MARMARA
UNIVERSITY

CSE4097 - Engineering Project I
Project Specification Document

USING HIERARCHIES IN REINFORCEMENT LEARNING FRAMEWORK WITH NON-STATIONARY ENVIRONMENTS

04.11.2016

Group Members
Ezdin ASLANCI 150112022
Ahmet Kutalmış COŞKUN 150113018

Project Advisor
Assoc. Professor M. Borahan TÜMER

1. Problem Statement

Reinforcement Learning (RL) is a behavioral learning approach to solve sequential decision making problems, in which the environment is generally assumed to be stationary. This assumption on stationarity is often considered to be optimistic for the reason that it holds for a limited amount of real world problems. In dynamic (non-stationary) environments, using RL is usually nontrivial since the agent is forced to re-learn the policy from scratch – and forgets the old policy – after changes. Making the agent able to detect and respond to the changes so as to learn and adapt to the current behavior of environment improves the learning performance [4]. In this work, we plan to improve the change detection mechanism of Hierarchical Reinforcement Learning with Context Detection (HRL-CD) by tracking the convergence tendency of First Order Markov (FOM) dependencies of action sequences.

2. Problem Description and Motivation

Reinforcement Learning (RL) is a learning method inspired from behaviorist psychology. An RL agent interacts with the *environment* that surrounds it, by taking *actions* which are defined in its current *state*. The environment responds to the action of agent by returning a reward signal and passing to a new state. Throughout this process, the aim of the agent is to reach a goal state by learning the optimal sequence of actions which maximizes the cumulative reward it obtains from the environment.

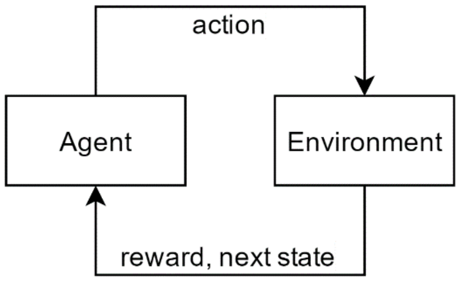

Figure 1: Interaction between agent and environment

A model-free, off-policy TD control algorithm, Q-Learning, estimates the Q values by taking actions and using related rewards for each step. In each step, Q value of a single state-action pair is updated by an update equation. In an improved model-based algorithm known as Prioritized Sweeping (PS), multiple eligible state-action pairs are updated with the same equation [3].

Both of these approaches become infeasible when the state space is large or continuous. In these kinds of problems, using hierarchies has advantages and makes RL feasible. Hierarchical Reinforcement Learning (HRL) approach divides the problem into solvable independent tasks instead of trying to solve the whole problem as a single task. An agent with HRL approach can combine the solutions of individual tasks to reach its goal [2, 3].

Typical real world problems are often non-stationary and have large state spaces. Although having the ability to handle large state spaces, HRL becomes infeasible when the environment is non-stationary since the agent is forced continuously to re-learn the policy from scratch. Silva et. al. proposed a method known as "Reinforcement Learning with Context Detection (RL-CD)" which detects the changes in the behavior of environment and creates new or selects the most appropriate pre-learned model that represents the current condition of the environment [1].

By combining HRL and RL-CD, Yücesoy, Yiğit E. and Tümer, M. Borahan proposed a method known as HRL-CD that is capable of applying RL on large state spaces and handling non-stationarity problem [4].

We plan to detect changes of the environment by tracking the convergence/divergence tendency of first order Markov (FOM) probabilities in sequences of actions. With this method we hope to improve HRL-CD by making it more sensitive to slight changes of the behavior of environment.

3. Aims of the Project

Our aims are listed below:

- Implementation of Hierarchical Reinforcement Learning with Context Detection (HRL-CD)
  In order to improve the change detection mechanism of HRL-CD, we should first have it as a module. We plan to test its performance with different non-stationary environments and different levels of changes.
- Implementation of Change Point Detection Algorithm
  We currently have a MATLAB version of the change detection algorithm that uses Stochastic Learning based Weak Estimation (SLWE) method. Since we will use it as a side module that detects changes in the environment simultaneously with learning, we plan to implement it with a language (such as C++) that is more suitable for real time applications.
- Development of Experiment Software
  We plan to develop a tool by which we can apply and test our approaches to the problems listed above. We also plan to measure the performance of our contribution against other proposed methods.
- Development of Environment Design Module
  We plan to measure the performance of proposed methods in different kinds of non-stationary environments. So we need a tool (or a module in the experiment software) that helps us design learning tasks for testing.

