Authorial Idioms for Target Distributions in TTD-MDPs

David L. Roberts, Sooraj Bhat, Kenneth St. Clair, and Charles L. Isbell
College of Computing
Georgia Institute of Technology
Atlanta, GA 30032
{robertsd,sooraj,krs3,isbell}@cc.gatech.edu

Abstract

In designing Markov Decision Processes (MDP), one must define the world, its dynamics, a set of actions, and a reward function. MDPs are often applied in situations where there is a clear choice of reward functions and in these cases significant care must be taken to construct a reward function that induces the desired behavior. In this paper, we consider an analogous design problem: crafting a target distribution in Targeted Trajectory Distribution MDPs (TTD-MDPs). TTD-MDPs produce probabilistic policies that minimize divergence from a target distribution of trajectories from an underlying MDP. They are an extension of MDPs that provide variety of experience during repeated execution. Here, we present a brief overview of TTD-MDPs with approaches for constructing target distributions. Then we present a novel authorial idiom for creating target distributions using prototype trajectories. We evaluate these approaches on a drama manager for an interactive game.

Introduction

Frequently in the design of AI systems, a human is faced with the task of constructing rules, environments, or instructions that agents use as the basis for their reasoning process. For example, consider the task of building an interactive game with an AI subsystem. Modern games are rich, complex systems requiring the subsystem to take on a variety of roles: tactical or strategic opponent, partner, support character, and commentator. Each role must be specified exactly, often by a designer unschooled in the art of AI. In the larger class of *interactive drama*, the task is even more difficult. A designer must create an environment where a user can explore and create her own story while at the same time ensure a coherent and entertaining experience. Further, such environments must support repeated play; that is, the user should be able to experience the story again and again without the story becoming too predictable. In practice, the effort of authoring such an interactive game can be monumental, often requiring years of development.

One recent technique for approaching this particular problem is to think of interactive drama as a Markov Decision Process (MDP): plot events correspond to states; actions taken by a central coordinator or *drama manager (DM)* correspond to MDP actions; player actions in the game world are modeled as probabilistic transitions between states; and an author-supplied evaluation function over stories is cast as the reward function. Thus, the problem of building an interactive drama becomes, in part, a problem of defining each of the components of the MDP, especially the reward function.

Here we consider the difficulties that arise when authoring *Targeted Trajectory Distribution Markov Decision Processes* (TTD-MDPs) (Roberts *et al.* 2006; Bhat *et al.* 2007). TTD-MDPs are a class of Markov Decision Processes (MDPs) specifically designed for the agent coordination problem that arises in interactive drama and similar domains. TTD-MDPs support *variety of experience*, allowing for repeated play that appears unpredictable to the user, but adheres to the game designer's aesthetics. In previous work, an earlier reinforcement learning approach that maximizes the author's evaluation function has been shown to target a small set of highly-rated stories that do not provide the variety of experience we desire. As we shall see, in TTD-MDPs, the problem of defining a reward function becomes instead the problem of defining a distribution of *trajectories* or possible stories.

In the next sections, we will present an overview of TTD-MDPs and describe two idioms for authoring target distributions. We will then present experiments in a real-world test domain: a drama management MDP for the interactive fiction *Anchorhead*.

TTD-MDPs

A traditional MDP is defined by a tuple $(\mathcal{S}, \mathcal{A}, P, R)$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of actions, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is a transition model, and $R : \mathcal{S} \rightarrow \mathbf{R}$ is a reward function. The solution to an MDP is a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$. An optimal policy ensures that the agent receives maximal long-term expected reward.

Likewise, a TTD-MDP is defined by a tuple $(\mathcal{T}, \mathcal{A}, P, p)$, where $\mathcal{A}$ and P are defined as above, $\mathcal{T}$ is the set of finite-length trajectories of MDP states, and $p : \mathcal{T} \rightarrow [0, 1]$ is a target distribution over complete trajectories. The target distribution in a TTD-MDP conceptually replaces the reward function in a traditional MDP. The solution to a TTD-MDP is a policy $\pi : \mathcal{T} \times \mathcal{A} \rightarrow [0, 1]$ providing a distribution over actions for every trajectory. An optimal policy results

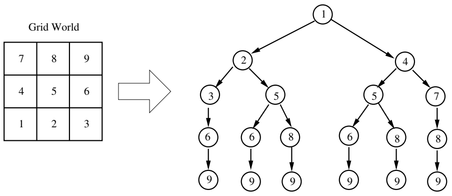

Figure 1: Sample 3×3 gridworld with deterministic actions *Right* and *Up*, along with the resulting trajectory tree.

in long-term behavior as close to the target distribution as possible.

