Robust pest management using reinforcement learning

JiaLin Hau*, Marek Petrik, Talha Siddique*, Shadi Atallah

University of New Hampshire (Department of Computer Science, Department of Natural Resources and the Environment)

Abstract

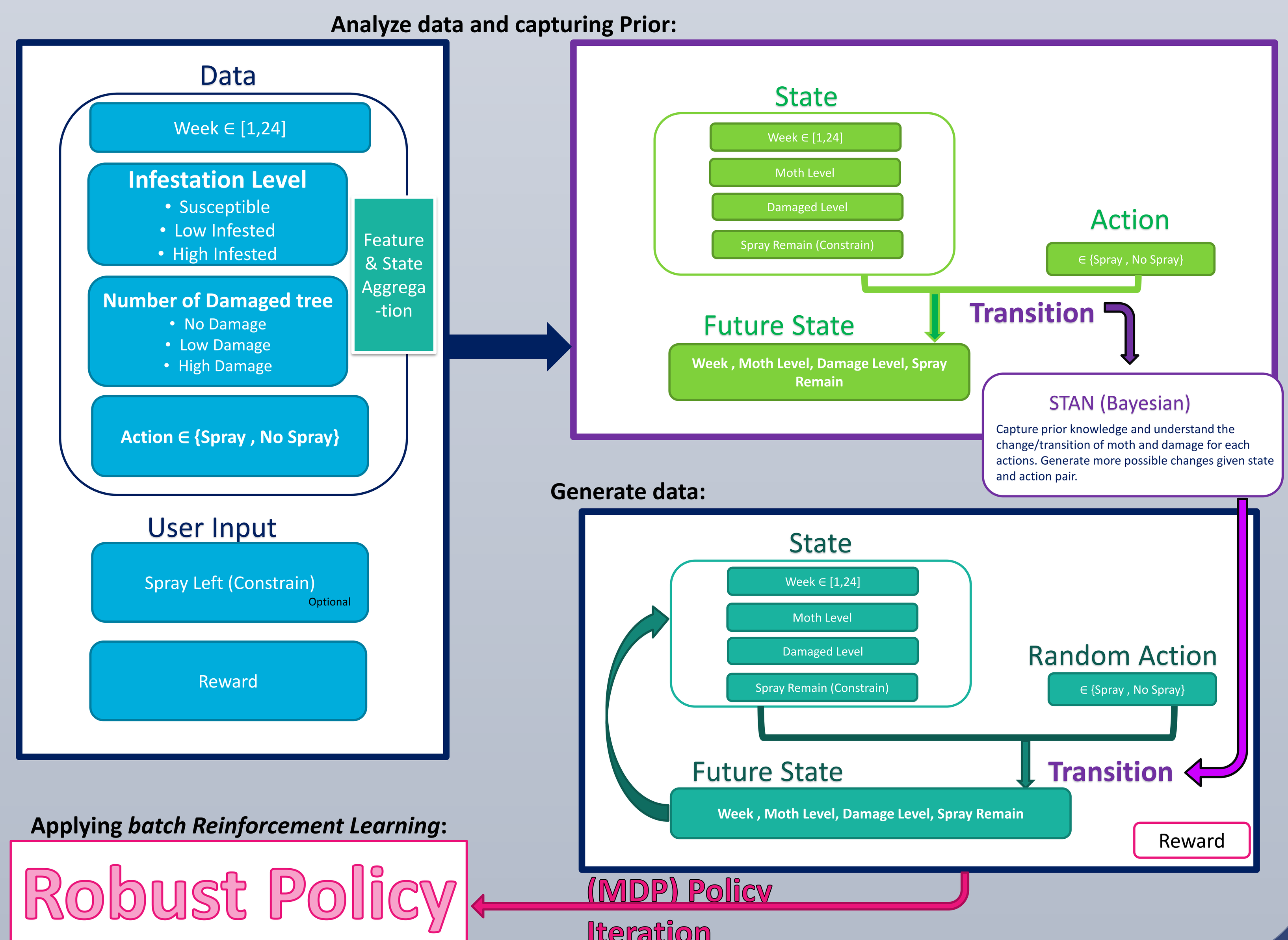
- In agriculture, natural disaster, and medical care having big dataset is impractical. Nevertheless, making a good decision is important.
- Our work is used to solve an issue of batch reinforcement learning where data might be insufficient to compute a good policy.
- We propose a robust reinforcement learning approach that can compute good solutions even when the models or rewards are not known precisely. We use Bayesian models to capture prior knowledge.
- In this work, we show an example of an apple orchardist who must decide how to control the population of codling moth, which is an important apple pest.
- Identifying the optimal timing to spray reduces pesticide use, which can decrease negative impacts on the environment and reduce costs.

