FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Exploring the impact of different behaviours on fairness and efficiency in MADRL

Margarida Silva

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Rosaldo J. F. Rossetti, PhD

Second Supervisor: Zafeiris Kokkinogenis, MSc

External Supervisor: Jeremy V. Pitt, PhD

July 22, 2021

Exploring the impact of different behaviours on fairness and efficiency in MADRL

Margarida Silva

Mestrado Integrado em Engenharia Informática e Computação

Chair: Prof. Rui Camacho, PhD
External Examiner: Prof. Pedro Campos, PhD
Supervisor: Prof. Rosaldo Rossetti, PhD

July 22, 2021

Abstract

Reinforcement Learning is the paradigm of learning to make good sequences of decisions. Throughout history, it appeared as a mechanism to maximise an expectation of benefit - reward - that may come in the future for taking some action at some point in time. More recently, the concept of fairness has been a consideration for the design of these models. In particular, for the multi-agent setting, some work attempts in finding a learning architecture that promotes equality in the distribution of rewards of the agents within the system. While there are works that attempt to achieve this goal in the most efficient way possible, there seems to be a gap in the literature for exploring the spectrum of behaviours that can be found between fairness and efficiency.

There are two primary motivations for studying this range. On the one hand, while it is natural to think that fairness and efficiency are not correlated objectives, the result from mixing fair and efficient policies is indeed still unknown in the multi-agent deep reinforcement learning setting. On the other hand, from a practical perspective, a user's intent for a system may be a goal in-between focusing entirely on efficiency and an as efficient as possible fair system. Indeed, it may be that the optimal solution for such a user is within the spectrum of these two options.

In this work, we study the performance of the state of the art fairness model - SOTO - in the fair-efficient spectrum. Moreover, we produce changes in this fairness architecture and evaluate the impact of each of these changes on the shape of the new spectrum. We change the learning strategy of this model and propose a different architecture - I-SOTO - that is an extension to the first by enabling a bidirectional recommendation between the team and self-oriented points of view during the decision making process.

We tested these approaches in the Matthew Effect and Traffic Light Control environments, which share in common the existence of dynamics that promote inequality in rewards obtained. The results show that, with the proposed training strategy changes for SOTO and I-SOTO, it is possible to find competitive options compared to the original SOTO setting and an independent baseline. In particular, at least one solution in each environment can outperform the SOTO model under the same social welfare goal.

Keywords: Multi-Agent Deep Reinforcement Learning, Fair Optimisation, Social Welfare

Resumo

Aprendizagem por Reforço é o paradigma de aprendizagem que se debruça sobre como fazer boas sequências de decisões. Ao longo da história, apareceu como um mecanismo para maximizar uma espectativa de benefício - recompensa - que pode advir no futuro vinda de uma acção tomada num determinado momento. Mais recentemente, o conceito de justiça tem sido considerado no desenho destes modelos. Em particular, no contexto multi-agente, alguns trabalhos tentam encontrar uma arquitetura que promova a equidade na distribuíção de recompensas que os agentes de um sistema recebem. Embora hajam trabalhos que tentam atingir este objetivo da forma mais eficiente possível, parece existir uma ausência de exploração na literatura no o expectro de comportamentos que podem ser encontrados entre estas duas opções.

Existem duas principais motivações para estudar este espectro. Por um lado, apesar de ser senso comum não ver a equidade e a eficiência como objectivos correlacionados, o resultado da mistura de políticas equitativas e efficientes é ainda assim desconhecido no âmbito de apredizagem por reforço profunda em sistemas multi-agente. Por outro lado, de um ponto de vista mais prático, a intenção do utilizador para o sistema pode ser um intermédio entre um foco completo em efficiência ou um sistema equitativo tão eficiente quanto possível. De facto, pode ser o caso que a solução ótima para esse utilizador esteja dentro do espectro entre estas duas opções.

Nesta dissertação nós estudamos a performance do modelo estado de arte em equidade - SOTO - dentro do espectro equidade-eficiência. Para além disso, nós fazemos modificações a esta arquitetura e avaliamos o impacto de cada uma destas alterações na forma do novo espectro. Nós modificamos a estratégia de aprendizagem do modelo e propomos uma architetura diferente - I-SOTO - que estende da primeira permitindo recommendação bidirecional entre pontos de vista equitativos e eficientes na tomada de decisão.

Nós testamos estas abordagens nos ambientes Matthew Effect e Traffic Light Control, que partilham em comum a existência de dinâmicas que promovem inequidade de reforços obtidos. Os resultados mostram que, com as mudanças propostas para a estratégia de treino no SOTO e I-SOTO, é possível encontrar soluções competitivas comparando com o modelo SOTO original e uma baseline independente. Em particular, existe pelo menos uma solução em cada ambiente que tem melhor performance tanto em eficiência como equidade comparada com o SOTO para o mesmo objetivo de bem-estar social.

Keywords: Aprendizagem por Reforço Profunda Multi-Agente, Optimisação Justa, Função de Bem-Estar Social

Acknowledgements

First of all, to my supervisors. To Rosaldo Rossetti for encouraging my ideas, giving me creative freedom and helping throughout developing this dissertation before it had even begun being thought of. To Zafeiris Kokkinogenis for helping me gain expertise on Reinforcement Learning by fostering ongoing discussion of concepts, guiding towards relevant literature and encouraging critical thinking. Last but not least, to Jeremy Pitt for accepting to collaborate in the project, expanding my ideas on the concept of fairness and providing insightful reflections on the purpose of developing fair systems. The contributions of all were essential for the development of the project. As a result of this collaboration, I got the opportunity to travel and work on this dissertation at Imperial College London. I am very grateful for this experience and will undoubtedly remember the months spent in London with unique warmth.

Secondly, to everyone who impacted my journey throughout the Masters and the development of this work. Despite not making a direct impact on its content, this work is undoubtedly a by-product of influences from these people and their contribution to my well-being and predisposition for creative and analytical thinking.

Margarida Silva

The worst form of inequality is to try to make unequal things equal.

Aristotle

Contents

List of Figures

List of Tables

Abbreviations

RL	Reinforcement Learning
MARL	Multi-agent Reinforcement Learning
MADRL	Multi-agent Deep Reinforcement Learning
TD	Temporal Difference
MC	Monte Carlo
DP	Dynamic Programming
DQN	Deep Q-Network
NN	Neural Network
MDP	Markov Decision Process
POMDP	Partially Observable Markov Decision Process
Dec-POMDP	Decentralised Partially Observable Markov Decision Process
SWF	Social Welfare Function

Notation

Calligraphic letters - $\mathcal{X}$ - denote alphabet sets, upper-case letters - X - denote random variables and constants in some cases, hats denote approximations - $\hat{X}$ - and lower-case letters - x - denote realizations.

$\doteq$	equality true by definition
$\Pr\{X = x\}$	probability of a random variable X taking value x
$X \sim p$	random variable X under distribution $p(x) \doteq \Pr\{X = x\}$

s	a state
a	an action
r	a reward
$\mathcal{S}$	set of possible s states
$\mathcal{A}$	set of possible a actions
$\mathcal{R}$	set of possible r rewards, a finite subset of $\mathbb{R}$

t	discrete time step
T	number of time steps of an episode
s_t	the state at t
a_t	the action at t
g_t	the return following time t
e	discrete episode
E	number of episodes of the training process

π	policy (decision-making rule)
$v(s)$	value of state s
$\hat{V}(s)$	approximation of the value of state s
$q(s,a)$	q-value of state s and action a
$\hat{Q}(s)$	approximation of the q-value of state s and action a
γ	discount-rate parameter

In the function approximation context:

$\mathbf{w}, \mathbf{w}_t$	vector of weights underlying a value function approximation
θ	vector of weights underlying a policy function approximation

In the SOTO context:

π^{IND}	self-oriented policy
θ^{IND}	vector of weights underlying the self-oriented policy function approximation
π^{SWF}	team-oriented policy
θ^{SWF}	vector of weights underlying the team-oriented policy function approximation

Chapter 1

Introduction

In this chapter, we introduce the object of study in this dissertation. We start by providing some context and motivation for the problem in the subject. Then, we present our goals for the work and questions we aim to answer, followed by a summary of the main contributions achieved. Finally, the remaining document is outlined in the last section.

1.1 Context and Motivation

Reinforcement Learning (RL) is the paradigm of making good sequences of decisions, provided the reward of an environment that reflects positive and/or negative impacts of actions. Despite being around for decades [92], more recently, RL has recently shown very promising results, which led to increased activity within its research community, particularly in Deep Learning approaches [82, 90]. Multi-Agent Deep Reinforcement Learning [32] (MADRL) is the problem of having multiple Deep RL agents within a shared environment. In such settings, additional challenges arise due to the system's number of agents, the uncertainty that it brings, and a potential need for decentralised models.

RL techniques have aimed to optimise the expected sum of rewards an agent gets for acting under its policy throughout history. More recently, fairness concerns have been brought into the Machine Learning literature, and a fairness-aware line of algorithms are emerging. In the multi-agent paradigm, some work attempts to optimise the equality in the distribution of rewards of the agents in the most efficient manner possible [40, 103]. This literature approaches fairness as a goal for the agents' policies within the system and uses architectural adaptations to help the agents still perform efficiently. The state of the art model in this setting, SOTO [103], includes learning self- and team-oriented policies with traditionally selfish and fair optimisation goals, respectively, in which the first provides recommendations to the latter, helping it become efficient apart than fair. However, there is a lack of work considering fairness and efficiency without a clear aim for optimising one of them. Indeed, in a real-world setting, such a setting may not be ideal. As a

system designer, one may prefer an in-between solution between the efficient and fair extremes. Indeed, the designer should be able to opt to sacrifice one of them, to a certain extent, for the other, and such a decision is not possible with the current literature.

Moreover, there is still no evidence of the outcome of mixing fair and efficient behaviours or even training these together in a MADRL system. While efficiency is considered for helping fair models be efficient, the opposite is never. It may be the case that, depending on the dynamics of the environment, these goals can lead to better solutions together.

1.2 Objectives

The general goal of this dissertation is to tackle the equality of rewards fairness problem in MADRL in an exploratory manner. By relaxing assumptions on which goal is intended for the system - efficiency or fairness - we aim to observe what solutions arise. We aim to observe whether any of these is better than the efficient and fair baselines in any of the goals - ideally, both.

We aim to tackle this challenge by employing three different techniques. The first is heterogenous testing, where the agents in the system act selfishly or fairly according to a probability, without updating their policies weights. This enables a direct mix in previously learned policies. The second technique is to experiment with different settings for the training strategy that controls how SOTO's self- and team-oriented policies are trained. Doing so, we believe different solutions will be found in both extremes and heterogeneous intermediates from testing. Finally, we aim to experiment with an extension to the SOTO architecture, enabling the team-oriented policy to provide fair action insights to the self-oriented one. Because each policy recommends the other intertwined in this setting, we call this method Intertwined SOTO (I-SOTO). To sum up, the questions we aim to approach are:

- Is it possible to find a range of fair-efficient behaviours from previously learned policies?
- Does SOTO perform better in either fairness or efficiency with different training strategy functions?
- If the fairness-oriented also gives action recommendations to a self-oriented one, does the performance of the SOTO model change? In what ways?

1.3 Contributions

This dissertation contributes to the state of the art on equality fairness in MADRL in a variety of manners. In a broad sense, our contribution mainly relies on presenting results for the exploratory attempts made in combining fair and efficient policies and training them jointly. These results are presented in popular environments in the equality fairness literature for MADRL because they include dynamics that foster unequal access or opportunities to obtain rewards, i.e., being efficient. We extend the art fairness model state and compare the results obtained with it, together with an

efficient baseline. In particular settings of training strategies and environments, we find that our models can perform better than the SOTO baseline. The list of contributions of this dissertation could be summarised as follows:

- We propose methods for mixing the efficient and fair goals in MADRL through testing - under heterogeneous behaviour - and training - exploring different training strategies for SOTO and a different architecture I-SOTO
- We study these methods in two environments popular in the literature of fairness: Matthew Effect and Traffic Light Control
- We show that, in both of these environments, particular instances of our methods outperform the SOTO baseline and others the efficient baseline
- We provide visual intuitions and results for all of the experiments accomplished in our exploratory methods
- We discuss differences in baseline results with SOTO authors and conclude on the adequate efficiency metric for Traffic Light Control
- We review fairness related work within the MADRL field

1.4 Outline

In the remainder of this document, we start by presenting a background and related work overview in Chapter 2. Then, we describe the methodological approach utilised in Chapter 3. The results of the experiments employed under such methods are present in Chapter 4. Finally, in the last chapter, we conclude on the final remarks regarding this work in Chapter 5.

Chapter 2

Literature Review

In this chapter, we provide the necessary background to fully understand the extent of our work together with related work. We start by providing some background on Reinforcement Learning and its core concepts, then move to methods that require function approximation and, finally, review existent literature within the multi-agent domain, both in settings with and without fairness concerns.

2.1 Reinforcement Learning

Reinforcement Learning (RL) is the Machine Learning paradigm of learning to make good sequences of decisions. As its name suggests, RL agents learn by reinforcement from the environment they are put under. Higher reinforcements signal positive behaviours from the agent as opposed to lower reinforcement values, almost as praise or punishment for actions. The aim of RL is then to make the most out of this reward signal and behave in the way that optimises it - whatever it is.

2.1.1 Markov Decision Processes

Early forms of Reinforcement Learning were methods for solving the problem of a Markov Decision Process [69] (MDP). An MDP is a stochastic control optimisation problem [87] characterised by the 5-tuple $\langle \mathcal{T}, \mathcal{S}, \mathcal{A}, p, r \rangle$, where

$\mathcal{T}$	is the set of decision epochs, i.e. opportunities in time for the agent to make decisions. Can be finite/infinite, discrete/continuous.
$\mathcal{S}$	is the state space i.e. set of possible states
$\mathcal{A}$	is the action space i.e. set of possible actions. $\mathcal{A}(s)$ is the set of possible actions in state s.
$r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$	is the reward $r(s_t, a_t, s_{t+1})$ for performing action a_t under state s_t and moving to the next state s_{t+1}
$p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$	is the transition probability $p(s_{t+1}\|s_t, a_t)$ of moving to state s_{t+1} as after taking action a_t under state s_t.
$\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$	is the policy or decision-making rule $\pi(s_t)$ as the probability of choosing action a_t under state s_t. Can be deterministic or stochastic.

Illustrated by Figure 2.1, at each timestep t, an agent under state s_t consults its policy π to perform an action $a_t \sim \pi(s_t)$. Then, the environment changes to state $s_{t+1} \sim p(s_{t+1}|s_t, a_t)$, and this agent receives the corresponding reward $r(s_t, a_t, s_{t+1})$.

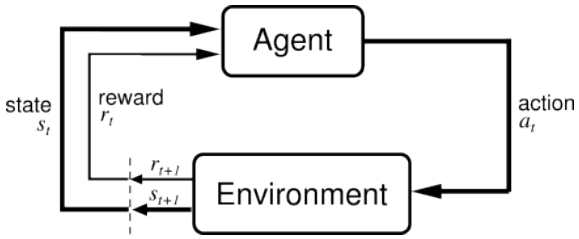


Figure 2.1: Agent-Environment interaction in a Markov Decision Process [87]

The name of these processes is inherited from the assumption they rely on. The Markov Assumption states that the next state is independent of past states, given the current state. Formally, this assumption is expressed in equation 2.1.

$$\Pr\{s_{t+1}, r_{t+1} \mid s_t, a_t, s_{t-1}, a_{t-1}, \ldots, s_0, a_0\} = \Pr\{s_{t+1}, r_{t+1} \mid s_t, a_t\} \tag{2.1}$$

As such, one can now clearly see why either of the functions that characterise an MDP - r, s and π - provide insights on the dynamics of the environment provided information on the present time step t, and only t.

To accommodate real-life challenges [24], the MDP definition has been extended to contemplate a more general case of conditions. One of these is incomplete or incorrect access to the world's state. Lack of sensors, sensory inaccuracy and environment limitations are some of many reasons that may impair the accessibility of state information to the agent. A Partially Observable Markov Decision Process [43, 84] (POMDP) is an extension the MDP fully observable model in which agents only perceive observations $o \in \mathcal{O}$, modelled by an observation function $\omega : \mathcal{S} \times \mathcal{A} \times \mathcal{O}$, instead of accessing the state $s \in \mathcal{S}$ directly. Let $h_t{}^i = o_t, a_{t-1}, o_{t-1}, \ldots, a_1, o_1, a_0$ be the history of

observations and actions of agent i since the start of the simulation [1]. Under these circumstances, each agent i should find alternative ways of enriching their perception of ${s_t}^i$ to overcome the lack of information in an observation. In the literature, popular options are

${s_t}^i = {h_t}^i$	Using the complete history of observations and actions available to the agent. This naturally leads to an explosion in the dimensions necessary to represent the state.
${s_t}^i \sim \mathbb{P}[s \mid {h_t}^i]$	Building a belief of which is the current state, given the history, as a probability distribution [62].
${s_t}^i = \sigma({s_{t-1}}^i W_s + o_t W_o)$	Encoding the state in a recursive manner by combining the previous state representation s_{t-1} with the current observation o_t with the weights W_s and W_o, respectively [30]. This encoding is commonly accomplished with the use of Artificial Neural Networks, whose functioning is to be further described in section 2.2.1.

2.1.2 Prediction

In the RL domain, prediction refers to the task of estimating the consequences of actions performed by agents. Oftentimes, the object of prediction in RL methods is associated with a nomenclature. In particular, methods that make estimations of the policy π, the transition function p and the reward function r or derivatives are named *policy-*, *model-* and *value-based* respectively. Due to the importance of value-based methods in the RL literature, we dedicate section 2.1.2.1 to it. Section 2.1.2.2 refers to how these estimations are made throughout time steps. Lastly, the commitment of making predictions with regards to the action provided by the policy, or lack thereof, is also a way of naming methods as *on-* or *off-policy*, respectively, as presented in Section 2.1.2.3.

2.1.2.1 Value Estimation

One of the most important outcomes to employ estimations on is, naturally, the reward value expected to arise from performing some action. The agent is then able to control which action to take by choosing the one that seems most promising. This potential reward outcome is commonly known as return g_t. An intuitive but naive way of defining this return is as the sum of rewards received $\sum_{k=1}^{T} r_k$, where T is the last time step. However, many real applications are made of continuous agent-environment interactions, making $\mathcal{T}$ infinite. In order to provide convergence properties to g_t [87], a discounting factor $0 \leq \gamma \leq 1$ is added and g_t is defined as the discounted sum of rewards - also referred to as the discounted return - as defined in Equation 2.2.

$$\begin{aligned} g_t &\doteq r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{T} \gamma^k r_{t+1+k} \\ &= r_{t+1} + \gamma g_{t+1} \end{aligned} \tag{2.2}$$

[1] Also known as trajectory

Because the ultimate goal of RL is to maximise this return, it is oftentimes referred to as value. If this value is estimated only with respect to the state the agent is under, the State-value function V^π of some policy π is given by Equation 2.3. Notice that, contrary to the aforementioned definition of return in Equation 2.2, because in this case, we are defining estimations of values that may depend on stochastic processes - as seen in the MDP section - return and reward are now random variables and, by extension, any estimations based on them.

$$V^\pi(s) = \mathbb{E}^\pi [G_t \mid S_t = s] = \mathbb{E}^\pi \left[\sum_{j=0}^{T} \gamma^j R_{t+j+1} \mid S_t = s \right] \tag{2.3}$$

If we extend V by also considering particular actions, we are now referring to the Quality-value, or Q-value in short. Such function provides the estimated return for performing a particular action a, given the agent's policy π, as shown in Equation 2.4.

$$Q^\pi(s,a) = \mathbb{E}^\pi [R_t \mid S_t = s, A_t = a] = \mathbb{E}^\pi \left[\sum_{k=0}^{\mathcal{T}} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a \right] \tag{2.4}$$

In deterministic policies, V has the same value of Q given the policy's action - unique in this case. However, in stochastic ones, the q-value is equal to the state-value distributed over the action distribution of the policy. This relationship is formally portrayed in Equations 2.5 and 2.6, respectively.

$$V^\pi(s) \doteq Q^\pi(s, \pi(s)) \tag{2.5}$$

$$V^\pi(s) \doteq \sum_{a \in \mathcal{A}} p(a \mid S_t = s) Q^\pi(s,a) \tag{2.6}$$

The availability of a reward function - or at least an accurate one - is not trivial in all applications. Indeed, designing a good reward function is a challenge per se in complex environments such as driving a vehicle. Potential approaches to solve this problem include inferring the reward function from already existing policies known to behave well, with Inverse Reinforcement Learning [59, 50], or even learning directly an approach from such examples, with Imitation Learning [35]. We could also tweak the existent reward function into a more reliable one with the use of Reward Shaping [105, 9]. While we will not be directly approaching this problem, we acknowledge the importance of having a good reward function to rely on while value predictions, independently of the estimation utilised.

2.1.2.2 Update Strategy

A crucial part of making estimations is how they are successively updated. There are three essential update paradigms in RL: Dynamic Programming (DP), Monte-Carlo methods (MC) and Temporal Difference (TD). While each of these has its own intricacies, we compare them in a high-level manner and direct the interested reader to [87]. We consider in this section the state-value for simplicity in understanding.

Dynamic Programming [22] was one of the first update methods that appeared in RL. In this method, the estimation is updated at each time step t using Equation 2.7, also known as the Bellman Equation [6, 23]. As can be seen, this method relies on the transition function, being inapt for *model-free* domains. Notice that this model requires the transition function p to function, being inapt for *model-free* domains. It is also important to note that it bootstraps the return value if the estimation of the value of the following state $V_t(s')$, which is naturally a source of bias.

$$\hat{V}_{t+1}(s) = \sum_{a \in \mathcal{A}} \pi(a \mid s) \left[r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s,a,s') \hat{V}_t\left(s'\right) \right] \tag{2.7}$$

Monte Carlo methods, on another note, function quite differently. Under this update paradigm, the agent waits for the end of the episode to update its estimation, making this method not suitable for continuing tasks as they may never come a time for the agent to learn. There are many variants to this method, but the general idea is that the value of each state is a function of the summation of returns generated after such state, $G(s)$ over the number of times such state appeared throughout episodes $N(s)$ - popularly, the average, as shown in Equation2.8. In the First-Visit variant, each state is only considered once per episode, so the summation of returns is averaged over the number of episodes.

$$\hat{V}(s) = \hat{G}(s)/\hat{N}(s) \tag{2.8}$$

Temporal difference is the most recent of all methods and the most popular in recent RL literature. Combining principles from DP and MC, TD not only supports continuing domains but also handles Model-free domains, i.e., problems in which the world model - transition and reward functions - are unknown. A example of a TD variant, TD(0), is illustrated in equation 2.9. Given the transition (s_t, a_t, r_t, s_{t+1}) from time step t to $t+1$, TD updates the estimation of the value of being in state s taking into account the reward received r_t and the value of the next state $V_{t+1}(s)$ after performing the action a determined with the policy. The TD-error δ_t is then defined by Equation 2.10 as the difference between the current reward plus value estimation for $t+1$ and the value estimation for t. Although this difference does not include the true value of return g_t, it is the approximation possible at time step t.

$$\hat{V}_{t+1}(s_t) = \hat{V}_t(s_t) + \alpha[r_t + \gamma \hat{V}_t(s_t) - \hat{V}_t(s_{t+1})] \tag{2.9}$$

$$\hat{\delta}_t = r_{t+1} + \gamma \hat{V}\left(S_{t+1}, \mathbf{w}_t\right) - \hat{V}\left(S_t, \mathbf{w}_t\right), r_{t+1} \sim R_{t+1} \tag{2.10}$$

A comparative summary of these three methods follows in presented table 2.1.

