# Incentive Engineering for Concurrent Games

David Hyland
University of Oxford
Oxford, United Kingdom
david.hyland@cs.ox.ac.uk

Julian Gutierrez
Monash University
Melbourne, Australia
julian.gutierrez@monash.edu

Michael Wooldridge
University of Oxford
Oxford, United Kingdom
mjw@cs.ox.ac.uk

We consider the problem of incentivising desirable behaviours in multi-agent systems by way of taxation schemes. Our study employs the concurrent games model: in this model, each agent is primarily motivated to seek the satisfaction of a goal, expressed as a Linear Temporal Logic (LTL) formula; secondarily, agents seek to minimise costs, where costs are imposed based on the actions taken by agents in different states of the game. In this setting, we consider an external principal who can influence agents' preferences by imposing taxes (additional costs) on the actions chosen by agents in different states. The principal imposes taxation schemes to motivate agents to choose a course of action that will lead to the satisfaction of their goal, also expressed as an LTL formula. However, taxation schemes are limited in their ability to influence agents' preferences: an agent will always prefer to satisfy its goal rather than otherwise, no matter what the costs. The fundamental question that we study is whether the principal can impose a taxation scheme such that, in the resulting game, the principal's goal is satisfied in at least one or all runs of the game that could arise by agents choosing to follow game-theoretic equilibrium strategies. We consider two different types of taxation schemes: in a *static* scheme, the same tax is imposed on a state-action profile pair in all circumstances, while in a *dynamic* scheme, the principal can choose to vary taxes depending on the circumstances. We investigate the main game-theoretic properties of this model as well as the computational complexity of the relevant decision problems.

## 1 Introduction

*Rational verification* is the problem of establishing which temporal logic properties will be satisfied by a multi-agent system, under the assumption that agents in the system choose strategies that form a game-theoretic equilibrium [12, 41, 17]. Thus, rational verification enables us to verify which desirable and undesirable behaviours could arise in a system through individually rational choices. This article, however, expands beyond verification and studies methods for incentivising outcomes with favourable properties while mitigating undesirable consequences. One prominent example is the implementation of Pigovian taxes, which effectively discourage agents from engaging in activities that generate negative externalities. These taxes have been extensively explored in various domains, including sustainability and AI for social good, with applications such as reducing carbon emissions, road congestion, and river pollution [24, 22, 32].

We take as our starting point the work of [40], who considered the possibility of influencing one-shot Boolean games by introducing taxation schemes, which impose additional costs onto a game at the level of individual actions. In the model of preferences considered in [40], agents are primarily motivated to achieve a goal expressed as a (propositional) logical formula, and only secondarily motivated to minimise costs. This logical component limits the possibility to influence agent preferences: an agent can never be motivated by a taxation scheme away from achieving its goal. In related work, Wooldridge et al. defined the following implementation problem: given a game $G$ and an objective $\Upsilon$, expressed as a propositional logic formula, does there exists a taxation scheme $\tau$ that could be imposed upon $G$ such that, in the resulting game $G^{\tau}$, the objective $\Upsilon$ will be satisfied in at least one Nash equilibrium [40].

R. Verbrugge (Ed.): Theoretical Aspects of
Rationality and Knowledge 2023 (TARK 2023)
EPTCS 379, 2023, pp. 344–358, doi:10.4204/EPTCS.379.28

We develop these ideas by applying models of finite-state automata to introduce and motivate the use of history-dependent incentives in the context of *concurrent games* [2]. In a concurrent game, play continues for an infinite number of rounds, where at each round, each agent simultaneously chooses an action to perform. Preferences in such a multiplayer game are defined by associating with each agent $i$ a Linear Temporal Logic (LTL) goal $\gamma_i$, which agent $i$ desires to see satisfied. In this work, we also assume that actions incur costs, and that agents seek to minimise their limit-average costs.

Since, in contrast to the model of [40], play in our games continues for an infinite number of rounds, we find there are two natural variations of taxation schemes for concurrent games. In a *static* taxation scheme, we impose a fixed cost on state-action profiles so that the same state-action profile will always incur the same tax, no matter when it is performed. In a *dynamic* taxation scheme, the same state-action profile may incur different taxes in different circumstances: it is history-dependent. We first show that dynamic taxation schemes are strictly more powerful than static taxation schemes, making them a more appropriate model of incentives in the context of concurrent games, characterise the conditions under which an LTL objective $\Upsilon$ can be implemented in a game using dynamic taxation schemes, and begin to investigate the computational complexity of the corresponding decision problems.

## 2 Preliminaries

Where $S$ is a set, we denote the powerset of $S$ by $\mathbf{2}^S$. We use various propositional languages to express properties of the systems we consider. In these languages, we will let $\Phi$ be a finite and non-empty vocabulary of Boolean variables, with typical elements $p, q, \ldots$. Where $a$ is a finite word and $b$ is also a word (either finite or infinite), we denote the word obtained by concatenating $a$ and $b$ by $ab$. Where $a$ is a finite word, we denote by $a^\omega$ the infinite repetition of $a$. Finally, we use $\mathbb{R}^+_n$ for the set of $n$-tuples of non-negative real numbers.

**Concurrent Game Arenas:** We work with concurrent game structures, which in this work we will refer to as *arenas* (to distinguish them from the game structures that we introduce later in this section) [2]. Formally a *concurrent game arena* is given by a structure

$$\mathscr{A} = (\mathscr{S}, \mathscr{N}, Ac_1, \ldots, Ac_n, \mathscr{T}, \mathscr{C}, \mathscr{L}, s_0),$$

where: $\mathscr{S}$ is a finite and non-empty set of *arena states*; $\mathscr{N} = \{1, \ldots, n\}$ is the set of *agents* – for any $i \in \mathscr{N}$, we let $-i = \mathscr{N} \setminus \{i\}$ denote the set of *all agents excluding* $i$; for each $i \in \mathscr{N}$, $Ac_i$ is the finite and non-empty set of unique actions available to agent $i$ – we let $Ac = \bigcup_{i \in \mathscr{N}} Ac_i$ denote the set of all *actions* available to all players in the game and $\vec{Ac} = Ac_1 \times \cdots \times Ac_n$ denote the set of all *action profiles*; $\mathscr{T} : \mathscr{S} \times Ac_1 \times \cdots \times Ac_n \to \mathscr{S}$ is the *state transformer function* which prescribes how the state of the arena is updated for each possible action profile – we refer to a pair $(s, \vec{\alpha})$, consisting of a state $s \in \mathscr{S}$ and an action profile $\vec{\alpha} \in \vec{Ac}$ as a *state-action profile*; $\mathscr{C} : \mathscr{S} \times Ac_1 \times \cdots \times Ac_n \to \mathbb{R}^n_+$ is the *cost function* – given a state-action profile $(s, \vec{\alpha})$ and an agent $i \in \mathscr{N}$, we write $\mathscr{C}_i(s, \vec{\alpha})$ for the $i$-th component of $\mathscr{C}(s, \vec{\alpha})$, which corresponds to the cost that agent $i$ incurs when $\vec{\alpha}$ is executed at $s$; $\mathscr{L} : \mathscr{S} \to \mathbf{2}^\Phi$ is a *labelling function* that specifies which propositional variables are true in each state $s \in \mathscr{S}$; and $s_0 \in \mathscr{S}$ is the *initial state* of the arena. In what follows, it is useful to define for every agent $i \in \mathscr{N}$ the value $c^*_i$ to be the maximum cost that $i$ could incur through the execution of a state-action profile: $c^*_i = \max\{\mathscr{C}_i(s, \vec{\alpha}) \mid s \in \mathscr{S}, \vec{\alpha} \in \vec{Ac}\}$.

**Runs:** Games are played in an arena as follows. The arena begins in its initial state $s_0$, and each agent $i \in \mathscr{N}$ selects an action $\alpha_i \in Ac_i$ to perform; the actions so selected define an action profile, $\vec{\alpha} \in Ac_1 \times$

$\cdots \times Ac_n$. The arena then transitions to a new state $s_1 = \mathscr{T}(s_0, \alpha_1, \ldots, \alpha_n)$. Each agent then selects another action $\alpha_i' \in Ac_i$, and the arena again transitions to a new state $s_2 = \mathscr{T}(s_1, \vec{\alpha}')$. In this way, we trace out an infinite interleaved sequence of states and action profiles, referred to as a *run*, $\rho : s_0 \xrightarrow{\vec{\alpha}_0} s_1 \xrightarrow{\vec{\alpha}_1} s_2 \xrightarrow{\vec{\alpha}_2} \cdots$.