4. Related Work

In the work of da Silva, Basso, Bazzan and Engel [1], authors introduced “RL-CD” which is a method for solving reinforcement learning problems in non-stationary environments. Their method is based on creating, updating and selecting one among several partial models of the environment and it makes the learning system capable of partitioning knowledge into models. The non-stationary environments that are covered by the authors on this work are those whose behavior is given by one among several different stationary dynamics. Each of these dynamics is called a *context.* The authors assumed that each context can be detected by only tracking the transitions and rewards. The contribution they made is that they overcame an important restriction of previously proposed methods, which is requiring a fixed number of models. RL-CD dynamically creates a new model if the *quality* of the best model is still worse than some minimum quality. At any time, only the model with the highest quality is activated. They presented empirical results of RL-CD for three different validation scenarios. One of them is “Ball Catching” which we would like to create a similar experiment for testing our method.

In the work of R. S. Sutton, D. Precup, S. Singh [2], authors worked on an important challenge of Artificial Intelligence, which is representing knowledge at multiple levels of temporal abstraction. Their research is focused on addressing this challenge within the mathematical framework of Reinforcement Learning and Markov Decision Processes (MDPs). They extended the usual notion of action in Reinforcement Learning framework to include temporally extended actions. Options can be considered as policies for taking actions to reach a sub-goal over a period of time. A set of options defined on MDPs creates a semi-Markov Decision Process (SMDP). Extending their work to deal with non-stationary environments is the main motivation for us on this project.

In the work of Y. E. Yücesoy and M. B. Tümer [4], the authors proposed an autonomous agent which learns a dynamic environment by taking the advantage of Hierarchical Reinforcement Learning. They presented their empirical results the grid world problem with different dynamics. We noticed that the weak point of this method is that it fails to detect slight changes in the environment. We hope to contribute their work by proposing a more sensitive method using our Change Point Detection algorithm which uses Stochastic Learning based Weak Estimation (SLWE). Combining these two studies is the main goal of our project.

5. Scope of the Project

Our research is focused on deterministic, discrete and dynamic environments with different dynamics that occur infrequently and independently from the agent. We plan to adopt grid world problem and run our simulations of Reinforcement Learning on such environments. In a grid world problem, an RL agent learns how to reach to a goal state from its start state by taking right, left, up and down actions on the grid. Dynamism will be provided by the changing the grid randomly at the training process of the agent. We expect the agent to realize and develop some strategies such as relearning or using its previous experiences, when some changes appear on the grid. The changes on the environment (grid) will be detected by observing the sequence of actions which is executed by the agent.

Our another study which is on the improvement process, is “Detection of Regime Switching Points in Non-Stationary Sequences using Stochastic Learning based Weak Estimation Method (SCD)”. Our main objective is to combine these two studies which means; changes on environment will be detected by our SCD method. We expect to detect very slight changes by using SCD method.

Simulating learning process on continuous environments currently is not in the scope of our project. We plan to conduct experiments only on discrete environments that can be represented with a grid. For a future work, our method may be improved so that it works on both discrete and continuous problems.

6. Success Factors and Benefits

There are few metrics being used for measuring the learning performance of Reinforcement Learning algorithms. Specifically, on the grid world problem, the agent’s objective is to reach the goal state with the minimum number of steps. This makes the number of steps a good concept for tracking convergence and divergence of learning process. We expect to get instant jumps on the step count curve after the current behavior of environment changes.

Moreover, we are planning to use the sequence of actions as an input to the change point detection algorithm we developed. The change points we get from the algorithm will be compared with the true change points we expect to get. We know these true change points because while conducting experiments, we will determine the points where the current behavior of the environment changes. We are planning to use Standard Accuracy Criteria (SAC) for measuring the success of behavior change detection.