2.1.2.3 On-Policy & Off-Policy

The terms *on-policy* and *off-policy* refer to value estimation methods that make estimations limited on the policy π decision-making rules or that go beyond such rules, respectively. Q-Learning [93,

Control	Model-free domains	Continuing domains	Non-Markov domains	Unbiased
DP		✓		
MC	✓		✓	✓
TD	✓	✓		

Table 2.1: Comparison of Prediction Update Strategies

92] is an example of an off-policy method, depicted in equation 2.11. While making q-value updates quite similar to those of SARSA. However, after an agent performs action a_t under state s_t and state is updated to s_{t+1}, the q-value update is made considering the next best action that can be taken under state s_{t+1}, which may not be necessarily equal to what the policy $\pi_{s_{t+1}}$ would dictate, since the update of $\hat{Q}(s_t, a_t)$ has not been completed yet. While having a very similar update equation, SARSA is an on-policy alternative to Q-Learning. As depicted in equation 2.12, the main difference is on the γ parcel that considers the action $a_{t+1} \sim \pi_{s_{t+1}}$, i.e. the action pointed by the policy with regards to the next state, and naturally never in its updates considers other actions from such.

$$\hat{Q}_{t+1}(s_t, a_t) \leftarrow \hat{Q}_t(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a \hat{Q}_t(s_{t+1}, a) - \hat{Q}_t(s_t, a_t)] \tag{2.11}$$

$$\hat{Q}_{t+1}(s_t, a_t) \leftarrow \hat{Q}_t(s_t, a_t) + \alpha[r_{t+1} + \gamma \hat{Q}_t(s_{t+1}, a_{t+1}) - \hat{Q}_t(s_t, a_t)] \tag{2.12}$$

2.2 Approximate Solution Methods

When $\mathcal{A}$ and $\mathcal{S}$ are finite sets, it is possible to make predictions using tables with entries for different states or state-action pairs, i.e. with *tabular methods* [98, 11]. However, when either $|\mathcal{A}|$ or $|\mathcal{S}|$ are very large - maybe even infinite - these lack in generalisation and efficiency. Not only would it be costly to store tables with a tremendous amount of entries, but it would also take much time to be able to produce reasonable approximations for all entries, and they are built from a sequence of several updates. Additionally, since each action/state is treated as an independent entity, tabular methods lack in generalisation. Given two very similar in meaning states s_i and s_j, if s_i is close to convergence and s_j is not, when the agent lands s_j it will still perform poorly, in theory, as it has not lived that exact state several times and there is no way of inferring $v(s_j)$ from $v(s_i$. An evident case where this becomes problematic is when either $\mathcal{A}$ or $\mathcal{S}$ are continuous sets, i.e. $\mathcal{A} \in \mathbb{R}^n, n \in \mathbb{N}$, $\mathcal{S} \in \mathbb{R}^n, n \in \mathbb{N}$, and thus it would be intractable to have a table with infinite entries.

In these cases, a parameterised function representation is used to approximate the estimations of policies, values or models. In the Machine Learning (ML) literature, linear models have been widely used to make these approximations. These models frame the approximation problem as finding the best set of weights $\mathbf{w}$ and bias b enable $\mathbf{f}(\mathbf{x})$ - as in Equation 2.13 - that better represents

the real values of some distribution $\mathbf{y}$. In order words, to minimise the difference $f(\mathbf{x}) - \mathbf{y}$.

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b \tag{2.13}$$

2.2.1 Neural Networks

While linear models can be adapted to also approximate non-linear functions - with the use of polynomials, Fourier series or Radial Basis Functions - this process requires manual transformation inputs and outputs ("feature engineering") and requires some insight to be successful. Artificial Neural Networks (ANNs), also known as Neural Networks (NNs), are more flexible models that are able to learn at least some of these non-linear transformations.

A Neural Network, in its simplest form, can be viewed as a set of linear layers where each layer l is placed after the other $l-1$. The values of the first layer also referred to as the input layer, are the inputs given to the function $\mathbf{h}^{(0)} = \mathbf{x}$. The subsequent layers are the result of applying an activation function $g^{(l)}$ to the outputs of the previous $\mathbf{h}^{(l-1)}$ layer multiplied by a set of weights $W^{(l)}$ plus the layer biases $\mathbf{b}$, as in Eq. 2.14. Neural Network architectures with a large number of layers - "deeply layered" - give the name to a sub-field of Machine Learning called Deep Learning.

$$\mathbf{h}^{(l)} = g^{(l)}\left(W^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\right) \tag{2.14}$$

While NNs enable the learning of more complex functions, its fitting process is much more challenging, requiring more data and time to converge successfully. Not only the parameter number is much larger, but the optimisation process is rather iterative through successive updates of the weights applying the chain rule for each neuron u_i, given that the previous layer in which u_i is located has D dimensionality2.15.

$$\frac{\partial f}{\partial u_i} = \sum_{d=1}^{D} \frac{\partial f}{\partial x_d}\frac{\partial x_d}{\partial u_i} \tag{2.15}$$

Neural Networks use Gradient-based optimisation to fit their parameters. The most naive optimisation process is the Steepest Gradient Descent 2.16 where the weights are updated by being subtracted with the cost function multiplied by a learning rate η. The cost function used as an example in Equation 2.16 is the squared error, having $\mathbf{r}$ be the vector of errors or differences between the predictions and targets. However, a common loss function could be the root mean squared error, averaging this error over the number of samples considered and then calculating the square of such value. The important thing is that this cost function represents how much the algorithm needs to "pay" for wrong predictions, such that its gradient on the weights of the network $\nabla_{\mathbf{w}}[\mathbf{r}^\top \mathbf{r}]$ directs it to a good direction towards minimising it.

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} \left[\mathbf{r}^\top \mathbf{r}\right] \tag{2.16}$$

An alternative is to use Gradient Ascent, which, as the name indicates, aims to maximise an objective function instead of minimising a cost function. The key idea is identical to gradient,

but instead, the product between the learning rate and the cost function is summed to the weights instead of subtracted.

However, this process demands great computational power as either the cost or objective function needs to be evaluated for each sample of a dataset - which may have thousands of thousands of entries in extreme cases - and for each step of the optimisation - which may also require a great number of steps. In order to solve this, Stochastic Gradient Descent selects on a random sample of the dataset to calculate the value of the cost function and multiplies it by the number of samples in order to estimate the actual dataset value. ADAM optimiser [44] is an instance of such a method that has become widely popular, especially for Deep Learning applications.

2.2.2 Eligibility Traces

An important concept to consider in Function Approximation updates is Eligibility Traces. Under this paradigm, a parameter λ controls the eligibility of each dimension of the weight vector of a function-approximated approximation to be updated. As such, it offers an in-between solution for DP and MC updates with regards to how many time steps in advance are considered. When $\lambda = 0$ - TD(0) - it only considers the immediately following state as DP does. On the other hand, when $\lambda = 1$ - TD(1) - it gets more similar to MC by considering all the following time steps until the end of the episode, being identical when $\gamma = 1$. The eligibility trace is a vector $\mathbf{Z}_t \in \mathbb{R}^d$, as defined in Equation 2.17, and the TD(λ) update is now defined as presented in Equation 2.18.

$$\begin{aligned} \mathbf{z}_{-1} &\doteq \mathbf{0} \\ \mathbf{z}_t &\doteq \gamma\lambda\mathbf{z}_{t-1} + \nabla\hat{V}(s_t, \mathbf{w}_t), \quad 0 \leq t \leq T \end{aligned} \tag{2.17}$$

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\delta_t\mathbf{z}_t \tag{2.18}$$

2.2.3 Advantage

A recent approach to estimating value is the concept of advantage. As shown in Equation 2.19, the advantage $A(s,a)$ is by definition the difference between V and Q for a given state-action pair. If we make a TD estimate for $\hat{Q}(s,a)$ as $r + \gamma \cdot \hat{V}(s')$, then the advantage can also be seen as the TD error $\hat{\delta}_t$, as previously seen. The intuition behind this idea is that if we remove to the Q-value the value of the state itself, we are left with the value of the action itself under such a state. Naturally, in cases where the state is very promising, all actions will have very high Q-values, so the decision-making process may be misled and compromised. As such, the advantage function is used to eliminate the bias for estimation of the value of an action that comes from the value of the state itself.

$$\begin{aligned} A(s,a) &\doteq Q(s,a) - V(s) \\ \hat{A}(s,a) &= r + \gamma \cdot \hat{V}(s') - \hat{V}(s) \\ &= \hat{\delta}_t \end{aligned} \tag{2.19}$$

However, this advantage definition is highly dependent on the value estimation. Should this estimation be biased, this will also be imprinted in the advantage estimation. In order to approach this issue, the Generalised Advantage Function [75] (GAE) brings the eligibility Traces concept to the definition of advantage. As shown in Equation 2.20, the GAE advantage $\hat{A}_t^{\text{GAE}}$ for a given transition time step is an exponential average of the advantage estimations for such transition but throughout the following n steps. This concept, of course, implies that the update on the value estimation is only done after a collection of n time steps.

$$\hat{A}_t^{\text{GAE}(\gamma,\lambda)} \doteq (1-\lambda)\left(\hat{A}_t^{(1)} + \lambda\hat{A}_t^{(2)} + \lambda^2\hat{A}_t^{(3)} + \ldots\hat{A}_t^{(n)}\right) = \sum_{l=0}^{n}(\gamma\lambda)^l\delta_{t+l}^V \tag{2.20}$$

2.2.4 Policy Gradients

Policy Gradient methods are gradient-based optimisations used for policy approximation. The general idea behind these algorithms is that at each timestep t, the policy is updated - originally, through gradient ascent - towards the direction that leads to the policy choosing the best action possible a^*, as shown in Equation 2.21.

$$\theta_{t+1} = \theta_t + \alpha\nabla\pi_{\theta_t}(a^* \mid s) \tag{2.21}$$

REINFORCE [97] is one of the earliest and simplest Policy Gradients algorithms. Naturally, one challenge of Policy Gradient methods is finding the direction where $\pi_{\theta_t}(a^*|s)$ gets closer to one. In order to address this, REINFORCE uses an estimation of the expected return $\hat{g}_t$. This may be in the form of a q-value or even an advantage value, as in Equation 2.22. However, because this estimation may provide very high values in earlier stages due to the initialisation utilised, the gradient is then divided by the current probability of the action in question $\pi(a_t|s_t,\theta_t)$.

$$\theta_{t+1} = \theta_t + \alpha\frac{\hat{Q}(s,a)\nabla\pi_{\theta_t}(a \mid s)}{\pi_\theta(a \mid s)} \tag{2.22}$$

Another very common way to present the REINFORCE equation is in the logarithm form, as in Equation 2.23, since $\frac{\nabla\pi_\theta(a|s)}{\pi_\theta(a|s)} = \nabla_\theta \log\pi_\theta(s \mid a)$.

$$\theta_{t+1} = \theta_t + \alpha\hat{A}(s,a)\nabla_\theta \log\pi_\theta(s \mid a) \tag{2.23}$$

Since the appearance of REINFORCE, many alternative policy optimisation algorithms have appeared [74, 49, 76]. Trust Policy Region Optimisation [74] (TRPO) adds KL divergence constraints to ensure that new updated policies are not largely deviating. In other words, the new policy is not far away from the old policy, or we can say that the new policy is within the trust region of the old policy. Proximal Policy Optimisation [76] (PPO) that simplifies the implementation of this diversion implementation by imposing the policy ratio, $r(\theta) = \frac{\pi_\theta(a|s)}{\pi_{\theta old}(a|s)}$ to stay within

a small interval around 1, as shown in Equation 2.24 of PPO objective function. The clip operator is what makes sure it won't deviate largely by clipping the update in the range.

$$J^{\text{CLIP}}(\theta) = \mathbb{E}\left[\min\left(r(\theta)\hat{A}_{\theta_{\text{old}}}(s,a), \text{clip}(r(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_{\theta_{\text{old}}}(s,a)\right)\right] \tag{2.24}$$

2.2.5 Actor-Critic

As the name implies, Actor-Critic [45] approaches involve a dynamic between two elements: an actor - the policy - and a critic - some value estimation. The actor performs an action, and the critic is used to access - "criticise" - the policy's choice in behaviour [87]. However, as previously seen, policy gradients also involve the use of both a policy and value estimation, where the latter impacts the next policy weights θ_{t+1}. From a technical point of view, the main difference between Actor-Critic methods and Policy Gradients is that, whichever method is used, Actor-Critic value estimations contemplate the prediction of the value of the state after performing an action while pure policy gradients do not. From an intuitive point of view, when I am going beyond estimating the value of the current state but also including a prediction of value for the next, the model is no longer only named a Policy Gradients approach but also an Actor-Critic one. An example of value estimation associated with Actor-Critic approaches is the definition of advantage, as previously seen in Section 2.2.3.

In function approximation approaches, the state-value or action-value yielded by the critic is typically part of the expression of the gradient used by the actor to estimate the policy [77, 27, 52, 55, 36, 72, 28], linking the q-value approximation goal with the policy.

2.3 Multi-Agent RL

When multiple RL agents are placed within a shared environment, we are now under a Multi-agent Reinforcement Learning (MARL) problem [11, 12, 64]. Multi-agent systems [79] bring complexity to the optimisation process when compared to the single-agent case. In this case, the system can no longer be considered stationary. Because the states of other agents' are unknown to each agent within the system, the Markov Property no longer holds. In other words, the future state depends on more variables than the present state and action. This breach of this property is problematic as some control methods - and especially TD - are built under this assumption, as seen before, and bootstrap estimations accordingly. The following sections outline a few approaches and respective concerns to take into consideration in this setting.

Independence

A natural approach to extending the well-known single-agent algorithms to the multi-agent setting is to provide each agent with an independent single-agent model [1, 88]. This method is naturally advantageous in the sense that any model could be very easily applied to the MARL setting. However, it also brings additional challenges. Because every agent is acting on its own, the emergence

of coordinate behaviours is more challenging. Moreover, the computational complexity of the model is higher, $|\mathcal{D}|$ times more demanding.

Joint Action

Another thread of RL models frames the action **a** as the vector of the actions chosen by each agent - a joint action. If a model is set to optimise a joint action, perhaps the non-stationarity problem is solved since - in centralised approaches - the actions of all agents can be seen before making the final joint action decision. However, under this framing, the state and action spaces exponentially grow with the number of agents within the system. Each state can now contemplate the states of n-1 other agents, and each action is now $|\mathcal{A}| \cdot |\mathcal{D}|$. This dimensionality explosion - often referred to as a curse - presents a serial problem and affects the RL algorithms' ability to converge. It affects the exploration-exploitation trade-off agents need to make since it is tough to explore the behaviour of all agents within the system.

On the other hand, should agents perform a joint action, it is only natural that the environment should return a joint reward for such action - typically, the sum of the individual is potentially different. Consequently, it may be difficult for agents to infer how much they are responsible for such a reward and even 'learn to rest' if other agents are already contributing very intensely to the global systems' reward. This challenge is popularly known as the *multi-agent credit assignment* [14] problem. In the literature, many approaches aim to solve this problem and transform this joint reward into a set of individual rewards. The most relevant techniques include neural network architectures for reward distribution [86, 70, 83] and the usage of estimation concepts such as Difference Rewards [27, 13] and Shapley Value [91]. In domains where the optimisation is multi-objective, this poses an even more complicated challenge since the reward is now a multi-dimensional vector: there is a credit to attribute for each of these dimensions [99].

Decentralisation & Dec-POMDPs

As seen before, when the agents only have access to a part of the environment state, they are called partially observable MDPs (POMDPs). Oliehoek et al. [62] introduces the concept of decentralised POMDPs (Dec-POMDPs) as a generalisation term for a system with multiple agents of a POMDP used to model a team of cooperative agents under a stochastic, partially-observable environment. [63, 46]]. Decentralisation implies there is a concern to some extent to the amount of information the agents may share between each other and that methodologies that require complete centralisation are to be avoided if possible. A typical approach to this scenario is either a centralised learning and decentralised execution approach - where the centralised learning components should be detachable for a posterior decentralised execution - or a fully decentralised [102] approach, where communication between neighbours of state information is typically used to overcome this problem [39, 40, 103, 54, 31]. These communication mechanisms may contemplate weights of the functions being approximated towards building more complete estimations [39], or even important state information that may help other agents make decision [103].

2.4 Fairness

As with any social construct, fairness is inherently subjective [48]. As a result, its notion has been extensively studied within various social science fields such as political philosophy [71], political science [8] and in economics [57]. Most concepts that emerge from the literature can be seen as particular instances of the concept of equal treatment of equals, including:

- **Impartiality**, as the lack of prejudice in making decisions towards specific individuals or groups [73].
- **Equality and Equity**, as a special case when the population is a set of equals. Under such a case, what is left is promoting an equal distribution of some resource - Equality - or equal opportunity in obtaining it - Equity. Such a distribution is oftentimes evaluated with the use of Social Welfare Functions (SWF) [57], which measure the welfare of the population as a function of the individuals' utilities.
- **Pareto efficiency**, as the state where no individual can improve their performance without declining the performance of others.
- **Envy-Freeness**, as the perception of an individual, is that the share of resource it has been allocated is, at least, as good as the share received by any other agent [16].

These notions of fairness have been brought to applied mathematics fields. In Operations Research, fair optimisation deals with promoting equitable allocation of resources [60] in the problems of sensor placement [58], resource allocation [7], MDPs [61] and, notably, in network routing [3, 33, 60]. Regarding Artificial Intelligence, most multi-agent systems [79] work focus on resource allocation. While some work uses pre-established fairness definitions, such as envy-freeness [16], others attempt to provide the agents with some self-organisation power towards achieving such a definition [68, 67]. Instead, the latter develop either centralised [68] or interaction-based [67] mechanisms for the agents to achieving rules that establish which concept of justice, and consequently fairness. Enabling a definition of fairness to emerge from agents also requires formal models of the world and rule generation. A popular use of the notions of fairness in multi-agent systems is towards achieving increased cooperative behaviour, or at least coordinate in competitive domains when believed to be beneficial [21, 29]. Finally, there have also been considerations on the importance of a priority-awareness model regarding fairness [41], in order to entangle a priority relationship between fairness and efficiency as the goal of the system. However, there seem not to be any empirical results testing this approach.

With the increased presence of Machine Learning models in real-life decision-making situations, the notion of fairness has gained important attention in such a field. Motivated by the impact of discriminatory harms from algorithmic bias [104], the most dominant notion of fairness to be incorporated in Machine Learning is of Impartiality, under many forms [73, 89], either in Classification [25, 47], Regression, Decision-Making [37] or Clustering [17] tasks. A more recent

approach uses a Social Welfare Function (SWF) to measure the impact of using models that optimise group impartiality on within-group inequality [85]. In Reinforcement Learning, this notion of fairness has been brought to the classification task on the decision-making task of choosing which action to take, both in the multi-armed bandit context [42] and the more general single-agent case [37]. This idea has been extended to the general single-agent case by imposing the constraint of never choosing one action over another unless its long-term return of choosing the first is higher or equal to the latter. In other words, making sure that the action is chosen actually has an estimation of better value - otherwise, it is a biased decision to act on it. A worse option is never favoured over a, apparently, better one.

2.4.1 Equal Distribution

More recently, the equality notion of fairness has been brought to MARL systems with regards to distributions of quantities - resources, opportunities, rewards, etc.

A line of work focuses on equitable resource allocation [53]. Some approaches solve the problem with domain-specific knowledge towards solving a known issue in it. Applications include scheduling [100] in order to minimise traffic, IoT devices [26] in order to take into account the preferences of various users in a shared system, networks, with regards to both connectivity [15] and routing [4, 78], and even stock trading strategies for a portfolio of clients with different goals [5]. There are also approaches that consider resource allocation in a more abstract manner. [101] use the max-min egalitarian notion of social welfare, in which it is defined as the lowest utility within the system. However, it does not consider learning. On another note [19, 18], tackle the multi-armed bandit domain by introducing constraints relative to the allocation process.

A different line of work focuses on including the equality concept in the model method itself. [81] work on the multi-objective MDP problem, where the reward is a multi-dimensional vector to be optimised. The approach taken was to make use of a social welfare function [10] as a way of ensuring each of these goals is being learned in a fair way, i.e., being given approximately equal opportunities to be learned. On another note, [91] approach the multi-agent credit assignment problem using the notion of Shapley value, which approximates the impact of a single agent in a coalition of agents. Each agent then is provided with this approximation of reward and is able to estimate the return for its action - a Shapley Q-value.

Finally, there is a line of reward which focuses on equality of the rewards between the set of agents within the MARL system. Because this is the fairness problem this work focuses on, we will be dedicating the next section to it.

2.4.2 Reward Equality

In this section, we address the problem of making agents of a MA system receive an approximately equal reward. Naturally, this cannot be forced, i.e. actually providing each agent with the same reward value. Under such a setting, we would find ourselves with a credit assignment problem,

which damages the capacity of each agent being able to learn to perform well, as already seen in Section .

Under competitive domains, [34] tackle this by adding to the agents' reward value parcels that encode aversion for inequality in two forms: advantageous and disadvantageous, controlled by parameters α and β, as shown in Equation 2.25. While this work is successful in promoting cooperation in competitive environments such as social dilemmas, they do not make any considerations fairness-wise.

$$U_i(r_i,\dots r_N) = r_i - \frac{\alpha_i}{N-1}\sum_{j\neq i}\max(r_j - r_i, 0) - \frac{\beta_i}{N-1}\sum_{j\neq i}\max(r_i - r_j, 0) \tag{2.25}$$

Regarding cooperative domains, the agents are training towards a common fairness goal - equality, in this case. However, the task of learning fair policies does not get easier. Indeed, if an agent is trying to optimise a global characteristic of the system, it may be the case that what they learn does not stimulate their individual performance as the fairness goal also depends on the actions of other agents. This issue shares many similarities with the credit assignment problem. The little existent literature in this set focuses on the architectural side of the models to mitigate this issue. To the best of our knowledge, there are only two approaches that attempt to work on this problem. Both of these measure fairness according to CV, the Coefficient of Variation [38]. As shown in Equation 2.26, CV is the standard deviation between the utilities of all n agents - typically, their accumulated reward throughout time $u_i = \sum_{t=1}^{T} r_{i,t}$ for each agent i - divided by the mean utility.

$$\text{CV} = \frac{s}{\bar{u}}, \text{ where } \bar{u} = \frac{\sum_{i=1}^{n} u_i}{n}, s = \sqrt{\frac{\sum_{i=1}^{n}(u_i - \bar{u})^2}{n-1}} \tag{2.26}$$

One of them, FEN [40] is a hierarchical policy approach consisting of a controller that ensembles several sub-policies. A key aspect of this architecture is that only one of the sub-policies is trained with the typical reward signal r_t and the remaining ones exploit an information-theoretic objective to explore alternative behaviours that may be further beneficial to achieve fairness. Giving name to the model, the controller is then optimised according to a *fair-efficient* reward, given by Equation 2.27 for each agent i. The results showed this model achieved better CV in a variety of domains. An ablation study also showed the importance of FEN's architecture as it significantly improves efficiency compared to a single policy optimised with a fair-efficient reward. The average agent utility $\bar{u}$ is estimated using a gossip algorithm between neighbours.

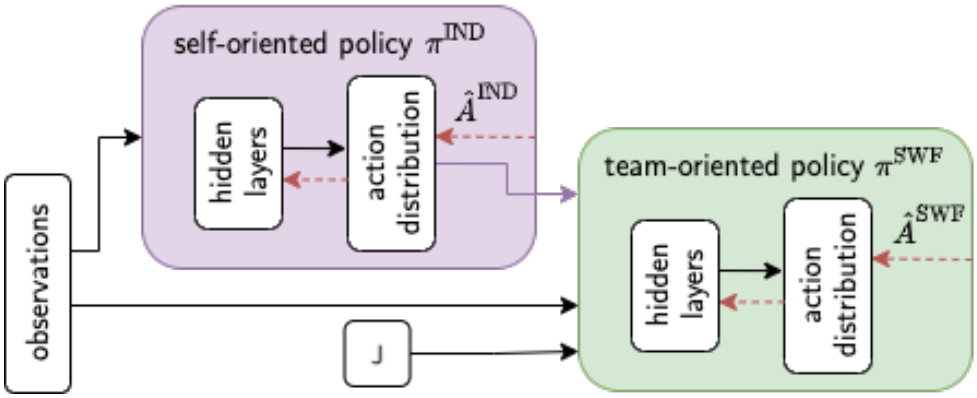
$$\hat{r}_t^i = \frac{\bar{u}_t / c}{\varepsilon + \left| u_t^i / \bar{u}_t - 1 \right|} \tag{2.27}$$

The second one, SOTO [103], is, to the best of our knowledge, state of the art in this problem. For this reason and because our work extends from it, we dedicate the next section to a more in-depth description of it.