Where $\rho$ is a run and $k \in \mathbb{N}$, we write $s(\rho,k)$ to denote the state indexed by $k$ in $\rho$, so $s(\rho,0)$ is the first state in $\rho$, $s(\rho,1)$ is the second, and so on. In the same way, we denote the $k$-th action profile played in a run $\rho$ by $\vec{\alpha}(\rho,k-1)$ and to single out an individual agent $i$'s $k$-th action, we write $\alpha_i(\rho,k-1)$.

Above, we defined the cost function $\mathscr{C}$ with respect to individual state-action pairs. In what follows, we find it useful to lift the cost function from individual state-action pairs to sequences of state-action pairs and runs. Since runs are infinite, simply taking the sum of costs is not appropriate: instead, we consider the cost of a run to be the *average* cost incurred by an agent $i$ over the run; more precisely, we define the average cost incurred by agent $i$ over the first $t$ steps of the run $\rho$ as $\mathscr{C}_i(\rho,0:t) = \frac{1}{t+1}\sum_{j=0}^{t}\mathscr{C}_i(\rho,j)$ for $t \geq 1$, whereby $\mathscr{C}_i(\rho,j)$ we mean $\mathscr{C}_i(s(\rho,j),\vec{\alpha}(\rho,j))$. Then, we define the cost incurred by an agent $i$ over the run $\rho$, denoted $\mathscr{C}_i(\rho)$, as the *inferior limit of means*: $\mathscr{C}_i(\rho) = \liminf_{t\to\infty}\mathscr{C}_i(\rho,0:\ t)$. It can be shown that the value $\mathscr{C}_i(\rho)$ always converges because the sequence of averages $\mathscr{C}_i(\rho,0:t)$ is Cauchy.

**Linear Temporal Logic:** We use the language of Linear Temporal Logic (LTL) to express properties of runs [33, 11]. Formally, the syntax of LTL is defined wrt. a set $\Phi$ of Boolean variables by the following grammar:

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi\,\mathbf{U}\,\varphi \tag{1}$$

where $p \in \Phi$. Other usual logic connectives ("$\bot$", " & ", "$\rightarrow$", "$\leftrightarrow$") are defined in terms of $\neg$ and $\vee$ in the conventional way. Given a set of variables $\Phi$, let $LTL(\Phi)$ be the set of LTL formulae over $\Phi$; where the variable set $\Phi$ is clear from the context, we simply write $LTL$. We interpret formulae of LTL with respect to pairs $(\rho,t)$, where $\rho$ is a run, and $t \in \mathbb{N}$ is a temporal index into $\rho$. Any given LTL formula may be true at none or multiple time points on a run; for example, a formula $\mathbf{X}q$ will be true at a time point $t \in \mathbb{N}$ on a run $\rho$ if $q$ is true on a run $\rho$ at time $t+1$. We will write $(\rho,t) \models \varphi$ to mean that $\varphi \in LTL$ is true at time $t \in \mathbb{N}$ on run $\rho$. The rules defining when formulae are true (i.e., the semantics of LTL) are defined as follows:

$$\begin{array}{lll}
(\rho,t) \models \top & & \\
(\rho,t) \models p & \text{iff} & p \in \mathscr{L}(s(\rho,t)) \quad \text{(where } p \in \Phi\text{)} \\
(\rho,t) \models \neg\varphi & \text{iff} & \text{it is not the case that } (\rho,t) \models \varphi \\
(\rho,t) \models \varphi \vee \psi & \text{iff} & (\rho,t) \models \varphi \text{ or } (\rho,t) \models \psi \\
(\rho,t) \models \mathbf{X}\varphi & \text{iff} & (\rho,t+1) \models \varphi \\
(\rho,t) \models \varphi\,\mathbf{U}\,\psi & \text{iff} & \text{for some } t' \geq t:\ (\rho,t') \models \psi \text{ and} \\
 & & \text{for all } t \leq t'' < t' : (\rho,t'') \models \varphi
\end{array}$$

We write $\rho \models \varphi$ as a shorthand for $(\rho,0) \models \varphi$, in which case we say that $\rho$ *satisfies* $\varphi$. A formula $\varphi$ is *satisfiable* if there is some run satisfying $\varphi$. Checking satisfiability for LTL formulae is known to be PSPACE-complete [38], while the synthesis problem for LTL is 2EXPTIME-complete [34]. In addition to the LTL tense operators $\mathbf{X}$ ("in the next state…") and $\mathbf{U}$ ("… until …"), we make use of the two derived operators $\mathbf{F}$ ("eventually…") and $\mathbf{G}$ ("always…"), which are defined as follows [11]: $\mathbf{F}\varphi = \top\,\mathbf{U}\,\varphi$ and $\mathbf{G}\varphi = \neg\mathbf{F}\neg\varphi$.

**Strategies:** We model strategies for agents as finite-state machines with output. Formally, strategy $\sigma_i$ for agent $i \in \mathscr{N}$ is given by a structure $\sigma_i = (Q_i, next_i, do_i, q_i^0)$, where $Q_i$ is a finite set of machine

states, $next_i : Q_i \times Ac_1 \times \cdots \times Ac_n \to Q_i$ is the machine's state transformer function, $do_i : Q_i \to Ac_i$ is the machine's action selection function, and $q_i^0 \in Q_i$ is the machine's initial state. A collection of strategies, one for each agent $i \in \mathcal{N}$, is a *strategy profile*: $\vec{\sigma} = (\sigma_1, \ldots, \sigma_n)$. A strategy profile $\vec{\sigma}$ enacted in an arena $\mathcal{A}$ will generate a unique run, which we denote by $\rho(\vec{\sigma}, \mathcal{A})$; the formal definition is standard, and we will omit it here [17]. Where $\mathcal{A}$ is clear from the context, we will simply write $\rho(\vec{\sigma})$. For each agent $i \in \mathcal{N}$, we write $\Sigma_i$ for the set of all possible strategies for the agent and $\Sigma = \Sigma_1 \times \cdots \times \Sigma_n$ for the set of all possible strategy profiles for all players.

For a set of distinct agents $A \subseteq \mathcal{N}$, we write $\Sigma_A = \prod_{i \in A} \Sigma_i$ for the set of partial strategy profiles available to the group $A$ and $\Sigma_{-A} = \prod_{j \in \mathcal{N} \setminus A} \Sigma_j$ for the set of partial strategy profiles available to the set of all agents excluding those in $A$. Where $\vec{\sigma} = (\sigma_1, \ldots, \sigma_i, \ldots, \sigma_n)$ is a strategy profile and $\sigma_i'$ is a strategy for agent $i$, we denote the strategy profile obtained by replacing the $i$-th component of $\vec{\sigma}$ with $\sigma_i'$ by $(\vec{\sigma}_{-i}, \sigma_i')$. Similarly, given a strategy profile $\vec{\sigma}$ and a set of agents $A \subseteq \mathcal{N}$, we write $\vec{\sigma}_A = (\sigma_i)_{i \in A}$ to denote a partial strategy profile for the agents in $A$ and if $\vec{\sigma}_A' \in \Sigma_A$ is another partial strategy profile for $A$, we write $(\vec{\sigma}_{-A}, \vec{\sigma}_A')$ for the strategy profile obtained by replacing $\vec{\sigma}_A$ in $\vec{\sigma}$ with $\vec{\sigma}_A'$.

**Games, Utilities, and Preferences:** We obtain a *concurrent game* from an arena $\mathcal{A}$ by associating with each agent $i$ a goal $\gamma_i$, represented as an LTL formula. Formally, a concurrent game $\mathcal{G}$ is given by a structure

$$\mathcal{G} = (\mathcal{S}, \mathcal{N}, Ac_1, \ldots, Ac_n, \mathcal{T}, \mathcal{C}, \mathcal{L}, s_0, \gamma_1, \ldots, \gamma_n),$$

where $(\mathcal{S}, \mathcal{N}, Ac_1, \ldots, Ac_n, \mathcal{T}, \mathcal{C}, \mathcal{L}, s_0)$ is a concurrent game arena, and $\gamma_i$ is the LTL goal of agent $i$, for each $i \in \mathcal{N}$. Runs in a concurrent game $\mathcal{G}$ are defined over the game's arena $\mathcal{A}$, and hence we use the notations $\rho(\vec{\sigma}, \mathcal{G})$ and $\rho(\vec{\sigma}, \mathcal{A})$ interchangeably. When the game or arena is clear from the context, we omit the $\mathcal{G}$ and simply write $\rho(\vec{\sigma})$. Given a strategy profile $\vec{\sigma}$, the generated run $\rho(\vec{\sigma})$ will satisfy the goals of some agents and not satisfy the goals of others, that is, there will be a set $W(\vec{\sigma}) = \{i \in \mathcal{N} : \rho(\vec{\sigma}) \models \gamma_i\}$ of *winners* and a set $L(\vec{\sigma}) = \mathcal{N} \setminus W(\vec{\sigma})$ of *losers*.