Introduction

- We propose to use (STAN) a full Bayesian statistical inference with MCMC sampling to capture prior knowledge and generate more possible data points.
- We use batch reinforcement learning by Lange et al (2012), in which a good policy needs to be computed from a logged dataset without interacting with a simulator to counter the issue where experimental is too time consuming or expensive.
- The decision maker selects a confident interval of the distribution for all parameters and determine a policy that chose the best worst case performance over the confidence region of all actions for every state.
- We decide to use Robust Markov Decision Process (RMDP) because it possesses several important properties: Policy is tractable, easy to understand and follow, include risk in sequential decision making, immune to data and parameter uncertainties.

Methodology

BAYESIAN BATCH RL MODEL



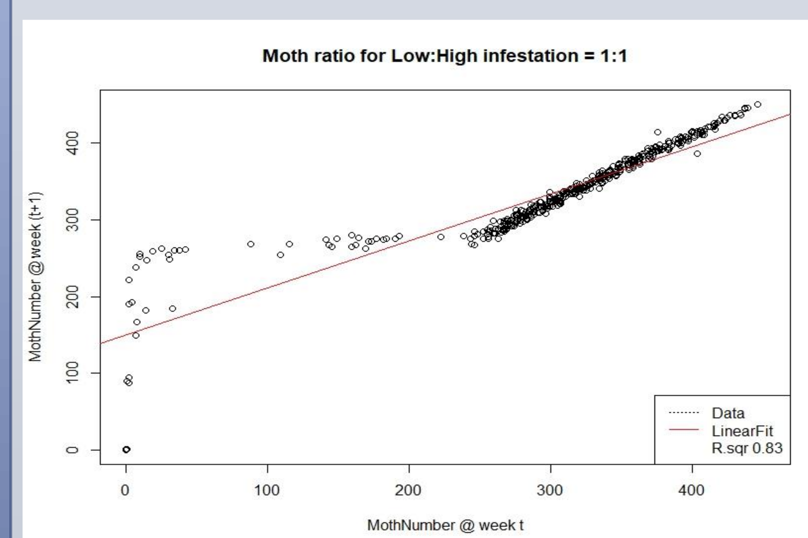
The Natural Resources and Environment expert made a **simulation model** that represent the nature. We will compute a robust optimal policy with **Batch RL** from the data given produce by the simulation model, **without interacting with the simulator and have no knowledge of the internal system of the simulation.**

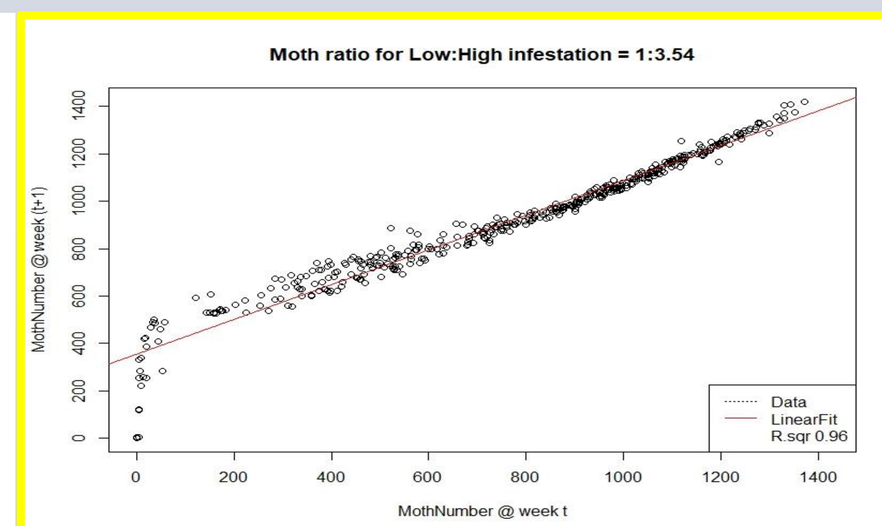
Analyze data and capturing Prior:

- We aggregate these 3 infestation level features into a feature called moth level which represent the number of moths in the farm. We then seek for a ratio of conversion that best explain and maintain the information from these 3 features. We apply the same ratio for the number of damaged trees into damage level.
- After we aggregate the features, we aggregate the state. We divide the features (moth level and damage level) to 10 different quantiles. We aggregate the state so that it turned into a finite discrete variable.