	Change Present	Change Absent	Total
Change Detected	a	b	$a + b$
Change not Detected	c	d	$c + d$
Total	$a + c$	$b + d$	$a + b + c + d$

Table 1: Symbol Definitions for Change Point Detection Results

Measure Name	Definition	Ideal Value
Sensitivity	$\frac{a}{a+c}$	1
Specificity	$\frac{d}{b+d}$	1
False Negative Rate	$\frac{c}{a+c}$	0
False Positive Rate	$\frac{b}{b+d}$	0
Predictive Value Positive (PVP)	$\frac{a}{a+b}$	1
Predictive Value Negative (PVN)	$\frac{d}{c+d}$	1
False Alarm Rate	$\frac{b}{a+b}$	0
False Reassurance	$\frac{c}{c+d}$	0

Table 2: Definitions of Standard Accuracy Criteria

One of our main goals for our method is to make the agent detect the current behavior of the algorithm besides detecting the point where it changed. This will make the agent avoid relearning a regime from scratch. We also plan to employ SAC for measuring the performance of detecting a regime correctly.

Getting expected values for these metrics we explained will indicate that we satisfied the requirements of this project. Also, if we get nice results, it will demonstrate the success of our work about change point detection. Hopefully, researchers working on Reinforcement Learning on non-stationary environments will benefit the results of our project.

When project is completed with success, real world applications which are continuous problems might be solved with this approach by using discretization techniques because it is hard to apply RL and detect changes in continuous problems. We expect that even very small changes can be detected with this approach. This project will provide a solution for dynamic problems which can be represented with a finite space of state-action pairs.

7. Methodology and Technical Approach

Q-Learning (QL) is the primary algorithm that explains the methodology of RL. QL is a model-free, off-policy Temporal Difference Control algorithm and it estimates the Q-values by taking actions and using related rewards for each step. In each step Q values are updated by the following equations.

```
$Initialize\ Q(s,a), \forall s \in S, a \in A(s), arbitrarily, and\ Q(terminal - state, \cdot) = 0$
$Repeat\ (for\ each\ episode)$
    $Initialize\ S$
    $Repeat\ (for\ each\ step\ of\ episode)$
        $Choose\ A\ from\ S\ using\ policy\ derived\ from\ Q(e.g., \epsilon - greedy)$
        $Take\ action\ A, observe\ R, S'$
        $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$
        $S \leftarrow S'$
    $Until\ S\ is\ terminal$
```

Algorithm 1: Q-Learning [3]

Prioritized Sweeping (PS) is a *memory-based* RL algorithm which creates a model of the environment during learning phase. With this information, PS effectively updates the value table of state-action pairs where only those pairs' values get adjusted which move the environment to a state-action pair with a non-zero value [3].

```
$Initialize\ Q(s,a), Model, for\ all\ s, a, and\ PQueue\ to\ empty$
$Do\ forever:$
    $(a)\ S \leftarrow current\ (nonterminal) state$
    $(b)\ A \leftarrow policy(S,Q)$
    $(c)\ Execute\ action\ A; observe\ resultant\ reward, R, and\ state, S'$
    $(d)\ Model(S,A) \leftarrow R, S'$
    $(e)\ P \leftarrow \left|R + \gamma \max_a Q(S',a) - Q(S,A)\right|$
    $(f)\ if\ P > \theta, then\ insert\ S, A\ into\ PQueue\ with\ priority\ P$
    $(g)\ Repeat\ n\ times, while\ PQueue\ is\ not\ empty:$
        $S, A \leftarrow first(PQueue)$
        $R, S' \leftarrow Model(S,A)$
        $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$
        $Repeat, for\ all\ \bar{S}, \bar{A}\ predicted\ to\ lead\ to\ S:$
            $\bar{R} \leftarrow predicted\ reward\ for\ \bar{S}, \bar{A}, S$
            $P \leftarrow \left|\bar{R} + \gamma \max_a Q(S,a) - Q(\bar{S},\bar{A})\right|$
            $if\ P > \theta\ then\ insert\ \bar{S}, \bar{A}\ into\ PQueue\ with\ priority\ P$
```

Algorithm 2: Prioritized Sweeping [3]

We plan to adopt the idea of using partial models to represent the current behavior of the environment, which is proposed in the work of Silva, Basso, Bazzan and Engel [1]. The method they proposed, RL-CD, performs well on a specific class of non-stationary environments, which is also the class that our research is focused on. RL-CD automatically creates a new partial model if the *quality* of the best model is still worse than some minimum quality. Quality of a model is inversely proportional its prediction error.