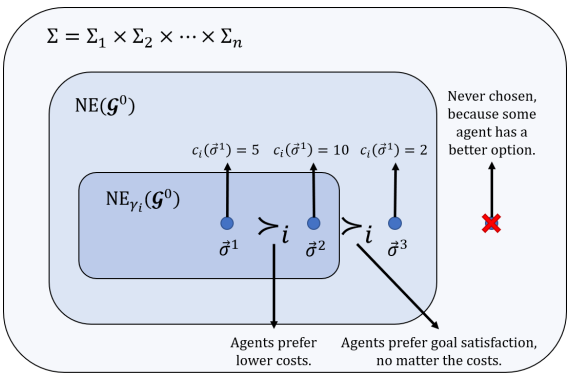
We are now ready to define preferences for agents. Our basic idea is that, as in [40], agents' preferences are structured: they first desire to accomplish their goal, and secondarily desire to minimise their costs. To capture this idea, it is convenient to define preferences via utility functions $u_i$ over runs, where $i$'s utility for a run $\rho$ is

$$u_i(\rho) = \begin{cases} 1 + c_i^* - \mathcal{C}_i(\rho) & \text{if } \rho \models \gamma_i \\ -\mathcal{C}_i(\rho) & \text{otherwise.} \end{cases}$$

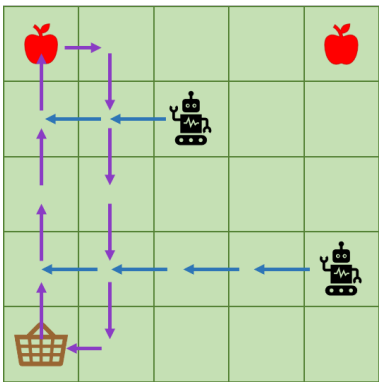
Defined in this way, if an agent $i$ gets their goal achieved, their utility will lie in the range $[1, c_i^* + 1]$ (depending on the cost she incurs), whereas if she does not achieve their goal, then their utility will lie within $[-c_i^*, 0]$. Preference relations $\succeq_i$ over runs are then defined in the obvious way: $\rho_1 \succeq_i \rho_2$ if and only if $u_i(\rho_1) \geq u_i(\rho_2)$, with indifference relations $\sim_i$ and strict preference relations $\succ_i$ defined as usual.

**Nash equilibrium:** A strategy profile $\vec{\sigma}$ is a (pure strategy) Nash equilibrium if there is no agent $i$ and strategy $\sigma_i'$ such that $\rho(\vec{\sigma}_{-i}, \sigma_i') \succ_i \rho(\vec{\sigma})$. If such a strategy $\sigma_i'$ exists for a given agent $i$, we say that $\sigma_i'$ is a *beneficial deviation* for $i$ from $\vec{\sigma}$. Given a game $\mathcal{G}$, let $\text{NE}(\mathcal{G})$ denote its set of Nash equilibria. In general, Nash equilibria in this model of concurrent games may require agents to play infinite memory strategies [8], but we do not consider these in this study [1]. Where $\varphi$ is an LTL formula, we find it useful to define $\text{NE}_\varphi(\mathcal{G})$ to be the set of Nash equilibrium strategy profiles that result in $\varphi$ being satisfied: $\text{NE}_\varphi(\mathcal{G}) = \{\vec{\sigma} \in \text{NE}(\mathcal{G}) \mid \rho(\vec{\sigma}) \models \varphi\}$. It is sometimes useful to consider a concurrent game that is modified so that no costs are incurred in it. We call such a game a *cost-free game*. Where $\mathcal{G}$ is a game, let $\mathcal{G}^{\mathbf{0}}$ denote

[1] Even in the purely quantitative setting where all agents' goals are $\top$, it is still possible that some Nash equilibria require infinite memory [16].




(a)



(b)

Figure 1: (a) Illustration of the lexicographic quantitative and qualitative preferences of agents. (b) A concurrent game where two robots are situated in a grid world and are programmed to 1) never crash into another robot and 2) to secondarily minimise their limit-average costs. The arrows indicate how a run may be decomposed into a non-repeating and an infinitely-repeating component.

the game that is the same as $\mathcal{G}$ except that the cost function $\mathscr{C}^{\mathbf{0}}$ of $\mathcal{G}^{\mathbf{0}}$ is such that $\mathscr{C}^{\mathbf{0}}_i(s,\vec{\alpha}) = 0$ for all $i \in \mathcal{N}$, $s \in \mathcal{S}$, and $\vec{\alpha} \in \vec{Ac}$. Given this, the following is readily established (cf., [17]):

**Theorem 1.** *Given a game $\mathcal{G}$, the problem of checking whether* $\mathrm{NE}(\mathcal{G}^{\mathbf{0}}) \neq \emptyset$ *is* 2ExpTime*-complete.*

The notion of Nash equilibrium is closely related to the concept of beneficial deviations. Given how preferences are defined in this study, it will be useful to introduce terminology that captures the potential deviations that agents may have [19]. Firstly, given a game $\mathcal{G}$, we say that a strategy profile $\vec{\sigma}^1 \in \Sigma$ is *distinguishable* from another strategy profile $\vec{\sigma}^2 \in \Sigma$ if $\rho(\vec{\sigma}^1,\mathcal{G}) \neq \rho(\vec{\sigma}^2,\mathcal{G})$. Then, for an agent $i$, a strategy profile $\vec{\sigma}$, and an alternative strategy $\sigma'_i \neq \sigma_i$, we say that $\sigma'_i$ is an *initial deviation* for agent $i$ from strategy profile $\vec{\sigma}$, written $\vec{\sigma} \rightarrow_i (\vec{\sigma}_{-i},\sigma'_i)$, if we have $i \in W(\vec{\sigma}) \Rightarrow i \in W(\vec{\sigma}_{-i},\sigma'_i)$ and strategy profile $\vec{\sigma}$ is distinguishable from $(\vec{\sigma}_{-i},\sigma'_i)$.

## 3 Taxation Schemes

We now introduce a model of incentives for concurrent games. For incentives to work, they clearly must appeal to an agent's preferences $\succeq_i$. As we saw above, incentives for our games are defined with respect to both goals and costs: an agent's primary desire is to see their goal achieved – the desire to minimise costs is strictly secondary to this. We will assume that we cannot change agents' goals: they are assumed to be fixed and immutable. It follows that any incentives we offer an agent to alter their behaviour must appeal to the costs incurred by that agent. Our basic model of incentives assumes that we can alter the cost structure of a game by imposing *taxes*, which depend on the collective actions that agents choose in different states. Taxes may increase an agent's costs, influencing their preferences and rational choices.

Formally, we model static taxation schemes as functions $\tau : \mathcal{S} \times \vec{Ac} \rightarrow \mathbb{R}^n_+$. A static taxation scheme $\tau$ imposed on a game $\mathcal{G} = (\mathcal{S},\mathcal{N},Ac_1,\ldots,Ac_n,\mathcal{T},\mathscr{C},\mathscr{L},s_0,\gamma_1,\ldots,\gamma_n)$ will result in a new game, which we denote by

$$\mathcal{G}^{\tau} = (\mathcal{S},\mathcal{N},Ac_1,\ldots,Ac_n,\mathcal{T},\mathscr{C}^{\tau},\mathscr{L},s_0,\gamma_1,\ldots,\gamma_n),$$

which is the same as $\mathcal{G}$ except that the cost function $\mathscr{C}^{\tau}$ of $\mathcal{G}^{\tau}$ is defined as $\mathscr{C}^{\tau}(s,\vec{\alpha}) = \mathscr{C}(s,\vec{\alpha}) + \tau(s,\vec{\alpha})$. Similarly, we write $\mathcal{A}^{\tau}$ to denote the arena with modified cost function $\mathscr{C}^{\tau}$ associated with $\mathcal{G}^{\tau}$ and $u^{\tau}_i(\rho)$

to denote the utility function of agent $i$ over run $\rho$ with the modified cost function $\mathscr{C}^{\tau}$. Given $\mathscr{G}$ and a taxation scheme $\tau$, we write $\rho_1 \succeq_i^{\tau} \rho_2$ iff $u_i^{\tau}(\rho_1) \geq u_i^{\tau}(\rho_2)$. The indifference relations $\sim_i^{\tau}$ and strict preference relations $\succ_i^{\tau}$ are defined analogously.