2.4.3 SOTO

SOTO stands for Self-Oriented Team-Oriented networks. As the name indicates, its architecture comprises a Self-Oriented Policy π^{IND} and Team-Oriented one π^{SWF}, trained for the traditionally selfish and the fair goals respectively, as shown in Figure 2.2 [2], with the use of distinct advantage value estimations $\hat{\mathbf{A}}^{\text{IND}}$ and $\hat{\mathbf{A}}^{\text{SWF}}$, respectively. Red arrows represent the back-propagation process of the network. As can be seen, SOTO operates under the partially observable domain (POMDP), giving as input to the policies the observations of the world rather than the states. It is also important to note that the team-oriented policy receives two extra parameters: $\mathbf{J}$ and $\pi^{IND}(a|o)$. The first one, corresponds to the distribution of discounted accumulated rewards by each agent $J_i(\theta) = \mathbb{E}_\theta\left[\sum_t \gamma^t r_{i,t}\right]$. The reason for including this vector as a feature for π^{SWF} is to provide this policy information on the welfare of other agents before making a decision. For instance, before deciding to share resources, it would be useful for an agent to know how many resources other agents possess. The second one is the distribution of probabilities of the self-oriented policy for the next action a, given observation o. The reason for the inclusion of this distribution of probabilities has to do with the definition of fairness and of $\hat{\mathbf{A}}^{\text{SWF}}$, which is further detailed in the following paragraphs.

Figure 2.2: SOTO architecture



Regarding fairness, the authors of SOTO consider three principles: Impartiality, Equity and Pareto-efficiency [2]. In order to represent these [3], they use Social Welfare Functions that satisfy these criteria [94]. The intricacies of these criteria are as follows:

- **Impartiality** is, in a way, intrinsically met since all agents are built equally in an RL system and thus entitled to equal treatment. However, it also means that the social welfare function should output the same value independently of the order in which the agents' utilities are currently.

[2] Although not explicit in the image, for simplicity purposes, each the advantages are estimated in an actor-critic setting with the use of separate value function approximation $\hat{V}^{\text{IND}}$ and $\hat{V}^{\text{SWF}}$, for the self- and team-oriented goal respectively.

[3] Notice that, despite aiming to respect these principles, they are used to design the Social Welfare Function utilised, which is a function of the utility - reward - of the agents. As such, the problem at stake is still equality in the distribution of rewards.

- **Equity** is materialised by the Pigou-Dalton Principle [65, 20]. This principle states that for any utility vector $\mathbf{u} \in \mathbb{R}^{\mathcal{D}}$, if $u_j - u_i > \varepsilon > 0$, then $\mathbf{u} + \varepsilon \mathbf{e_i} - \varepsilon \mathbf{e_j}$, where $\mathbf{e_i}, \mathbf{e_j} \in \mathbb{R}^{\mathcal{D}}$ are the standard basis vectors on components i and j respectively [4], is considered a fairer distribution of utilities than $\mathbf{u}$. Intuitively, this states that if an agent has higher utility than another by some difference ε, then transferring some of its utility to the poorer agent is considered to be a fairer solution. While this Pigou-Dalton transfer is not enforced in any way within SOTO, the authors require that the SWF respects this principle, such that when two solutions in the learning process are encountered, the fairest is chosen. As such, by the principle's definition, ϕ must be strictly Schur-concave, i.e., strictly monotonic with respect to Pigou-Dalton transfers. An instance of these criteria is symmetric and strictly concave functions.

- **Pareto-efficiency** refers to the concept of pareto dominance i.e. for any pair of utility vectors $(\mathbf{u}, \mathbf{u}') \in \mathbb{R}^{\mathcal{D} \times \mathcal{D}}$, $\mathbf{u}$ Pareto-dominates $\mathbf{u}'$ (denoted $\mathbf{u} \succ \mathbf{u}'$) if $\forall i, u_i \geq u'_i$ and $\exists j, u_j > u'_j$. In other words, that a utility vector needs to be either weakly or strictly preferred by agents in order to be chosen as solution. Regarding a social welfare function $\phi(\mathbf{u})$, it means that it should be strictly monotonic with regards to Pareto-efficiency, i.e. $\mathbf{u} \succ \mathbf{u}' \Rightarrow \phi(\mathbf{u}) > \phi(\mathbf{u}')$. Although this principle may be harder to understand as a fairness concept, it could be interpreted as a guarantee that independently of the social welfare function chosen, solutions that are weakly more efficient solutions will always have higher social welfare value. In other words, the social welfare function will not prevent agents from choosing more efficient solutions - otherwise, it would be unfair.

Respecting these criteria, two families of Social Welfare Functions are considered for the set $\mathcal{D}$ of agents. The first family is the Generalised Gini Function [95] (GGF), depicted in Equation 2.28, which is a linear combination of the agents utilities. Under this definition $\mathbf{w} \in [0,1]^{\mathcal{D}}$ is a fixed strictly decreasing weight vector (i.e., $w_1 > w_2 > \ldots > w_{\mathcal{D}}$) and $\mathbf{u}^{\uparrow}$ is the vector agents utilities $\mathbf{u}$ sorted in decreasing order. Instances of this family include the utilitarian notion - $\forall i, w_i = 1$ - and the maxmin egalitarian approach - $w_1 = 1, w_2 = \ldots = w_{\mathcal{D}} = 0$. The second family is depicted in Equation 2.29. It is the summation of the transformation of the agents utilities by a strictly decreasing concave function $U : \mathbb{R} \to \mathbb{R}$. Instances of this function include proportional fairness [66], the generalised entropy index [80], and, the one utilised by the authors, α-fairness [56] where $U_\alpha(x) = \frac{x^{1-\alpha}}{1-\alpha}$ if $\alpha \neq 1$ and $U_\alpha(x) = \log(x)$ otherwise. The $\alpha \in \mathbb{R}^+$ parameter controls the aversion to inequality.

$$G_{\mathbf{w}}(\mathbf{u}) = \sum_{k \in [\mathcal{D}]} w_k u_k^{\uparrow} \qquad (2.28) \qquad S_U(\mathbf{u}) = \sum_{k \in [\mathcal{D}]} U(u_k) \qquad (2.29)$$

Independently of the social welfare function utilised, while $\hat{\mathbf{A}}^{\text{IND}}$ is identical to the typical advantage definition, the definition of the advantage $\hat{\mathbf{A}}^{\text{SWF}}$ is a function of the social welfare function $\phi(\mathbf{u})$ utilised as portrayed in Equations 2.30 and 2.31 respectively.

[4] Null vectors except in component i and j where they are 1

$$\hat{\mathbf{A}}_i^{\text{IND}} = \hat{\mathbf{A}}_{I_i}(o_i, a_i) \quad (2.30) \qquad \hat{\mathbf{A}}^{\text{SWF}} = \nabla_{\mathbf{u}}\phi(\hat{\mathbf{J}}(\theta))^{\top} \cdot \hat{\mathbf{A}}(\mathbf{o}, \mathbf{a}) \quad (2.31)$$

The team-oriented advantage is the dot product between the gradient of the SWF with regards to the agents utilities and the global advantage of the system. The gradient term, after derivation, is then equal to:

- $\nabla_{\mathbf{u}} G_{\mathbf{w}}(\mathbf{J}(\theta)) = \mathbf{w}_{\sigma}$, where σ is a permutation that sorts $\mathbf{J}$ in an increasing order, as the definition of this SWF family requires.
- $\nabla_{\mathbf{u}} S_{U_\alpha}(\mathbf{J}(\theta)) = \mathbf{J}(\theta)^{-\alpha}$, for $\alpha \in [0, 1[$, which is the setting in consideration.

Since this method is fully decentralised, the agents resort to communication between neighbours in order to estimate J. As a consequence, an agent i may receive conflicting information about the quality of π^{SWF}, making the learning of a policy under the advantage $\hat{A}^{\text{SWF}}$ potentially unstable. Furthermore, the value of a Social Welfare function is by definition the same for all the agents within the system - or, in this case, tends to be the same as the number of neighbours increases. These challenges motivated the authors to include a self-oriented policy in the architecture, providing an insight to the team-oriented policy on whether its successes/failures are due to the individual behaviour of the agent or to behaviours of neighbours. The distribution of probabilities estimated by such policy is provided as input to the team-oriented policy, serving as an individualistic recommendation of how to act.

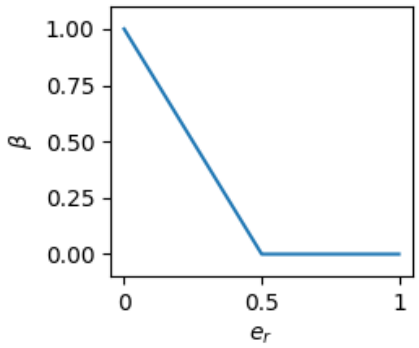
The dynamics of the SOTO training procedure of this architecture is fully detailed in Algorithm 1. At each batch of steps, throughout episodes, a policy is chosen according to the value of the β parameter. The agents act according to such policy for the length of the batch in time steps and, by the end of it, updates the corresponding policy's weights. As such, the training of each policy depends on the evolution of β throughout episodes $\beta(e_r)$, where $e_r = \frac{e}{E}$ is the ratio of episodes that have already occurred, i.e. the number of the current episode e divided by the total number of episodes E. The function chosen by the authors is $\beta(e_r) = \max(1 - 2e_r, 0)$, where e is the episode number and E is the number of the last episode. As visually illustrated in Figure 2.3, under this $\beta(e_r)$ function, the value of β starts as 1, decreases to 0 until it reaches half of the episodes designated for train and stabilises on 0 until the end of the training process.

2.5 Summary

In the multi-agent domain, there is work approaching important threats to efficiency such as non-stationarity, partial-observability and decentralisation. In the fairness domain, it is clear that it is still a novel thread of research appearing in Machine Learning, and is especially understudied in the MARL paradigm for equitable distributions of rewards between agents.

The concept of equality is a characteristic of the whole system and not tied to the particular performance of individual agents. As such, this goal may be hard to learn without falling into the credit assignment problem. There are two solutions in the literature that approach this problem -

Figure 2.3: β in function of relative episode number



FEN and SOTO. Both tackle this problem by making agents learn a set of policies with different goals, instead of a single policy, to not lose sight of what good individual behaviour looks like.

However, there seems to be a gap in the literature. Attempting to learn to behave fairly may undermine the efficiency of the agents. This may explain why the goal of fairness and efficiency are, to the best of our knowledge, never considered altogether. With this underlying assumption, there seems to be a lack of studies that consider these goals holistically and present solutions in-between them. In a way, it seems that models are either built wholeheartedly for fairness or for efficiency.

Algorithm 1 SOTO

Initialize $\pi_i^{\text{SWF}}, \pi_i^{\text{IND}}, v_i^{\text{SWF}}, v_i^{\text{IND}}$, respectively the team-oriented/self-oriented policies, team-oriented/self-oriented critics.
for each episode e **do**
  $\beta = \max(1-2e_r, 0)$, where $e_r = \frac{e}{E}$
  for each agent i **do**
    Initialize $J_{I_i} = \mathbf{0}$
    $(\pi_i, v_i) \leftarrow \begin{cases} \left(\pi_i^{\text{IND}}, v_i^{\text{IND}}\right) \text{ with probability } \beta \\ \left(\pi_i^{\text{SWF}}, v_i^{\text{SWF}}\right) \text{ otherwise} \end{cases}$
  end for
  while episode e is not completed **do**
    Collect M a minibatch of transitions with μ while updating and sharing
    for each agent i **do**
      Update v_i with TD(λ) on M
      Compute $\hat{A}_{I_i}(o_i, a_i)$ on M with v_i and TD(λ) and send it to everyone
      if $\pi_i = \pi_i^{\text{IND}}$ **then**
        Update π_i^{IND} with $\hat{A}_i^{\text{IND}}$
      else
        Collect and form $\hat{A}(o, a)$
        Update π_i^{SWF} with $\hat{A}^{\text{SWF}}$
      end if
      $(\pi_i, v_i) \leftarrow \begin{cases} \left(\pi_i^{\text{IND}}, v_i^{\text{IND}}\right) \text{ with probability } \beta \\ \left(\pi_i^{\text{SWF}}, v_i^{\text{SWF}}\right) \text{ otherwise} \end{cases}$
    end for
  end while
end for

Chapter 3

Methodological Approach

In this chapter, we describe the methodological approach devised for this dissertation. In order to understand the problem at stake and the methods employed, the reader must be familiar with SOTO [103], the state of the art fairness model, before reading this chapter. We defer the reader to Section 2.4.3 if this is not the case.

We start by describing the problem we are studying, together with the questions we aim to answer. Then, we move to the description of our methodology. We employ three methods: heterogeneous testing, changes in SOTO's original training strategy and an architecture adaptation for intertwined predictions - I-SOTO - each described in a separate section. In the last section, there are general considerations on the performance of these models.

3.1 Problem Statement

Reinforcement Learning agents learn to optimise the return g_t they expect to get from their behaviour. Their ultimate goal is to behave in the way that provides them with the most favourable outcomes, in the form of a reward. If the fairness concept of Equality is considered, it brings additional challenges to this process. In particular, learning to be fair implies that the evaluation of an agent's behaviour depends on the outcomes of the actions of the other agents within the system. Intuitively, it is hard to visualise what fair behaviour is, especially if the agents are placed in a decentralised setting where they decide based on partial observation of the world.

Consequently, in MADRL literature, fairness is approached as a goal that needs to be carefully learned to prevent the agents from not being efficient at all. In particular, SOTO, the state of the art fairness model, employs an architecture that uses a self-oriented policy towards attempting to help the fairness-oriented policy not forget what an efficient behaviour looks like.

While this approach makes logical sense towards finding the most fair-efficient policies, there is, to the best of our knowledge, a lack in research on exploring what considering these two goals

independently but simultaneously actually looks like. In real applications, a system designer may prefer an efficient solution that is as fair as possible. Moreover, it is still unclear on what sort of impact learning fair and efficient behaviours have on the final result that each of these ends up learning. Motivated by this problem, we aim to answer the following questions:

1. **Can a range of behaviours be generated between two trained policies?** Given any two MARL trained sets of policies, it is possible to mix the policies being utilised, making agents act heterogeneously. Is this range of performances linear both in the fairness and efficiency dimensions? Are its extremes the ones that provide better system's performance in their respective dimension?

2. **In the SOTO architecture, can different $\beta(e_r)$ training strategies generate solutions that Pareto-front the original model or selfish baseline?** The role of β in SOTO is functional - it is the probability of agents choosing to act under and train the self-oriented policy instead of the team-oriented one. If β is high, the behaviour of agents is then more selfish, and the self-oriented policy is trained more often. The reverse happens when β is low. It is also important to note that, while the authors provided ablation studies to justify this option, they only tested a few options. On the one hand, they tested pre-training π^{IND} for 20% and 50% of the episodes and concluded it only delayed the convergence process. They also tested a constant $\beta = 0$ (equivalent to having π^{IND} always a randomly initiated network). They showed the information provided by a somewhat trained π^{IND} was significantly important to the optimisation of π^{SWF}. This evidence goes to show that the definition of this function is important. We aim to see if we can find settings that can generate solutions that Pareto-front SOTO or even the independent baseline.

3. **In such a model, what happens if the predictions of the team-oriented policy are also provided to the self-oriented one?** More specifically, we want to know whether this new architecture generate solutions that Pareto-front the original model or selfish baseline. The inclusion of insights from π^{IND} as input π^{SWF} is for efficiency purposes, specially for agents that are under-performing. But what if these insights were mutual, and the self-oriented policy also received fair insights from the team-oriented policy? Is it possible that the inverse could be beneficial? Are there any improvements in either fairness or efficiency? Does this architecture function better under a particular β training strategy?

3.2 Testing Heterogeneous Behaviour

Throughout our experiments, we repeatedly apply this testing technique of heterogeneous behaviour. Inspired by SOTO's learning algorithm, it enables agents to act either selfishly or fairly according to β. As depicted in Algorithm 2, β a fixed parameter is provided as an argument that determines the probability of the agent being attributed the self-oriented policy. The agents act for a mini-batch of M transitions, and its policy is changed again according to the same β criteria.

It is important that under this testing method, no agent acts only selfishly or fairly - the intent is to expose the behaviour of interactions of each policy kind for each agent, as each trained multi-agent architecture trains a policy per agent. Moreover, notice that the policies are never updated under this method. As such, this is only a testing method, putting in evidence the policy resultant from training without changes.

Algorithm 2 Heterogeneous Policy Testing

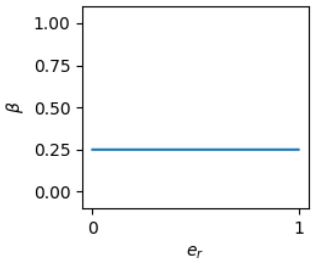
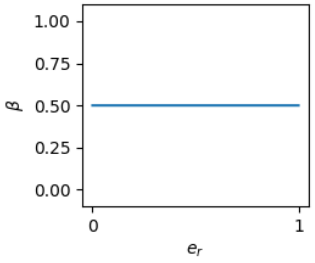
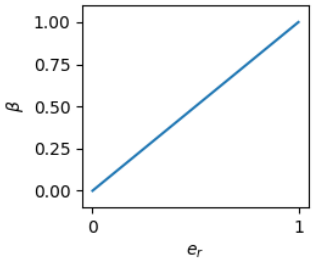
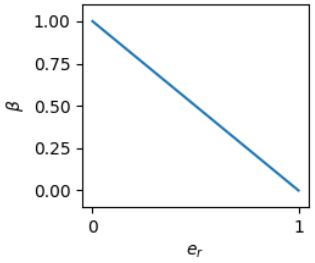
Initialize $\pi_i^{\text{SWF}}, \pi_i^{\text{IND}}, v_i^{\text{SWF}}, v_i^{\text{IND}}$, respectively the pre-trained team-oriented/self-oriented policies, team-oriented/self-oriented critics.
for each episode e **do**
  for each agent i **do**
    Initialize $J_{I_i} = \mathbf{0}$
  end for
  while episode e is not completed **do**
    $(\pi_i, v_i) \leftarrow \begin{cases} (\pi_i^{\text{IND}}, v_i^{\text{IND}}) & \text{with probability } \beta \\ (\pi_i^{\text{SWF}}, v_i^{\text{SWF}}) & \text{otherwise} \end{cases}$
    Collect M a minibatch of transitions with π_i
  end while
end for

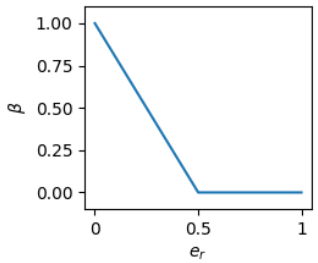
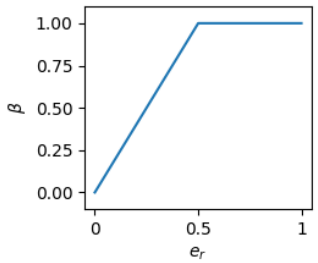
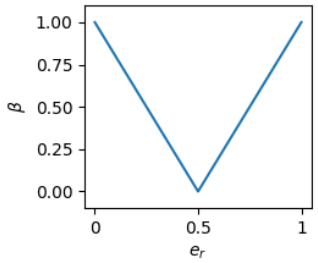
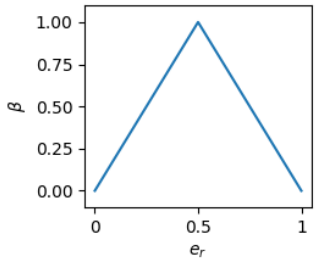
3.3 Training strategy functions

In the context of this work, training strategy refers to the function of $\beta(e_r)$, which determines the probability of each agent choosing to act under the self-oriented policy π^{IND} on a given episode for and on before mini-batch of M transitions. Higher β makes the agents train more the self-oriented policy and then vice-versa for the team-oriented policy. We test a variety of functions $\beta(e_r)$, where $e_r = \frac{e}{E}$ is the episode rate, i.e. the number of the current episode e divided by the total number of episodes E. In the original training setting of SOTO, $\beta(e_r)$ is present visually in Figure 2.3 and formally in its caption. It is a linearly decreasing function until half of the episodes $\frac{1}{e}e_r$ and constant from such point on-wards on 0.

Our methodology tests 4 different beta families: constant, linear, baseline reverse and v-shaped. A summary of all of the functions and respective variants is depicted in Table 3.1. Each family of functions is then described in separate subsections.

Table 3.1: Strategy Functions by family: Constant, Linear, Baseline and V-shaped. Variants within these families are given specific labels. For the Constant family this label is the constant itself. For the remaining families it is lin, b and v, respectively. The prefix "r" is added to denote the reverse version of such family. The $\pi^{\text{IND}}/\pi^{\text{SWF}}$ sample ratio is the percentage of area under and above the function line, since β constrols the samples dedicate to π^{IND}

Family	Variant	π^{IND}/π^{SWF} sample ratio	Equation
Constant	0.25	25/75	$\beta = 0.25$ 
	0.5	50/50	$\beta = 0.5$ 
Linear	lin	50/50	$\beta(e_r) = e_r$ 
	rlin	50/50	$\beta(e_r) = 1 - e_r$ 

Baseline	b	25/75	$\beta(e_r) = \max(1-2e_r, 0)$ 
	rb	75/25	$\beta(e_r) = 1-\max(1-2e_r, 0)$ 
V-shaped	v	50/50	$\beta(e_r) = \max(1-2e_r, 2e_r-1)$ 
	rv	50/50	$\beta(e_r) = 1-\max(1-2e_r, 2e_r-1)$ 

Constant

This is the most elementary family of β functions, which is a constant function. Under this family, β is not dependant on e_r, and the agents train selfishly/fairly according to the same probability

throughout episodes. SOTO performs an ablation study where $\beta = 0$, which is indeed an instance of this family. However, we aim to test with values greater than 0 to expose both policies' training simultaneously.

In particular, we want to study two values of β: 0.25 and 0.5. Studying $\beta = 0.5$ is important as it gives the same opportunity for each policy π^{IND} and π^{SWF} to converge. On the other hand, studying $\beta = 0.25$ is also interesting since it provides the same area below the curve as the baseline. This implies that the number of mini-batches when the agents act under each policy is approximately the same, by the law of large numbers.

Linear

Linear alternatives prolong the switch between choosing one or the other policy to double the period compared to the baseline setting. We aim to test whether the stabilising period of baseline after $\frac{T}{2}$ is necessary or if most of its performance gains come from slowly switching from self-oriented to team-oriented. In a way, this family of functions is an intermediate between baseline and linear in the sense that it is not constant but has a lower slope than the line of baseline until $e_r = 0.5$. An important thing to note is that is alternative gives 50% of samples to each policy.

Baseline

This family comprises the baseline setting of SOTO and its reverse. Notice that the reverse variant ends up being very similar to the independent baseline except that 25% of the samples are, in the first half of episodes, directed to the team-oriented policy during. The test will make evident how a later train of the self-oriented policy makes changes on either policies. Moreover, it is a mirror of the baseline function utilised in SOTO, so it is important to include as an ablation study.

V-shaped

Derived from the baseline alternative, v-shaped functions are intended to provide the same function as baseline until $\frac{E}{2}$ episodes and then return to the initial value linearly in a V-shaped manner. The interest of this test relies on providing a view on, should the first episodes look similar to baseline alternatives, if the policies' number of training samples is compensated at the end to reach 50% on each if both policies end up performing better, or if the team-oriented policy performance is harmed since from a theoretical point of view - as previously seen - it is more prone to divergence.

3.4 Intertwining self- and team-oriented policies

In the SOTO architecture, the inclusion of a self-oriented policy is intended to provide a recommendation on how to act efficiently to the team-oriented one. However, it is still unknown if the recommendations of a team-oriented policy could improve the performance of the first as well.

We propose a new architecture in which the self-oriented policy also receives insights from the team-oriented policy, generating an intertwined sharing of recommendations, which we name as

Intertwined Self-Oriented Team-Oriented policy - I-SOTO. The team-oriented policy then receives as input the action distribution resultant from forwarding π^{IND}, as shown in Figure 3.1.

One problem that arises from this model is the dependency of each policy on the other in order to provide forwarded action distributions. Indeed, if we want to get the action of policy π^{SWF}, we need to first forward the policy π^{IND}. However, to forward this policy, we need the forward output of the first, which is dependent on the latter. In order to address this, whenever some policy π is being used, it forwards the other π' substituting the expected inputs from π with a null vector $\mathbf{0}^{|\mathcal{A}|}$.

Note that the distribution of wealth $\mathbf{J}$ is still not passed to the self-oriented policy. Indeed, we aim to isolate the effect of this intertwined sharing of action recommendations.

