AUTONOMOUS SKILL ACQUISITION IN REINFORCEMENT LEARNING USING LOCALITY BASED GRAPH THEORETIC FEATURES

Semiha Hazel Çavuş
semihazel@gmail.com

Zeynep Kumralbaş
zeynepkumralbas@gmail.com

Advisor: Assoc. Prof. Borahan Tümer

Abstract

In this project, we aim to reduce the time complexity of subgoal detection in Hierarchical Reinforcement Learning (HRL).

Problem: There are different approaches for detecting subgoals. Betweenness Centrality a graph based approach, is one of the well performed techniques. Since the time complexity for subgoal detection is $O(n^3)$ for each episode, it brings a computational burden.

Solution: Dynamic Community Detection algorithm which runs in $O\left(|\Delta E| \cdot \frac{|E|}{|v|} + |E|^*\right)$ can be used for subregion detection. Hence, subgoals are detected using these subregions.

Problem of Community Detection:

- It can further partition a subregion (over-segmentation),
- It can combine two or more subregions as one subregion (under-segmentation).

Solution:

- Since subgoals are detected to construct options, option merging can solve this over-segmentation.
- Under-segmentation is mostly solvable by adjusting the parameters.

Reinforcement Learning (RL)

RL is a machine learning technique where an agent takes an action in an environment, moves to the next state and receives rewards or punishments regarding this new state.

So, it can learn a satisfactory (hopefully optimal or possibly a near optimal) policy that leads the agent to the goal state in the environment.

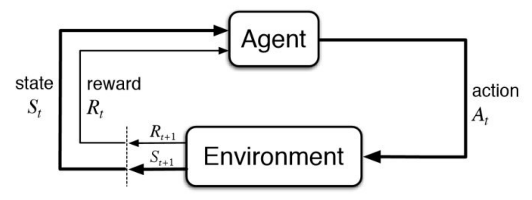


Fig. 1: RL diagram

Hierarchical Reinforcement Learning (HRL)

As the environment grows too large, converging to a satisfactory policy for regular RL algorithms such as flat Q–learning becomes quickly infeasible.

In HRL:

- Environment is split into subregions.
- A subpolicy (sequence of primitive actions) is learned for each subregion.
- The sequence of primitive actions is called skill/option.

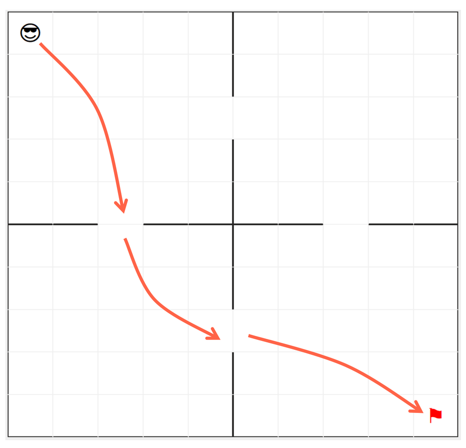


Fig. 2: Skills in HRL

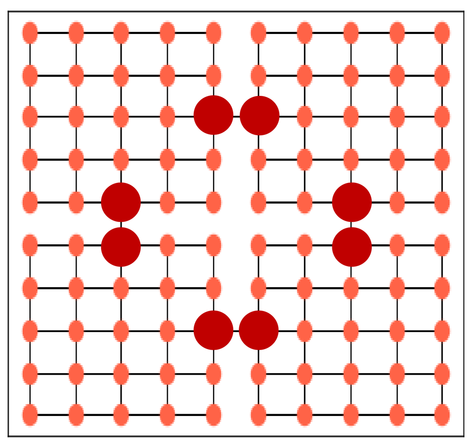


Fig. 3: Subgoals on graph representation of Fig. 2

An option consists of three components:

- A policy π: S×A → [0,1]
- An initiation set I ⊆ S
- A termination condition β: S+ → [0,1]

Community Detection

Community Detection algorithms try to maximize the modularity which is used as an objective function.