The model of static taxation schemes has the advantage of simplicity, but it is naturally limited in the range of behaviours it can incentivise—particularly with respect to behaviours $\Upsilon$ expressed as LTL formulae. To overcome this limitation, we therefore introduce a *dynamic* model of taxation schemes. This model essentially allows a designer to impose taxation schemes that can choose to tax the same action in different amounts, depending on the history of the run to date. A very natural model for dynamic taxation schemes is to describe them using a finite state machine with output—the same approach that we used to model strategies for individual agents. Formally, a *dynamic taxation scheme* $T$ is defined by a tuple $T = (Q_T, next_T, do_T, q_T^0)$ where $Q_T$ is a finite set of taxation machine states, $next_T : Q_T \times Ac_1 \times \cdots \times Ac_n \to Q_T$ is the transition function of the machine, $q_T^0 \in Q_T$ is the initial state, and $do_T : Q_T \to (\mathscr{S} \times \vec{Ac} \to \mathbb{R}_+^n)$ is the output function of the machine. With this, let $\mathscr{T}$ be the set of all dynamic taxation schemes for a game $\mathscr{G}$. As a run unfolds, we think of the taxation machine being executed alongside the strategies. At each time step, the machine outputs a static taxation scheme, which is applied at that time step only, with $do_T(q_T^0)$ being the initial taxation scheme imposed.

When we impose dynamic taxation schemes, we no longer have a simple transformation $\mathscr{G}^{\tau}$ on games as we did with static taxation schemes $\tau$. Instead, we define the effect of a taxation scheme with respect to a run $\rho$. Formally, given a run $\rho$ of a game $\mathscr{G}$, a dynamic taxation scheme $T$ induces an infinite sequence of static taxation schemes, which we denote by $t(\rho, T)$. We can think of this sequence as a function $t(\rho, T) : \mathbb{N} \to (\mathscr{S} \times \vec{Ac} \to \mathbb{R}_+^n)$. We denote the cost of the run $\rho$ in the presence of a dynamic taxation scheme $T$ by $\mathscr{C}^T(\rho)$:

$$\mathscr{C}^T(\rho) = \liminf_{u \to \infty} \frac{1}{u} \sum_{v=0}^{u} \mathscr{C}(\rho, v) + \underbrace{t(\rho, T)(v)(s(\rho, v), \vec{\alpha}(\rho, v))}_{(*)}$$

The expression $(*)$ denotes the vector of taxes incurred by the agents as a consequence of performing the action profile which they chose at time step $v$ on the run $\rho$. The cost $\mathscr{C}_i^T(\rho)$ to agent $i$ of the run $\rho$ under $T$ is then given by the $i$-th component of $\mathscr{C}^T(\rho)$.

**Example 1.** *Two robots are situated in a grid world (Figure 1b), where atomic propositions represent events where a robot picks up an apple (label $a_{ij}$ represents agent i picking up apple j), has delivered an apple to the basket (label $b_i$ represents agent i delivering an apple to the basket), or where the robots have crashed into each other (label c). Additionally, suppose that both robots are programmed with LTL goals $\gamma_1 = \gamma_2 = \mathbf{G}\neg c$. In this way, the robots are not pre-programmed to perform specific tasks, and it is therefore the duty of the principal to design taxes that motivate the robots to perform a desired function, e.g., pick apples and deliver them to the basket quickly. Because the game is initially costless, there is an infinite number of Nash equilibria that could arise from this scenario and it is by no means obvious that the robots will choose one in which they perform the desired function. Hence, the principal may attempt to design a taxation scheme to eliminate those that do not achieve their objective, thus motivating the robots to collect apples and deliver them to the basket. Clearly, using dynamic taxation schemes affords the principal more control over how the robots should accomplish this than static taxation schemes.*

## 4 Nash Implementation

We consider the scenario in which a principal, who is external to the game, has a particular goal that they wish to see satisfied within the game; in a general economic setting, the goal might be intended

to capture some principle of social welfare, for example. In our setting, the goal is specified as an LTL formula $\Upsilon$, and will typically represent a desirable system/global behaviour. The principal has the power to influence the game by choosing a taxation scheme and imposing it upon the game. Then, given a game $\mathscr{G}$ and a goal $\Upsilon$, our primary question is whether it is possible to design a taxation scheme $T$ such that, assuming the agents, individually and independently, act rationally (by choosing strategies $\vec{\sigma}$ that collectively form a Nash equilibrium in the modified game), the goal $\Upsilon$ will be satisfied in the run $\rho(\vec{\sigma})$ that results from executing the strategies $\vec{\sigma}$. In this section, we will explore two ways of interpreting this problem.

**E-Nash Implementation:** A goal $\Upsilon$ is *E-Nash implemented* by a taxation scheme $T$ in $\mathscr{G}$ if there is a Nash equilibrium strategy profile $\vec{\sigma}$ of the game $\mathscr{G}^T$ such that $\rho(\vec{\sigma}) \models \Upsilon$. The notion of E-Nash implementation is thus analogous to the E-Nash concept in rational verification [14, 15]. Observe that, if the answer to this question is "yes" then this implies that the game $\mathscr{G}^T$ has at least one Nash equilibrium. Let us define the set $\text{ENI}(\mathscr{G}, \Upsilon)$ to be the set of taxation schemes $T$ that E-Nash implements $\Upsilon$ in $\mathscr{G}$:

$$\text{ENI}(\mathscr{G}, \Upsilon) = \{T \in \mathscr{T} \mid \text{NE}_\Upsilon(\mathscr{G}^T) \neq \emptyset\} .$$

The obvious decision problem is then as follows:

E-NASH IMPLEMENTATION:
*Given*: Game $\mathscr{G}$, LTL goal $\Upsilon$.
*Question*: Is it the case that $\text{ENI}(\mathscr{G}, \Upsilon) \neq \emptyset$?

This decision problem proves to be closely related to the E-NASH problem [14, 15], and the following result establishes its complexity:

**Theorem 2.** E-NASH IMPLEMENTATION *is* 2EXPTIME*-complete, even when* $\mathscr{T}$ *is restricted to static taxation schemes.*

*Proof.* For membership, we can check whether $\Upsilon$ is satisfied on any Nash equilibrium of the cost-free concurrent game $\mathscr{G}^{\mathbf{0}}$ obtained from $\mathscr{G}$ by effectively removing its cost function using a static taxation scheme which makes all costs uniform for all agents. This then becomes the E-NASH problem, known to be 2EXPTIME-complete. The answer will be "yes" iff $\Upsilon$ is satisfied on some Nash equilibrium of $\mathscr{G}^{\mathbf{0}}$; and if the answer is "yes", then observing that $\text{NE}(\mathscr{G}^T) \subseteq \text{NE}(\mathscr{G}^{\mathbf{0}})$ for all taxation schemes $T \in \mathscr{T}$ [40], the given LTL goal $\Upsilon$ can be E-Nash implemented in $\mathscr{G}$. For hardness, we can reduce the problem of checking whether a cost-free concurrent game $G$ has a Nash equilibrium (Theorem 1). Simply ask whether $\Upsilon = \top$ can be E-Nash implemented in $\mathscr{G}^{\mathbf{0}}$.

For the second part of the result, observe that the reduction above only involves removing the costs from the game and checking the answer to E-NASH, which can be done using a simple static taxation scheme. Hardness follows in a similar manner. □

**A-Nash Implementation:** The universal counterpart of E-Nash implementation is *A-Nash Implementation*. We say that $\Upsilon$ is *A-Nash implemented* by $T$ in $\mathscr{G}$ if we have both 1) $\Upsilon$ is E-Nash implemented by $T$ in game $\mathscr{G}$; and 2) $\text{NE}(\mathscr{G}^T) = \text{NE}_\Upsilon(\mathscr{G}^T)$. We thus define $\text{ANI}(\mathscr{G}, \Upsilon)$ as follows:

$$\text{ANI}(\mathscr{G}, \Upsilon) = \{T \in \mathscr{T} \mid \text{NE}(\mathscr{G}^T) = \text{NE}_\Upsilon(G^T) \neq \emptyset\}$$

The decision problem is then:

A-NASH IMPLEMENTATION:
*Given*: Game $\mathscr{G}$, LTL goal $\Upsilon$.
*Question*: Is it the case that $\text{ANI}(\mathscr{G}, \Upsilon) \neq \emptyset$?