Trajectories represent total history traces of an online decision making process. Consider Figure 1, a 3×3 gridworld where there are two deterministic actions: *move right (R)* and *move up (U)*. All legal trajectories have initial state 1 and terminating state 9. The right side of the figure depicts the corresponding *trajectory tree*. A trajectory tree is simply a graphical representation of the valid trajectories and the prefix relationship that governs partial trajectories. Consider the partial trajectory $1 \xrightarrow{R} 2$, for instance. It is a prefix of two immediate subsequent partial trajectories ($1 \xrightarrow{R} 2 \xrightarrow{R} 3$ and $1 \xrightarrow{R} 2 \xrightarrow{U} 5$) as well as three other partial trajectories and three complete trajectories. A policy π would provide a distribution over available actions for each internal node in the tree.

Unfortunately, it is not always possible to find a stochastic policy that exactly solves a TTD-MDP (*e.g.*, in the case where the transition probabilities lead to non-zero probability mass on a state where we desire to never go). Thus, we must consider approximations to the desired distribution of trajectories. The following approach was first presented in (Bhat *et al.* 2007). The approach involves finding a policy that minimizes the *Kullback-Leibler divergence* between the desired distribution of trajectories p and the distribution q that is realized through the combination of the policy and the transition model:

$$\begin{aligned} D_{KL}(p\|q) &= \sum_{\tau} p(\tau) \log \frac{p(\tau)}{q(\tau)} \\ &= \sum_{\tau} p(\tau) \log p(\tau) - \sum_{\tau} p(\tau) \log q(\tau) \end{aligned}$$

KL-divergence is not a distance, as it is asymmetric; however, it is well-understood with several important properties. In particular, it is consistent, always non-negative and zero only when p and q are equal. For TTD-MDPs, p is given, and q is defined as follows:

$$q(\tau) = \prod_{t \preceq \tau} w(t) \qquad (1)$$

$$\text{where} \quad w(t') = \sum_{a} P(t'|a,t) \cdot \pi(t,a) \qquad (2)$$

Here, τ represents complete trajectories, t and t' represent (partial or complete) trajectories, and the symbol $\preceq$ is used to denote the prefix relationship. The probability $w(t')$ represents the frequency with which t' is targeted when the process is at t. It combines information about the probabilistic policy and world dynamics at t. Thus, the product of these one-step transition frequencies, $w(t')$, yields the probability of a complete trajectory, $q(\tau)$.

Our objective function for optimization now becomes:

$$\begin{aligned} \max_{\pi} \sum_{\tau} p(\tau) \log q(\tau) &= \max_{\pi} \sum_{\tau} p(\tau) \log \prod_{t \preceq \tau} w(t) \\ &= \max_{\pi} \sum_{\tau} p(\tau) \sum_{t \preceq \tau} \log w(t) \\ &= \max_{\pi} \sum_{\tau} \sum_{t \preceq \tau} p(\tau) \log w(t) \end{aligned}$$

Note that a partial trajectory t contributes $p(\tau) \log w(t)$ to the sum for each complete trajectory τ for which it is a prefix. We can define a function over complete trajectories summarizing the factor of $\log w(t)$ that t contributes, $m(t) = \sum_{t \preceq \tau} p(\tau)$. Note that this definition implies the recursive definition $m(t) = \sum_{t \to t'} m(t')$, *i.e.* the mass of a trajectory t is the sum of the masses of its children t' in the trajectory tree. Our objective function is then:

$$\max_{\pi} \sum_{t} m(t) \log w(t) \qquad (3)$$

Note that $m(t)$ represents the total probability mass contained in the subtree rooted at t.

Though the explanation is beyond the scope of this paper, the objective function in Equation 3—encapsulating a search over global policies π—can be reformulated in terms of a series of independent local optimizations searching over local policies π_t—one for each internal node in the trajectory tree. We achieve the globally KL-optimal solution regardless of the order in which we perform these local optimizations; thus, the TTD-MDP can be solved online. One can start by processing the root of the trajectory tree, then process the children of the root, and so on, to compute a policy for each node in the tree. Better, if $m(t)$ can be computed efficiently, we can solve only the local optimizations that we actually encounter, interleaving the local optimization steps with taking actions in the world. The local policy π_t produced by the local optimization tells us how to take the next action, and the action places us at a new node in the trajectory tree.