Modularity: A metric that represents how strongly nodes are connected to each other in each community.

$$Q = \frac{1}{2m}\sum_{i,j}\left[A_{ij} - \gamma\frac{k_i k_j}{2m}\right]\delta(c_i, c_j)$$

A_{ij}: the number of the edges between nodes i and j
k_i: the degree of node i, k_j: the degree of node j
$\delta(c_i, c_j)$: 1 if nodes i and j belong to same community, 0 otherwise
m: total number of edges in the network
γ: resolution parameter

DynaMo: A dynamic community detection algorithm which updates communities locally.

$\underbrace{\text{Partial graphs change in time} \rightarrow \text{A dynamic approach}}_{\text{DynaMo}}$

Communities and Subgoals

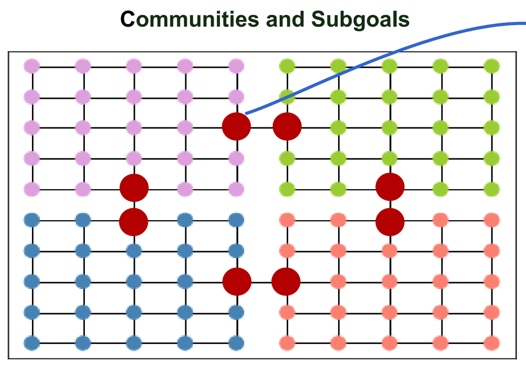


Fig. 4: Subgoals on accurately detected communities

Initiation Sets

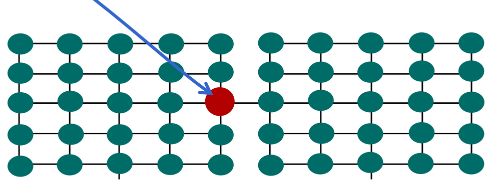


Fig. 5: Initiation set of the shown subgoal

Communities may not be found correctly, since *resolution parameter* which changes the size of the communities is not set adaptively according to the structure of the graph.

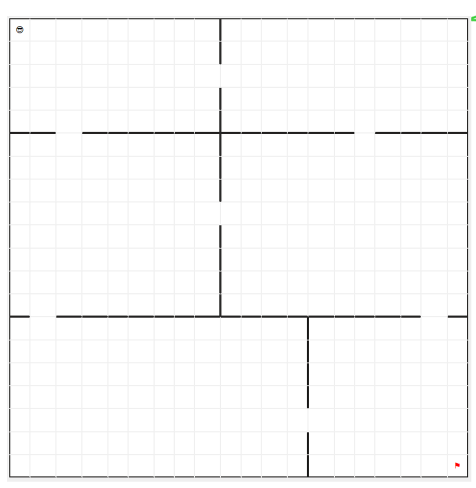


Fig. 6: 20x20 environment

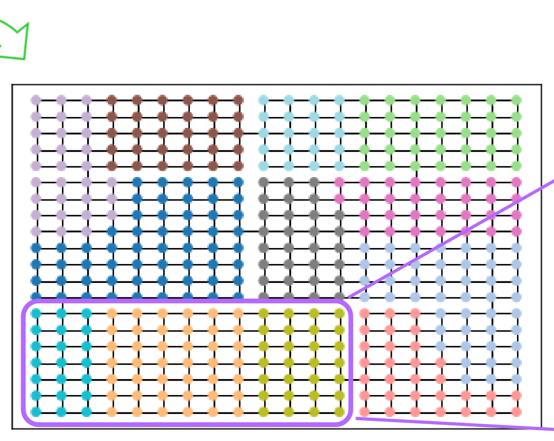


Fig. 7: Poorly detected communities (over-segmentation)