Feature aggregation:

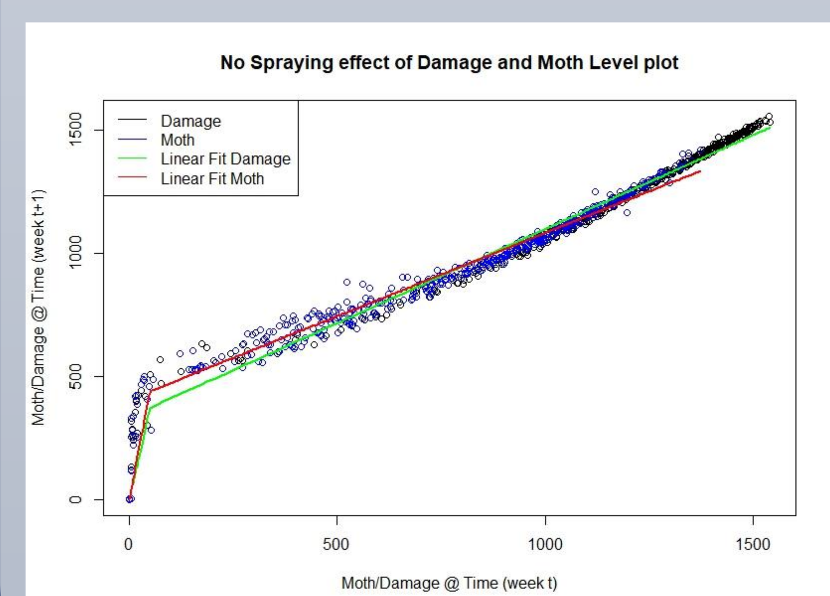
- We aggregate the feature so that we could incorporate our knowledge of exponential growth model for codling moth. Feature aggregation also allow us to decrease the number of states and avoid the curse of dimensionality.
- We know exponential model has a form of $N_t = N_0\, r^t$ ➔ $N_{t+1} = N_t r$ for N : Number of moth, r : growth rate, t : week t^{th}. Therefore, we select ratio where the best fit line of ratio for Low Infestation: High Infestation has the highest R-squared. For each action we generalized and understand how moth and damage transition for each action.

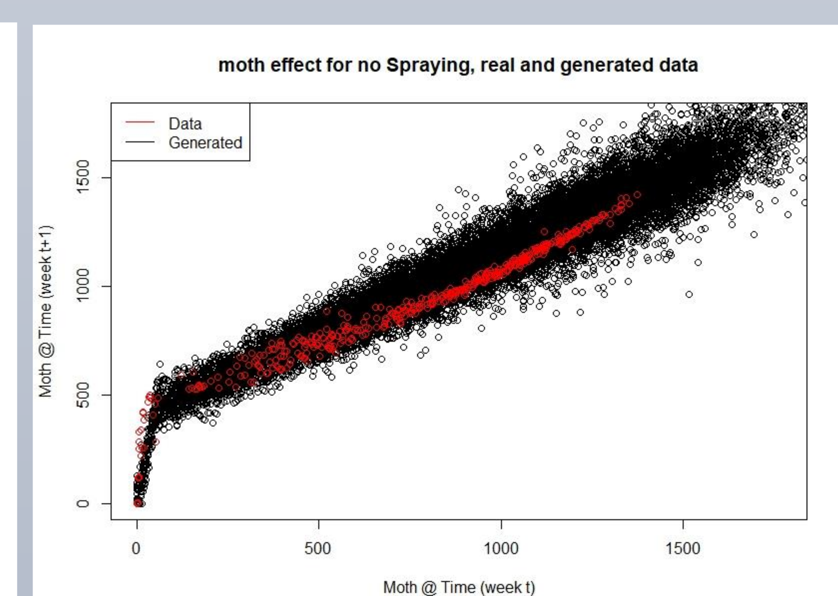




Generate data:

- To acquire generalized transition probability, we use STAN to model the changes in the number of moths and damages. We use the advantages of this programming language to generate more data.