They assumed that the following properties hold for the environment:

1- environmental changes are confined to a small number of contexts, which are stationary environments with distinct dynamics
2- the current context cannot be directly observed, but can be estimated according to the types of transitions and rewards observed
3- environmental context changes are independent of the agent's actions
4- context changes are relatively infrequent

The partial models that RL-CD learns contain functions which estimate transition probabilities and rewards. At each time step, the current partial model is updated by adjusting its transition model T_m and its reward model R_m.

RL-CD calculates a *quality signal* for each partial model in order to detect changes in the behavior of the environment. Also, a *confidence value*, which reflects the number of times the agent chose and performed an action in a state. Quality of the model is proportional to the confidence value computed. Another important concept in RL-CD is the *instantaneous quality*. It is a linear combination of two quality predictions, which are the quality of reward prediction and the quality of transition prediction. These values are linearly interpolated by a value Ω, which reflects the relative importance of the two values combined.

In this work, authors also presented their empirical results on three different validation scenarios. They used a model-based algorithm, Prioritized Sweeping (PS) to build partial models for each context. In one of these experiments, "Ball Catching", the agent is a cat whose goal is to catch a moving ball on a toroidal discrete grid. The moving ball can take 4 different behaviors which are moving left, up, right or down. The environment is non-stationary since the direction of the moving ball changes over time. RL-CD learns the partial models of each of these behaviors and when the direction of the ball is changed to a known one, the learning curve (number of steps over time) does not rise with a sudden jump which is a nice indication of detecting the behavior correctly [1]. We plan to conduct an experiment with the same idea of changing the position of goal state over time.

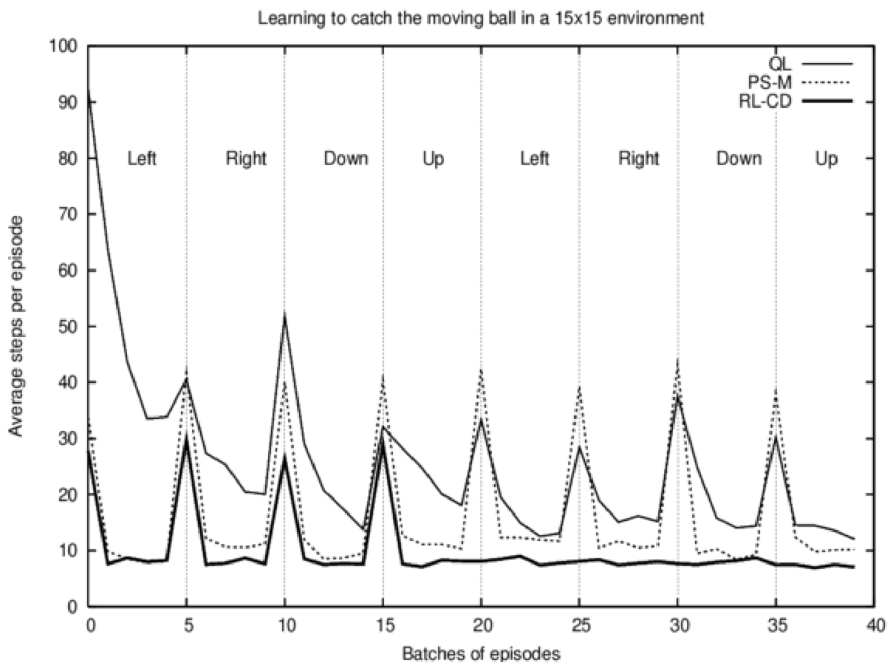

Figure 2: A graph from [1] showing the performance of QL, PS-M and RL-CD

Another concept we plan to adopt in our research is using hierarchies in RL problems. Applying RL is infeasible on environments that are continuous or have large state spaces since the agent needs to reach goal state multiple times (in each episode) to learn the optimal policy. This challenge brought the idea of dividing the learning problem into different solvable smaller tasks. R. S. Sutton, D. Precup, S. Singh proposed a method with this idea. Authors extended the concept of action in RL framework to include actions that are temporally extended, also known as, *options*. An option can be considered as a solution to a smaller task, a sub-policy to reach a sub-goal. Once the problem is divided into smaller problems whose solutions are options, on a higher hierarchy, the optimal solution can be expressed as a combination of options [2, 4].