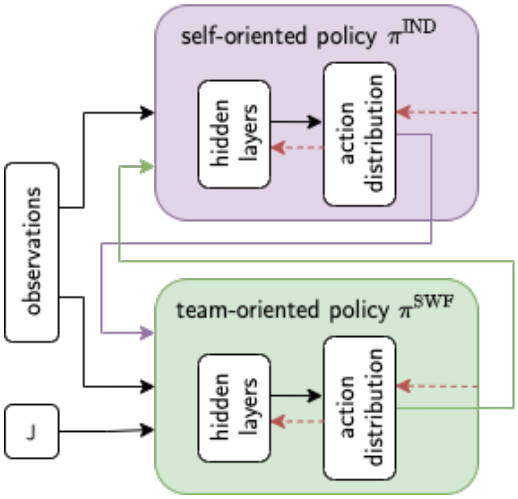


Figure 3.1: Intertwined-SOTO architecture

3.5 Evaluation

The evaluation of any of these settings is done through simulation. The environment of the simulation and the reward it provides drive the agents to do better next time—as such, choosing appropriate environments is crucial, especially in the fairness domain. We choose two environments in which is the resource opportunity is unequal - Matthew Effect and Traffic Light Control. In the first, the agents are placed on a map and receive rewards for each consumption. However, after consumption, they also get a bigger size and higher speed. This dynamic catalyses the effect of those who are "rich" become richer, and those who are "poor" become poorer - the Matthew Effect. On the other hand, Traffic Light Control is an environment where multiple intersections have to decide which light state is most beneficial to reduce the waiting time. However, because some intersections lead to others, there is a high dependency on the rewards obtained within the agents of the system, which translates into an unequal opportunity of performing well. The common aspect of these environments is this tendency for inequality, which is the desired challenge for the methods to be tested.

As baselines, we use the original SOTO model and an Independent baseline, which has the same architecture as SOTO's self-oriented policy for comparative reasons. For each environment, we first run these baseline models to ensure that the results are coherent with those reported by the authors of SOTO. Then, if so, we proceed to test our methods.

Running a model comprises two steps: training and testing. Training is a longer process in which the agents update their policies weights. Testing is simply executing the policies obtained from training. In each environment, a domain-specific measure per time step and agent m_t^i is considered and four types of metrics per episode are considered:

- **Total** which sums this measure for all time steps and agent within the system $\sum_t^T \sum_i^{|\mathcal{D}|} m_t^i$. This metric represents efficiency.
- **CV**, the Coefficient of Variation, which is the standard deviation between the total reward received by all agents, devised by the mean total reward per agent, as previously shown in Equation 2.26 but having $u_i = \sum_t^T m_t^i$. This metric represents fairness. The lower the CV, the fairer the system is.
- **Min** as the minimum accumulated reward in the system, i.e. $\min(\mathbf{u}), u = \{u_i, i \in \mathcal{D}\}, u_i = \sum_t^T m_t^i$.
- **Max** as the maximum accumulated reward in the system, i.e. $\max(\mathbf{u}), u = \{u_i, i \in \mathcal{D}\}, u_i = \sum_t^T m_t^i$.

3.6 Summary

Reinforcement Learning models traditionally optimise the expected outcome of their actions, which is seen as an efficiency goal. More recently, models that approach the equality concept of fairness have appeared in MADRL, bringing about other performance measures rather than solely efficiency. However, there seems to be a gap in the literature devoid of assumptions on which of these goals is ultimately intended for the system.

Our methodology approaches the problem of exploring fair and efficient behaviours in an independent but simultaneous manner. Although the equality concept of fairness has been tackled in the literature, it is approached as a goal that requires careful training and never – to the best of our knowledge – in a joint way with the efficiency goal.

We employ three different methods to approach this problem. The first consists in mixing two policies, which we refer to as Heterogeneous behaviour testing. The second two are changes in the model. First, we change the strategy for training the self- and team-oriented policies of SOTO. Then, we propose a novel architecture in which the team-oriented policy also recommends a probability distribution of actions to the self-oriented policy.

All of these models are trained and tested to be evaluated in efficiency and fairness (CV). We utilise environments that intrinsically do not provide agents the same access to resources,

catalysing inequality phenomena. The question we aim to answer can be generally summarised as searching for solutions that Pareto-front SOTO or the independent baseline in the efficiency-fairness space.

Chapter 4

Experimental Setup and Result Analysis

This chapter describes the settings utilised in our set of experiments and provides the results obtained together with its analysis. We start by describing the core characteristics of all experiments and then dedicate a separate chapter for each environment tested. We first present the baseline results for each environment and then show the results of our methods. Finally, the last section summarises the main experimental outcomes.

4.1 Core Setup

The experiments provided share with a core of setup settings. For comparative purposes, this core setup is identical to the one reported by SOTO. For the same reason, the development of the proposed methods was carried out by extending the original code base of SOTO [1] and can be found at https://github.com/MargaridaSilva/Dissertation-Experimenting.

All policies use PPO optimisation. The importance sampling has a 0.03 exploration bonus and 0.1 clipping ratio. The learning rate utilised was 10^{-3} for the critic and 2.5^{-3} for the actor. Generalised Advantage Estimation was utilised with $\lambda = 0.97$. The neural networks have two hidden layers with 256 ReLU units each. We used 50 time step batches of transitions. As for the social welfare functions, we utilised an instance of the Generalised Gini Function, with $w_i = \frac{1}{2^i}$ and an instance of α-fairness with $\alpha = 0.9$

All of the results presented are the average from the execution of 50 episodes. When applicable, the values of β used for heterogeneous behaviour were $\{0.02i, \forall i \in \{0, 1, ...50\}\}$.

4.2 Matthew Effect

In the Matthew Effect environment [40], agents are initialised with different property values in position, size and speed and coexist in a map throughout the simulation. Three stationary ghosts

[1] https://github.com/matthieu637/DFRL

are also put in the grid. The aspect of this environment and its elements, as rendered from the code base, is illustrated in Figure 4.1.

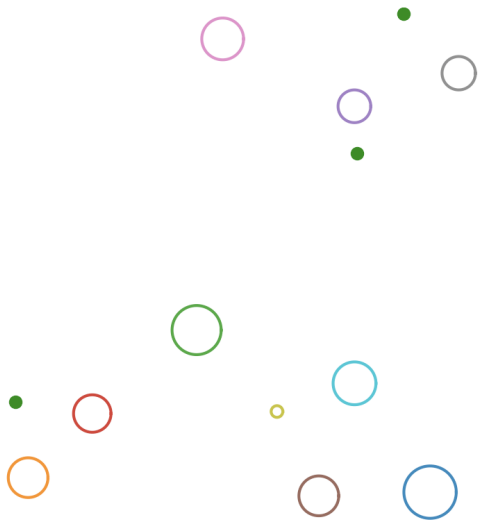


Figure 4.1: Matthew Effect Environment

The dynamics between agents are as follows. Each agent can only observe the nearest three other agents and the nearest ghost. Available actions include moving to one of the four directions or stay in the same position. If a ghost is next to an agent, the ghost is consumed, and the agent gets a reward of 1. A new ghost is then generated and attributed to a random position. As such, the relationship between the number of ghosts consumed by agent i at time step t n_t^i, the reward r_t^i and the evaluation measure m_t^i for each agent i is as defined in Equation 4.1.

$$m_t^i = n_t^i = r_t^i, \text{ referred to as income} \tag{4.1}$$

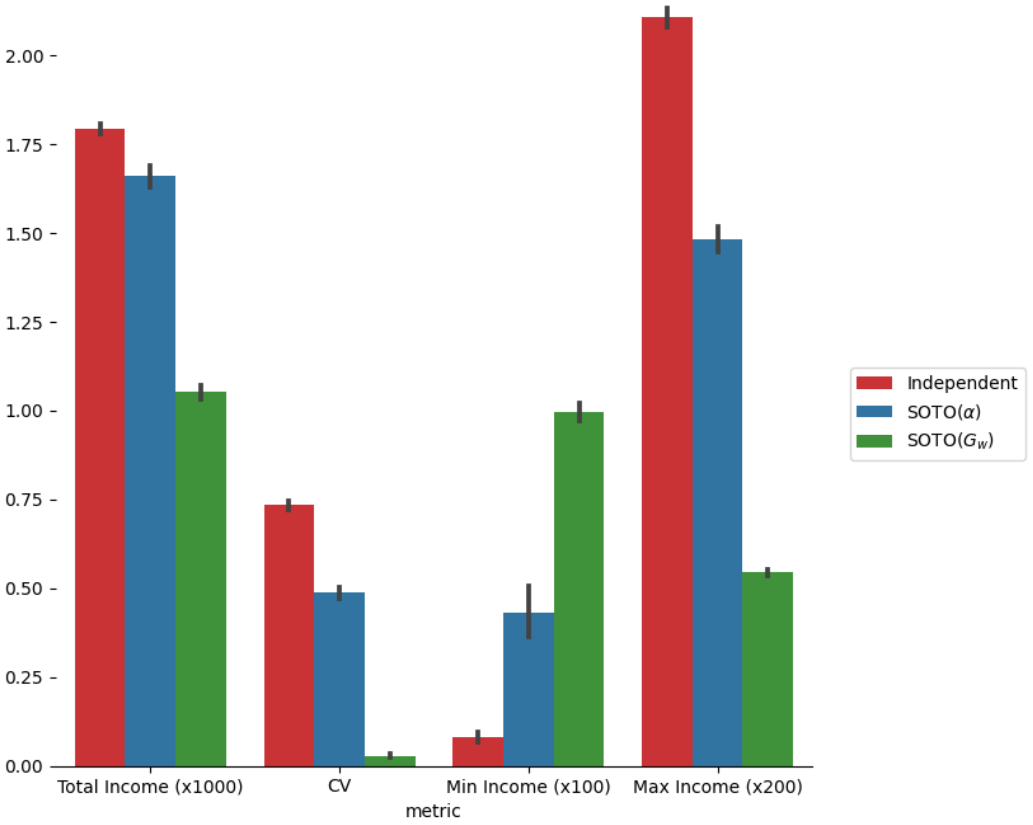
Whenever an agent consumes a ghost, it receives a reward and its size and speed will increase correspondingly until the pre-established upper bounds are reached. This dynamic makes it prone that agents who have already consume will become more likely to consume again since they have been provisioned with faster movement capabilities - while others will be in advantage. This effect where the rich get richer, and the poor get poorer is what gives name to the environment - The Matthew Effect.

In the experiments of this environment, we consider ten agents, 10000 episodes, 1000 steps per episode and a discount factor γ of 0.98.

4.2.1 Baseline Performance

The performance of SOTO and Independent baseline is depicted in Figure 4.2. These results show that the Independent baseline is the most efficient, while the SOTO models are the fairest. While SOTO(G_w) has a CV very close to 0, it is also approximately 1/3 less efficient. A side by side

Figure 4.2: Baseline performance in Matthew Effect.



display of our results with those reported by the authors of SOTO can be found in the Section A.1. The results agree with the ones reported by the authors of SOTO, so we decided to carry on with our set of experiments using them as baselines.

4.2.2 Heterogeneous Behaviour

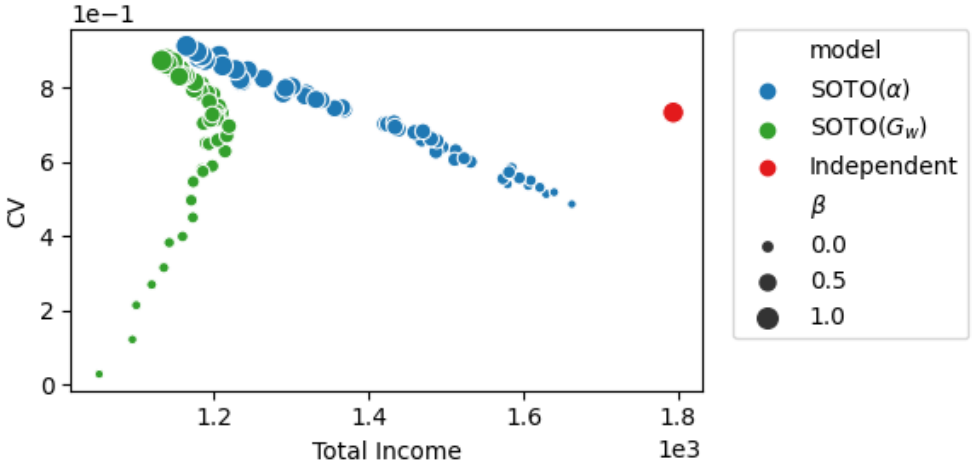
The first set of experiments carried out is the testing of heterogeneous behaviour within different policies.

Self- and Team- Oriented Policy

SOTO trains two policies with different aims: a self- and a team-oriented goal. The results of the heterogeneous behaviour produced by these policies in the Matthew Effect environment is depicted in Figure 4.3. It seems that the two SWFs utilised can generate ranges of behaviour, according to β, in two different directions.

Regarding fairness, as expected, the lower values of β seem to be associated with lower CV values. This means, the higher the probability of each agent to act under π^{SWF}, the fairer the system is, globally. On the other hand, with regards to efficiency, a more complex scenario occurs.

Figure 4.3: Heterogeneous Behaviour between SOTO self-(π^{IND}) and team-oriented (π^{IND}) policies in Matthew Effect. Higher diameter points correspond to higher probability of acting under π^{IND}.



Contrary to our expectations, for SOTO(α), there seems to be an inverse relationship between β and the value of total income. Indeed, for this metric, the most efficient policy is also the fairest. As for SOTO(G_w), such a relationship is no longer linear. The most efficient policy is an intermediate behaviour between the two extremes π^{IND} and π^{SWF}.

Overall, it is possible to conclude that these two SWFs have quite distinct behaviours under this environment. However, a similarity between them is the proximity in performance between self-oriented extremes in each SWF. We believe this may be because such policy only trains for 25% of the training samples. This would also justify why the efficiency of such policy seems mediocre, despite its training goal being directly oriented towards efficiency.

SOTO and Independent Baseline

As previously seen, under this environment, the performance of π^{IND} is mediocre when compared to π^{SWF}. As such, the following set of experiments include attempts in improving the efficiency of the first, with hopes of observing potential improvements in the latter. In other words, we aim to observe whether a more efficient self-oriented policy leads to a more efficient and/or fair policy.

The first of these attempts is the production of heterogeneous behaviour between SOTO's π^{SWF} and the Independent Baseline. The results of this attempt are depicted in Figure 4.4. As can be seen, although a range of in-between solutions is formed, none of them Pareto-fronts the fair and efficient extremes.

The second attempt consisted in setting the weights of π^{IND} as the weights of the Independent baseline $\theta^{\text{IND}} = \theta^{\text{Independant}}$ and then observing the heterogeneous behaviour formed. Notice that this attribution of weights is only possible since the architectures of both policies are identical, and thus there is exactly one neuron to place each weight in θ^{Ind} to SOTO. Had we used any other architecture, concerns with loss of information would have to be considered.

Figure 4.4: Heterogeneous Behaviour between SOTO and the independent baseline in Matthew Effect

This behaviour, labeled as SOTO($\alpha, \theta^{\text{Ind}}$) and SOTO($G_w, \theta^{\text{Ind}}$) for the corresponding SWFs is illustrated in Figure 4.5. As expected, instances with larger values of β got performance results closer to the Independent baseline. However, for the other extreme, efficiency abruptly decreased. This drastic change in performance is interesting to take intuitions from these models. One of them could be that, whatever π^{IND} ends up learning by the end of the training process, it is certainly something very different from a purely selfish insight, which $\pi^{\text{Independant}}$ gives. Moreover, it is also possible to conclude that a trained SOTO model does not get Pareto-front solutions by simply plugging in a trained Independent model.

In order to further extend this line of thought, we also tested a version of SOTO initialised with $\theta^{\text{Independant}}$ but trained for E episodes. This version is entitled SOTO($\alpha, \theta^{\text{Ind}})_{2E}$ and SOTO($G_w, \theta^{\text{Ind}})_{2E}$ respectively since, in theory, they take twice the amount of episodes to be trained than SOTO - E episodes for the training of SOTO plus E episodes for the training of SOTO($\theta^{\text{Ind}})_{2E}$. For comparison sake, we also include a version of the Independent baseline trained in 2E episodes, Independent$_2E$. Another way to interpret the question at stake in this experiment would be: given an already trained Independent model, what is the difference in performance between continuing to train for E episodes or integrating it into the SOTO architecture to train it? As can be seen from the results presented in Figure 4.5, it seems that this training brings π^{SWF} towards a more efficient performance. In particular, for α-fairness, it seems to be producing on the team-oriented end an as efficient but significantly fairer performance. Under this metric, it seems that using SOTO leads to improving fairness while not decreasing much efficiency. The great drawback of this solution is that it takes twice the amount of episodes and is thus not a proper alternative to the Independent baseline. It is also important to note that, should the user choose to keep training this baseline within the same model, they would obtain even higher efficiency and a slight improvement in fairness. In the end, it is up to the user or system designer to opt which alternative is the most appropriate from a personal perspective.

Figure 4.5: Heterogeneous Behaviour between $\pi^{IND}(\theta^{\text{Ind}})$ and π^{SWF} of SOTO in Matthew Effect

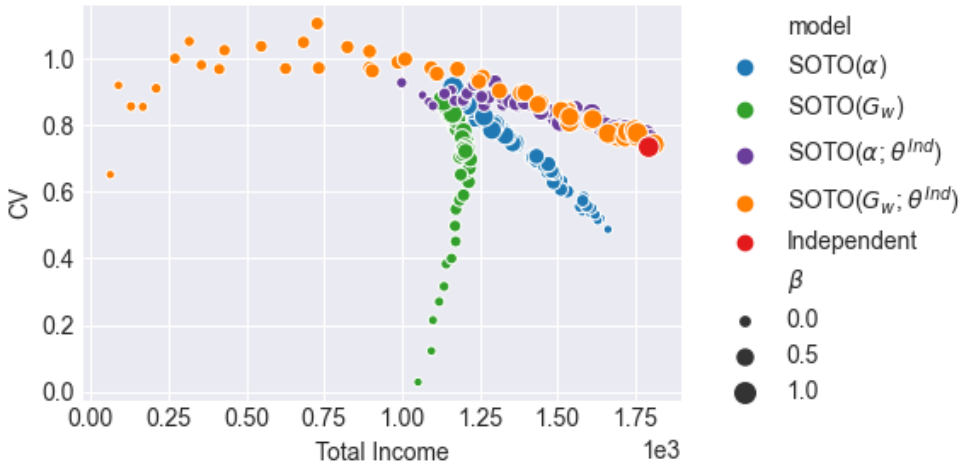


Figure 4.6: Heterogeneous Behaviour between $\pi^{IND}(\theta^{\text{Ind}})$ and π^{SWF} of SOTO trained with additional E episodes in Matthew Effect

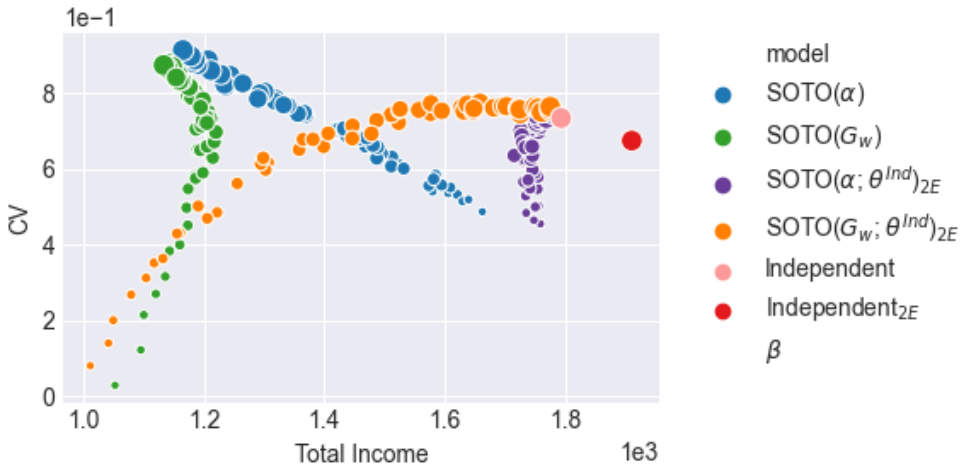
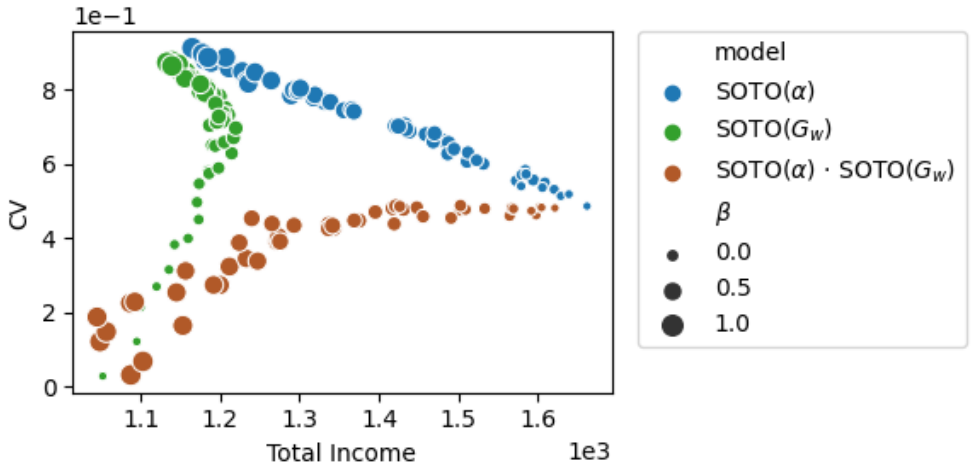


Figure 4.7: Heterogeneous Behaviour between SOTO(α) and SOTO(G_w) in Matthew Effect



SOTO(α) and SOTO(G_w)

This test, compared to the previous, aims to assess a different range of behaviours. Instead of exploring ranges between a competitive and a cooperative end, in this experiment, we test whether the two team-oriented policies under different SWFs together produce a Pareto-from solution. As shown in Figure 4.7, it is possible to conclude that it was not the case. Indeed, the range produced is almost a linear trade-off in performance between the two fair extremes, thus not producing any fairer and/or efficient solution without decreasing one of these goals.

To conclude, this initial set of experiments was important to observe that, in this environment, the behaviours generated between pre-trained policies do not seem to generate Pareto-front solutions compared to the original policies utilised to generate them. The only solutions that achieved this result were trained for double the number of episodes and thus can't be considered as proper alternatives. As such, changes in the training method of SOTO are applied in the following sections towards finding such solutions.

4.2.3 Training Strategy

The next set of experiments employed is the test of different training strategies for the SOTO model. These strategies encode, as previously seen, which policy is more likely to be trained for each agent throughout episodes. The results for each of each strategy under each SWF are depicted in Table 4.1. The set of ranges generated from this transformation can be found in Figure A.2.

With regards to income, it seems that none of the SWF alternatives can surpass the respective SWF baseline. The IND alternatives, naturally, are the ones that have the highest incomes. As for fairness, the constant function of 0.25 achieved a slightly better fairness value in 0.02 CV. However, this little improvement also leads to a decrease of $\approx$ 82 income units, which is around 5% of loss from the original value.

Table 4.1: Training strategies (β) performance under the self- (IND) and team-oriented (SWF) policies in Matthew Effect. **Bold** values are entries which perform equal or better than the baseline (b) of the respective model, in Total Income (higher) or CV (lower), i.e. efficiency or fairness.

model	β	π	Total Income	CV	Min Income	Max Income
SOTO(α)	0.25	IND	1464	0.73	12.76	369
		SWF	1581	**0.47**	43.16	280
	0.5	IND	1654	0.65	24.16	392
		SWF	1452	0.50	27.31	257
	b	IND	1178	0.90	5.88	342
		SWF	**1663**	**0.49**	43.29	297
	rb	IND	**1786**	0.72	5.18	431
		SWF	41	1.26	0.06	15
	lin	IND	**1746**	0.66	22.60	410
		SWF	1014	0.68	7.50	193
	rlin	IND	1480	0.75	9.21	367
		SWF	1628	0.51	29.96	294
	v	IND	1638	0.70	18.32	398
		SWF	1155	0.61	14.01	229
	rv	IND	1660	0.66	24.88	391
		SWF	1268	0.65	26.52	242
SOTO(G_w)	0.25	IND	**1135**	0.89	6.11	326
		SWF	800	0.22	44.86	95
	0.5	IND	**1457**	0.72	16.97	363
		SWF	371	0.59	10.30	63
	lin	IND	**1719**	0.70	16.65	416
		SWF	18	1.10	0.04	5
	rlin	IND	**1188**	0.85	7.58	327
		SWF	480	0.43	18.71	68
	b	IND	**1139**	0.86	9.00	324
		SWF	**1052**	**0.03**	99.68	109
	rb	IND	**1724**	0.78	5.69	427
		SWF	9	1.51	0.00	4
	v	IND	**1649**	0.68	23.90	394
		SWF	235	0.90	2.41	58
	rv	IND	**1256**	0.81	12.29	341
		SWF	563	0.33	21.89	74
Independent	N.A.	N.A.	**1793**	**0.73**	8.11	421

4.2.4 I-SOTO

Finally, we apply the Intertwined version of SOTO and test in this environment for the same set of training strategies utilised in the previous section. The results are presented in Figure 4.2. The set of ranges generated from this model can be found in Figure A.3. Regarding the α-fairness SWF, one solution was found to Pareto-front the baseline, both in fairness and efficiency - the setting of I-SOTO with the baseline function strategy. As for the G_w baseline, while no solution was found to improve both efficiency and fairness, the baseline SWF function version of I-SOTO - once again - obtained a slightly better CV in 0.02 at the cost of $\approx$ 18 income points. Although this result does not surpass the baseline, it is interesting that the CV of SOTO(G_w) was already very close to 0.