Community Borders

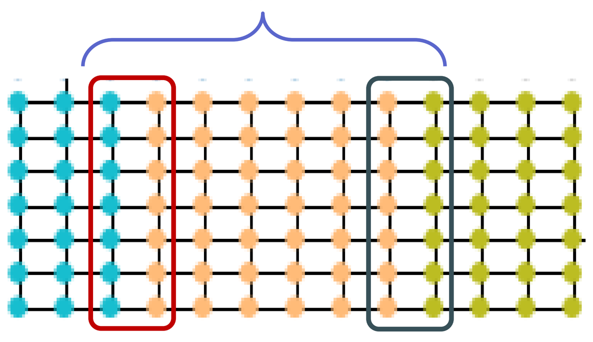
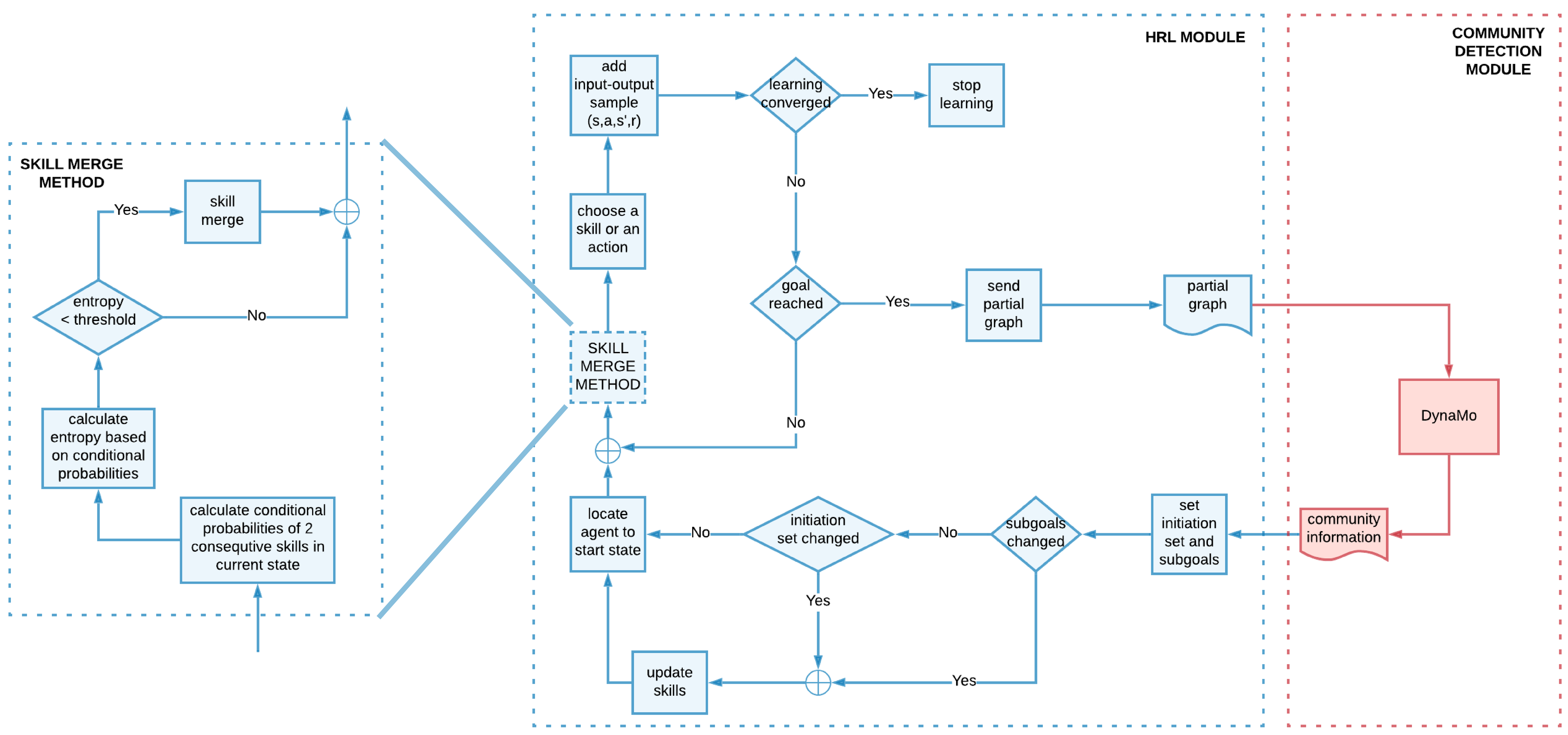


Fig. 8: Examples of community borders

Flow Chart



Time Complexity

Betweenness Centrality: $g(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$

σ_{st}: total number of shortest paths from node s to node t
$\sigma_{st}(v)$: number of those paths that pass through v

Average case: $O(|v|^3)$

DynaMo:

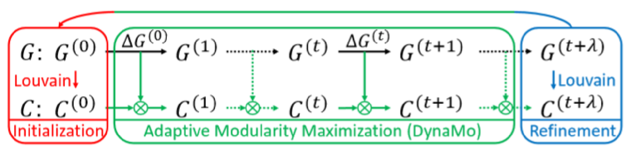


Fig. 9: The overview of DynaMo algorithm. Taken from [3]

C: a set of communities associated with G
G: a sequence of graph snapshots

Best Case:
$O(|\Delta E| + |E|^*)$

Worst Case:
$O\left(|\Delta E| \cdot \frac{|E|}{|v|} + |E|^*\right)$

$|E|$: total number of edges
$|v|$: total number of nodes
$|\Delta E|$: number of added/removed edges
$|E|^*$: the number of edges evaluated in the second phase of the algorithm
$|E|^* << |E|$

Experimental Results

We have done approximately 30 experiments on different types of environments and get similar results. The figures below are the representative ones.

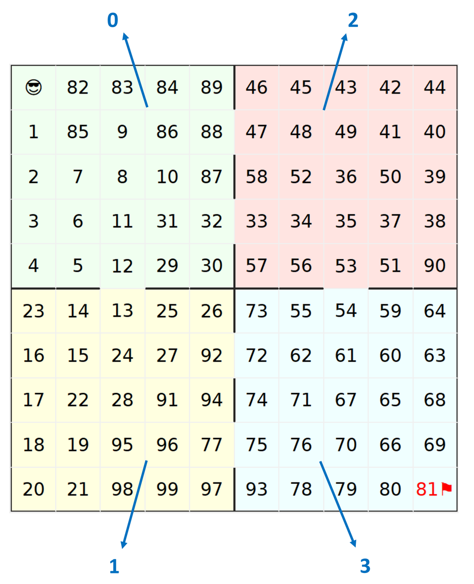


Fig. 10: Communities of 10x10 environment

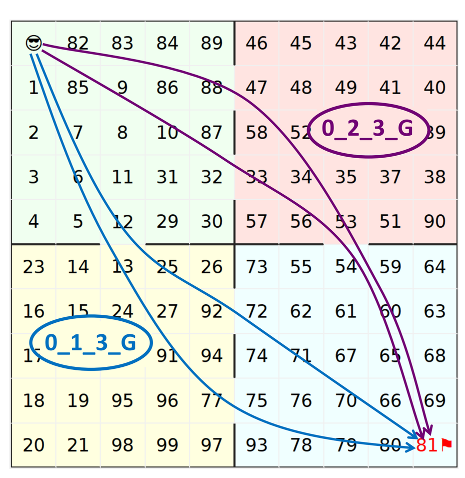


Fig. 11: Merged options

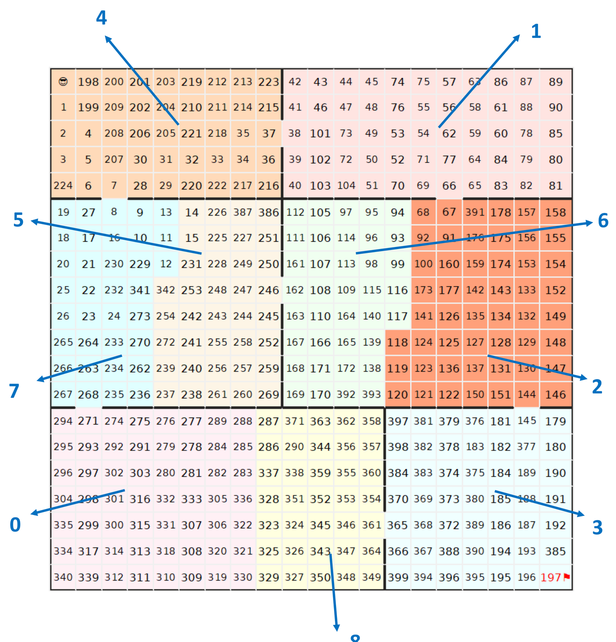


Fig. 12: Communities of 20x20 environment

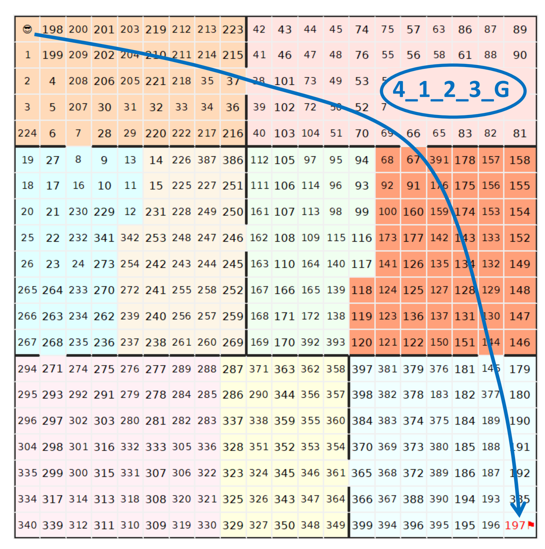


Fig. 13: Merged options

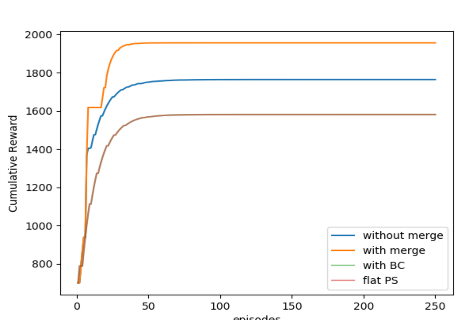


Fig. 14: Cumulative reward graph

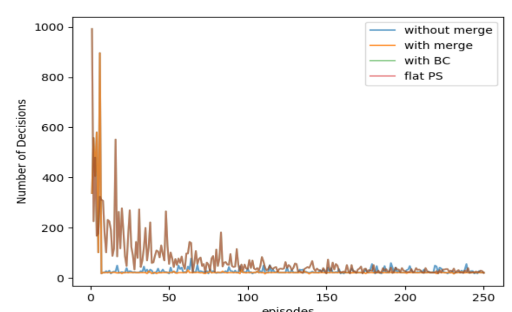


Fig. 15: Number of decisions graph