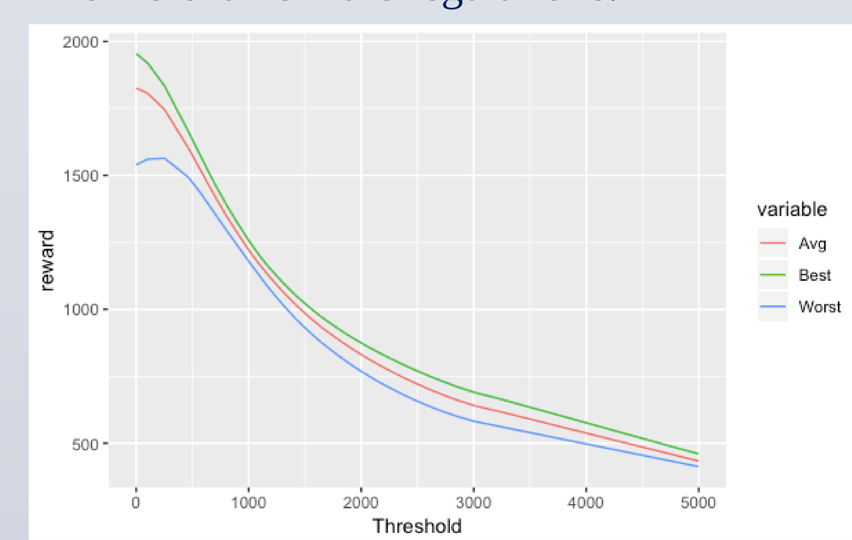
Applying batch Reinforcement Learning (RMDP):

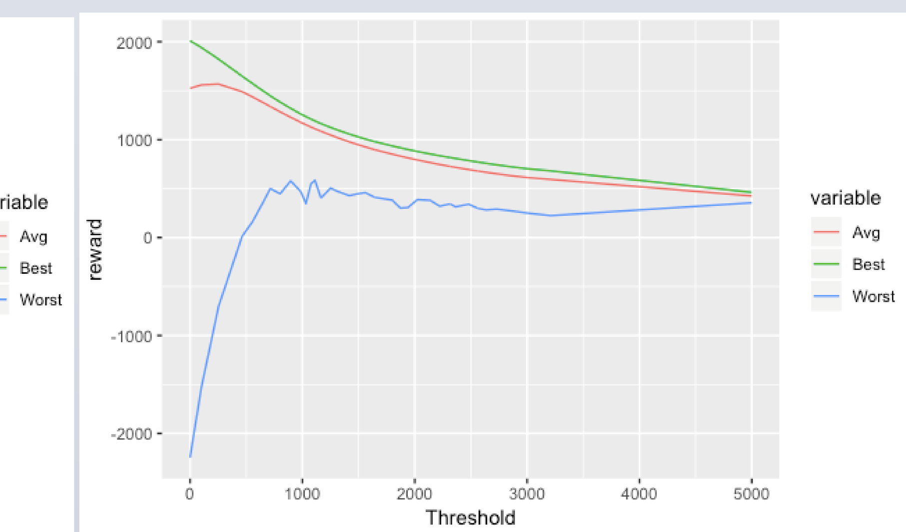
$$p(\quad) \rightarrow is\ a\ performance\ measure/net\ revenue$$
$$r \rightarrow is\ the\ set\ of\ transition\ and\ \lambda\ is\ an\ element\ from\ this\ set$$
$$\pi \rightarrow\ is\ the\ set\ of\ policy$$

We use policy iteration to obtain the optimal policy. Start with an arbitrary action and value, evaluate and improve the policy until the policy is stable. The robust method will focus on the worst case return $\boldsymbol{min_{\lambda\epsilon r} p(\pi_i, \lambda)}$, instead of the average return $\boldsymbol{E_{\lambda\epsilon r}[p(\pi_i, \lambda)]}$. Robust return is defined as $\boldsymbol{max_\pi min_{\lambda\epsilon r} p(\pi, \lambda)}$.

Importance of Consider all Uncertainty for Robust Method

- We show an example where solution is sensitive to the distribution for reward.
- (Left) uses the expected reward, the regular and robust optimal policy are very similar.
- (Right) uses the whole distribution of possible reward, the robust optimal policy are different from the regular one.





- From the solutions determined by our model where specifying distributions for the reward, we can see an obvious change in policy suggested by the method.

Conclusion

- We show the importance to identify and define the distribution of all important factors and reward in a model for Bayesian approach, which otherwise could lead to a very different solution.
- Applying Bayesian Inference to create more possible data could be a technique for Reinforcement Learning on Small Data to counter the issue where experimental is too time consuming or expensive, if we have prior knowledge on the data.

Future Work

- Incorporate understanding rectangular & non-rectangular trade off.
- Evaluate which model and reward uncertainties have the greatest impact on solution quality. In another word, we show a procedure to obtain a confident interval which is not too pessimistic nor optimistic.
- Incorporate understanding rectangular & non-rectangular trade off.