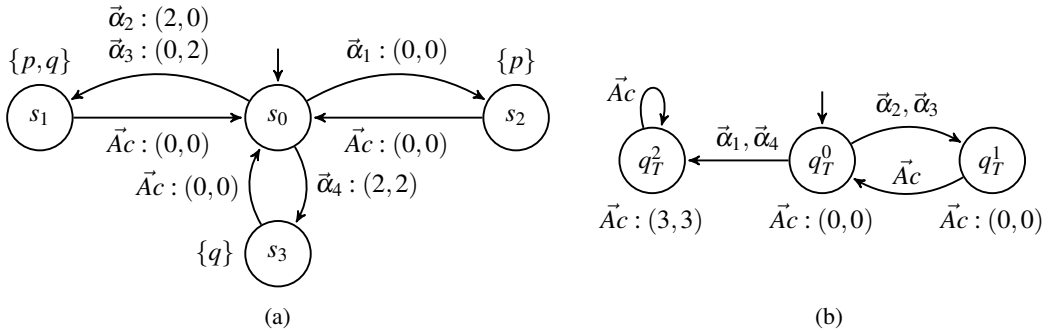



Figure 2: (a): A two-agent concurrent game $\mathscr{G}$ with action sets $Ac_1 = \{a,b\}$ and $Ac_2 = \{c,d\}$ and goals $\gamma_1 = \gamma_2 = \mathbf{GF}p$, where we let $\vec{\alpha}_1 = (a,c), \vec{\alpha}_2 = (a,d), \vec{\alpha}_3 = (b,c), \vec{\alpha}_4 = (b,d)$. Cost vectors associated with sets denote that all action profiles within the set are assigned those costs. (b): A dynamic taxation scheme that could be imposed on the agents in the game from (a). Labels below the states represent a static taxation scheme that applies a uniform tax for all agents and all action profiles.

The following result shows that, unlike the case of E-Nash implementation, dynamic taxation schemes are *strictly more powerful* than static taxation schemes for A-Nash implementation. It can be verified that the game in Figure 2a, the taxation scheme in Figure 2b, and the principal's goal being $\Upsilon = G(p \leftrightarrow q)$ are witnesses to this result (see Appendix for the full proof):

**Proposition 1.** *There exists a game $\mathscr{G}$ and an LTL goal $\Upsilon$ such that* $\text{ANI}(\mathscr{G}, \Upsilon) \neq \emptyset$*, but not if $\mathscr{T}$ is restricted to static taxation schemes.*

Before proceeding with the A-NASH IMPLEMENTATION problem, we will need to introduce some additional terminology and concepts, beginning first with deviation graphs, paths, and cycles. A *deviation graph* is a directed graph $\Gamma = (\mathscr{V}, E)$, where $\mathscr{V} \subseteq \Sigma$ is a set of nodes which represent strategy profiles in $\Sigma$ and $E \subseteq \{(\vec{\sigma}, \vec{\sigma}') \in \mathscr{V} \times \mathscr{V} \mid \vec{\sigma} \rightarrow_i \vec{\sigma}' \text{ for some } i \in \mathscr{N}\}$ is a set of directed edges between strategy profiles that represent initial deviations. Additionally, we say that a dynamic taxation scheme $T$ *induces* a deviation graph $\Gamma = (\mathscr{V}, E)$ if for every $(\vec{\sigma}, \vec{\sigma}') \in \mathscr{V} \times \mathscr{V}$, it holds that $\vec{\sigma}' \succ_i^T \vec{\sigma}$ for some $i \in \mathscr{N}$ if and only if $(\vec{\sigma}, \vec{\sigma}') \in E$. In other words, if the edges in a deviation graph precisely capture all of the beneficial deviations between its nodes under $T$, then the deviation graph is said to be induced by $T$.[2] Then, a *deviation path* is simply any path $P = (\vec{\sigma}^1, \ldots, \vec{\sigma}^m)$ within a deviation graph $\Gamma$ where $(\vec{\sigma}^j, \vec{\sigma}^{j+1}) \in E$ for all $j \in \{1, \ldots, m-1\}$.

Because the principal is only able to observe the actions taken by the agents and not their strategies directly, any taxation scheme that changes the cost of some strategy profile $\vec{\sigma}$ will also change the cost of all strategy profiles that are indistinguishable from $\vec{\sigma}$ by the same amount. This naturally suggests that we modify the concept of a deviation path to take indistinguishability into account. To this end, we say that a sequence of runs $P_o = (\rho^1, \rho^2, \ldots, \rho^m)$ is an *observed deviation path* in a deviation graph $\Gamma = (\mathscr{V}, E)$ if there exists an *underlying tuple* $(\vec{\sigma}^1, \vec{\sigma}^2, \ldots, \vec{\sigma}^m)$ such that for all $j \in \{1, \ldots, m\}$, it holds that 1) $\rho^j = \rho(\vec{\sigma}^j)$, and 2) if $j < m$, then $(\vec{\sigma}^j, \vec{\sigma}^{j+1'}) \in E$ for some $\vec{\sigma}^{j+1'}$ such that $\rho(\vec{\sigma}^{j+1'}) = \rho(\vec{\sigma}^{j+1})$. Then, a *deviation cycle* is a deviation path $(\vec{\sigma}^1, \ldots, \vec{\sigma}^m)$ where $\rho(\vec{\sigma}^1) = \rho(\vec{\sigma}^m)$. A deviation path $P = (\vec{\sigma}^1, \vec{\sigma}^2, \ldots, \vec{\sigma}^m)$ is said to *involve* an agent $i$ if $\vec{\sigma}^j \rightarrow_i \vec{\sigma}^{j+1}$ for some $j \in \{1, \ldots, m-1\}$ and similarly, an observed deviation path $P_o$ in a deviation graph involves agent $i$ if the analogous property holds for all of its underlying sets. Given a game $\mathscr{G}$ and a set of strategy profiles $X$, a taxation scheme $T$ *eliminates*

[2] This definition implies that a taxation scheme may induce many possible deviation graphs in general, depending on the nodes selected to be part of the graph.

$X$ if $NE(\mathscr{G}^T) \cap X = \emptyset$. Finally, a set of strategy profiles $X$ is said to be *eliminable* if there exists a taxation scheme that eliminates it. With this, we can characterise the conditions under which a finite set of strategy profiles is eliminable:

**Proposition 2.** *Let $\mathscr{G}$ be a game and $X \subset \Sigma$ be a finite set of strategy profiles in $\mathscr{G}$. Then, $X$ is eliminable if and only if there exists a finite deviation graph $\Gamma = (\mathscr{V}, E)$ that satisfies the following properties: 1) For every $\vec{\sigma} \in X$, there is some $\vec{\sigma}' \in \mathscr{V}$ such that $(\vec{\sigma}, \vec{\sigma}') \in E$; and 2) Every deviation cycle in $\Gamma$ involves at least two agents.*

*Proof Sketch.* The forward direction follows by observing that if all deviation graphs fail to satisfy at least one of the two properties, then every deviation graph will either fail to eliminate some $\vec{\sigma} \in X$ if induced, or will not be inducible by any dynamic taxation scheme. The backward direction can be established by constructing a dynamic taxation scheme $T^\Gamma$ that induces a deviation graph $\Gamma$ satisfying the two properties. Using these properties, it follows that $T^\Gamma$ eliminates $X$. □

To conclude our study of dynamic taxation schemes, we present a characterisation of the A-Nash implementation problem.[3]

**Theorem 3.** *Let $\mathscr{G}$ be a game and $\Upsilon$ be an LTL formula. Then $\text{ANI}(\mathscr{G}, \Upsilon) \neq \emptyset$ if and only if the following conditions hold:*

1. $\text{ENI}(\mathscr{G}, \Upsilon) \neq \emptyset$;
2. $\text{NE}_{\neg\Upsilon}(\mathscr{G}^{\mathbf{0}})$ *is eliminable.*

*Proof.* For the forward direction, it follows from the definition of the problem that if $\text{ENI}(\mathscr{G}, \Upsilon) = \emptyset$, then $\text{ANI}(\mathscr{G}, \Upsilon) = \emptyset$. Moreover, it is also clear that if $\text{NE}_{\neg\Upsilon}(\mathscr{G}^{\mathbf{0}})$ is not eliminable, then it is impossible to design a (dynamic) taxation scheme such that only good equilibria remain in the game and hence, $\text{ANI}(\mathscr{G}, \Upsilon) = \emptyset$.