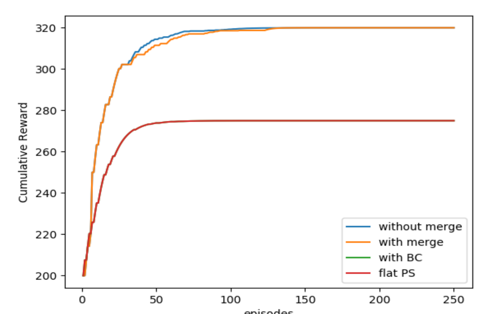


Fig. 16: Cumulative reward graph

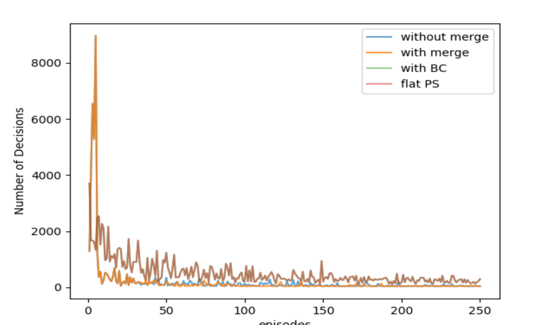


Fig. 17: Number of decisions graph

Conclusion

- Autonomous skill acquisition is achieved.
- Complexity of subgoal detection is improved.
- Skill merge approach converges faster.
- The number of decisions agent has to take is decreased.

Future Work

- Setting resolution parameter adaptively.
- Finding more robust function to select probabilities of skills to be merged.

Technologies Used

Acknowledgement

We would like to thank **Kutalmış Coşkun** for his precious contributions to Project.

Selected References

[1] Sutton, Richard S., Doina Precup, and Satinder Singh. "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning." Artificial intelligence 112.1-2 (1999): 181-211.

[2] Şimşek, Özgür, and Andrew G. Barto. "Skill characterization based on betweenness." Advances in neural information processing systems. 2009.

[3] ZHUANG, Di; CHANG, J. Morris; LI, Mingchen. DynaMo: Dynamic Modularity-based Community Detection in Evolving Social Networks. arXiv preprint arXiv:1709.08350, 2017.