Authoring TTD-MDPs

As with any AI technique, one must specify the components of a TTD-MDP. We are inspired by interactive dramas, and the mechanisms for specifying a good story, so we focus here on the problem of authoring target distributions in TTD-MDPs. The number of valid trajectories is often large, so one cannot simply enumerate all possible trajectories and manually assign each one a probability weight. To be authorially feasible, there must be a compact way of specifying the distribution.

In the original formulation of drama management as an optimization problem (prior to TTD-MDPs), authors of interactive dramas were expected to provide an evaluation function that encapsulated the quality of a complete story. Insofar as that is a reasonable requirement, it is possible to use such an evaluation function to induce a reasonable distribution over stories. For instance, we may wish that stories occur with a probability proportional to their evaluation score: $p(\tau) \propto R(\tau)$.[1] Unfortunately, such an approach still does not eliminate a difficult hurdle: solving a TTD-MDP efficiently and optimally requires $m(t)$ to be computed quickly. Below, we will describe two techniques that address this difficulty.

Sampling

The first authorial idiom we consider constructs an estimate for $p(\tau)$, from which we will compute $m(t)$. When an author has defined an evaluation function, we can use it to construct a distribution $p(\tau)$ as above; however, this approach is infeasible for large trees. Instead, we can approximate $p(\tau)$ by sampling a subset of trajectories $\mathcal{T}_s \subset \mathcal{T}$ (via simulation of gameplay, for instance) and then using $\widetilde{p}(\tau)$ as a replacement for $p(\tau)$, where $\widetilde{p}(\tau) \propto R(\tau)$ for $\tau \in \mathcal{T}_s$ and $\widetilde{p}(\tau) = 0$ otherwise. We construct $m(t)$ from this estimate; there will be an $m(t)$ value for each node in the trajectory tree induced by $\mathcal{T}_s$. Because we control the size of $\mathcal{T}_s$, we can adjust it to fit our memory requirements.

There are several choices for generating samples. Following previous work, we could first select uniformly from the set of possible actions and then select uniformly from the set of successor trajectories, to generate a complete trajectory. An alternative is *Markov Chain Monte Carlo (MCMC)* sampling, a rejection sampling technique used to draw *i.i.d.* samples from a distribution that is difficult to sample directly. In our experiments, we use the Metropolis-Hastings algorithm (Metropolis *et al.* 1953; Hastings 1970). The pseudo-uniform sampling procedure described above is used as the (unconditional) MCMC proposal distribution.

It is important to include the action in the sampling process as it constrains the set of states that can be reached. Consider actions a_1 and a_2, and partial trajectories t, t_1 and t_2, where t is parent of t_1 and t_2 in the trajectory tree. If $P(t_1|a_1, t) = 0.2$ and $P(t_2|a_1, t) = 0.8$, then both t_1 and t_2 are valid successor trajectories; however, if $P(t_1|a_2, t) = 0.0$ and $P(t_2|a_2, t) = 1.0$, then care must be taken because t_1 can never actually occur with action a_2. Further, in some domains reward is based on both the sequence of states and the actions taken by the system. In drama management, for example, the author seeks to avoid the perception by the player that the drama manager is overly manipulative, therefore penalizing instrusive actions.

The sampling approach has its drawbacks. Due to non-determinism in $P(t'|a, t)$ and the sheer size of the trajectory space, it is quite likely that an unsampled part of the full trajectory tree will be encountered during an episode. Presumably this is more likely in the low probability portions of the tree, so one may have already been doing poorly to have entered into that part of the space. Further, if the deviation occurs near the leaves of the trees, it may be possible to perform online resampling to recover. In the drama management domain, it appears that good stories often have common prefixes (Nelson *et al.* 2006), so it may be that one is most likely to deviate only after one has already ensured a good story.