For the backward direction, suppose that the two conditions hold and let $T$ be a taxation scheme that only affects the limiting-average costs incurred by agents under strategy profiles in $\text{NE}_{\neg\Upsilon}(\mathscr{G}^{\mathbf{0}})$, and eliminates this set. Such a taxation scheme is guaranteed to exist by the assumption that condition (2) holds and because it is known that no good equilibrium is indistinguishable from a bad one. Now consider a static taxation scheme $\tau$ such that $c_i(s, \vec{\alpha}) + \tau_i(s, \vec{\alpha}) = \hat{c}$ for all $i \in \mathscr{N}$, $(s, \vec{\alpha}) \in \mathscr{S} \times \vec{Ac}$, and some $\hat{c} \geq \max_{i \in \mathscr{N}} c_i^*$. Combining $\tau$ with $T$ gives us a taxation scheme $T^*$ such that for each state $q \in Q_{T^*} = Q_T$ and $(s, \vec{\alpha}) \in \mathscr{S} \times \vec{Ac}$, we have $do_{T^*}(q)(s, \vec{\alpha}) = do_T(q)(s, \vec{\alpha}) + \tau(s, \vec{\alpha})$. Now, because $T$ eliminates $\text{NE}_{\neg\Upsilon}(\mathscr{G}^{\mathbf{0}})$, and $\text{NE}(\mathscr{G}^\tau) = \text{NE}(\mathscr{G}^{\mathbf{0}})$, it follows that $T^*$ eliminates $\text{NE}_{\neg\Upsilon}(\mathscr{G}^{\mathbf{0}})$. Finally, note that because the satisfaction of an LTL formula on a given run is solely dependent on the run's trace, it follows that all good equilibria, i.e., strategy profiles in $\text{NE}_\Upsilon(\mathscr{G}^{\mathbf{0}})$, are distinguishable from all bad equilibria, so we have $\text{NE}_\Upsilon(\mathscr{G}^{\mathbf{0}}) \cap \text{NE}(\mathscr{G}^{T^*}) \neq \emptyset$. □

It is straightforward to see that A-NASH IMPLEMENTATION is 2EXPTIME-hard via a simple reduction from the problem of checking whether a Nash equilibrium exists in a concurrent game – simply ask if the formula $\top$ can be A-Nash implemented in $\mathscr{G}^{\mathbf{0}}$. However, it is an open question whether a matching upper bound exists and we conjecture that it does not. This problem is difficult primarily for two reasons. Firstly, it is well documented that Nash equilibria may require infinite memory in games with

[3]Note that, in general, Proposition 2 cannot be directly applied to Theorem 3, because it is assumed that the set to be eliminated is finite, whereas $\text{NE}_{\neg\Upsilon}(\mathscr{G}^{\mathbf{0}})$ is generally infinite. However, this can be reconciled if some restriction is placed on the agents' strategies so that $\Sigma$ is finite, which is the case in many game-theoretic situations of interest, e.g., in games with memoryless, or even bounded memory, strategies – both used to model bounded rationality.

lexicographic $\omega$-regular and mean-payoff objectives [8], and the complexity of deciding whether a Nash equilibrium even exists in games with our model of preferences has yet to be settled [15]. Secondly, Theorem 3 and Proposition 2 suggest that unless the strategy space is restricted to a finite set, a taxation scheme that A-Nash implements a formula may require reasoning over an infinite deviation graph, and hence require infinite memory. Nevertheless, our characterisation under such restrictions provides the first step towards understanding this problem in the more general setting.

## 5 Related Work and Conclusions

This work was motivated by [40], and based on that work, presents four main contributions: the introduction of *static and dynamic* taxation schemes as an extension to concurrent games expanding the model in (one-shot) Boolean games [40, 18, 19]; a study of the *complexity* of some of the most relevant computational decision problems building on previous work in rational verification [14, 17, 15]; evidence (formal proof) of the strict *advantage of dynamic taxation schemes* over static ones, which illustrates the role of not just observability but also *memory* to a principal's ability to (dis)incentivise certain outcomes [13, 20]; and a full characterisation of the *eliminability of sets of strategy profiles* under dynamic taxation schemes and the A-Nash implementation problem.

The incentive design problem has been studied in many different settings, and [35] group existing approaches broadly into those from the economics, control theory, and machine learning communities. However, more recent works in this area adopt multi-disciplinary methods such as automated mechanism design [30, 27, 37, 3], which typically focus on the problem of constructing incentive-compatible mechanisms to optimise a particular objective such as social welfare. Other approaches in this area reduce mechanism design to a program synthesis problem [29] or a satisfiability problem for quantitative strategy logic formulae [25, 28]. The notion of dynamic incentives has also been investigated in (multi-agent) learning settings [7, 26, 36, 42, 10]. These works focus solely on adaptively modifying the rewards for quantitative reward-maximising agents. In contrast, our model of agent utilities more naturally captures fundamental constraints on the principal's ability to (dis)incentivise certain outcomes due to the lexicographic nature of agents' preferences [4].

Another area closely related to incentives is that of norm design [23]. Norms are often modelled as the encouragement or prohibition of actions that agents may choose to take by a regulatory agent. The most closely related works in this area are those of [21, 31, 1], who study the problem of synthesising dynamic norms in different classes of concurrent games to satisfy temporal logic specifications. Whereas norms in these frameworks have the ability to *disable* actions at runtime, our model confers only the power to *incentivise* behaviours upon the principal. Finally, other studies model norms with violation penalties, but differ from our work in how incentives, preferences, and strategies are modelled [6, 5, 9].

In summary, a principal's ability to align self-interested decision-makers' interests with higher-order goals presents an important research challenge for promoting cooperation in multi-agent systems. The present study highlights the challenges associated with incentive design in the presence of constraints on the kinds of behaviours that can be elicited, makes progress on the theoretical aspects of this endeavour through an analysis of taxation schemes, and suggests several avenues for further work. Promising directions include extensions of the game model to probabilistic/stochastic or learning settings, finding optimal complexity upper bounds for the A-Nash implementation problems, and consideration of different formal models of incentives. We expect that this and such further investigations will positively contribute to our ability to develop game-theoretically aware incentives in multi-agent systems.

## References


[1] Natasha Alechina, Giuseppe De Giacomo, Brian Logan & Giuseppe Perelli (2022): *Automatic Synthesis of Dynamic Norms for Multi-Agent Systems*. In: *19th International Conference on Principles of Knowledge Representation and Reasoning, KR 2022: KR 2022*, doi:10.24963/kr.2022/2.

[2] Rajeev Alur, Thomas A. Henzinger & Orna Kupferman (2002): *Alternating-time temporal logic*. *J. ACM* 49(5), pp. 672–713, doi:10.1145/585265.585270.

[3] Jan Balaguer, Raphael Koster, Christopher Summerfield & Andrea Tacchetti (2022): *The Good Shepherd: An Oracle Agent for Mechanism Design*. *arXiv preprint arXiv:2202.10135*, doi:10.48550/arXiv.2202.10135.

[4] Michael Bräuning, Eyke Hüllermeier, Tobias Keller & Martin Glaum (2017): *Lexicographic preferences for predictive modeling of human decision making: A new machine learning method with an application in accounting*. *European Journal of Operational Research* 258(1), pp. 295–306, doi:10.1016/j.ejor.2016.08.055.

[5] Nils Bulling & Mehdi Dastani (2016): *Norm-based mechanism design*. *Artificial Intelligence* 239, pp. 97–142, doi:10.1016/j.artint.2016.07.001.

[6] Henrique Lopes Cardoso & Eugénio Oliveira (2009): *Adaptive Deterrence Sanctions in a Normative Framework*. In: *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, 2, pp. 36–43, doi:10.1109/WI-IAT.2009.123.

[7] Roberto Centeno & Holger Billhardt (2011): *Using incentive mechanisms for an adaptive regulation of open multi-agent systems*. In: *Twenty-Second International Joint Conference on Artificial Intelligence*, doi:10.5591/978-1-57735-516-8/IJCAI11-035.

[8] Krishnendu Chatterjee, Thomas A. Henzinger & Marcin Jurdzinski (2005): *Mean-Payoff Parity Games*. In: *20th IEEE Symposium on Logic in Computer Science (LICS 2005), 26-29 June 2005, Chicago, IL, USA, Proceedings*, IEEE Computer Society, pp. 178–187, doi:10.1109/LICS.2005.26.