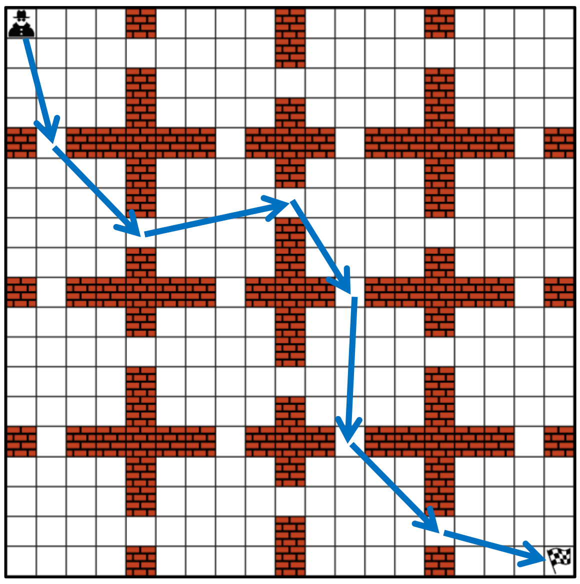

Figure 3: An example grid of 19x19 states, blue arrows can be considered as options that lead the agent from one room to another

An option has three components, a policy $\pi: S \times A \rightarrow [0, 1]$, a termination condition $\beta: S^+ \rightarrow [0, 1]$ and an initiation set $I \subseteq S$. An option is available in a state s_t if and only if $s_t \in I$. When the agent selects an option at state s_t, it starts to perform actions according to the policy of that option until the option terminates according to β. When the option terminates, agent may select another option, or continue to select primitive actions according to its main policy [2].

The environment is not fully observable by the agent in RL problems. This lack of information brings another challenge, detecting sub-goals simultaneously with learning. There are few ways to detect sub-goals by using the partial model that the agent generates while interacting with the environment. Particularly, in the grid world problem, options can be thought as extended actions that lead the agent from one room to another. Also, sub-goals can be thought as the doors between consecutive rooms. These doors are like bottlenecks and can be separated from other states by the amount of shortest paths passing through them. A graph property, *betweenness centrality*, can be used to detect sub-goals by counting the shortest paths passing through [5].

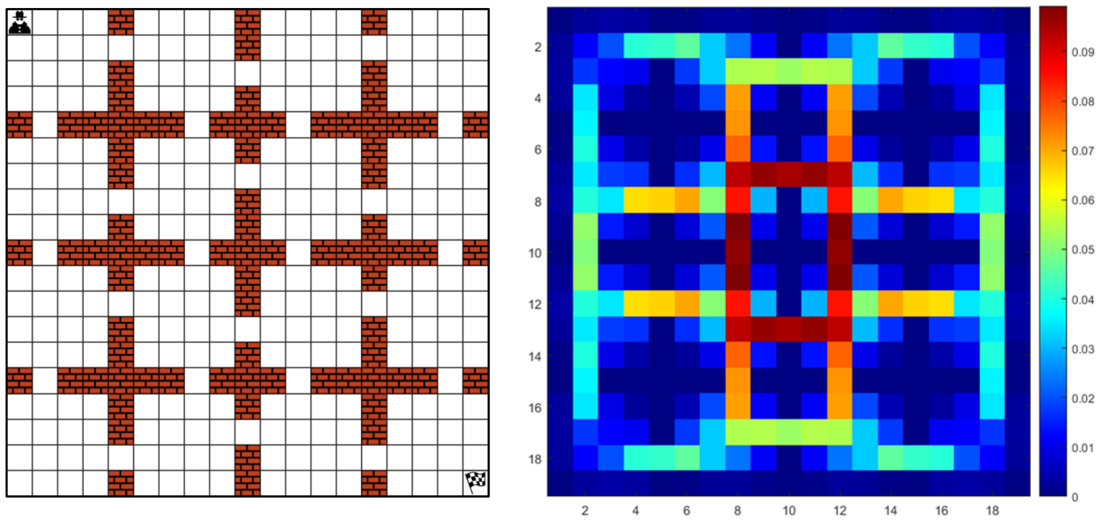

Figure 4: An example grid of 19x19 states (left) and a colormap showing betweennes centrality values of each state (right)