Prototypes

The second authorial idiom we consider computes $m(t)$ directly (which induces a target distribution $p(\tau)$ that is never represented explicitly) and is based on (*i*) a set of prototypical "good" trajectories and (*ii*) a distance metric over trajectories. Combining the distance metric with the prototypes can induce a probability distribution over all possible trajectories. One such method is to construct a Gaussian mixture model (GMM) over the set of prototypes:

$$m(t) = \sum_{i=1}^{N} w(\mu_i) \cdot \mathcal{N}(t; \mu_i, \sigma_i) \qquad (4)$$

where

$$\mathcal{N}(t; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-d(t,\mu)^2/2\sigma^2}, \qquad (5)$$

μ_i is a prototype and the centroid of a Gaussian distribution with variance σ_i^2, $w(\mu_i)$ is the prior weight given to each centroid, and d is some distance measure on trajectories. There are a number of choices one could make for a distance metric. We explore three classes.

The first is *Levenshtein distance* or *edit distance* (Levenshtein 1966). The edit distance is a computationally efficient generalization of the Hamming distance (Hamming 1950) that is defined over strings of unequal length and handles insertions, deletions, and substitutions. Consider three trajectories: $t_1 = 1 \xrightarrow{R} 2 \xrightarrow{U} 5$, $t_2 = 1 \xrightarrow{U} 4 \xrightarrow{U} 5$, and $t_3 = 1 \xrightarrow{U} 2 \xrightarrow{U} 5$. The edit distance between t_1 and t_2 is $d_E(t_1, t_2) = 2$ because they differ in the first action and second state. By contrast, $d_E(t_1, t_3) = 1$.

There are several variations of edit distance. Let $l(t)$ be the length of a trajectory t and $\rho(t, n)$ be the prefix of t with length n; if $l(t) < n$, we define $\rho(t, n) = t$. Using $\rho(t, n)$, we can begin to construct measures of distance that are better suited to different domains. For example, in the drama management domain, deviations from desirable trajectories near the root of the trajectory tree are potentially more costly than deviations later. Thus, we may wish to consider a *scaled edit distance* between trajectories t and μ: $d_{SE}(t, \mu) = 1 + |l(t) - l(\mu)| \cdot d_E(t, \rho(\mu, l(t)))$.

A second class of distance measures involves variations of the *longest common subsequence*. A subsequence of a trajectory is another trajectory formed by deleting some of the elements of the original trajectory without disturbing the relative position of the states (and actions). The longest common subsequence between two trajectories is the longest subsequence that appears in both strings.

A third class of distance measures uses the evaluation function directly when it is available. Typically, such functions are implemented as a linear combination of features about the story: $R(t) = \sum_k w_k \cdot f_k(t)$ (we refer the interested reader to (Weyhrauch 1997; Nelson & Mateas 2005)

[1] For ease of explanation, we have required $m(t)$ to represent the total probability mass located at the subtree rooted at t, but in actuality only a *relative* measure (w.r.t. the siblings of t) is needed, so in practice, normalization of the probabilities is unnecessary.

for details). Here, distance from a prototype is simply defined as $d_F(t,\mu) = |R(t) - R(\mu)|$, which we shall call the *feature distance*. We could also construct a vector representation of these features $\vec{R}(t) = [w_1 \cdot f_1(t), w_2 \cdot f_2(t), \ldots]$ and use those vectors in a multivariate GMM. The weights on the features have an effect similar to changing the covariance matrix of the GMM, providing an interesting prospect for authorial control.

One problem with this approach is that story features are not necessarily well defined over partial stories. We overcome this by defining a *blended feature distance* function:

$$d_{BF}(t,\mu) = \min\left[1, \frac{l(t)}{l(\mu)}\right] \cdot d_F(t, \rho(\mu, l(t))) \tag{6}$$

$$+ \max\left[0, \left(1 - \frac{l(t)}{l(\mu)}\right)\right] \cdot d_{\widetilde{E}}(t, \rho(\mu, l(t))) \tag{7}$$

where d_F is a function based on the features and $d_{\widetilde{E}}$ is some form of the edit distance. Equation 6 represents increasing contributions of the drama management features as the length of trajectory t approaches that of μ. Similarly, Equation 7 represents decreasing contributions from the edit distance as the length of the trajectories become similar.

Using prototypes provides a number of distinct advantages over sampling-based approaches. In comparison to authoring a reward function for an MDP, hand selecting a small number of prototypes may be significantly easier. Further, the prototype approach—especially using GMMs—allows efficient computation of $m(t)$ for partial trajectories. Even better, this approach provides a smooth distribution such that no trajectory has zero mass. Thus, it is not possible to fall out of the sampled space.