[9] Davide Dell'Anna, Mehdi Dastani & Fabiano Dalpiaz (2020): *Runtime Revision of Sanctions in Normative Multi-Agent Systems*. *Autonomous Agents and Multi-Agent Systems* 34(2), doi:10.1007/s10458-020-09465-8.

[10] Mahmoud Elbarbari, Florent Delgrange, Ivo Vervlimmeren, Kyriakos Efthymiadis, Bram Vanderborght & Ann Nowé (2022): *A framework for flexibly guiding learning agents*. *Neural Computing and Applications*, pp. 1–17, doi:10.1007/s00521-022-07396-x.

[11] E. Allen Emerson (1990): *Temporal and Modal Logic*. In Jan van Leeuwen, editor: *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics*, Elsevier and MIT Press, pp. 995–1072, doi:10.1016/b978-0-444-88074-1.50021-4.

[12] Dana Fisman, Orna Kupferman & Yoad Lustig (2010): *Rational synthesis*. In: *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Springer, pp. 190–204, doi:10.1007/978-3-642-12002-2_16.

[13] Sanford J. Grossman & Oliver D. Hart (1992): *An analysis of the principal-agent problem*. In: *Foundations of insurance economics*, Springer, pp. 302–340, doi:10.1007/978-94-015-7957-5_16.

[14] Julian Gutierrez, Paul Harrenstein & Michael J. Wooldridge (2017): *Reasoning about equilibria in game-like concurrent systems*. *Annals of Pure and Applied Logic* 169(2), pp. 373–403, doi:10.1016/j.apal.2016.10.009.

[15] Julian Gutierrez, Aniello Murano, Giuseppe Perelli, Sasha Rubin, Thomas Steeples & Michael J. Wooldridge (2021): *Equilibria for games with combined qualitative and quantitative objectives*. *Acta Informatica* 58(6), pp. 585–610, doi:10.1007/s00236-020-00385-4.

[16] Julian Gutierrez, Muhammad Najib, Giuseppe Perelli & Michael J. Wooldridge (2019): *Equilibrium Design for Concurrent Games*. In: *30th International Conference on Concurrency Theory*, doi:10.4230/LIPIcs.CONCUR.2019.22.

[17] Julian Gutierrez, Muhammad Najib, Giuseppe Perelli & Michael J. Wooldridge (2020): *Automated temporal equilibrium analysis: Verification and synthesis of multi-player games*. *Artificial Intelligence* 287, p. 103353, doi:10.1016/j.artint.2020.103353.

[18] Paul Harrenstein, Paolo Turrini & Michael J. Wooldridge (2014): *Hard and soft equilibria in boolean games*. In Ana L. C. Bazzan, Michael N. Huhns, Alessio Lomuscio & Paul Scerri, editors: *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '14, Paris, France, May 5-9, 2014*, IFAAMAS/ACM, pp. 845–852, doi:10.5555/2615731.2615867. Available at http://dl.acm.org/citation.cfm?id=2615867.

[19] Paul Harrenstein, Paolo Turrini & Michael J. Wooldridge (2017): *Characterising the Manipulability of Boolean Games*. In Carles Sierra, editor: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, ijcai.org, pp. 1081–1087, doi:10.24963/ijcai.2017/150.

[20] Bengt Holmstrom (1982): *Moral hazard in teams*. *The Bell journal of economics*, pp. 324–340, doi:10.2307/3003457.

[21] Xiaowei Huang, Ji Ruan, Qingliang Chen & Kaile Su (2016): *Normative Multiagent Systems: A Dynamic Generalization*. In: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, AAAI Press, p. 1123–1129.

[22] Kenneth S. Lyon & Dug Man Lee (2001): *Pigouvian tax and the congestion externality: a benefit side approach*. *Economics Research Institute Study Paper* 10, p. 1.

[23] Moamin A. Mahmoud, Mohd Sharifuddin Ahmad, Mohd Zaliman Mohd Yusoff & Aida Mustapha (2014): *A review of norms and normative multiagent systems*. *The Scientific World Journal* 2014, doi:10.1155/2014/684587.

[24] N. Gregory Mankiw (2009): *Smart taxes: An open invitation to join the pigou club*. *Eastern Economic Journal* 35(1), pp. 14–23, doi:10.1057/EEJ.2008.43.

[25] Bastien Maubert, Munyque Mittelmann, Aniello Murano & Laurent Perrussel (2021): *Strategic reasoning in automated mechanism design*. In: *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, 18, pp. 487–496, doi:10.24963/kr.2021/46.

[26] David Mguni, Joel Jennings, Emilio Sison, Sergio Valcarcel Macua, Sofia Ceppi & Enrique Munoz de Cote (2019): *Coordinating the Crowd: Inducing Desirable Equilibria in Non-Cooperative Systems*. In: *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '19, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, p. 386–394.

[27] David Mguni & Marcin Tomczak (2019): *Efficient reinforcement dynamic mechanism design*. In: *GAIW: Games, agents and incentives workshops, at AAMAS, Montreal, Canada*.

[28] Munyque Mittelmann, Bastien Maubert, Aniello Murano & Laurent Perrussel (2022): *Automated synthesis of mechanisms*. In: *31st International Joint Conference on Artificial Intelligence (IJCAI-22)*, International Joint Conferences on Artificial Intelligence Organization, pp. 426–432, doi:10.24963/ijcai.2022/61.

[29] Sai Kiran Narayanaswami, Swarat Chaudhuri, Moshe Vardi & Peter Stone (2022): *Automating Mechanism Design with Program Synthesis*. *Proc. of the Adaptive and Learning Agents Workshop (ALA 2022)*.

[30] David C Parkes, Ruggiero Cavallo, Florin Constantin & Satinder Singh (2010): *Dynamic incentive mechanisms*. *Ai Magazine* 31(4), pp. 79–94, doi:10.1609/aimag.v31i4.2316.

[31] Giuseppe Perelli (2019): *Enforcing equilibria in multi-agent systems*. In: *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 188–196, doi:10.5555/3306127.3331692.

[32] Arthur C. Pigou & Nahid Aslanbeigui (2017): *The Economics of Welfare*. Routledge, doi:10.4324/9781351304368.

[33] Amir Pnueli (1977): *The Temporal Logic of Programs*. In: *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*, IEEE Computer Society, pp. 46–57, doi:10.1109/SFCS.1977.32.

[34] Amir Pnueli & Roni Rosner (1989): *On the Synthesis of an Asynchronous Reactive Module*. In Giorgio Ausiello, Mariangiola Dezani-Ciancaglini & Simona Ronchi Della Rocca, editors: *Automata, Languages*

*and Programming, 16th International Colloquium, ICALP89, Stresa, Italy, July 11-15, 1989, Proceedings*, *Lecture Notes in Computer Science* 372, Springer, pp. 652–671, doi:10.1007/BFb0035790.

[35] Lillian J Ratliff, Roy Dong, Shreyas Sekar & Tanner Fiez (2019): *A perspective on incentive design: Challenges and opportunities*. *Annual Review of Control, Robotics, and Autonomous Systems* 2, pp. 305–338, doi:10.1146/ANNUREV-CONTROL-053018-023634.

[36] Lillian J Ratliff & Tanner Fiez (2020): *Adaptive incentive design*. *IEEE Transactions on Automatic Control* 66(8), pp. 3871–3878, doi:10.1109/tac.2020.3027503.

[37] Weiran Shen, Pingzhong Tang & Song Zuo (2019): *Automated Mechanism Design via Neural Networks*. In: *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '19, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, p. 215–223, doi:10.5555/3306127.3331696.

[38] A. Prasad Sistla & Edmund M. Clarke (1985): *The Complexity of Propositional Linear Temporal Logics*. *J. ACM* 32(3), pp. 733–749, doi:10.1145/3828.3837.

[39] Michael Ummels & Dominik Wojtczak (2011): *The complexity of Nash equilibria in limit-average games*. In: *International Conference on Concurrency Theory*, Springer, pp. 482–496, doi:10.1007/978-3-642-23217-6_32.

[40] Michael J. Wooldridge, Ulle Endriss, Sarit Kraus & Jérôme Lang (2013): *Incentive engineering for Boolean games*. *Artif. Intell.* 195, pp. 418–439, doi:10.1016/j.artint.2012.11.003.

[41] Michael J. Wooldridge, Julian Gutierrez, Paul Harrenstein, Enrico Marchioni, Giuseppe Perelli & Alexis Toumi (2016): *Rational Verification: From Model Checking to Equilibrium Checking*. In Dale Schuurmans & Michael P. Wellman, editors: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, AAAI Press, pp. 4184–4191, doi:10.1016/J.ARTINT.2017.04.003. Available at http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12268.