Another interesting result obtained was that the reverse baseline IND version of I-SOTO was able to surpass the baseline in efficiency in both metrics. With regards to efficiency, this result is in agreement with the fact that reverse baseline is the strategy function with most samples dedicated to π^{IND} of all the strategies explored. However, it is still interesting that the inclusion of a fairness oriented policy was beneficial for both goals, and most significantly with the α-fairness setting with a gain of $\approx$ 66 income points and decrease in 0.08 CV units.

4.3 Traffic Light Control

In the Traffic Light Control [96] environment, there is a grid of roads. Each intersection joins four roads, each with two lanes in different directions - being a total of 8 lanes with different numbers of vehicles per intersection, as illustrated in Figure 4.8.

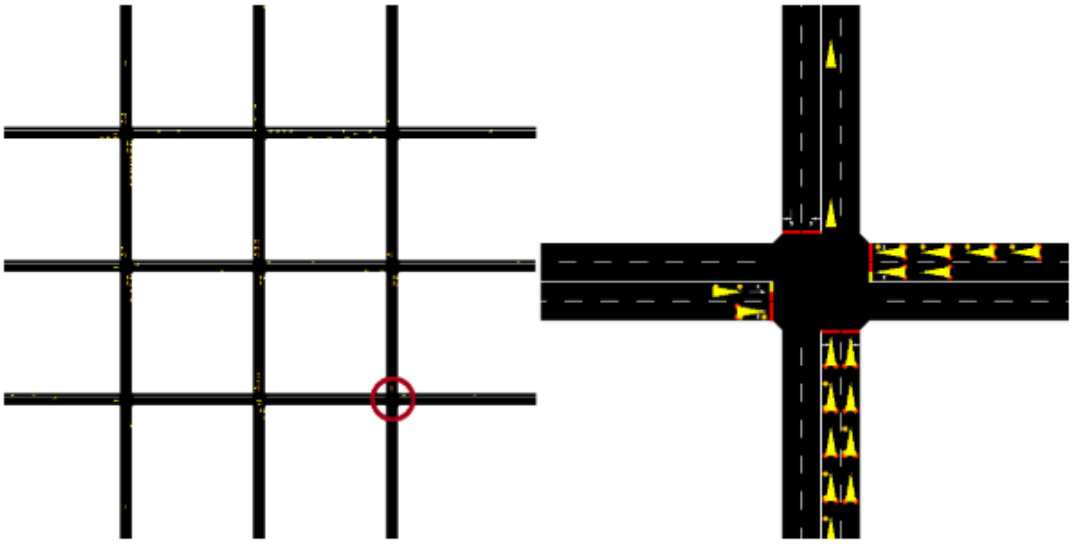


Figure 4.8: Traffic Light Control Environment [103]

Each agent controls the state of its traffic light. The traffic light phase controls the traffic and specifies which lanes have the green light and which don't. We assume that each intersection has 4

Table 4.2: I-SOTO performance under the self- (IND) and team-oriented (SWF) policies in Matthew Effect. **Bold** values are entries which perform equal or better than the baseline (b) of the respective model, in Total Income (higher) or CV (lower), i.e. efficiency or fairness.

model	β	π	Total Income	CV	Min Income	Max Income
I-SOTO(α)	0.25	IND	1440	0.75	11.80	356
		SWF	1529	**0.48**	35.22	271
	0.5	IND	1626	0.66	24.18	385
		SWF	1378	0.53	15.67	240
	lin	IND	**1771**	0.64	31.42	417
		SWF	888	0.69	4.77	167
	rlin	IND	1617	0.64	23.68	372
		SWF	**1680**	**0.48**	38.34	306
	b	IND	1138	0.94	3.69	338
		SWF	**1756**	**0.44**	58.02	316
	rb	IND	**1859**	0.65	23.31	423
		SWF	99	1.28	0.16	37
	v	IND	1641	0.64	22.59	372
		SWF	1473	0.56	29.96	293
	rv	IND	1573	0.68	17.12	374
		SWF	1550	0.52	21.75	282
SOTO(α)	b	IND	1178	0.90	5.88	342
		SWF	**1663**	**0.49**	43.29	297
I-SOTO(G_w)	0.25	IND	**1178**	0.86	6.91	327
		SWF	733	0.21	43.32	87
	0.5	IND	**1552**	0.64	25.88	358
		SWF	130	0.89	0.90	34
	lin	IND	**1724**	0.67	22.82	407
		SWF	37	1.09	0.07	10
	rlin	IND	**1210**	0.84	7.47	332
		SWF	649	0.33	24.33	87
	b	IND	**1156**	0.88	5.63	326
		SWF	1035	**0.01**	101.06	106
	rb	IND	**1799**	0.71	15.05	433
		SWF	11	1.58	0.01	5
	v	IND	**1479**	0.77	13.61	372
		SWF	355	0.88	2.68	100
	rv	IND	**1241**	0.81	8.97	325
		SWF	581	0.37	18.50	81
SOTO(G_w)	b	IND	**1139**	0.86	9.00	324
		SWF	**1052**	**0.03**	99.68	109
Independent	N.A.	N.A.	**1793**	**0.73**	8.11	421

phases. The state space is composed of the current traffic light phase, and for each lane, its queue and density of cars stopped at the intersection. The action corresponds to choosing which traffic light phase is going to be up next. At each time step, new vehicles enter into the intersection with a fixed destination.

We use SUMO [51](Simulation of Urban Mobility) to simulate the waiting time of the vehicles in a particular time step and intersection w_t^i during 5000 seconds. Notice that while in Matthew Effect, agents aim to maximise the income, in this case, they aim to minimise the waiting. As such, at each time step, the reward provided to the agents is the difference between the waiting time of the last time step w_{t-1} and the waiting time of the current w_t. This definition motivates the agent to minimise the accumulated waiting time, which is our goal in this domain. It is important to note that because this difference may be negative, this environment is not compatible with the α-fairness social welfare function and was thus not considered in the set of experimental results for this environment.

The relationship between waiting time w_t^i, the reward r_t^i and the evaluation measure m_t^i for each agent i is as defined in Equation 4.2.

$$\begin{aligned} r_t^i &= w_{t-1}^i - w_t^i \\ m_t^i &= w_t^i, \text{ referred to as waiting time} \end{aligned} \tag{4.2}$$

If the agents' decisions are too bad in this environment, a situation of complete blockage may occur. If this is the case, then regardless of the actions taken by the agents for the rest of the episode, traffic will remain blocked everywhere. Indeed, even if a not so extreme case occurs, there is a high dependence between agents' success under this environment as delays in other intersections may impact following intersections. Moreover, the flux of vehicles is not purposely equally distributed, being an extra source potentiating inequality. Finally, an added intricacy of this environment is that the notion of equality, and the value of CV, could be calculated in function of the waiting time per lane. Indeed, it could make sense to consider equality in the waiting time that agents take to pass through a lane and not an intersection. This option was not considered in this work, but we acknowledge the existence of this added concern. We consider only the original SOTO setting of fairness within intersections.

Experiments under this environment consider a 3x3 intersection grid, with a total of 9 agents, 2000 episodes, 500 steps per episode, and a discount factor γ is 0.99.

4.3.1 Baseline Performance

The performance of SOTO and the Independent baseline is depicted in Figure 4.2. These results show that while the SOTO models are the fairest, they seem to be approximately as efficient as the Independent baseline.

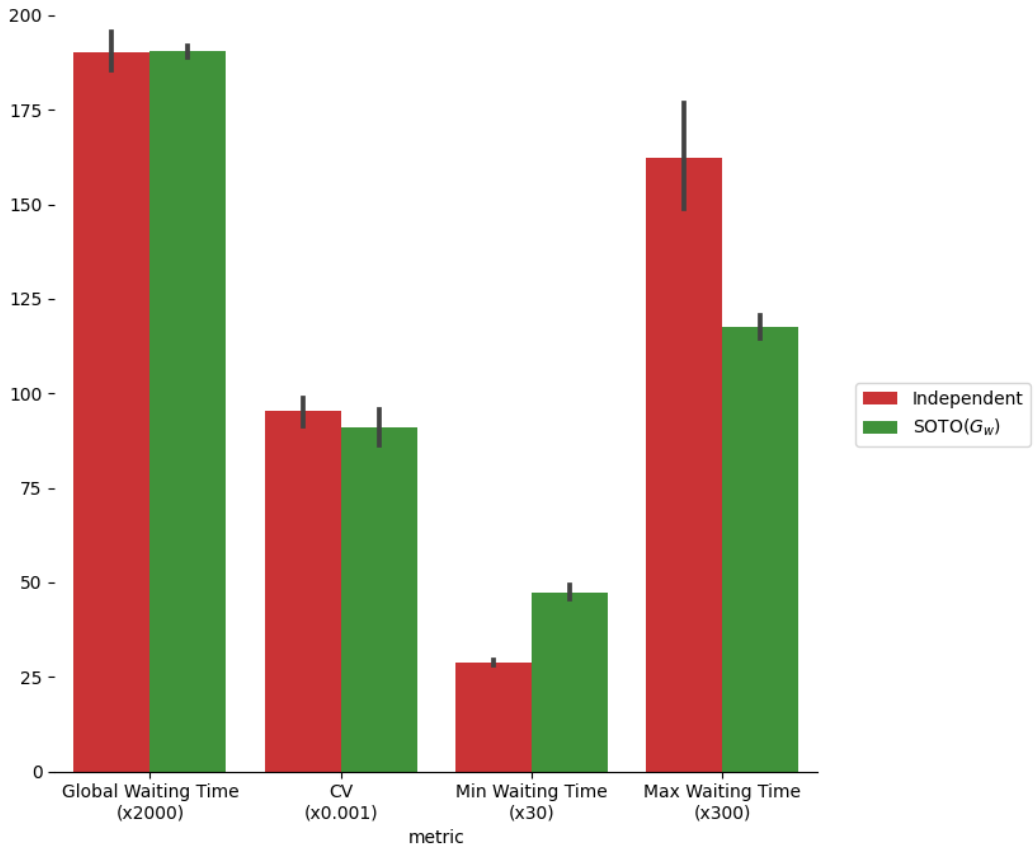


Figure 4.9: Baseline performance in Traffic Light Control. Logarithmic scale is being used.

These results are not in agreement with those reported by the authors of SOTO [103], since in their setting SOTO was approximately as fair as the Independent model but with waiting time lower in 3 orders of magnitude. A side by side display of our results with those reported by the authors of SOTO can be found in the Section B.1. In order to better understand why these differences could be happening, we decided to contact the authors and debate potential reasons for such differences being verified. The full discussion can be found at https://github.com/matthieu637/DFRL/issues/2. One of the outcomes of this contact was that, as an evaluation metric, the authors were considering the reward value per time step t as the evaluation measure $m_t = r_t$. However, we argue that, taking into account the reward formulation utilised, the aggregated global waiting time per episode W_e would be equal to the waiting time of the last time step negative w_T, as demonstrated in Equation 4.3. As such, it should not be used as an evaluation measure as it is not a representative value of the whole episode. This is what motivated us to record m_t as the waiting time such that $W_e = \sum_{t=1}^{T} w_t$, and not the reward as in the case of the Matthew Effect. After presenting this argument to the authors, they agreed that the measure they were using was not reliable and will be re-uploading their results for this environment. We moved forward

with the planned set of experiments using $m_t = w_t$ for this reason.

$$\begin{aligned} W_e &= \sum_{t=1}^{T} r_t = \sum_{t=1}^{T} w_{t-1} - w_t \\ &= (w_0 - w_1) + (w_1 - w_2) + ... + (w_{T-1} - w_T), \quad w_0 = 0 \\ &= -w_T \end{aligned} \tag{4.3}$$

4.3.2 Heterogeneous Behaviour

The initial experimental setup and results relate to the testing of heterogeneous behaviour generated from different policies.

Self- and Team- Oriented Policy

The results of the heterogeneous behaviour produced by the self- and team-oriented policies of SOTO are depicted in Figure 4.10. It is possible to observe that the range of SOTO behaviours generated is approximately linear in the efficiency-fairness space. The team-oriented end is both the most efficient and fair. When compared to the previous environment, this phenomenon also occurred for the α-fairness metric. In the G_w, the performance range was not linear in the efficiency dimension, so we can conclude that the behaviour of the same SWF can be different under different environments.

Regarding the range of behaviours generated by π^{IND} and π^{SWF}, it seems to be sparser than in the Matthew effect, with an outlier with a high global waiting time. Nonetheless, some solutions seem to be able to be more efficient than the independent baseline, or as efficient with but fairer. However, because these options - together with the seam-oriented extreme - have a close performance to the independent baseline, the heterogeneous behaviour line of testing was not applied for this environment.

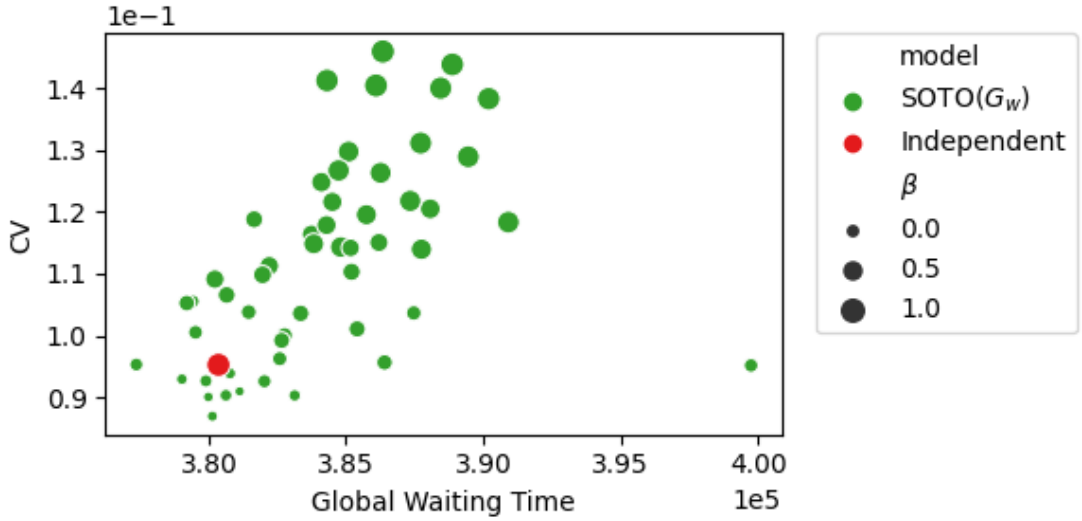


Figure 4.10: Heterogeneous Behaviour between SOTO self-(π^{IND}) and team-oriented (π^{SWF}) policies in Traffic Light Control. Higher diameter points correspond to higher probability of acting under π^{IND}.

4.3.3 Training Strategy

The results of each of the training strategies utilised are depicted in Table 4.3. The set of ranges generated from this transformation can be found in Section B.2. Regarding waiting time, it seems that many IND policies can surpass the SWF training strategy baseline, i.e., have lower waiting times. This aligns with the past domain, where we saw that the IND extremes, naturally, are more efficient. The most efficient result is the constant option of 0.5. Regarding fairness, interestingly, the only policy that outperforms the baseline is the very same model under the IND extreme. We conclude that, in this environment, it is possible to achieve between performing solutions in both optimisation dimensions just by changing the training strategy.

4.3.4 I-SOTO

The last set of experiments is the testing of I-SOTO. Results are depicted in Table 4.4. The set of ranges generated from this model can be found in Section B.3. Once again, in this environment, the IND extremes seem to be the most efficient, with the exception of the baseline training strategy. In particular, in the constant setting of 0.5, as in the previous case, waiting time is the lowest and outperforms the Independent baseline in this goal. On the other hand, with regards to fairness, it seems that rv in the IND setting achieves almost 50% of the baseline CV, 0.05 against 0.09, and can still be more efficient and reduce the waiting time. Once again, a better solution than SOTO was found.

Table 4.3: Training strategies performance of SOTO(G_w) in Traffic Light Control for the self- (IND) and team-oriented (SWF) policies. **Bold** values are entries which surpass the baseline, b, for the respective SWF, in Total Waiting Time (lower) or CV (lower), representing better efficiency or fairness respectively.

model	β	π	Global Time	CV	Min Time	Max Time
SOTO(G_w)	0.25	IND	**3.39e+05**	**0.09**	9.5e+02	3.2e+04
		SWF	**3.76e+05**	0.09	1.1e+03	3.5e+04
	0.5	IND	**3.27e+05**	**0.07**	8.9e+02	3.0e+04
		SWF	4.21e+05	0.09	1.4e+03	3.4e+04
	lin	IND	**3.57e+05**	0.20	9.0e+02	2.8e+04
		SWF	4.68e+05	0.15	1.9e+03	3.4e+04
	rlin	IND	**3.60e+05**	0.13	8.1e+02	3.1e+04
		SWF	4.48e+05	0.17	1.4e+03	3.5e+04
	b	IND	3.86e+05	0.14	8.8e+02	4.3e+04
		SWF	**3.81e+05**	**0.09**	1.4e+03	3.5e+04
	rb	IND	**3.66e+05**	0.14	8.5e+02	4.5e+04
		SWF	1.06e+06	0.14	4.6e+03	1.4e+05
	v	IND	**3.57e+05**	0.13	9.4e+02	3.7e+04
		SWF	4.67e+05	0.20	1.5e+03	4.6e+04
	rv	IND	**3.56e+05**	0.13	8.9e+02	3.1e+04
		SWF	4.68e+05	0.16	1.6e+03	3.8e+04
Independent	N.A.	N.A.	**3.80e+05**	**0.10**	8.6e+02	4.9e+04

4.4 Summary

We provide an experimental set of results within two environments and regarding heterogeneous behaviour from pre-trained policies, training strategies that change the training process of SOTO, and a novel architecture I-SOTO which intertwines self- and team-oriented policy recommendations. We are now able to answer the questions raised in Chapter 1 and detailed in Section 3.1.

Regarding heterogeneous behaviour, we conclude that it is indeed possible to generate ranges of behaviours between two pre-trained policies. However, we also show that these ranges do not represent linear trade-offs but more complex non-linear tendencies. Furthermore, in the Traffic Light Control environment, the solutions generated seem to have higher frequencies of outliers. Finally, this generation of intermediate behaviours seems not to produce solutions that outperform the extremes that originated them, with the exception of Traffic Light Control, where a few solutions are found but with little improvement.

As for the methods utilised to change the SOTO model - different training strategies and I-SOTO - a summary of the most relevant results is presented in Tables 4.5 and 4.6 for the Matthew and Traffic Light Control environments, respectively. In these tables were included entries that either (1) are the most efficient (2) are the fairest (3) Pareto-front the baseline SOTO model in the same SWF.

As can be seen, the most efficient and fair solutions were, in both environments, found in

Table 4.4: Training strategies performance of SOTO(G_w) in Traffic Light Control for the self- (IND) and team-oriented (SWF) policies. **Bold** values are entries which surpass the baseline, b, in Total Waiting Time (lower) or CV (lower), representing better efficiency or fairness respectively.

model	β	π	Global Time	CV	Min Time	Max Time
I-SOTO(G_w)	0.25	IND	3.78e+05	0.11	9.1e+02	3.7e+04
		SWF	3.80e+05	0.08	1.4e+03	2.9e+04
	0.5	IND	**2.89e+05**	0.48	5.6e+02	3.8e+04
		SWF	3.97e+05	0.32	1.6e+03	3.7e+04
	lin	IND	3.58e+05	0.10	8.3e+02	3.9e+04
		SWF	4.82e+05	0.10	1.2e+03	4.0e+04
	rlin	IND	3.79e+05	0.23	1.2e+03	3.2e+04
		SWF	4.48e+05	0.16	1.4e+03	3.7e+04
	b	IND	3.56e+05	0.10	1.0e+03	3.0e+04
		SWF	**3.50e+05**	**0.08**	9.8e+02	3.1e+04
	rb	IND	**3.48e+05**	**0.07**	7.5e+02	4.2e+04
		SWF	7.64e+05	0.10	3.4e+03	7.1e+04
	v	IND	**3.46e+05**	**0.07**	9.1e+02	3.3e+04
		SWF	4.22e+05	0.09	1.2e+03	3.4e+04
	rv	IND	**3.45e+05**	**0.05**	9.2e+02	3.2e+04
		SWF	4.44e+05	0.09	1.2e+03	3.3e+04
SOTO(G_w)	b	IND	3.86e+05	0.14	8.8e+02	4.3e+04
		SWF	**3.81e+05**	**0.09**	1.4e+03	3.5e+04
Independent	N.A.	N.A.	**3.80e+05**	**0.10**	8.6e+02	4.9e+04

the I-SOTO setting. This means that specific training strategies in I-SOTO can outperform the Independent baseline, which is fully focused on selfish behaviour optimisation. Furthermore, with this model, we were also able to find solutions that are both fairer and more efficient than the corresponding baseline for Matthew Effect under the α SWF and for Traffic Light Control under the Generalised Gini Function with I-SOTO(α;rlin)$^{\text{SWF}}$ and I-SOTO(G_w;rb)$^{\text{IND}}$ respectively. Interestingly, in the Traffic domain, the solution that was better than SOTO was on the self-oriented extreme. No solutions we found to outperform SOTO(G_w) in Matthew effect without sacrificing one of the goals. One solution, however, is able to get a better CV than this baseline. This result is nevertheless important to consider since the baseline CV was already very low, to begin with - 0.03 - and the baseline instance of I-SOTO reduced it to 0.01.

Furthermore, there is a broader pattern entailed in the experimental results data. With regards to the self- and team-oriented policies, it seems to be the case that either (1) π^{SWF} is the most fair and π^{SWF} is the most efficient policy or that (2) π^{SWF} is both the most efficient and fair. The latter case occurs for SOTO(α) in Matthew Effect and SOTO(G_w) in Traffic Light Signal. While we are not able to provide an exact explanation of why this happens, there are two potential intuitions for this reason.

On the one hand, it may have to do with the nature of the environment. As previously seen, in Traffic Light Control, the success of an agent (intersection) is highly dependent on the success

Table 4.5: Summary of result performances in Matthew Effect. **Bold** values are the best performing within its SWF.

model	β	π	Total Income	CV
I-SOTO(α)	b	SWF	1756	**0.44**
	rb	IND	**1859**	0.65
	rlin	SWF	1680	0.48
SOTO(α)	b	SWF	1663	0.49
I-SOTO(G_w)	b	SWF	1035	**0.01**
	rb	IND	**1799**	0.71
SOTO(G_w)	b	SWF	1052	0.03
Independent	N.A.	N.A.	1793	0.73

Table 4.6: Summary of result performances in Traffic Light Control. **Bold** values are the best performing within its SWF.

model	β	π	Global Time	CV
I-SOTO(G_w)	0.5	IND	**2.89e+05**	0.48
	rv	IND	3.45e+05	**0.05**
SOTO(G_w)	b	SWF	3.81e+05	0.09
Independent	N.A.	N.A.	3.80e+05	0.10

of other agents. This leads to the intuition that finding a fair solution, in this environment, is also finding a fair one. A result that is in agreement with this is the fact that the best performing model in this environment is I-SOTO(G_w) with the constant 0.5 strategy function, in which self- and team-oriented insights are shared between policies and in a balanced (50/50) way between goals.