The combination of these two valuable methods, is proposed by Y. E. Yücesoy and M. B. Tümer in “Hierarchical Reinforcement Learning with Context Detection (HRL-CD)”. The authors presented the advantages of using hierarchies in a non-stationary environment. The agent they designed can learn a dynamic environment based on RL-CD technique also taking the advantage of HRL for constructing the algorithm which is used to speed up the learning process. HRL-CD shows significant improvement especially for environments with a large state-space. Our main goal is to contribute their work by improving the detection mechanism of HRL-CD by using SCD [4].

Apart from RL-CD and HRL-CD, our change detection approach is based on convergence tendency of First Order Markov (FOM) probabilities. We want to detect change points by using our another project; “Detection of Regime Switching Points in Non-Stationary Sequences using Stochastic Learning based Weak Estimation Method (SCD)”, which is currently in progress.

SCD algorithm detects change points in a non-stationary sequence. A non-stationary sequence is a linear combination of stationary segments, regimes. Each regime is a combination of repeated cycles. In SCD, in order to detect change points, we watch the estimated FOM probabilities of the sequence and we expect that FOM probabilities of each regime should reveal the character of significant structures or the cycles of the regimes, which means there is a change point if we observe a sudden increase/decrease in FOM probabilities.

The main reason of combining HRL with SCD is similarities between the learning principle of HRL and change detection approach of SCD. In the learning process, an RL agent learns the environment by performing actions and getting rewards from the environment. The main objective of the agent is finding an optimal path (i.e., a path with the highest cumulative reward) from the start state to the goal state. According to this objective, after a while, the agent will start to perform similar sequences of actions in each episode. If we observe all actions of the agent as a sequence depending on time during the learning process, there will be many similar subsequences of actions. This subsequence of actions will be repeated while the environment changes. Depending on the inference; each stationary environment has a significant sequence of actions. If we think the same for the change detection mechanism of SCD; the entire sequence of actions selected during the learning in a dynamic environment is our non-stationary sequence. Action sequences of each stationary part in the environment will behave like a stationary regime because each stationary environment has a significant action sequence. Based on this implication, we expect to detect changes on the environment by using SCD algorithm.

8. Professional Considerations

We used GANTT charts to present our management plan which is covered in section 9 of this document.

During the development phase of the software we are planning to have for conducting experiments on environments with different dynamics, we will use Source Code Control via Git.

We decided to use Python programming language in order to develop the software since it has a large library database on the area we are working on. It also has nice data visualization tools that will be beneficial for us to debug and track the progress of our project.

Our research is focused on dynamic systems which are everywhere in modern society. By definition, these systems can be characterized by the non-stationary data sequences they emit. Working with these kind of systems has a potential economic impact since like most of the systems we use currently, production systems can also be considered as dynamic systems and analyzing these systems to increase efficiency is a major challenge in these days.

Another aspect of making a production system better is about health and safety. It is possible to save the life of the operator that uses the system, the production line itself and product that is being processed.

Most of the factors that harm the environment are also about dynamic systems. These systems can be designed in a way that the damage they cause is minimized.

There are no legal permissions to be taken to realize our project.

9. Management Plan

Phase 1: Literature Survey about HRL, HRL-CD, SCD and Multi-Threaded Programming.

Phase 2: Implementation of basic RL algorithms such as QL, PS.

Phase 3: Improving HRL-CD and adapting a real world application.

Phase 4: Improving SCD and making usable for real data input.

Phase 5: Re-implementation of SCD with multi-thread support.

Phase 6: Combining HRL-CD with SCD.

Phase 7: Testing the final version HRL-SCD with different inputs and parameters.