[42] Jiachen Yang, Ethan Wang, Rakshit Trivedi, Tuo Zhao & Hongyuan Zha (2021): *Adaptive Incentive Design with Multi-Agent Meta-Gradient Reinforcement Learning*. *arXiv preprint arXiv:2112.10859*, doi:10.5555/3535850.3536010.

# 6 Supplementary Material

**Proposition 1.** *There exists a game $\mathscr{G}$ and an LTL goal $\Upsilon$ such that* $\text{ANI}(\mathscr{G},\Upsilon) \neq \emptyset$*, but not if $\mathscr{T}$ is restricted to static taxation schemes.*

*Proof.* Consider the concurrent game $\mathscr{G}$ in Figure 2a. Intuitively, both agents desire to always eventually visit either $s_1$ or $s_2$. Suppose that the principal's objective is $\Upsilon = G(p \leftrightarrow q)$, i.e., they would like the agents to never visit $s_2$ or $s_3$. Firstly, observe that there is no *static* taxation scheme which can A-Nash implement $\Upsilon$, as any modification to the costs of the game will not eliminate any Nash equilibria where the agents visit $s_2$ or $s_3$ a finite number of times. This is due to the prefix-independence of costs in infinite games with limiting-average payoffs [39]. However, the dynamic taxation scheme depicted in Figure 2b A-Nash implements $\Upsilon$. To see this, observe that for any strategy profile that visits $s_2$ or $s_3$ a finite number of times, there exists a deviation for some agent to ensure that $s_2$ and $s_3$ are never visited. Such a deviation will result in all agents $i \in \{1,2\}$ satisfying their goals $\gamma_i$ and strictly reducing their average costs from at least $c_i^* + 1$ to some value strictly below this. This constitutes a beneficial deviation and hence, there is no Nash equilibrium under $T$ that does not satisfy $\Upsilon$. Moreover, any strategy profile $\vec{\sigma}$ that leads to the sequence of states $s(\rho(\vec{\sigma}),0:) = (s_0 s_1)^\omega$ is a Nash equilibrium of $\mathscr{G}^T$ and hence goal $\Upsilon$ is A-Nash implemented by $T$ in this game. □

**Proposition 2.** *Let $\mathscr{G}$ be a game and $X \subset \Sigma$ be a finite set of strategy profiles in $\mathscr{G}$. Then, $X$ is eliminable if and only if there exists a finite deviation graph $\Gamma = (\mathscr{V},E)$ that satisfies the following properties: 1) For every $\vec{\sigma} \in X$, there is some $\vec{\sigma}' \in \mathscr{V}$ such that $(\vec{\sigma},\vec{\sigma}') \in E$; and 2) Every deviation cycle in $\Gamma$ involves at least two agents.*

*Proof.* For the forward direction, suppose that there is no deviation graph $\Gamma$ satisfying both properties (1) and (2) in the statement. Then, for all deviation graphs $\Gamma$, either for some $\vec{\sigma} \in X$, there is no $\vec{\sigma}' \in \mathscr{V}$ such that $(\vec{\sigma},\vec{\sigma}') \in \mathscr{V}$, or there is some deviation cycle in $\Gamma$ involving only one agent. Now consider any deviation graph $\Gamma = (\mathscr{V},E)$, where $\mathscr{V} = X \cup \{\vec{\sigma}' \mid \vec{\sigma} \rightarrow_i \vec{\sigma}' \text{ for some } \vec{\sigma} \in X \text{ and } i \in \mathscr{N}\}$. In the first case, it is clear that any taxation scheme that induces $\Gamma$ does not eliminate $\{\vec{\sigma}\}$ and hence $X$. In the second case, no taxation scheme can induce the deviation graph $\Gamma$. To see why, suppose for contradiction that some taxation scheme $T$ induces $\Gamma$ and let $i$ be the agent for which there is a deviation cycle $C = \{\vec{\sigma}^1,\ldots,\vec{\sigma}^m\}$ in $\Gamma$ involving only agent $i$. Then, we have $\vec{\sigma}^1 \succ_i^T \vec{\sigma}^2 \succ_i^T \ldots \succ_i^T \vec{\sigma}^m$ and by transitivity of the preference relation $\succ_i^T$, we can conclude that $\vec{\sigma}^1 \succ_i^T \vec{\sigma}^m$. However, by definition of a deviation cycle, $\vec{\sigma}^1$ and $\vec{\sigma}^m$ are indistinguishable, so agent $i$ will always receive the same utility under both $\vec{\sigma}^1$ and $\vec{\sigma}^m$, no matter what taxation scheme is imposed on them and hence, we have a contradiction. From this, we can conclude that every deviation graph that can be induced by a taxation scheme does not eliminate $X$ and hence, $X$ is not eliminable, proving this part of the statement.

For the backward direction, assume that there is a deviation graph $\Gamma$ that satisfies both properties. Under this assumption, we will construct a dynamic taxation scheme $T$ that eliminates $X$. To assign the appropriate costs to different strategy profiles, we will make use of the lengths of deviation paths within $\Gamma$. For every $i \in \mathscr{N}$, let $\ell_i$ denote the length of the longest observed deviation path in $\Gamma$ that involves only agent $i$. Additionally, for all $\vec{\sigma} \in \mathscr{V}$, let $d_i(\rho(\vec{\sigma}))$ denote the length of the longest *observed* deviation path in $\Gamma$ that starts from $\rho(\vec{\sigma})$ and involves only $i$. The difference between these two quantities will serve as the basis for how much taxation an agent $i$ will incur for any given strategy profile in $\mathscr{V}$. Observe that because it is assumed that no deviation cycle involves only one agent, both quantities are well-defined and finite for all agents and strategy profiles. Then, for a deviation graph $\Gamma$ and a run $\rho$, let $\text{IN}(\rho,\Gamma)$ be the set of agents $i \in \mathscr{N}$ for which there is some pair of strategy profiles $\vec{\sigma},\vec{\sigma}' \in \mathscr{V}$ such that we have both

$(\vec{\sigma},\vec{\sigma}') \in E_D$ and $\rho = \rho(\vec{\sigma}')$. In other words, $\text{IN}(\rho,\Gamma)$ represents the set of agents who have an initial deviation from some other strategy profile in $\mathscr{V}$ to one that generates the run $\rho$. With this, we would like to construct a dynamic taxation scheme such that for any strategy profile $\vec{\sigma}$, the following criteria are satisfied:

- $C_i^T(\rho(\vec{\sigma})) \geq (\ell_i - d_i(\rho(\vec{\sigma}))) \cdot (c_i^* + 1)$ if $i \in \text{IN}(\rho,\Gamma)$;
- $C_i^T(\rho(\vec{\sigma})) = C_i(\rho(\vec{\sigma}))$ otherwise.

Intuitively, the idea is to ensure that for every edge $(\vec{\sigma},\vec{\sigma}') \in E$, the agent $i \in \mathscr{N}$ for whom $\vec{\sigma} \rightarrow_i \vec{\sigma}'$ gets taxed by a significantly higher amount for choosing $\vec{\sigma}$ compared to when they choose $\vec{\sigma}'$. To see why it is possible to construct such a taxation scheme, first observe that if $\rho \neq \rho'$ for any two runs $\rho,\rho'$, then there is some dynamic taxation scheme that can distinguish between the two by simply tracing out the two runs up to the first point in which they differ and then branching accordingly. From this point onwards, the dynamic taxation scheme can then output static taxation schemes, which assign different limiting average costs to the agents according to the above criteria. Extending this approach to a taxation scheme that distinguishes between all unique runs generated by elements of $\mathscr{V}$, it follows that there is a dynamic taxation scheme $T$ that satisfies the two criteria. Consequently, for all $(\vec{\sigma},\vec{\sigma}') \in E$, it follows that $\vec{\sigma}' \succ_i^T \vec{\sigma}$ because $\rho(\vec{\sigma}) \neq \rho(\vec{\sigma}')$ by definition of the initial deviation relation $\rightarrow_i$. Moreover, because it is assumed that no deviation cycle involves only one agent, $T$ gives rise to a strict total ordering $\succ_i^T$ on the elements of $\mathscr{V}$ for each $i \in \mathscr{N}$. Finally, by property $(1)$, it holds that for every $\vec{\sigma} \in X$, some agent has a beneficial deviation from $\vec{\sigma}$ to another $\vec{\sigma}' \in \mathscr{V}$ under $T$ and hence, $T$ eliminates $X$. □