On the other hand, this may also have to do with the nature of the social welfare function utilised. As seen in Chapter 2, the self- and team-oriented advantages utilised in the training process are a product of the derivative of the SWF with respect to the agents utilities, $\nabla_{\mathbf{u}}\phi(\hat{\mathbf{J}}(\theta))^{\top}$, with the original advantage. In the α-fairness scenario, this derivative is $\mathbf{u}^{0.9}$, while on the G_w setting it is $\mathbf{w} = \{2^{-i}, \forall_{i \in \mathcal{D}}\}$. This means that the team-oriented advantage for the first case is a sum of an exponential function to the agents utilities as opposed to a weighted sum based on their ranking. The fact that this function interprets social welfare as an independent concept from the ranking of individual utilities within the system perhaps deposits more confidence in individual success - efficiency - as a means towards fairness. On the other hand, considering the utility order overall produces much fairer results as it ensures no agent is being left behind. This, however, comes at the cost of a great deal in efficiency.

Chapter 5

Conclusions

With the increasing presence of Machine Learning models in our lives, fairness concerns are in order towards a trustworthy use of them. In Multi-Agent Reinforcement Learning domains (MARL), there is a concern for making agents receive approximately the same reward. However, this cannot be forced a priory - otherwise, agents would not receive reward reflecting the quality of their actions. The existent literature in the field approaches the problem as finding the fairest policy that, ideally, is also efficient. While this is a legitimate aim for a system, the impact of joint fair-efficient behaviours and training procedures is unknown, to the best of our knowledge. In a world where a complete commitment to fairness may not be feasible, fairness-efficiency holistic approaches are compelling to study.

5.1 Main Contributions

Our work approaches this problem in an exploratory manner in environments that expose the agents to unequal access or opportunity to resources - the Matthew Effect and Traffic Light Control.

First, we propose using an algorithm for testing fair and efficient behaviours heterogeneously in a MARL system. The intermediate solutions observed in both social welfare functions seem to not Pareto-front the extremes visibly but demonstrate viable solutions for system designers to consider having trained a SOTO model.

Then we evaluate SOTO with different training strategies for the self- and team-oriented goals and propose a change to SOTO where these intertwine recommendations for better predictions - I-SOTO. In every environment, we were able to find at least one I-SOTO solution that Pareto-fronts SOTO in a specific training strategy and social welfare goal. However, we acknowledge the limitation that if the same solutions in different environments perform differently, applying these methods to other environments potentially needs to be reevaluated.

Our main contributions can be summarised as follows:

- We propose an algorithm for testing different policies in a MADRL system
- We propose different training. strategies ($\beta(e_r)$) for SOTO.
- We propose I-SOTO, an architecture that intertwines recommendations from the self- and team-oriented policies.
- We show the ranges of behaviours generated by any of these alternatives and find solutions that are either fairer, more efficient or, in particular cases, that Pareto-front the baselines in each of the environments.
- We debate the performance of the baselines obtained by us with those reported by the authors of SOTO. It led to the conclusion that the metric they were using was inadequate and that their results are in need to be updated.
- We present a literature review in MADRL architectures and on the existent related work in fairness.

5.2 Future Work

As future work, we propose two main lines of work: expanding the testing space and architectural changes.

Perhaps the most natural path, expanding the space in which our methods are tested, would be interesting to accomplish in many dimensions. If there is one thing that became very evident in our results, it is that the use of different SWFs to train SOTO makes an impact on the performance direction it moves towards in the Fairness-Efficiency space. A natural direction would be including training SOTO and I-SOTO under different SWF definitions. Even if they do not meet the fairness criteria proposed by the authors of SOTO, it would be interesting to provide results comparing the performance and the benefits (or not) of using a carefully designed SWF definition. On the other hand, the β training strategy could also be rethought to accommodate more the needs of particular SWFs. While some of our experiments worked well for the α-fairness case, the same did not happen for GGF. These new definitions could include more dynamic definitions where the value of β could be determined by properties of the system itself (e.g. total score, CV, min score, avg score, etc.), apart from the episode ratio e_r, which is only an indicator of time. Another interesting dimension to test would be testing with new environments. Fairness can naturally behave differently if aimed in conditions where agents have more or less equal access to resources and under different environment dynamics. FEN and SOTO test their approaches in a total of 5 different domains. Only 2 of those were considered in this work. Furthermore, we are also curious to observe the behaviour of these models in environments not directly related to fairness and equality concerns. A question that could be considered is if it is easier to learn fair behaviour in environments that do not promote as much unequal resource access as the environments tested since unequal solutions are less likely to be found. Finally, the inclusion of other actor-critic

algorithms could be considered. For instance, with the use of A3C [55], the SOTO architecture would have to be tested in the CLDE domain, and potentially other improvements to it could be made to make the most out of the centralised learning setting.

On a different note, there could be attempts in changing the architecture of SOTO and I-SOTO towards improving their performance. For instance, the vector $\mathbf{J}$ could be considered as input of π^{IND}. Although this inclusion is not intuitive, since $\mathbf{J}$ contains information more relevant to the fair goal, it is unknown what the behaviour of the model would look like with this addition. Furthermore, the architecture could be adapted to include more communication mechanisms in the policies learned. Much like in DGN [39], there could be a communication of weights between neighbour agents to build more insightful networks to approximate either policy or value functions. In order to reduce the communication during execution, a viable approach could be performing this additional communication only on the value functions - such that, during execution, agents only need to communicate $\mathbf{J}$ which is at worse $|\mathcal{D}|$ dimensional.

References

[1] Bilal H Abed-Alguni, David J Paul, Stephan K Chalup, and Frans A Henskens. A comparison study of cooperative q-learning algorithms for independent learners. *Int. J. Artif. Intell*, 14(1):71–93, 2016.

[2] Matthew Adler. *Well-being and fair distribution: beyond cost-benefit analysis*. Oxford University Press, 2012.

[3] Edoardo Amaldi, Stefano Coniglio, Luca G Gianoli, and Can Umut Ileri. On single-path network routing subject to max-min fair flow allocation. *Electronic Notes in Discrete Mathematics*, 41:543–550, 2013.

[4] Edoardo Amaldi, Stefano Coniglio, Luca G Gianoli, and Can Umut Ileri. On single-path network routing subject to max-min fair flow allocation. *Electronic Notes in Discrete Mathematics*, 41:543–550, 2013.

[5] Wenhang Bao. Fairness in Multi-agent Reinforcement Learning for Stock Trading. *arXiv*, 12 2019.

[6] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[7] Dimitris Bertsimas, Vivek F Farias, and Nikolaos Trichakis. The price of fairness. *Operations research*, 59(1):17–31, 2011.

[8] Steven J Brams, Steven John Brams, and Alan D Taylor. *Fair Division: From cake-cutting to dispute resolution*. Cambridge University Press, 1996.

[9] Tim Brys, Anna Harutyunyan, Peter Vrancx, Ann Nowé, and Matthew E Taylor. Multi-objectivization and ensembles of shapings in reinforcement learning. *Neurocomputing*, 263:48–59, 2017.

[10] Róbert Busa-Fekete, Balázs Szörényi, Paul Weng, and Shie Mannor. Multi-objective bandits: Optimizing the generalized gini index. In *International Conference on Machine Learning*, pages 625–634. PMLR, 2017.

[11] Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008.

[12] Lucian Buşoniu, Robert Babuška, and Bart De Schutter. Multi-agent reinforcement learning: An overview. *Innovations in multi-agent systems and applications-1*, pages 183–221, 2010.

[13] Jacopo Castellini, Sam Devlin, Frans A. Oliehoek, and Rahul Savani. Difference Rewards Policy Gradients. 12 2020.

[14] Yu-Han Chang, Tracey Ho, and Leslie Kaelbling. All learning is local: Multi-agent learning in global reward games. 03 2004.

[15] Dezhi Chen, Qi Qi, Zirui Zhuang, Jingyu Wang, Jianxin Liao, and Zhu Han. Mean Field Deep Reinforcement Learning for Fair and Efficient UAV Control. *IEEE Internet of Things Journal*, 8(2):813–828, 1 2021.

[16] Yann Chevaleyre, Paul E Dunne, Ulle Endriss, Jérôme Lang, Michel Lemaitre, Nicolas Maudet, Julian Padget, Steve Phelps, Juan A Rodrígues-Aguilar, and Paulo Sousa. Issues in multiagent resource allocation. 2005.

[17] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. Fair clustering through fairlets. *arXiv preprint arXiv:1802.05733*, 2018.

[18] Houston Claure, Yifang Chen, Jignesh Modi, Malte Jung, and Stefanos Nikolaidis. Multi-Armed Bandits with Fairness Constraints for Distributing Resources to Human Teammates. *ACM/IEEE International Conference on Human-Robot Interaction*, pages 299–308, 6 2019.

[19] Houston Claure, Yifang Chen, Jignesh Modi, Malte Jung, and Stefanos Nikolaidis. Reinforcement learning with fairness constraints for resource distribution in human-robot teams. *arXiv preprint arXiv:1907.00313*, 2019.

[20] Hugh Dalton. The measurement of the inequality of incomes. *The Economic Journal*, 30(119):348–361, 1920.

[21] Steven De Jong, Karl Tuyls, and Katja Verbeeck. Fairness in multi-agent systems. *The Knowledge Engineering Review*, 23(2):153–180, 2008.

[22] Avinash K Dixit, John JF Sherrerd, et al. *Optimization in economic theory*. Oxford University Press on Demand, 1990.

[23] Stuart Dreyfus. Richard bellman on the birth of dynamic programming. *Operations Research*, 50(1):48–51, 2002.

[24] Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of Real-World Reinforcement Learning. *arXiv*, 4 2019.

[25] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012.

[26] Salma Elmalaki. Fair-iot: Fairness-aware human-in-the-loop reinforcement learning for harnessing human variability in personalized iot. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, pages 119–132, 2021.

[27] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual Multi-Agent Policy Gradients. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pages 2974–2982, 5 2017.

[28] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[29] Jianye Hao and Ho-fung Leung. *Fairness in Cooperative Multiagent Systems*, pages 27–70. 04 2016.

[30] Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable MDPs. In *AAAI Fall Symposium - Technical Report*, volume FS-15-06, pages 29–37. AI Access Foundation, 7 2015.

[31] Matthew John Hausknecht. *Cooperation and communication in multiagent deep reinforcement learning*. PhD thesis, 2016.

[32] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E. Taylor. A Survey and Critique of Multiagent Deep Reinforcement Learning. *Autonomous Agents and Multi-Agent Systems*, 33(6):750–797, 10 2018.

[33] SHI Huaizhou, R Venkatesha Prasad, Ertan Onur, and IGMM Niemegeers. Fairness in wireless networks: Issues, measures and challenges. *IEEE Communications Surveys & Tutorials*, 16(1):5–24, 2013.

[34] Edward Hughes, Joel Z. Leibo, Matthew G. Phillips, Karl Tuyls, Edgar A. Duéñez-Guzmán, Antonio García Castañeda, Iain Dunning, Tina Zhu, Kevin R. McKee, Raphael Koster, Heather Roff, and Thore Graepel. Inequity aversion improves cooperation in intertemporal social dilemmas. *Advances in Neural Information Processing Systems*, 2018-Decem:3326–3336, 3 2018.

[35] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35, 2017.

[36] Shariq Iqbal and Fei Sha. Actor-attention-critic for multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 2961–2970. PMLR, 2019.

[37] Shahin Jabbari, Matthew Joseph, Michael Kearns, Jamie Morgenstern, and Aaron Roth. Fairness in reinforcement learning. In *International conference on machine learning*, pages 1617–1626. PMLR, 2017.

[38] Rajendra K Jain, Dah-Ming W Chiu, William R Hawe, et al. A quantitative measure of fairness and discrimination. *Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA*, 1984.

[39] Jiechuan Jiang, Chen Dun, Tiejun Huang, and Zongqing Lu. Graph Convolutional Reinforcement Learning. *arXiv*, 10 2018.

[40] Jiechuan Jiang and Zongqing Lu. Learning Fairness in Multi-Agent Systems. *arXiv*, 10 2019.

[41] S. Jong, K. Tuyls, K. Verbeeck, and N. Roos. Considerations for fairness in multi-agent systems. *undefined*, 2007.

[42] Matthew Joseph, Michael Kearns, Jamie Morgenstern, and Aaron Roth. Fairness in learning: Classic and contextual bandits. *arXiv preprint arXiv:1605.07139*, 2016.

[43] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2):99–134, 5 1998.

[44] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[45] Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014, 2000.

[46] Landon Kraemer and Bikramjit Banerjee. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190:82–94, 5 2016.

[47] Matt J Kusner, Joshua R Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. *arXiv preprint arXiv:1703.06856*, 2017.

[48] Kai Lamertz. The social construction of fairness: Social influence and sense making in organizations. *Journal of Organizational Behavior*, 23(1):19–37, 2002.

[49] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, 9 2016.

[50] Siyuan Liu, Miguel Araujo, Emma Brunskill, Rosaldo Rossetti, Joao Barros, and Ramayya Krishnan. Understanding sequential decisions via inverse reinforcement learning. In *2013 IEEE 14th International Conference on Mobile Data Management*, volume 1, pages 177–186, 2013.

[51] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2575–2582. IEEE, 2018.

[52] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *Advances in Neural Information Processing Systems*, 2017-December:6380–6391, 6 2017.

[53] Hanan Luss. *Equitable Resource Allocation: Models, Algorithms and Applications*, volume 101. John Wiley & Sons, 2012.

[54] Laëtitia Matignon, Laurent Jeanpierre, and Abdel-Illah Mouaddib. Coordinated multi-robot exploration under communication constraints using decentralized markov decision processes. In *Twenty-sixth AAAI conference on artificial intelligence*, 2012.

[55] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous Methods for Deep Reinforcement Learning. *33rd International Conference on Machine Learning, ICML 2016*, 4:2850–2869, 2 2016.

[56] Jeonghoon Mo and Jean Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on networking*, 8(5):556–567, 2000.

[57] Hervé Moulin. *Fair division and collective welfare*. MIT press, 2003.

[58] Arnie Neidhardt, Hanan Luss, and KR Krishnan. Data fusion and optimal placement of fixed and mobile sensors. In *2008 IEEE Sensors Applications Symposium*, pages 128–133. IEEE, 2008.

[59] Andrew Y. Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *in Proc. 17th International Conf. on Machine Learning*, pages 663–670. Morgan Kaufmann, 2000.

[60] Wlodzimierz Ogryczak, Hanan Luss, Michał Pióro, Dritan Nace, and Artur Tomaszewski. Fair optimization and networks: A survey. *Journal of Applied Mathematics*, 2014, 2014.

[61] Wlodzimierz Ogryczak, Patrice Perny, and Paul Weng. A compromise programming approach to multiobjective markov decision processes. *International Journal of Information Technology & Decision Making*, 12(05):1021–1053, 2013.

[62] Frans A. Oliehoek and Christopher Amato. *A Concise Introduction to Decentralized POMDPs*. SpringerBriefs in Intelligent Systems. Springer International Publishing, Cham, 2016.

[63] Frans A. Oliehoek, Matthijs T.J. Spaan, and Nikos Vlassis. Optimal and approximate Q-value functions for decentralized POMDPs. *Journal of Artificial Intelligence Research*, 32:289–353, 2008.

[64] Afshin OroojlooyJadid and Davood Hajinezhad. A Review of Cooperative Multi-Agent Deep Reinforcement Learning. *arXiv*, 8 2019.

[65] Arthur Cecil Pigou. *Wealth and welfare*. Macmillan and Company, limited, 1912.

[66] Michal Pioro, Gabor Malicsko, and Gabor Fodor. Optimal link capacity dimensioning in proportionally fair networks. In *NETWORKING 2002: Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications*, volume 2345 of *Lecture Notes in Computer Science*, pages 277–288, Germany, 2002. Springer. Second International IFIP-TC6 Networking Conference , May 19–24, 2002 ; Conference date: 19-05-2002 Through 24-05-2002.

[67] Jeremy Pitt. Interactional Justice and Self-Governance of Open Self-Organising Systems. In *Proceedings - 11th IEEE International Conference on Self-Adaptive and Self-Organizing Systems, SASO 2017*, pages 31–40. Institute of Electrical and Electronics Engineers Inc., 10 2017.

[68] Jeremy Pitt, Dídac Busquets, and Sam Macbeth. Distributive justice for self-organised common-pool resource management. *ACM Transactions on Autonomous and Adaptive Systems*, 9(3):1–39, 10 2014.

[69] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[70] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. 3 2018.

[71] John Rawls. *A Theory of Justice*. Harvard University Press, 1971.

[72] Heechang Ryu, Hayong Shin, and Jinkyoo Park. Multi-Agent Actor-Critic with Hierarchical Graph Attention Network. *arXiv*, 9 2019.

[73] Nripsuta Ani Saxena, Karen Huang, Evan DeFilippis, Goran Radanovic, David C Parkes, and Yang Liu. How do fairness definitions fare? examining public attitudes towards algorithmic definitions of fairness. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 99–106, 2019.

[74] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

[75] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[76] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[77] Han Shen, Kaiqing Zhang, Mingyi Hong, and Tianyi Chen. Asynchronous Advantage Actor Critic: Non-asymptotic Analysis and Linear Speedup. 12 2020.

[78] Huaizhou Shi, R Venkatesha Prasad, Ertan Onur, and IGMM Niemegeers. Fairness in wireless networks: Issues, measures and challenges. 2014.

[79] Yoav Shoham and Kevin Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2008.

[80] A F Shorrocks. The class of additively decomposable inequality measures. *Econometrica*, 48:613–25, 1980.

[81] Umer Siddique, Paul Weng, and Matthieu Zimmer. Learning Fair Policies in Multiobjective (Deep) Reinforcement Learning with Average and Discounted Rewards. *arXiv*, 8 2020.

[82] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[83] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 5887–5896. PMLR, 2019.

[84] Matthijs T.J. Spaan. Partially observable markov decision processes. In *Adaptation, Learning, and Optimization*, volume 12, pages 387–414. Springer Verlag, 2012.

[85] Till Speicher, Hoda Heidari, Nina Grgic-Hlaca, Krishna P Gummadi, Adish Singla, Adrian Weller, and Muhammad Bilal Zafar. A unified approach to quantifying algorithmic unfairness: Measuring individual &group unfairness via inequality indices. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2239–2248, 2018.

[86] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, 3:2085–2087, 6 2018.

[87] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018.

[88] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pages 330–337, 1993.

[89] Sahil Verma and Julia Rubin. Fairness definitions explained. In *2018 ieee/acm international workshop on software fairness (fairware)*, pages 1–7. IEEE, 2018.

[90] 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.

[91] Jianhong Wang, Yuan Zhang, Tae-Kyun Kim, and Yunjie Gu. Shapley Q-value: A Local Reward Approach to Solve Global Reward Games. *arXiv*, 7 2019.

[92] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[93] Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.

[94] Paul Weng. Fairness in Reinforcement Learning. *34th International Conference on Machine Learning, ICML 2017*, 4:2542–2557, 7 2019.

[95] John A Weymark. Generalized gini inequality indices. *Mathematical Social Sciences*, 1(4):409–430, 1981.

[96] Marco A Wiering. Multi-agent reinforcement learning for traffic light control. In *Machine Learning: Proceedings of the Seventeenth International Conference (ICML'2000)*, pages 1151–1158, 2000.

[97] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 5 1992.

[98] E. Yang and D. Gu. Multiagent Reinforcement Learning for Multi-Robot Systems: A Survey. *undefined*, 2004.

[99] Logan Yliniemi and Kagan Tumer. Multi-objective multiagent credit assignment through difference rewards in reinforcement learning. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8886:407–418, 12 2014.

[100] Mingqi Yuan, Qi Cao, Man-on Pun, and Yi Chen. Fairness-Oriented Scheduling for Bursty Traffic in OFDMA Downlink Systems Using Multi-Agent Reinforcement Learning. 12 2020.

[101] Chongjie Zhang and Julie A Shah. Fairness in multi-agent sequential decision-making. 2014.

[102] Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Basar. Fully decentralized multi-agent reinforcement learning with networked agents. In *International Conference on Machine Learning*, pages 5872–5881. PMLR, 2018.

[103] Matthieu Zimmer, Claire Glanois, Umer Siddique, and Paul Weng. Learning Fair Policies in Decentralized Cooperative Multi-Agent Reinforcement Learning. *arXiv*, 12 2020.

[104] Indrė Žliobaitė. Measuring discrimination in algorithmic decision making. *Data Mining and Knowledge Discovery*, 31(4):1060–1089, 2017.

[105] Haosheng Zou, Tongzheng Ren, Dong Yan, Hang Su, and Jun Zhu. Reward Shaping via Meta-Learning. 1 2019.

Appendix A

Matthew Effect

This chapter details further results with regards to the Matthew Effect Environment.

A.1 State of the art

In this section we present the results obtained for the baseline models when compared to those reported by the authors of SOTO.

A.2 Training Strategy Behaviour Specter

In this section we provide the full set of behavioural ranges generated from heterogeneous testing of SOTO under the different training strategies considered.

Figure A.1: Performance of SOTO, FEN and the Independent baseline obtained (top) compared to the performance stated in SOTO (bottom) in Matthew Effect. The correspondence between models, top to bottom, is as follows: Independent as Independent, SOTO(α) as FD SOTO(ϕ_α), SOTO(G_w) as FD SOTO(G_w), FEN as FEN-g. The SOTO(G_w) alternative in the bottom plot is referent to the CLDE scenario.