	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun
Phase 1	🟩	🟩	🟩	🟩	🟩	🟩	🟩			
Phase 2	🟦	🟦								
Phase 3		🟨	🟨	🟨						
Phase 4		⬛	⬛	⬛						
Phase 5			🟧	🟧	🟧					
Phase 6					🟩	🟩	🟩			
Phase 7								🟦	🟦	🟦

Figure 5: Gantt chart for Management Plan

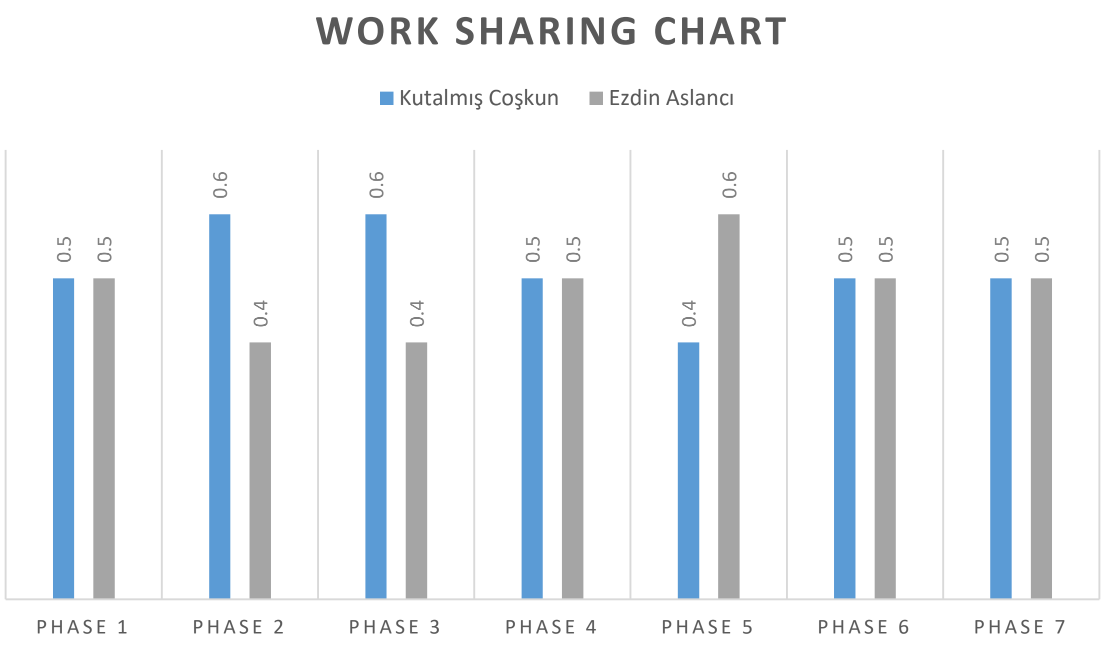

Figure 6: Work Sharing Chart

One possible risk that we may encounter throughout the project is that it may only be possible to detect slight changes with a specific latency by tracking FOM dependencies. We expect to have this latency between change detection module and the learning process. If this latency gets beyond a practical limit (may change depending on the learing task), we plan to use another change detection approach that is based on tracking backup diagrams, more generally, graph matching.

Another possible risk we foresee is that we may not handle huge action sequence files with our current hardware. We may encounter this type of problems if we conduct experiments on environments with large state spaces. We plan to use option sequences to detect changes on the environment to deal with this situation.

10. References

[1] Silva, B.D. da, Basso, E.W., Bazzan, A.L.C. & Engel, P.M., Dealing with Non-Stationary Environments using Context Detection. *23rd International Conference on Machine Learning (ICML)*, 2006.

[2] Sutton, R.S., Precup, D. & Singh, S., Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, pages 181-211, 1999.

[3] Sutton, R.S. & Barto, A.G., Reinforcement learning. *Learning* **3**, 322, 2012.

[4] Yücesoy, Y.E. & Tümer, M.B., Hierarchical Reinforcement Learning with Context Detection (HRL-CD). *International Journal of Machine Learning and Computing*, 7763, 2015.

[5] Freeman, L., A set of measures of centrality based on betweenness. *Sociometry*, **40**: 35–41, 1977.

[6] Oommen, B.J. & Rueda, L. Stochastic learning-based weak estimation of multinomial random variables and its applications to pattern recognition in non-stationary environments. *Pattern Recognition* **39**, 328-341, 2006.