

Fairness in Cooperative Multiagent Multiobjective Reinforcement Learning using the Expected Scalarized Reward Criterion – Appendix

Anonymous Author(s)
Submission Id: 1025

ACM Reference Format:

Anonymous Author(s). 2025. Fairness in Cooperative Multiagent Multiobjective Reinforcement Learning using the Expected Scalarized Reward Criterion – Appendix. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025)*, Detroit, Michigan, USA, May 19 – 23, 2025, IFAAMAS, 1 page.

A DESCRIPTION OF THE TEST ENVIRONMENT

In this section, we describe the environment used to evaluate our agents. This environment was inspired by the Pressureplate¹ environment and models cooperative multiobjective multiagent sequential decision-making tasks. The agents have to solve the task of resource distribution to different types of households. The environment is modeled as a 2d grid world of custom size and is set to 10 × 10 by default. At each timestep, the agents can move north, south, east, and west on the grid or stay still. A moving action is only successful if the cell the agent is trying to reach does not contain another agent on it. At each new episode, for each household type a number between 4 and 7 is picked uniformly at random, representing the number of households of that type that will be present in the environment. These households are then placed in the environment by sampling positions uniformly. The agents are randomly positioned on the grid cells that do not already contain households. The goal of the agent is to distribute resources among households fairly. An agent can serve as a resource to a household by moving toward the cell where the household is located. Each agent can serve at most 4 households per episode. The episode ends when all agents exhaust all the resources they had available or after 1000 timesteps of the environment. An agent’s observations consist of its relative position in the grid, the total amount of resources it still has not served, and the amount of resources it served for each type of household. Agents also use sensors to observe other agents and all the households that haven’t been accommodated yet. The sensor radius can be customized and is set to 2 by default. Under the global reward setting, after each timestep, each agent receives a vector reward of size d (with d representing the number of objectives in the environment), where the i -th component of that vector is the number of households of type i that all the agents accommodated during that timestep. Under the local reward setting the difference is that the i -th component of the vector represents the number of households of type i that were accommodated only by the agent

during that timestep. Note that the reward is normalized by the maximum reward achievable by the agent to stabilize training.

B COMPUTING OPTIMAL PROPORTIONAL ALLOCATION

To compute an optimal proportional allocation the following algorithm is used.

Algorithm 1: Max-Min Proportion

Input: $n_objectives$: int, $demand$: List of int,
 $total_bag_size$: int
Output: max-min proportion: float

```
1 if  $total\_bag\_size < n\_objectives$  then
2   raise Exception("Bag size not sufficient to compute max
   proportion");
3 end
4  $unit\_prop \leftarrow 1/demand$ ;
5  $max\_min\_prop \leftarrow 1/demand$ ;
6 for  $i \leftarrow n\_objectives$  to  $total\_bag\_size - 1$  do
7    $min\_indices \leftarrow \text{argmin}(max\_min\_prop)$ ;
8    $max\_min\_prop[min\_indices] \leftarrow$ 
      $max\_min\_prop[min\_indices] +$ 
      $unit\_prop[min\_indices]$ ;
9 end
10 return  $\min(max\_min\_prop)$ ;
```

¹<https://github.com/uoe-agents/pressureplate>