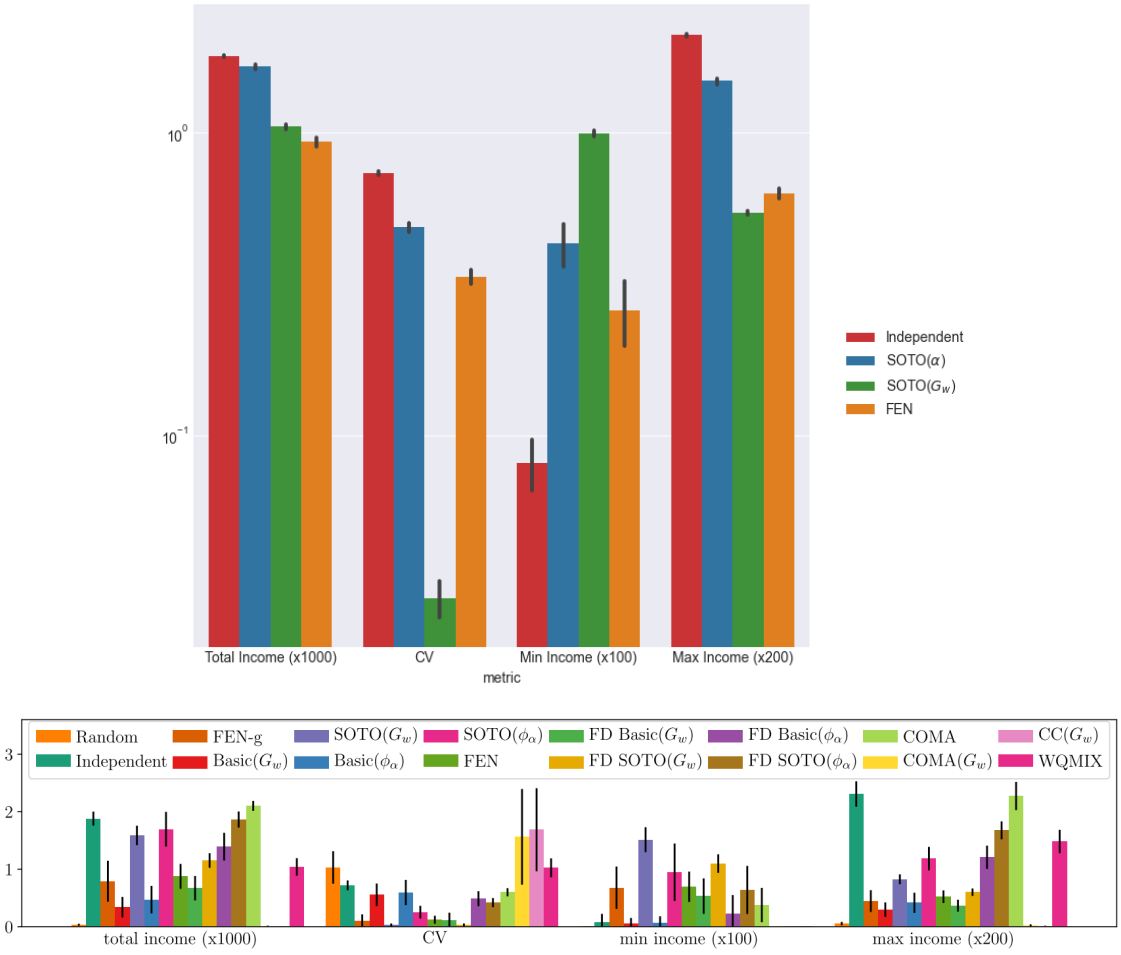
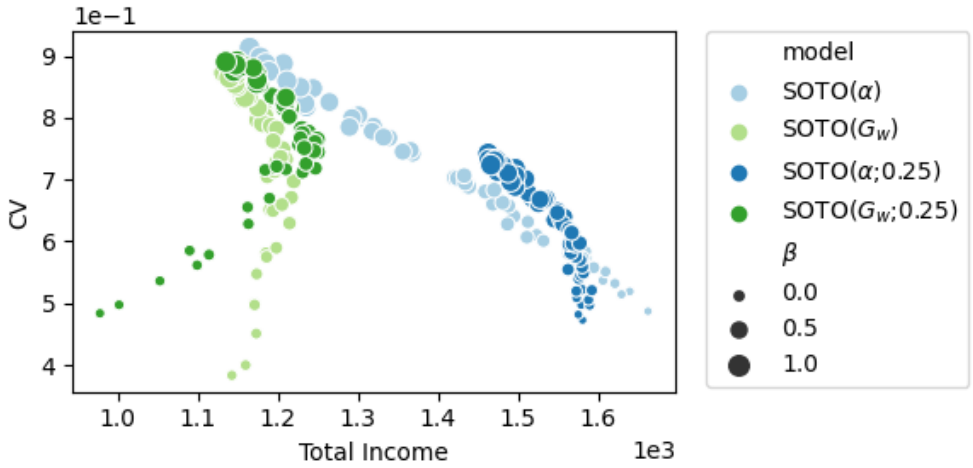
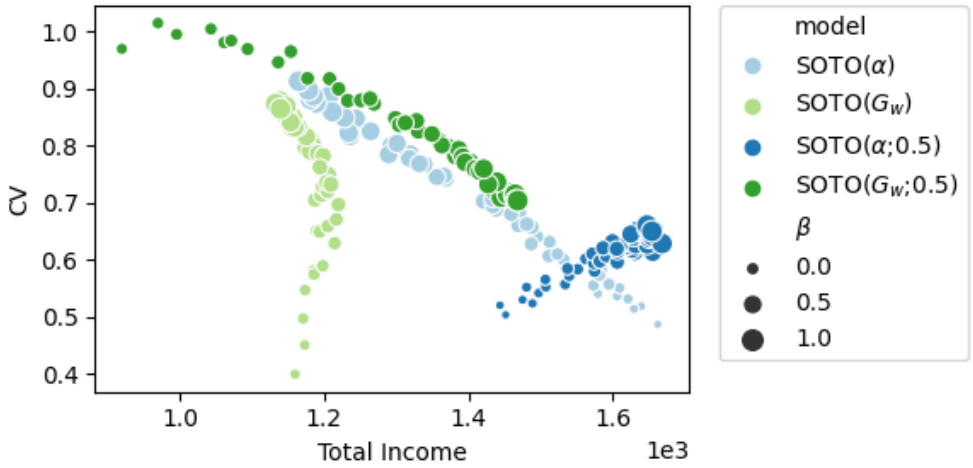


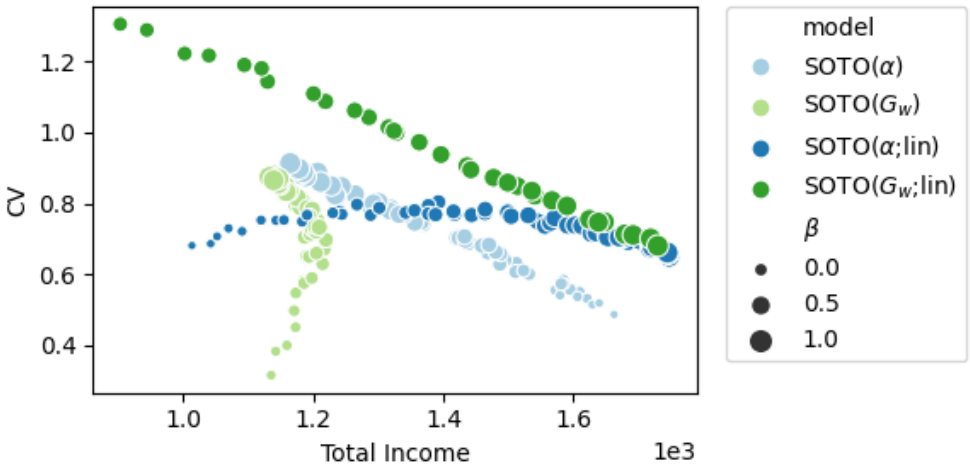
Figure A.2: Performance of SOTO in the efficiency-fairness space with different functions of β during training



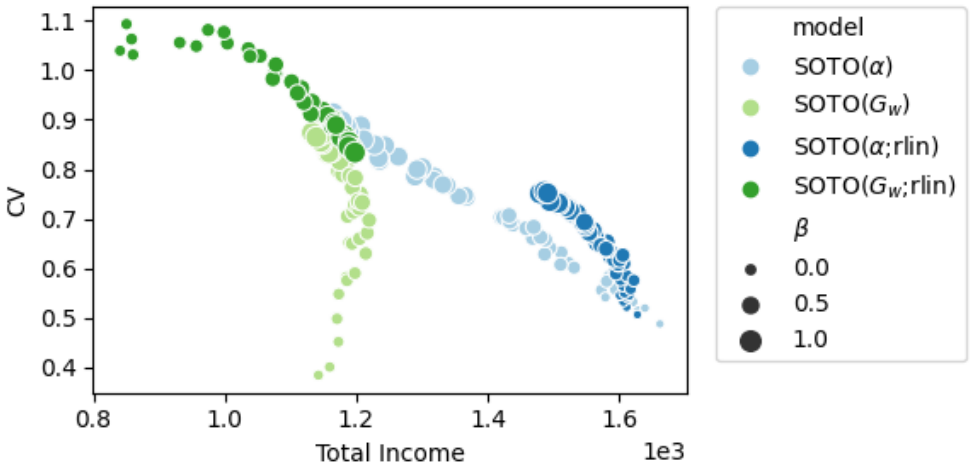
(a) Constant 0.25



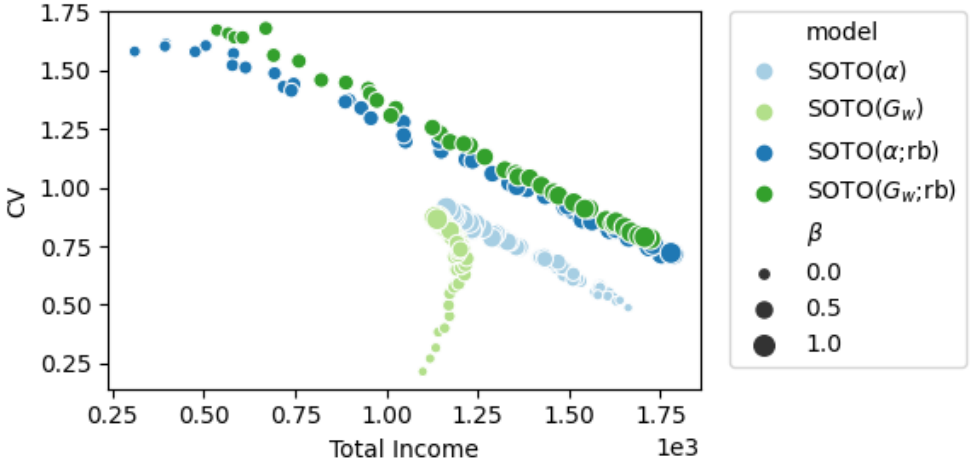
(a) Constant 0.5



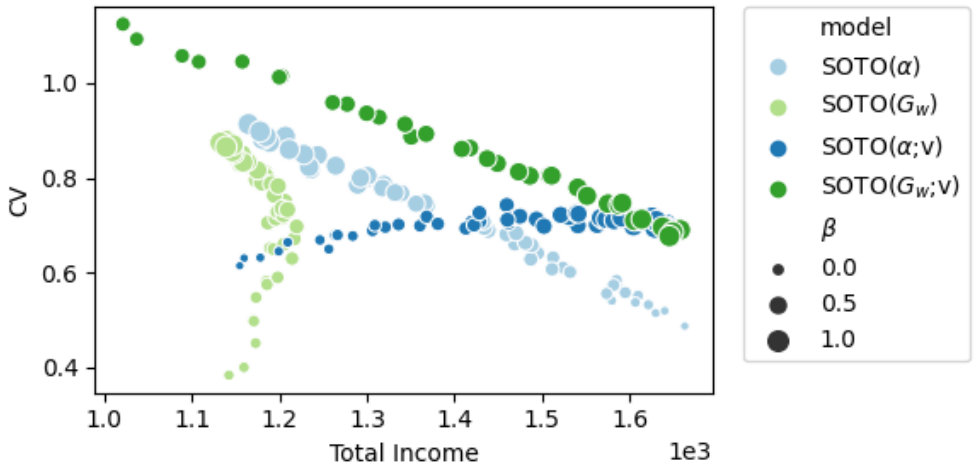
(a) Linear Ascending



(a) Linear Descending

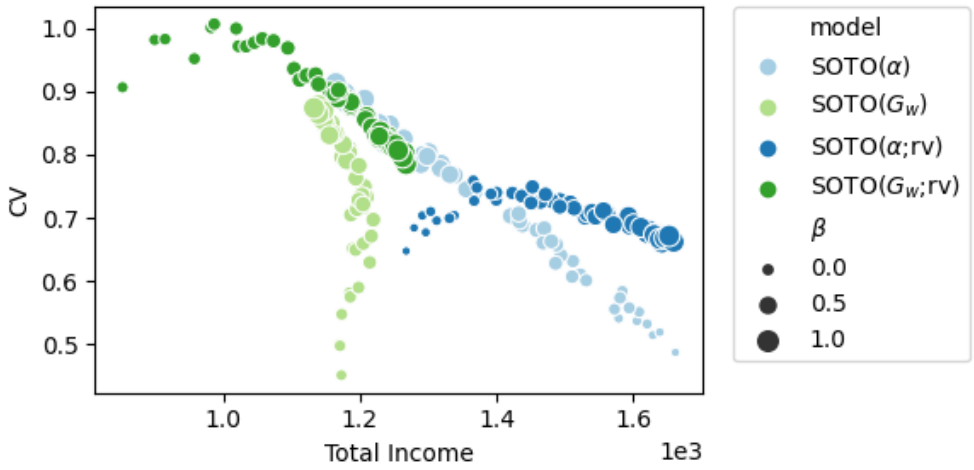


(a) Reverse Baseline



(a) V-shaped

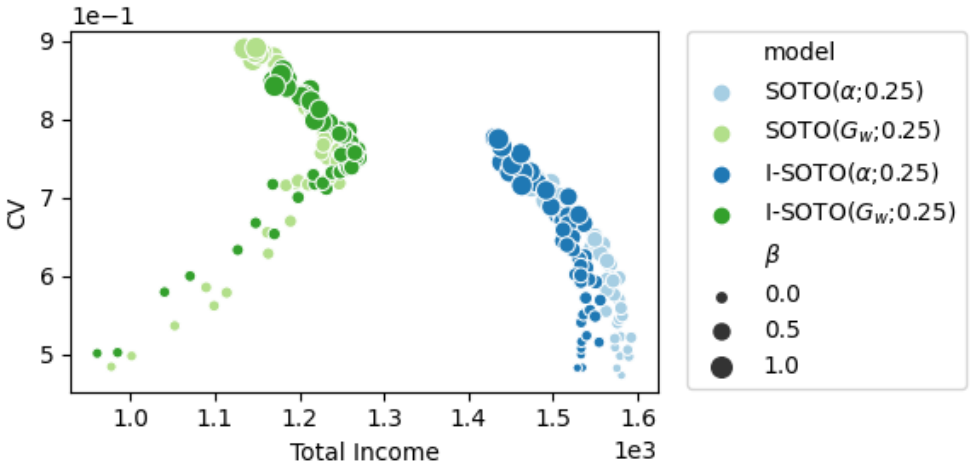
(a) Reverse V-shaped



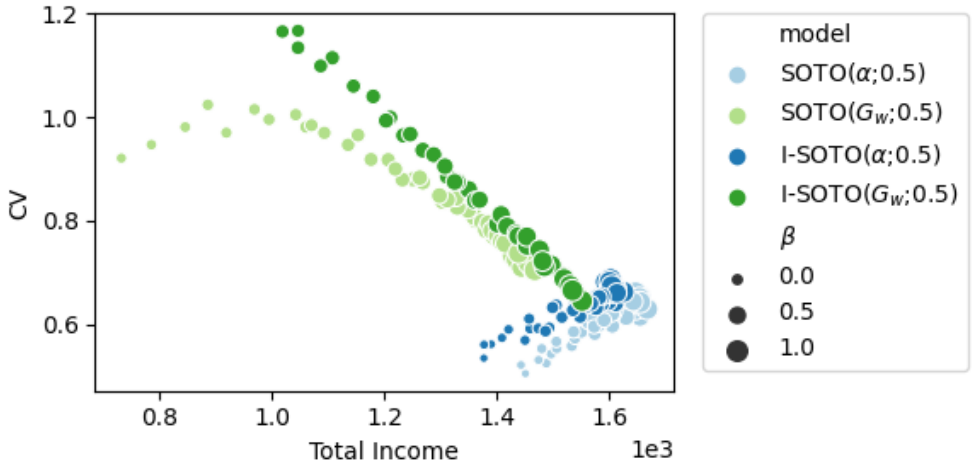
A.3 I-SOTO Behaviour Specter

In this section we provide the full set of behavioural ranges generated from heterogeneous testing of I-SOTO under the different training strategies considered.

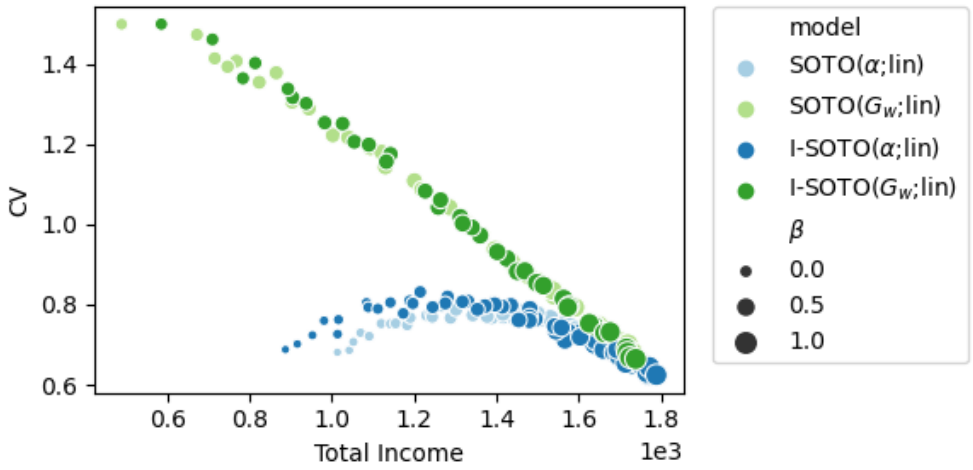
Figure A.3: Performance of I-SOTO in the efficiency-fairness space with different functions of β during training in Matthew Effect



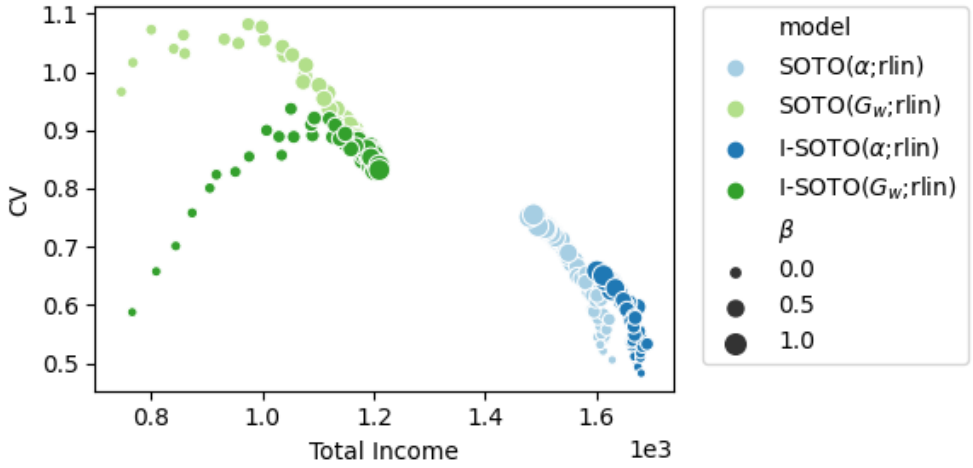
(a) Constant 0.25



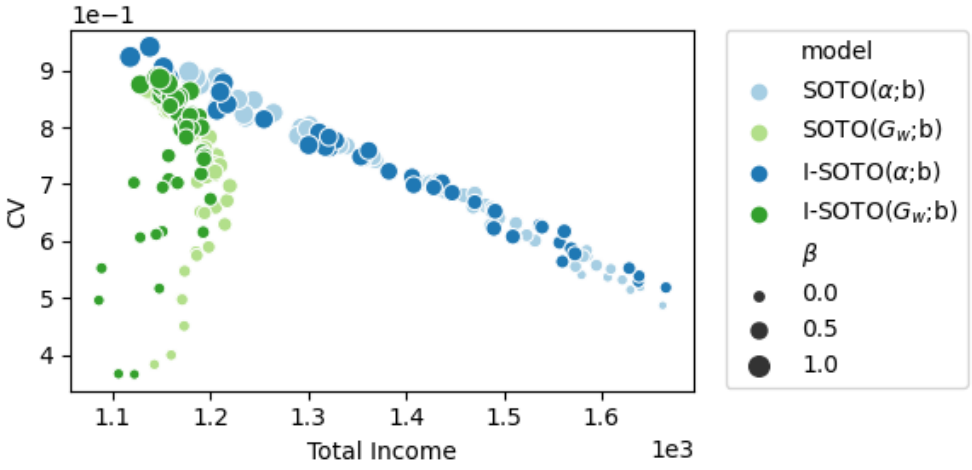
(a) Constant 0.5



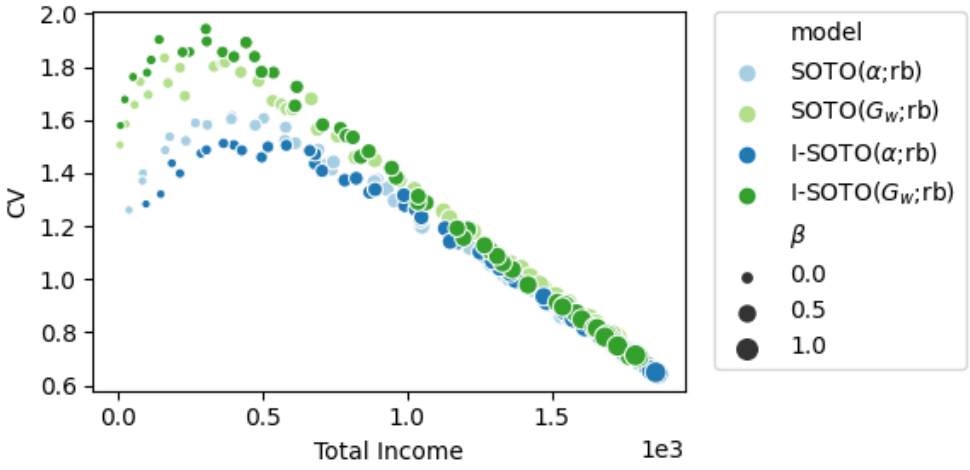
(a) Linear Ascending



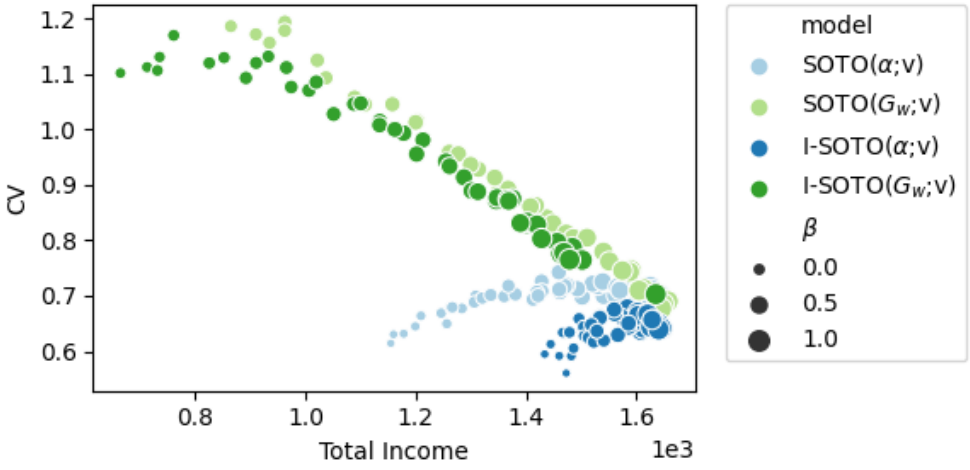
(a) Linear Descending



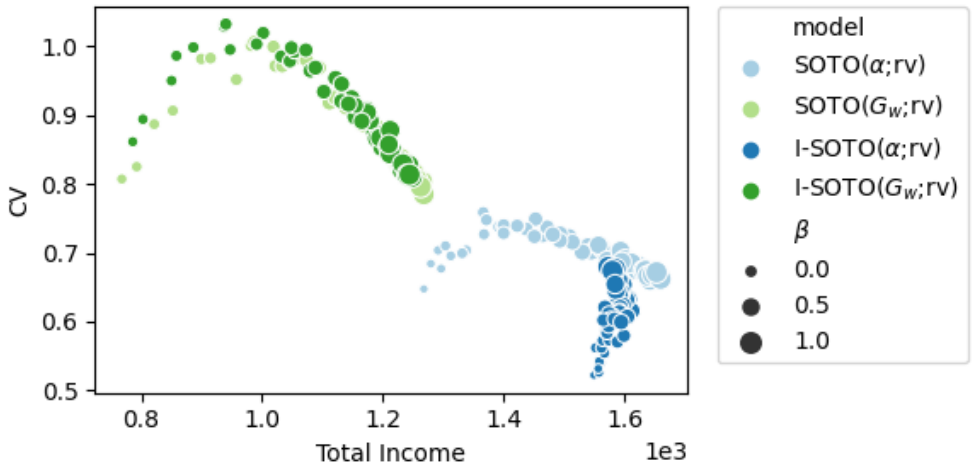
(a) Baseline



(a) Reverse Baseline



(a) V-shaped



(a) Reverse V-shaped

Appendix B

Traffic Light Control

This chapter details further results with regards to the Traffic Light Control Environment.

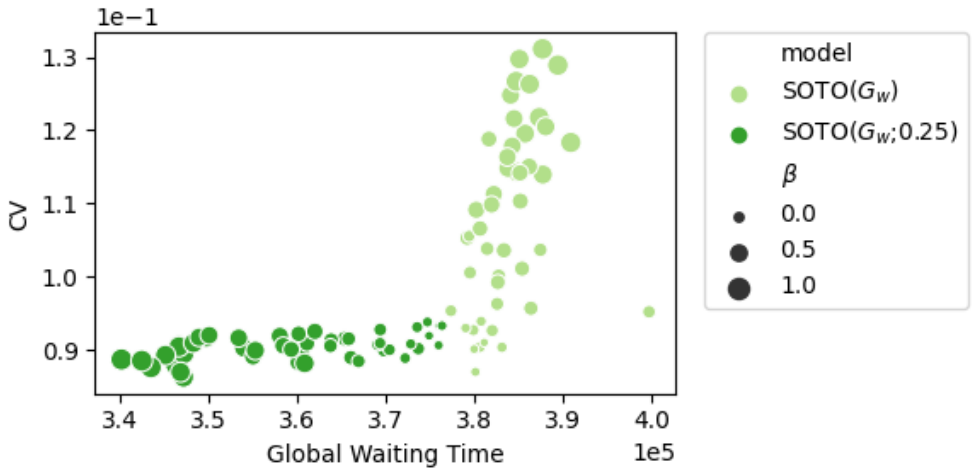
B.1 State of the art

In this section we present the results obtained for the baseline models when compared to those reported by the authors of SOTO.

