Measure Task-Level Inter-Agent Interaction Difficulty in Decentralized Scenarios, with Scenario Generalization Approximation

Gang Qiao[1*] Kaidong Hu[1] Seonghyeon Moon[1] Sejong Yoon[2] Mubbasir Kapadia[1] Vladimir Pavlovic[1]
[1]Rutgers University [2]The College of New Jersey
{gq19, sm2062, mk1353, vladimir}@cs.rutgers.edu kaidong.hu@gmail.com yoons@tcnj.edu

Abstract

Motivated by the need to understand the central complexity in problems like multi-agent path finding and generalizing models in new scenarios, we present an algorithm that measures task-level inter-agent interaction difficulties in decentralized crowd scenarios. The algorithm (i) views a crowd simulation as a transformation of the parameter of a steering model, (ii) captures the reduced amount of abstracted trajectory classes. We exploit the measurement to approximate the scenario generalizations of learning models. Experiments validate the efficacy of the measurement in characterizing interaction difficulty, and the potential to select domains before actually training and testing a model.

1 Motivation

In the context of decentralized crowd simulation [Qiao *et al.*, 2018], a scenario refers to the configuration of obstacles in an environment and the tasks of all agents in that environment. The task of an agent refers to the initial and destination positions of the agent, the starting time when the agent presents into the environment, the maximal number of simulation steps allowed, and the radius of the agent. The interaction among agents refers to the effort made by the agents to (i) follow their individual planned paths (nodes) and (ii) avoid collisions from each other. Given a decentralized scenario, some tasks will inevitably "encounter" more agents, hence potential inter-agent collisions, than other tasks despite the steering model been chosen. Such inherent task-level inter-agent interaction difficulty introduces the central complexity for problems including Multi-Agent Pathfinding, and scenario generalization (SG) of learning models [Qiao *et al.*, 2019]. To this end, an algorithm is presented. It takes the full information about a scenario as input and outputs a scalar for each agent (task) in the scenario, indicating the task-level interaction difficulty for that agent with all other agents, despite the steering model being chosen. We further use the measurement to approximate scenario generalization.

2 Approach to Measurement

An overview of the proposed method is illustrated in Figure 1.

2.1 Crowd Simulation

Given a scenario with obstacle configuration $\mathcal{E}$, suppose there are n agents in the scenario. Denote all agents as $X_{1\sim n} := (X_1, X_2, ..., X_n)$[1]. We aim to estimate the task-level interaction between an agent X_i and all other agents in the scenario, which are denoted as $X_{-i} := (X_1, ..., X_{i-1}, X_{i+1}, ..., X_n)$ [2], for i = 1, 2, ..., n.

Since our goal is to estimate task-level rather than model-level (how a particular model performs) interaction between agent X_i and agents X_{-i}, it is essential to avoid specifying a steering model with a fixed parameter for each agent. Instead, we model the uncertainty of the behavior of agent X_i by assuming the parameter of the model being a random variable, denoted as Θ_i, that obeys some distribution $p_i(\theta_i)$, i = 1, 2, ..., n. For notation simplicity, denote $\Theta_{1\sim n} := (\Theta_1, \Theta_2, ..., \Theta_n)$.

For a given scenario, at task-level crowd simulation, the input of the crowd system is $\Theta_{1\sim n}$ and the output is a tuple of interactive trajectories, denoted as $T_{1\sim n} := (T_1, T_2, ..., T_n)$, where T_i is the trajectory of agent X_i driven by $\Theta_{1\sim n}$. The relationship between the input and the output can be represented as $f(\Theta_{1\sim n}|\mathcal{E}, X_{1\sim n}, \mathcal{M}) = T_{1\sim n}$, conditioned on the obstacle configuration $\mathcal{E}$, the tasks of all agents $X_{1\sim n}$ from the given scenario, and the steering model family $\mathcal{M}$. The transformation function $f(\cdot)$ provides an alternative view of the input and output of a crowd simulation.

2.2 Trajectory Abstraction

To improve the representational efficiency for characterizing the relationships among trajectories of different agents, it is necessary to abstract the trajectory T_i of agent X_i to a finite number of classes. Each class contains realizations of T_i of the same modality, while realizations from different classes present different modalities, i=1, 2, ..., n.