On the other hand, the problem of authoring has become the problem of choosing an appropriate distance function. When an evaluation function is available we can use it to capture subtleties in the values of states; however, when such functions are difficult to construct, it is not clear how well methods like edit distances can do. Finally, prototypes must come from somewhere. They may be provided by the author, but they could also be generated by a sampling process similar to the ones described above.[2]

Results

We report on experiments designed to illustrate the overall performance characteristics of the two authorial idioms discussed above as well as show how some of the variations perform. As TTD-MDPs were originally developed for drama management, we evaluate the approaches on the two drama management TTD-MDPs described by Roberts *et al.* (2006), namely, Anchorhead and Alphabet City. Because existing work has already indicated the potential for sampling approaches to be effective, we choose to simply highlight the relationship between MCMC and uniform sampling (rather than provide a detailed study of sampling performance), and we instead focus the bulk of our attention on

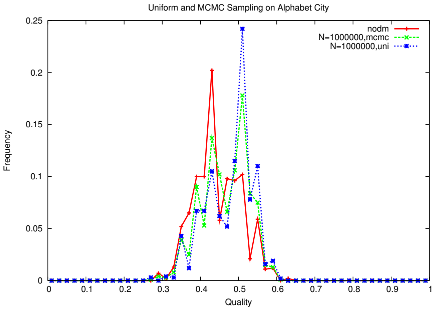

Figure 2: A comparison of uniform and MCMC sampling on Alphabet City.

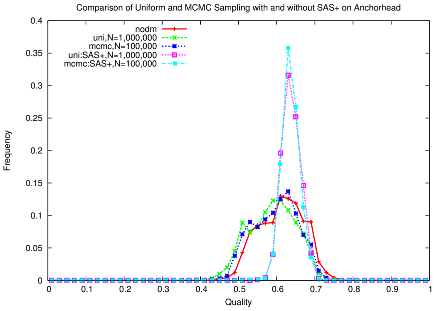

Figure 3: A comparison of uniform and MCMC sampling with and without SAS+ recovery on Anchorhead.

experiments in the prototype-distance idiom. In those cases where we evaluate the prototype-based approaches, we ignore hand-authored models to avoid skewing the results too much by our particular choice of prototypes.

Comparison of Sampling Approaches

In Figures 2 & 3, we present results on both the Anchorhead and Alphabet City domains in the form of a story quality histogram. Such histograms have been used for qualitative analysis of drama management systems in earlier work. In these figures, we examine three different techniques: uniform sampling, MCMC sampling, and sampling with SAS+ recovery (Roberts *et al.* 2006). The key `nodm` refers to stories for which no drama management was applied and is used as a baseline for assessing the effect of applying drama management. Of interest in these plots is the relative shape of the histogram curves. Qualitatively, the goal of the drama manager is to shift the distribution "right and up" (increasing the *quality* of stories) while preserving its "width" (ensuring the *variety* of stories).

First, we discuss the results presented in Figure 2. The three curves in this figure correspond to the `nodm` baseline as well as uniform and MCMC sampling with 1,000,000 samples (and a burn-in of 1,000 in the case of MCMC). The `nodm` baseline is relatively higher toward the bottom end of the evaluation scale and lower toward the top end of the scale than the other two curves. This `nodm` baseline is obtained by simulating gameplay without any DM actions taken, so this result is consistent with our expectations. On

[2] There are subtleties. In the uniform sampling case, the evaluation function provides the prior probability of each centroid. In the case of MCMC, the set of prototypes are already chosen according to the correct probability so the priors should be uniform.

the other hand, we found that MCMC performed slightly worse than uniform sampling, as evidenced by the MCMC quality curve being mostly between nodm and uniform (*i.e.* below nodm but above uniform at the bottom of the scale and above nodm but below uniform at the top end). Uniform sampling performing better than MCMC will be common to most of the experimental results presented in this paper. We believe this relative performance gap occurs as a result of MCMC sampling tending to "hang around" good parts of the space whereas uniform explores more thoroughly.

