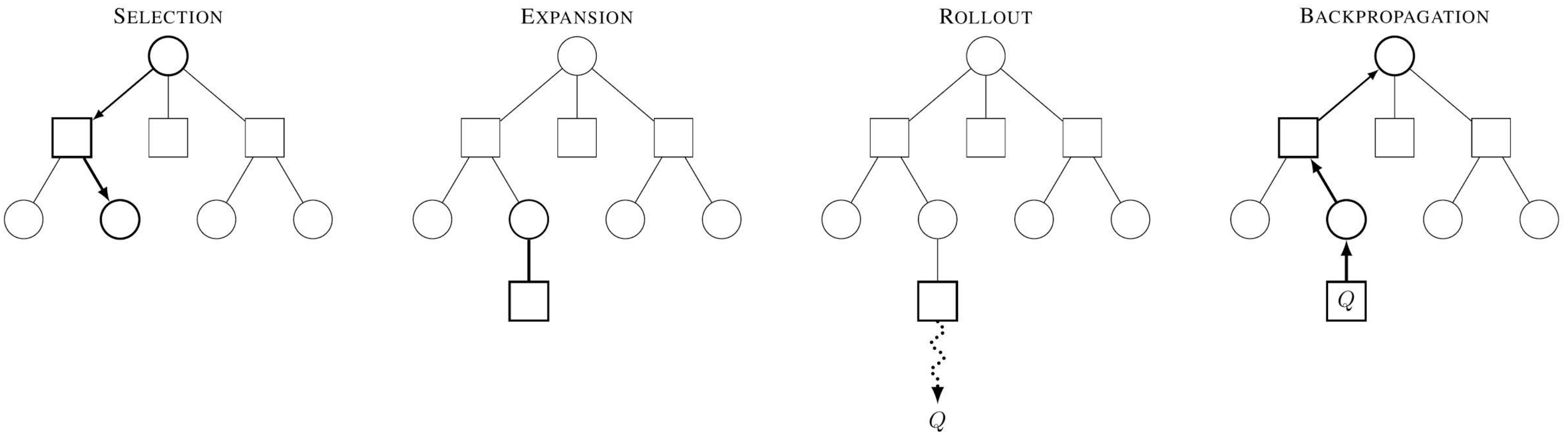
Allstate Updates: Outline

November 16, 2020 — Robert Moss

- **Primer on Monte Carlo tree search (MCTS)**
- **Primer on the cross-entropy method (CEM) for optimization**
- **Updates on optimized MCTS rollouts**
- **Next steps/timeline**

Allstate Updates: Monte Carlo Tree Search (MCTS)

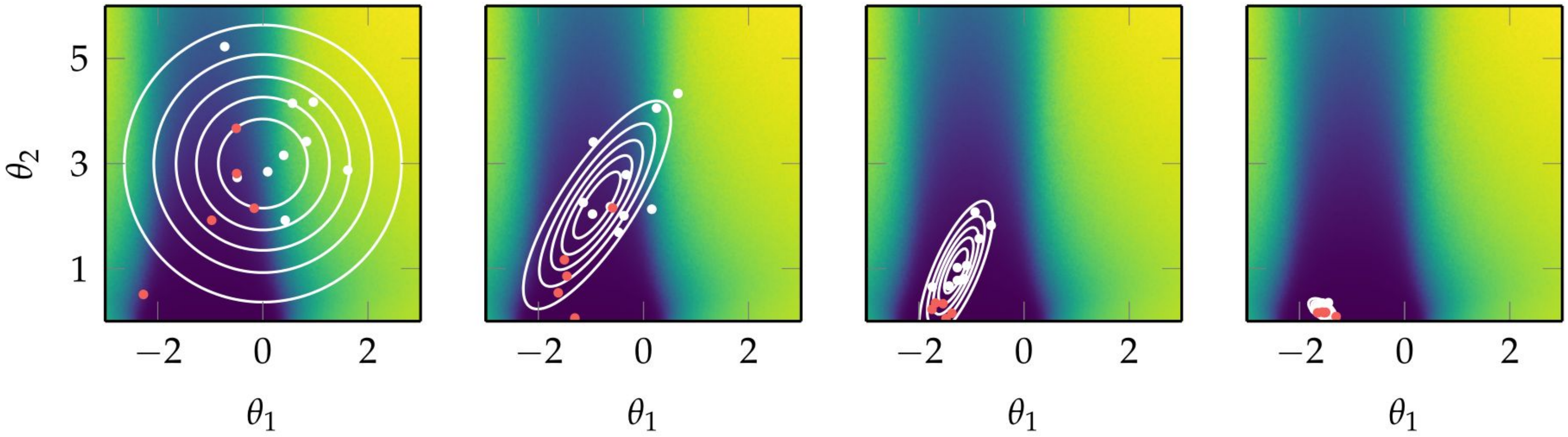
Can we make better action selection choices during the MCTS rollout phase?



