachandran, D., and E. Amir. 2007. "Bayesian inverse reinforcement learning". *Urbana* 51 (61801): 1–4.

Rand, W. 2006. "Machine learning meets agent-based modeling: When not to go to a bar". In *Conference on Social Agents: Results and Prospects*.

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 eleventh annual conference on Computational learning theory*, 101–103. ACM.

Thiele, J. C., W. Kurth, and V. Grimm. 2014. "Facilitating parameter estimation and sensitivity analysis of agent-based models: A cookbook using NetLogo and R". *Journal of Artificial Societies and Social Simulation* 17 (3): 11.

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 fourteenth 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 *AAAI*, Volume 8, 1433–1438. Chicago, IL, USA.

AUTHOR BIOGRAPHIES

Kamwoo Lee is a doctoral 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 e-mail address is kl9ch@virginia.edu.

Mark Rucker is a master's student at the University of Virginia in Systems Engineering. He received a B.S. in Computer Science from Harding University in 2008. He spent 8 years working in the technology industry as a lead software developer. In 2016 he returned to grad school full time. He can be reached at mr2an@virginia.edu.

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

Peter Beling is an Associate Professor of Systems and Information Engineering at the University of Virginia. His research interests are in the area of decision making in complex systems, with emphasis on Bayesian scoring models, machine learning, and mathematical optimization. His research has found application in a variety of domains, including lender and consumer credit decision-making, analysis of high frequency trading strategies, and man-machine decision systems. His email address is pb3a@virginia.edu.

Matthew Gerber is an Assistant Professor of Systems and Information Engineering at the University of Virginia. His research interests are in sequential decision making, reinforcement learning, and supervised (statistical) learning, with applications to natural language processing and spatiotemporal prediction of risk. His e-mail 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 developing and applying operations research and applied statistics methods to provide model-based, implementable solutions for complex systems. Currently, she is interested in developing simulation and optimization models to improve patient flow, design system-level interventions, and evaluate performance measures. Her e-mail address is hkang@virginia.edu.