In Figure 3, the results of experiments on Anchorhead similar to those performed on Alphabet City are presented. First, we point out that the nodm case slightly beats the performance of uniform and MCMC sampled TTD policies. This is in contrast to the results obtained on Alphabet City. There is, however, a simple explanation for this difference in performance. Although not presented in detail, the set of actions available to the DM in both story worlds have slightly different characteristics. Most notable is the use of a temp_denies action in Anchorhead, where the DM can take an action to temporarily deny a plot point from occurring in the game. At some point later in the game, the DM must reenable that plot point with another action. This would not be a problem for the DM if we could guarantee that the policy is completely specified for every partial story; however, because we construct the policy based on a sampled trajectory tree, there are frequently deviations from that tree before the reenable action can be taken by the DM.

For example, the Alphabet City story world has an average story length of roughly 9 plot points whereas the average story length in Anchorhead is approaching 30. In both cases, the average depth of deviation (*i.e.* number of plot events that occur during an episode before an unsampled part of the trajectory space is encountered) is approximately five. Thus, the Anchorhead domain is at a disadvantage for the following reason: when a plot event is temporarily denied by the first few DM actions, if it is not reenabled before deviation from the tree occurs, then it cannot occur in the story.

To more fully characterize the effect of falling off the tree, we additionally show the result of using Weyhrauch's SAS+ online sampling search (Weyhrauch 1997). There are two interesting things to notice. First, the addition of SAS+ significantly improves the story qualities, compared to the nodm baseline and the TTD policies without a recovery strategy.[3] Second, the curves are nearly identical, indicating that the deterministic search of SAS+ is able to realize its goals with high probability. This structure in the quality histogram (a steep, impulse-like curve) indicates potential issues for replayability.

Comparison of Prototype-Distance Models

To examine the performance of various prototype-distance models, we conducted a number of experiments to test some of the many free parameters of the system. In particular, we looked at different distance metrics, different Gaussian widths and different numbers of prototypes. Due to space constraints, we opt to present a selection of the results, which are intended to be representative of the other experiments we conducted as well as to provide insight into the characteristics of the approach.

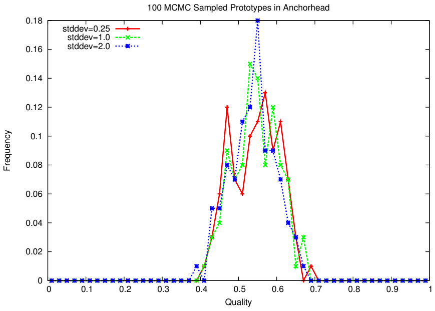

Figure 4: A comparison of models built with 100 prototypes generated by MCMC sampling with various standard deviations.

Because we are most interested in understanding how the models react to changes in parameters (*e.g.* changes in how the author specifies the TTD-MDP), we will focus the bulk of our analysis in this section on the drama-management-specific distances (*feature distance* and *blended feature distance*) described before. Although not presented here, we have identified similar characteristics in other test domains where the more generic variants of Levenshtein distance and longest common subsequence are more applicable.

In Figure 4 we plot quality histograms for three different prototype models on the Anchorhead domain. The models were constructed with 100 MCMC sampled prototypes after a 1,000 step burn-in and used the feature distance measure. We tested three different standard deviations: 0.25, 1.0, and 2.0. The results we obtained were somewhat counter-intuitive. Specifically, we found that as the width of the Gaussians increased, the width of the resulting quality histogram decreased. We believe the reason for this is related to the idea of a "plateau" in optimization problems. Specifically, with narrow mixture components in the GMM, it is likely that the space between them will have relatively stable and low probability mass; however, as the width increases, one would find that the tails of the Gaussians tend to overlap, forming a nice neighborhood of trajectories that are common to a number of centroids. Thus, during an episode with small-width Gaussians, if the nondeterminism in the environment causes the current episode to enter the flat space between centroids, the result is likely to end up resembling a random walk through the space. Thus, the quality histogram for experiments with small standard deviation tend to have more mass at the tails. Larger standard deviations do not seem to suffer from this effect.

Next, we consider the effect that the number of prototypes has on the resulting quality distribution. In Figure 5, we present two prototype models. The prototype models used for this plot were constructed using 100 prototypes after 1,000 sample burn-in and used the blended feature distance measure.

We found that the rejection step of the MCMC sam-

[3] Earlier work has shown that SAS+ alone does not perform well on Anchorhead (Nelson *et al.* 2006).