- **Four stages of the Monte Carlo tree search (MCTS) algorithm:**
  - ***Selection***: select the best action seen so far using an exploration strategy—generally use upper-confidence bound for trees (UCT)
  - ***Expansion***: apply the selected action a to the current state s in the tree, expand to a new state node s'
  - ***Rollout***: estimate the Q-value (i.e. state-action utility) of a specific branch by applying actions from a random policy to some depth d
  - ***Backpropagation***: record the Q-value back through the trace up the tree

Allstate Updates: Cross-Entropy Method for Optimization

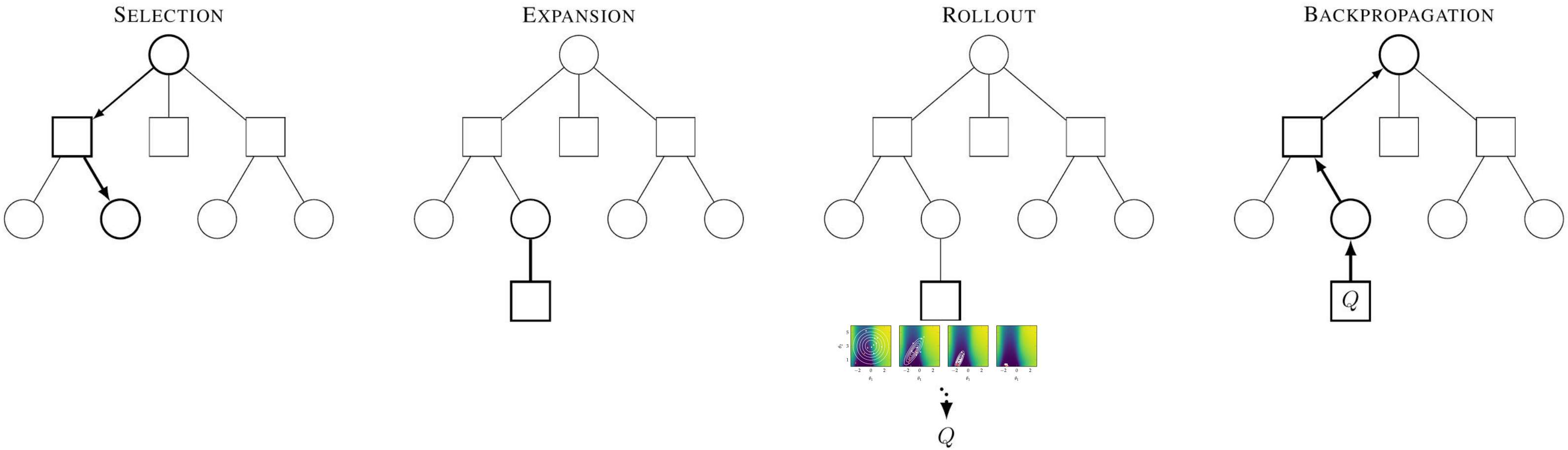
Can we make better action selection choices during the MCTS rollout phase?



- **Stages of the cross-entropy method (CEM) algorithm for optimization:**
  - Given an input importance sampling distribution, randomly sample *m* individuals from this distribution
  - Evaluate the *m* individuals using an objective function (we use the *Q*-value as we want to maximize this)
  - Select the top m_{elite} individuals
  - Fit the importance sampling distribution on those m_{elite} samples

Allstate Updates: Optimized Rollouts

Can we make better action selection choices during the MCTS rollout phase?



- **Optimized rollouts:** use a stochastic optimization algorithm (i.e. CEM) to narrow down the search over environment distributions that lead to failures:
    - Crosswalk problem results: random policy rollouts (~2.6% failure rate, 55083 evals) vs. MCTS+CEM rollouts (~6% failure rate, 30324 evals)
    - Our CEM produces a timeseries importance distribution (i.e. an optimized set of distributions per time step—where time is the rollout depth)
    - Optimization approach is sensitive to hyperparams and can be sample *inefficient*—leading to *more* computational cost on the underlying system
- **ϵ-greedy rollouts:** take random action with probability ϵ, otherwise use current best action (how to define best?)
- **Surrogate model-based approximations of the Q-value function:**
    - **Gaussian process surrogate model:** sample functions from a multivariate Gaussian, fit those functions to the seen datapoints
    - **Neural network-based surrogate model:** train a neural network given samples $Q(s,a)$ or $Q(d,a)$ for distance (generalize for partial observability)

Allstate Updates: Next Steps/Timeline

- **November timeline:**
  - Analyze CEM-based optimized rollouts further
  - Create simple neural network-based Q-approximator
- **December timeline:**
  - Incorporate STL into AST framework:
    - Use STL to define failure (i.e. collision)
    - Use STL robustness to define *distance* metric
- Next meeting (Nov. 30th)