Unlocking New Strategies: Intrinsic Exploration for Evolving Macro and Micro Actions

Sourav Panda
College of IST
Pennsylvania State University
sbp5911@psu.edu

Aviral Srivastava
College of IST
Pennsylvania State University
aks7873@psu.edu

Jonathan Dodge
College of IST
Pennsylvania State University
dodge@psu.edu

Abstract

We propose combining intrinsic exploration with extrinsic motivations to inspire RL agents to develop strategies at both macro and micro levels, enhancing adaptability.

Introduction and Related Work DeepMind recognizes StarCraft II (SC2) as having an immense action space, approximately 10^{26} [10, 9, 8]. One challenge of SC2 is its hierarchical action space constrained by game rules, demanding complex planning to compensate for long-horizon delayed rewards; players typically learn these rules through in-game tutorials rather than discovery through trial and error. Moreover, SC2 includes numerous trivial choices that minimally impact game outcomes, an inefficient use of learning resources. RL algorithms lacking higher-level abstractions face substantial challenges due to inefficient exploration [12], and the sparse, delayed feedback (win/loss at the game's end) [4, 6] extends training time and unnecessarily complicates learning.

One way to address SC2's challenges is via shaping rewards [1–3, 11], which risks suboptimal solutions. Alternatively, incorporating human knowledge through a set of macro-actions composed of micro-actions can streamline the learning process [7]. This encodes the game's intrinsic rules directly into the agent's knowledge, reducing reliance on trial and error.

Current RL problems operate within a predefined state-action space, so then the problem becomes to find good actions to take in each state. However, humans naturally seek to discover new actions. For example, having learned a recipe to make a Salad and Fried Rice, a cooking agent might recombine the micro actions to make Stir-Fried Veggies. Alternatively, the agent might add a new micro action to make Roast Veggies (see Appendix). For the agent to learn intrinsically, it should explore the environment and the state-action space in a way that it discovers new groupings of micro-actions, effectively forming new macro-actions.

Proposed Framework and Challenges Intrinsic motivation in artificial agents is inherently limited; they do not spontaneously develop new goals. For example, a fully autonomous car does not independently decide "I want to drive" [5]; it follows programmed instructions to perform its tasks. Therefore, we propose modifying the agent's objective by introducing *extrinsic* motivations that mimic intrinsic motivation. With objectives like "learn a large, diverse set of actions," we provide the agent with a multi-objective goal: not only to win the game but also to master an ever-expanding set of macro-actions. When the agent successfully assembles a set of micro-actions into a new macro-action, it should incorporate this new macro-action into its knowledge base and continue learning. The Appendix contains examples illustrating how the proposed approach influences the progression of the knowledge base.

Accepted at the Intrinsically Motivated Open-ended Learning workshop at NeurIPS 2024.

Exploration strategies, such as epsilon-greedy, help an agent explore/exploit existing macro-actions; however, the agent needs to learn (1) new micro-actions and (2) new macro-actions by combining micro-actions. We also need to incorporate the multi-objective goal into the reward signal. Depending on the specific agent architecture, we may face challenges incorporating new actions into its policy, value function, and model.

References

[1] Tim Brys, Anna Harutyunyan, Peter Vrancx, Matthew E Taylor, Daniel Kudenko, and Ann Nowé. Multi-objectivization of reinforcement learning problems by reward shaping. In *2014 international joint conference on neural networks (IJCNN)*, pages 2315–2322. IEEE, 2014.

[2] Marek Grzes. Reward shaping in episodic reinforcement learning. 2017.

[3] Shengyi Huang and Santiago Ontañón. Action guidance: Getting the best of sparse rewards and shaped rewards for real-time strategy games. *arXiv preprint arXiv:2010.03956*, 2020.

[4] Matthias Hutsebaut-Buysse, Kevin Mets, and Steven Latré. Hierarchical reinforcement learning: A survey and open research challenges. *Machine Learning and Knowledge Extraction*, 4(1): 172–221, 2022.

[5] Thomas G. Dietterich Kagan Tumer. https://engineering.oregonstate.edu/engineering-out-loud/season-9/beautiful-music-robotics-and-ai.

[6] Thanh Thi Nguyen, Ngoc Duy Nguyen, and Saeid Nahavandi. Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications. *IEEE transactions on cybernetics*, 50(9):3826–3839, 2020.

[7] Peng Sun, Xinghai Sun, Lei Han, Jiechao Xiong, Qing Wang, Bo Li, Yang Zheng, Ji Liu, Yongsheng Liu, Han Liu, et al. Tstarbots: Defeating the cheating level builtin ai in starcraft ii in the full game. *arXiv preprint arXiv:1809.07193*, 2018.

[8] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, John Quan, Stephen Gaffney, Stig Petersen, Karen Simonyan, Tom Schaul, Hado van Hasselt, David Silver, Timothy Lillicrap, Kevin Calderone, Paul Keet, Anthony Brunasso, David Lawrence, Anders Ekermo, Jacob Repp, and Rodney Tsing. Starcraft ii: A new challenge for reinforcement learning, 2017. URL `https://arxiv.org/abs/1708.04782`.

[9] Oriol Vinyals, Igor Babuschkin, Junyoung Chung, Michael Mathieu, Max Jaderberg, Wojtek Czarnecki, Andrew Dudzik, Aja Huang, Petko Georgiev, Richard Powell, Timo Ewalds, Dan Horgan, Manuel Kroiss, Ivo Danihelka, John Agapiou, Junhyuk Oh, Valentin Dalibard, David Choi, Laurent Sifre, Yury Sulsky, Sasha Vezhnevets, James Molloy, Trevor Cai, David Budden, Tom Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Toby Pohlen, Dani Yogatama, Julia Cohen, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Chris Apps, Koray Kavukcuoglu, Demis Hassabis, and David Silver. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. `https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/`, 2019.