B.2 Training Strategy Behaviour Specter

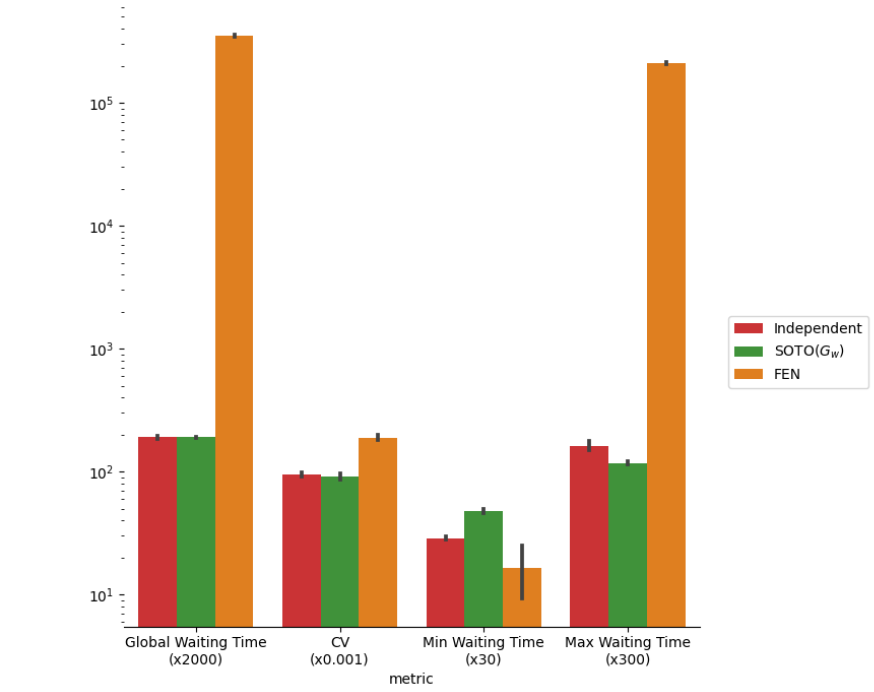
In this section we provide the full set of behavioural ranges generated from heterogeneous testing of SOTO under the different training strategies considered.

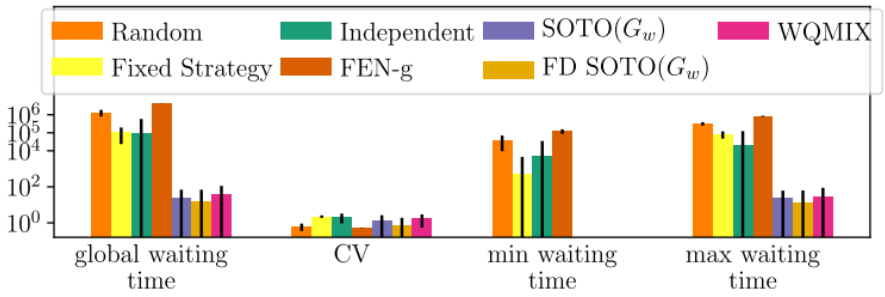
Figure B.2: Performance of SOTO in the efficiency-fairness space with different functions of β during training in Traffic Light Control

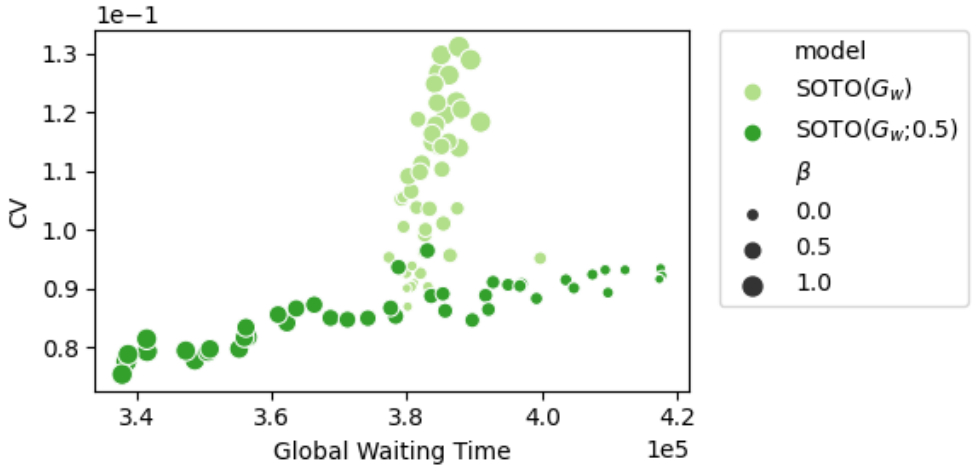


(a) Constant 0.25

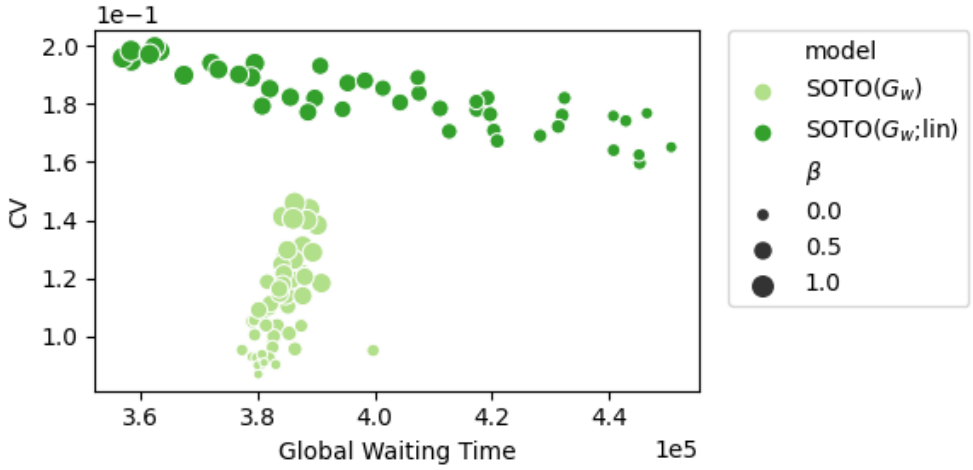
Figure B.1: Performance of SOTO, FEN and the Independent baseline obtained (top) compared to the performance stated in SOTO (bottom) in Traffic Light Control. The correspondence between models, top to bottom, is as follows: Independent as Independent, SOTO(G_w) as FD SOTO(G_w), FEN as FEN-g. The SOTO(G_w) alternative in the bottom plot is referent to the CLDE scenario.



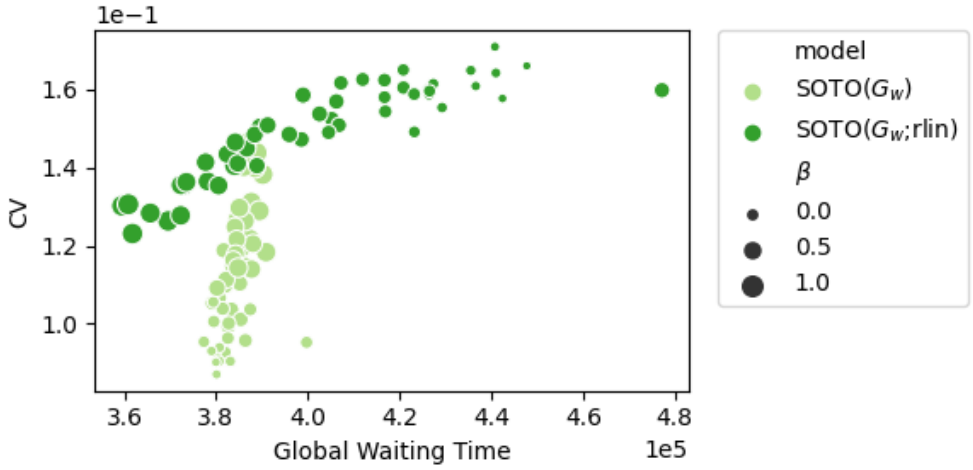




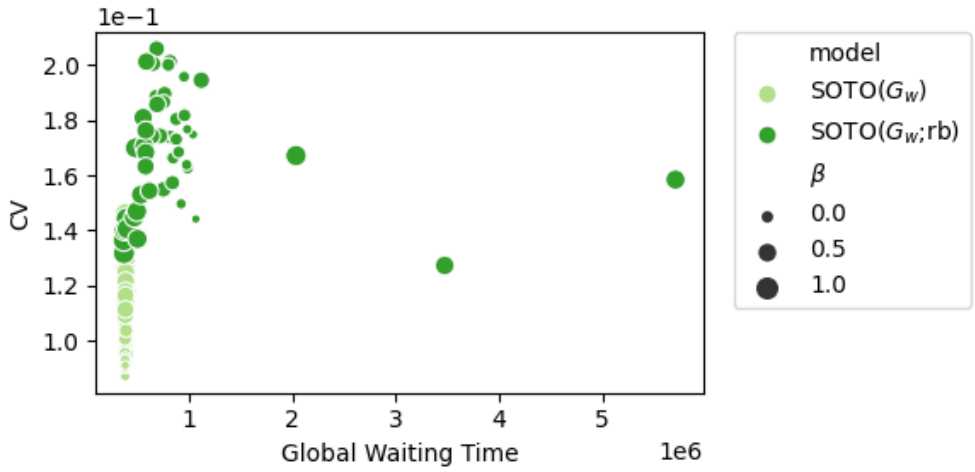
(a) Constant 0.5



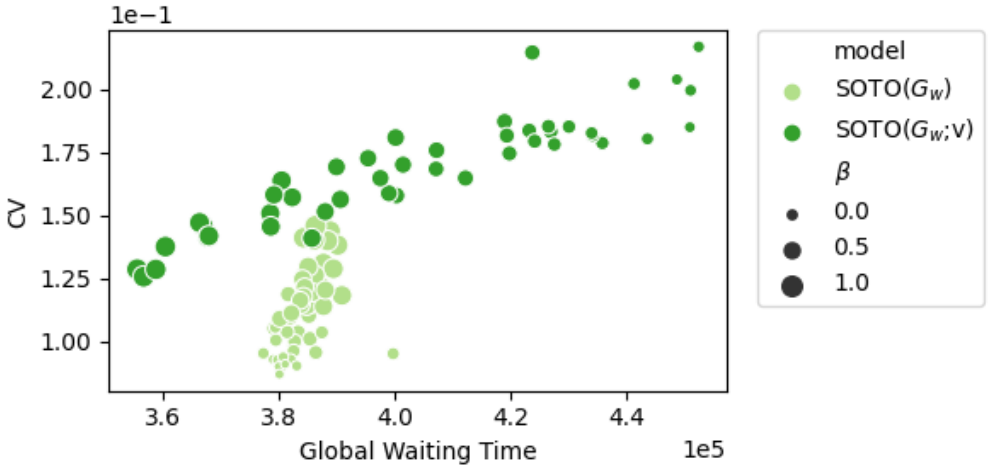
(a) Linear Ascending



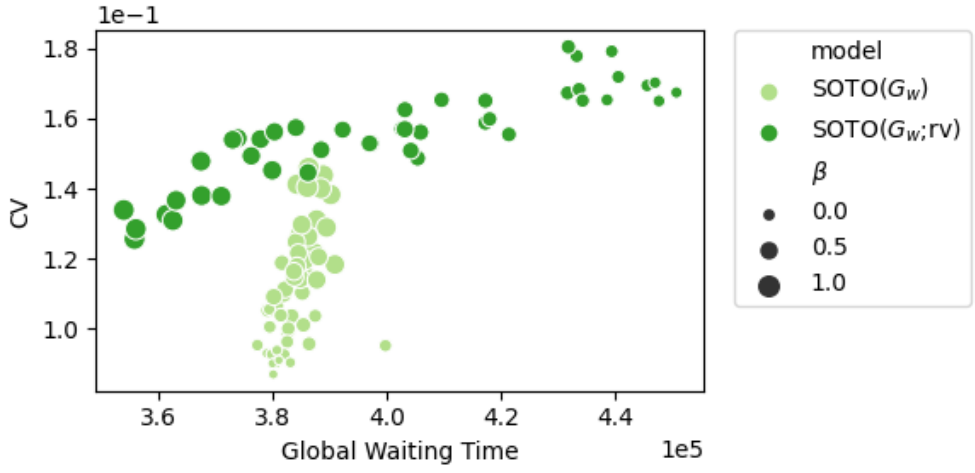
(a) Linear Descending



(a) Reverse Baseline



(a) V-shaped

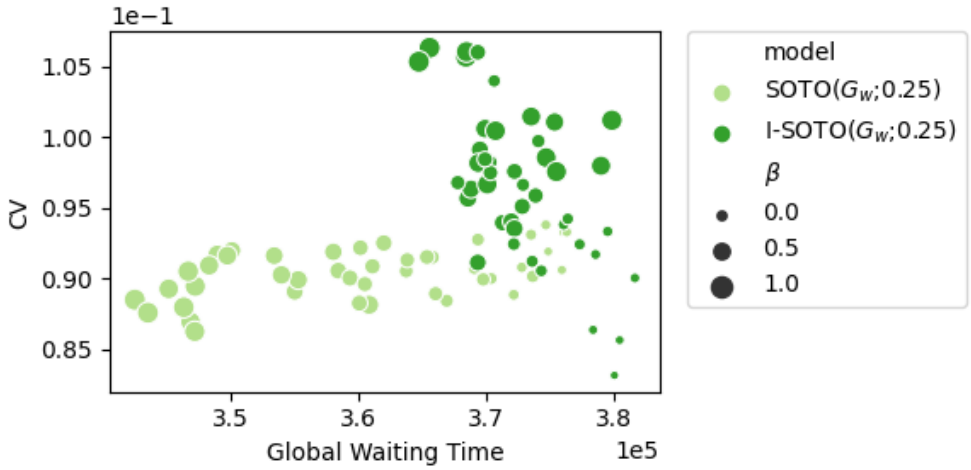


(a) Reverse V-shaped

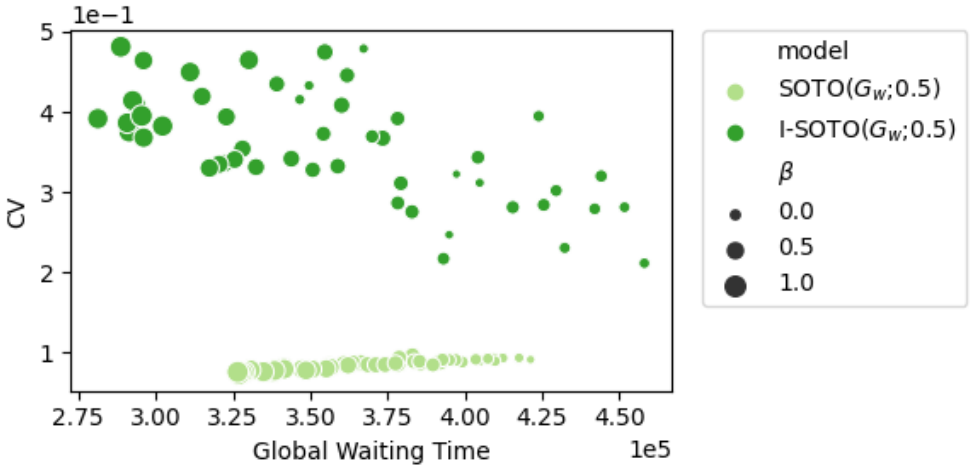
B.3 I-SOTO Behaviour Specter

In this section we provide the full set of behavioural ranges generated from heterogeneous testing of I-SOTO under the different training strategies considered.

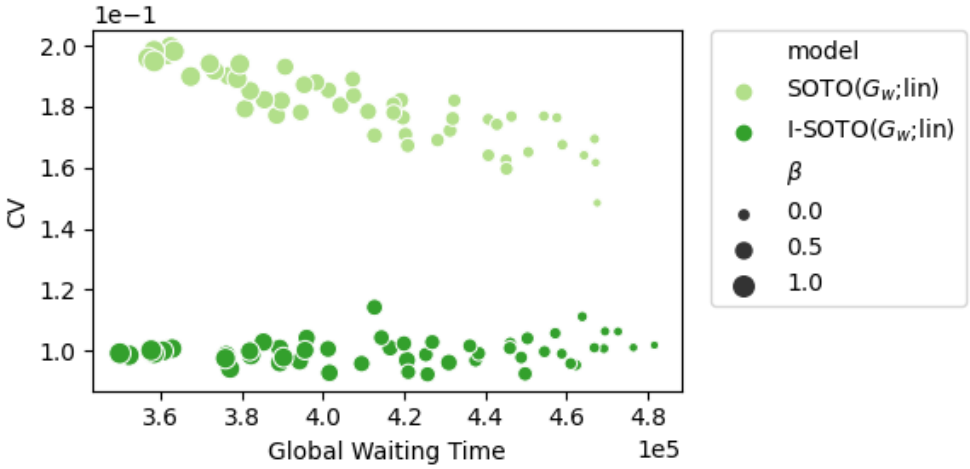
Figure B.3: Performance of I-SOTO in the efficiency-fairness space with different functions of β during training



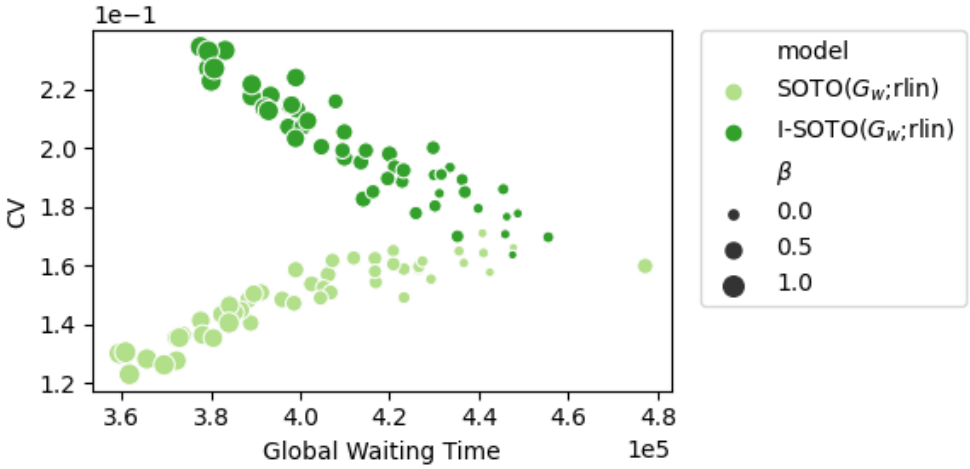
(a) Constant 0.25



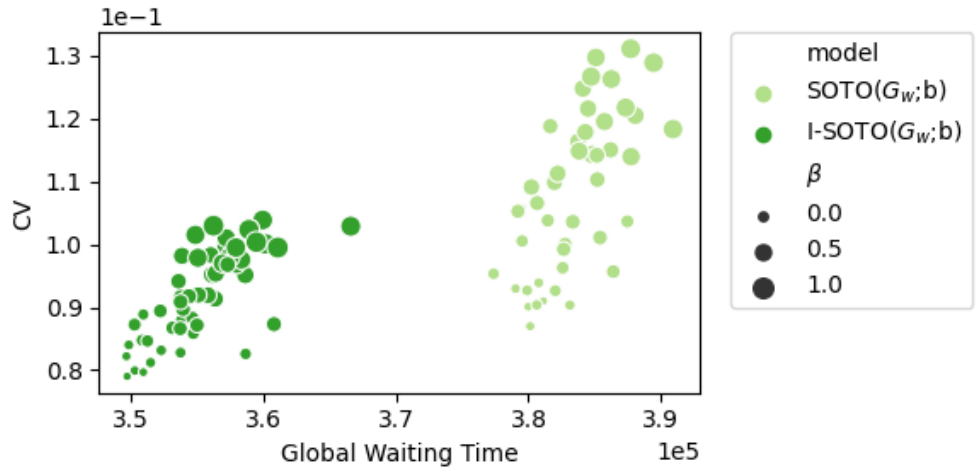
(a) Constant 0.5



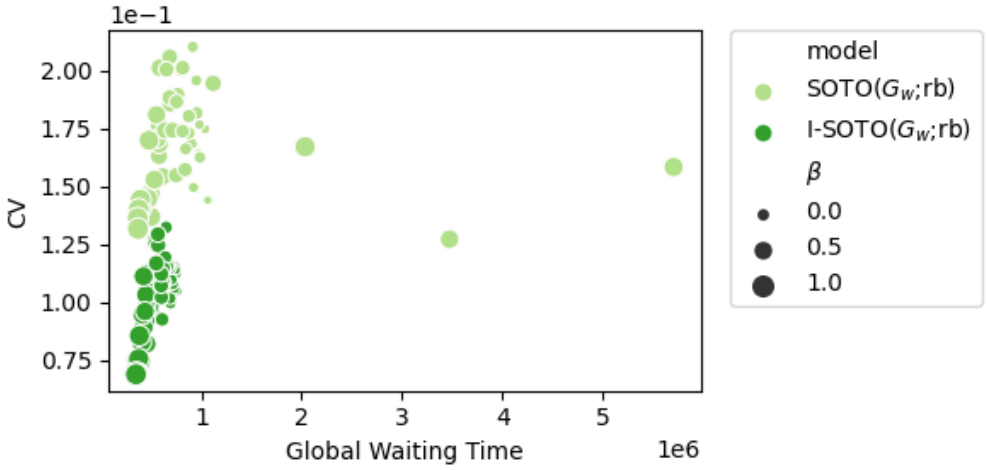
(a) Linear Ascending



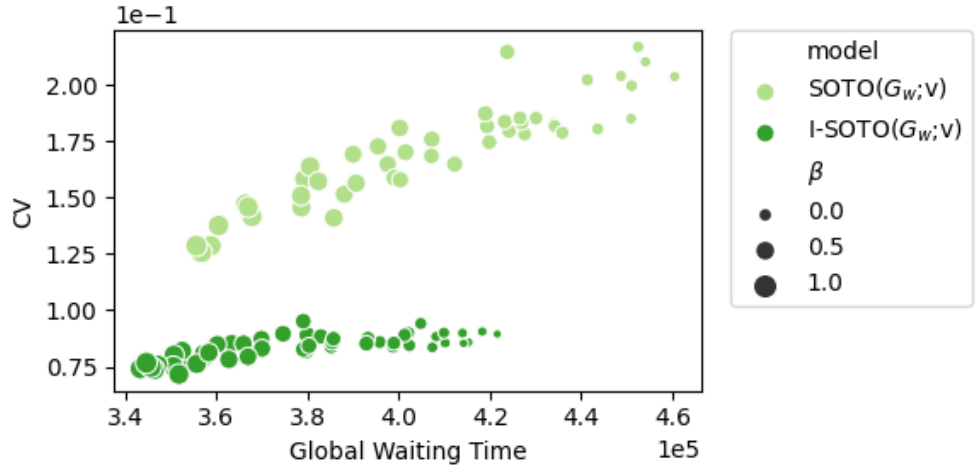
(a) Linear Descending



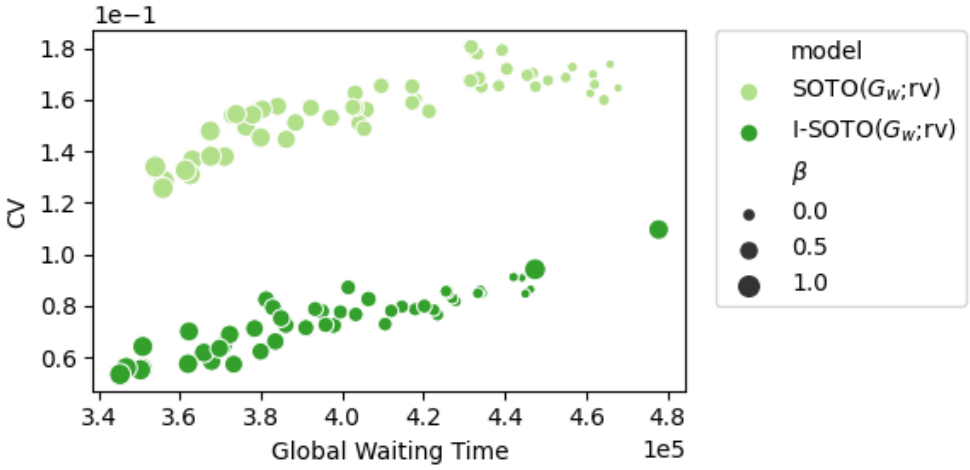
(a) Baseline



(a) Reverse Baseline



(a) V-shaped



(a) Reverse V-shaped