To achieve the trajectory abstraction, besides running the simulation involving all agents at the random parameter $\Theta_{1\sim n}$, we additionally run one simulation for agent X_i from its initial to its destination position, with obstacle configurations but no other agents, at an appropriately selected steering parameter θ_i^*, for i = 1, 2, ..., n. This results in a deterministic solo trajectory for agent X_i, denoted as s_i, which could be viewed as agent X_i's ideal trajectory in the sense that no extra effort is needed to avoid inter-agent collisions.

Therefore a generic function can be applied to compute the difference between the trajectory T_i of agent X_i and its solo trajectory s_i, which is a quantification of the effort that agent X_i makes for interaction with the rest agents X_{-i}. As a typical choice, Dynamic Time Warping (DTW) that accumulates

[1]At task level, X_i refers to both the i-th agent and the task of the i-th agent, and we use the two meanings of X_i interchangeably.

[2]This does not mean that during the crowd movement, there are no interactions among agents X_1, ..., X_{i-1}, X_{i+1}, ..., X_n.

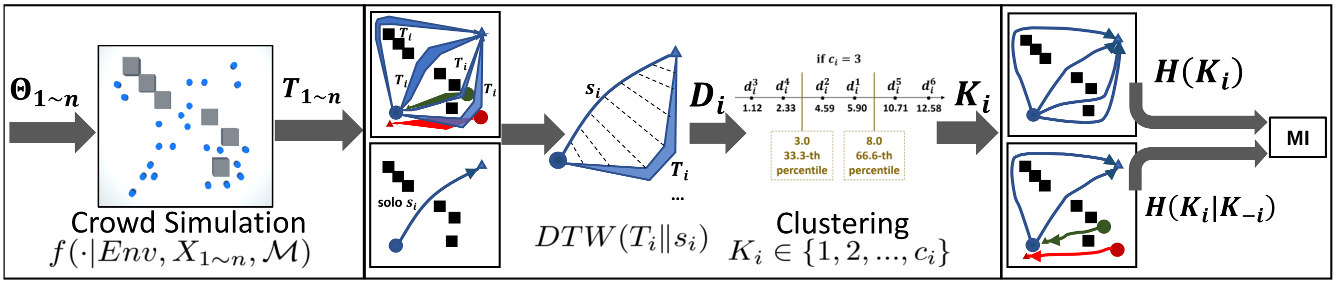

Figure 1: Diagram for measuring task-level inter-agent interaction (in this example, between the blue agent and other agents). It consists of a crowd simulation block, a trajectory abstraction block and a Mutual Information (MI) block. The crowd simulation block takes in a random parameter and generates random interactive trajectories of all agents. The trajectory abstraction block compares the trajectory of an agent (e.g., the blue agent) and its solo trajectory to yield DTW difference, which goes through a clustering process and outputs an abstracted trajectory index. The MI block measures the reduced amount of the abstracted trajectory classes of the agent resulting from the interaction with other agents. An option of an agent is shown with a curve, representing a class of similar trajectories for the agent. The destination of an agent is shown with a triangle of the same color. Black rectangles in the maps are obstacles.

distances over all pairs of aligned states [Salvador and Chan, 2007], yields the DTW difference: $D_i = DTW(T_i||s_i)$, which embodies the extra expenditure of agent X_i in the trajectory T_i due to the interaction with the rest agents X_{-i}.

D_i is further transformed to an abstracted trajectory index $K_i \in \{1, 2, \ldots, c_i\}$ by a clustering process, denoted by $K_i = g(D_i)$. We choose a simple method that defines the clusters as the proper percentiles on the cumulative distribution function (CDF) of D_i. The uniform partitions along the probability dimension of the CDF of D_i create the corresponding non-uniform partitions (bins) along the DTW dimension, and the index of the partition (bin) that D_i falls into on the DTW dimension is the K_i. Thus, K_i is an abstraction of the possible trajectory T_i of agent X_i. For instance, in the trajectory abstraction block of Figure 1, K_i abstracts T_i of the blue agent into four classes (an abstracted class of an agent represents a set of similar trajectories for the agent, in the sense of their DTW differences), and indexes them with the support $\{1, 2, 3, 4\}$.