[10] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

[11] Nicholas Waytowich, Sean L Barton, Vernon Lawhern, and Garrett Warnell. A narration-based reward shaping approach using grounded natural language commands. *arXiv preprint arXiv:1911.00497*, 2019.

[12] Yang Yu. Towards sample efficient reinforcement learning. In *IJCAI*, pages 5739–5743, 2018.

Appendix

Example: Learning New Macro-Actions and Micro-Actions in a Cooking Scenario

In this example, we illustrate the concept of the agent starting with an initial knowledge base of macro-actions composed of pre-defined micro-actions. Over time, the agent learns new macro-actions by combining existing micro-actions and, with extended training, discovers new micro-actions to build more complex macro-actions.

1. Initial Knowledge Base

The agent starts with the following set of macro-actions and their associated micro-actions:

Salad	**Fried Rice**	**Stir-Fried Veggies**
1. Chop Veggies	1. Make Rice	1. Chop Veggies
2. Add Dressing	2. Pan Fry	2. Add Dressing
3. Mix		3. Stir-fry

Table 1: Initial knowledge base of macro-actions and their corresponding micro-actions

2. Updated Knowledge Base (Learning a New Macro-Action)

The agent then learns a new macro-action, *Veggie Fried Rice*, by combining existing micro-actions from the previously known actions. This shows how the agent can explore the space of available actions to form new, more useful combinations.

Salad	**Fried Rice**	**Stir-Fried Veggies**	**Veggie Fried Rice**
1. Chop Veggies	1. Make Rice	1. Chop Veggies	1. Chop Veggies
2. Add Dressing	2. Pan Fry	2. Add Dressing	2. Make Rice
3. Mix		3. Stir-fry	3. Stir-fry

Table 2: Knowledge base after learning a new macro-action

3. Expanded Knowledge Base (Discovering a New Micro-Action)

After prolonged training, the agent discovers a new micro-action, *Roast*. This newly discovered micro-action allows the agent to form more complex macro-actions, such as *Roast Veggies*.

Salad	**Fried Rice**	**Stir-Fried Veggies**	**Veggie Fried Rice**	**Roast Veggies**
1. Chop Veggies	1. Make Rice	1. Chop Veggies	1. Chop Veggies	1. Chop Veggies
2. Add Dressing	2. Pan Fry	2. Add Dressing	2. Make Rice	2. Add Dressing
3. Mix		3. Stir-fry	3. Stir-fry	3. *Roast*

Table 3: Knowledge base after discovering new micro-actions

This example demonstrates how the agent can start with a predefined set of micro-actions and macro-actions, then incrementally learn new macro-actions by reusing existing micro-actions, and eventually discover entirely new micro-actions that further expand the agent's knowledge and decision-making capabilities.

Example: Learning New Macro-Actions and Micro-Actions in StarCraft II

StarCraft II involves three general action categories: Build, Harvest, and Attack. These categories are comprised of different combinations of micro-actions. For this example, we will focus on building, for simplicity. Initially, the agent starts with a predefined set of macro-actions formed by these micro-actions:

Hatchery	**Extractor**	**Worker**
1. Select Worker	1. Select Worker	1. Select Hatchery
2. Move Camera	2. Move Camera	2. Build Worker
3. Select Location	3. Select Location	
4. Build Hatchery	4. Build Extractor	

Table 4: Initial knowledge base of macro-actions and their corresponding micro-actions

Over time, the agent learns to combine existing micro-actions into a new macro-action. For example, after exploration, the agent discovers a new macro-action formed by a sequence of existing micro-actions from all three categories. In this case, the new macro-action is composed of all three whole macro-actions, which we do not expand to micro-actions for brevity. The updated knowledge base is:

Hatchery	**Extractor**	**Worker**	**Expansion**
1. Select Worker	1. Select Worker	1. Select Hatchery	1. **Hatchery**
2. Move Camera	2. Move Camera	2. Build Worker	2. **Extractor**
3. Select Location	3. Select Location		3. **Workers**
4. Build Hatchery	4. Build Extractor		

Table 5: Knowledge base after learning a new macro-action

As the agent continues to train and learn, it may discover entirely new micro-actions, which can then be used to form even more macro-actions. For instance, the agent might learn a new micro-action, *Build a Tower*, which it incorporates into a new macro-action **Build Tower** (omitted from the table, but it has the same "select worker, move camera, select location, build" structure as the others). Subsequently, the agent creates a new macro-action **Build a Defended Expansion**, which we again represent with only macro-actions in the table, but the only new ingredient is the micro-action *Build a Tower*. The extended knowledge base is:

Hatchery	**Extractor**	**Worker**	**Expansion**	**Def. Expansion**
1. Select Worker	1. Select Worker	1. Select Hatchery	1. **Hatchery**	1. **Hatchery**
2. Move Camera	2. Move Camera	2. Build Worker	2. **Extractor**	2. **Extractor**
3. Select Location	3. Select Location		3. **Workers**	3. **Workers**
4. Build Hatchery	4. Build Extractor			4. ***Build Tower***

Table 6: Knowledge base after discovering new micro-actions

This example demonstrates how the agent's knowledge base evolves as it explores the environment, learns new combinations of actions, and even discovers entirely new micro-actions.