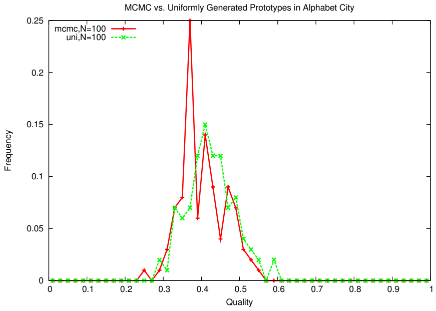

Figure 5: A comparison of models built with 100 prototypes generated by uniform and MCMC sampling.

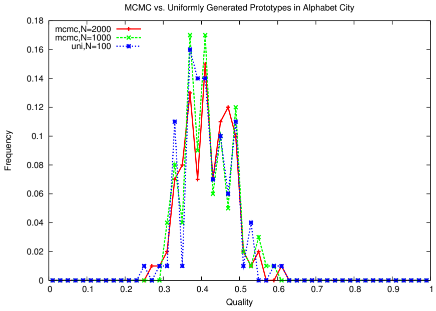

Figure 6: A comparison of prototype models built from 100 uniformly sampled prototypes and 2000 MCMC sampled prototypes.

pling procedure often leads to clusters of samples, especially when fewer samples are used as prototypes. For example, consider Figure 5. The quality of the MCMC-sampled prototype model is substantially lower than that of the uniform sampled model. Now consider Figure 6, where the same 100-sample uniform model is compared to a 1,000-sample MCMC model as well as a 2,000-sample MCMC model. The quality of the 1,000-sample MCMC model is roughly equivalent to that of the 100-sample uniform model—an order of magnitude increase in the number of prototypes is required for the performance of MCMC sampling prototypes to match that of uniform sampled prototypes. Further, notice how the additional increase in performance obtained by doubling the number of prototypes to 2,000 is noticeable, but not particularly pronounced.

Concluding Thoughts

In this paper, we have discussed the power and performance of two authorial idioms for target distributions in TTD-MDPs. Our discussion and analysis is firmly entrenched in the domain of interactive narrative. Our focus on interactive narrative is motivated by the unique challenges it provides to designers of AI technologies as well as the challenges provided to the author. We have shown the effectiveness of these authorial idioms through an empirical analysis of two interactive narrative domains previously studied in the literature. Specifically, we have shown that our novel approach to MCMC sampling of trajectory trees performs approximately as well as the previous uniform sampling approach. In addition, we have shown that the novel prototype-distance approach and the distance metrics we define, when constructed using a set of sampled prototypes, show promise for continued development.

Acknowledgments

This research was performed while on appointment as a U.S. Department of Homeland Security (DHS) Fellow under the DHS Scholarship and Fellowship Program, a program administered by the Oak Ridge Institute for Science and Education (ORISE) for DHS through an interagency agreement with the U.S Department of Energy (DOE). ORISE is managed by Oak Ridge Associated Universities under DOE contract number DE-AC05-06OR23100. All opinions expressed in this paper are the author's and do not necessarily reflect the policies and views of DHS, DOE, or ORISE. We also acknowledge the support of DARPA under contract No. HR0011-06-1-0021.

References

Bhat, S.; Roberts, D. L.; Nelson, M. J.; Isbell, C. L.; and Mateas, M. 2007. A globally optimal online algorithm for TTD-MDPs. In *Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS07)*.

Hamming, R. W. 1950. Error-detecting and error-correcting codes. *Bell System Technical Journal* 29(2):147–160.

Hastings, W. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57(1):97–109.

Levenshtein, V. I. 1966. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady* 10(8):707–710.

Metropolis, N.; Rosenbluth, A. W.; Rosenbluth, M. N.; Teller, A. H.; and Teller, E. 1953. Equations of state calculations by fast computing machines. *Journal of Chemical Physics* 21(6):1087–1092.

Nelson, M. J., and Mateas, M. 2005. Search-based drama management in the interactive fiction Anchorhead. In *Proceedings of the First Annual Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-05)*.

Nelson, M. J.; Roberts, D. L.; Isbell, C. L.; and Mateas, M. 2006. Reinforcement learning for declarative optimization-based drama management. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-06)*.

Roberts, D. L.; Nelson, M. J.; Isbell, C. L.; Mateas, M.; and Littman, M. L. 2006. Targetting specific distributions of trajecotries in MDPs. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI-06)*.

Weyhrauch, P. 1997. *Guiding Interactive Drama*. Ph.D. Dissertation, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. Technical Report CMU-CS-97-109.