The trajectory abstraction block outputs the abstracted index tuple $K_{1\sim n} := (K_1, K_2, ..., K_n)$ at $\Theta_{1\sim n}$[3]. The rationale is that in a scenario, we represent the possible abstracted class of a task X_i with an index K_i, and a simulation as a co-occurrence of these abstracted indexes $K_{1\sim n}$.

2.3 Computing interaction

To exploit the co-occurrence of the abstracted indexes among agents in estimating the inter-agent interaction, we use the mutual information (MI):

$$
\begin{aligned}
I(K_i; K_{-i}) &= H(K_i) - H(K_i|K_{-i}) \\
&= \mathop{\mathbb{E}}_{(K_i, K_{-i})}\left[\log \frac{P(K_i, K_{-i})}{P(K_i)P(K_{-i})}\right]
\end{aligned}
\tag{1}
$$

In Equ.(1), $H(K_i)$ measures the uncertainty in predicting agent X_i's abstracted trajectory class, while $H(K_i|K_{-i})$ measures the uncertainty in predicting agent X_i's abstracted trajectory class influenced by the classes of all the rest agents. Thus MI characterizes the influence of knowing agents X_{-i}'s abstracted trajectory index tuple in predicting agent X_i's abstracted trajectory class. This intuition is further pictured in the MI block of Figure 1, where the figure above illustrates the possible trajectory classes (indexes) for agent X_i (the blue agent) when the other agents' possible trajectory classes are unknown, while the figure below shows how agent X_i's possible trajectory classes are restricted by other agents' trajectory classes. The reduced amount of the abstracted trajectory classes of agent X_i reflects the interaction between agent X_i and agents X_{-i}, in the given scenario.

[3] Note that K_1=1 and K_2=1 are two different events. The abstracted index enumerates the event set of each individual agent.

3 Experiment Design for Evaluation

We design two sets of experiments to systematically verify the efficacy and utility of the algorithm. The first set of experiments verifies the efficacy of the measurement in characterizing the interaction difficulty of a scenario. This set of experiments consists of two parts. Part-I compares the measurement and a baseline on designated scenarios to demonstrate that the measurement presents advantages over the baseline on anticipated aspects. Part-II compares the measurement and the baseline in three data domains, both qualitatively and quantitatively. The second set of experiments aims to verify the utility of the measurement to approximate scenario generalization, and conducts a comparison with the baseline.

In the future submission, we will describe how to exploit the proposed measurement to estimate scenario generalization. We will also provide full experimental results on Egocentric Representative (G) domain, by qualitatively comparing the measurement with the baseline [Berseth *et al.*, 2013].

4 Summary

We propose an algorithm to measure task-level inter-agent interaction in decentralized crowd scenarios, and exploit the measurement to estimate the scenario generalization of a learning model in crowd simulation. Experiment results validate the efficacy in characterizing interaction difficulty. In addition, the consistency with the ranking of true scenario generalizations on multiple candidate domains implies the potential of the measurement in helping to select suitable training and testing domains, before actually training and testing a model.

References

[Berseth *et al.*, 2013] Glen Berseth, Mubbasir Kapadia, and Petros Faloutsos. Steerplex: Estimating scenario complexity for simulated crowds. In *Proceedings of Motion on Games*, pages 67–76. 2013.

[Qiao *et al.*, 2018] Gang Qiao, Sejong Yoon, Mubbasir Kapadia, and Vladimir Pavlovic. The role of data-driven priors in multi-agent crowd trajectory estimation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[Qiao *et al.*, 2019] Gang Qiao, Honglu Zhou, Sejong Yoon, Mubbasir Kapadia, and Vladimir Pavlovic. Scenario generalization of data-driven imitation models in crowd simulation. In *ACM SIGGRAPH Conference on Motion, Interaction and Games (MIG)*, 2019.

[Salvador and Chan, 2007] Stan Salvador and Philip Chan. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580, 2007.