A Graph-Theoretical Perspective on Law Design for Multiagent Systems

Qi Shi, Pavel Naumov

University of Southampton, United Kingdom
qi.shi@soton.ac.uk; p.naumov@soton.ac.uk

Abstract

A law in a multiagent system is a set of constraints imposed on agents' behaviours to avoid undesirable outcomes. The paper considers two types of laws: useful laws that, if followed, completely eliminate the undesirable outcomes and gap-free laws that guarantee that at least one agent can be held responsible each time an undesirable outcome occurs. In both cases, we study the problem of finding a law that achieves the desired result by imposing the minimum restrictions.

We prove that, for both types of laws, the minimisation problem is NP-hard even in the simple case of one-shot concurrent interactions. We also show that the approximation algorithm for the vertex cover problem in hypergraphs could be used to efficiently approximate the minimum laws in both cases.

Extended Version — https://arxiv.org/abs/2511.06361

1 Introduction

Suppose that three factories a, b, and c need to dump the same type of pollutant into a river. Each factory must dump once every three days. The assimilative capacity of the river endures at most two factories dumping per day. Otherwise, the fish in the river would be killed. To avoid the death of the fish, the local government would like to set a law that regulates the dumping activity. Building on the well-known legal maxim "everything which is not forbidden is allowed", we assume that *a law serves solely to identify the actions from which agents must abstain*. This is in line with the liberal rule-of-law perspective: a key virtue of the rule of law is the protection of individual freedom (Hayek 1944; Raz 1979).

In a simple form, a dumping law can assign each factory a fixed dumping day within a recurring three-day cycle by banning it from dumping on the other two days. For instance, the law specifies the set $L_0 = \{d_a^1, d_a^2, d_b^2, d_b^3, d_c^1, d_c^3\}$ of *banned* dumping actions, where d_x^i represents the action that factory x dumping on the i^{th} day in each three-day cycle. Under this law, factory a dumps on the third day (d_a^3), factory b dumps on the first day (d_b^1), and factory c dumps on the second day (d_c^2) of each three-day cycle. In other words, only one factory dumps on each day, and it is guaranteed that the fish will not be killed. In this case, we say that law L_0 is *useful* in terms of prohibiting the death of the fish. In general, we say that a law is ***useful*** if *the prohibited outcomes shall never appear when every agent obeys the law*. The term "useful" is adopted from (Shoham and Tennenholtz 1995), a pioneering work on law design in multiagent systems.

It is easily observable that the law L_0 above unnecessarily constrains the dumping behaviour of the factories. Note that, to avoid the death of the fish, it suffices to ensure that not all three factories dump on the same day. In other words, for any given day, the law only needs to prevent one factory from dumping. This means that the law $L_1 = \{d_a^1, d_b^2, d_c^3\}$ is also useful. However, law L_1 allows each factory one more day to dump in each three-day cycle, providing more flexibility toward the factories' production activities.

Observe that laws L_0 and L_1 are both useful, while L_1 sets fewer constraints than L_0 (*i.e.* $L_1 \subsetneq L_0$). In this case, we say that L_1 is a ***useful reduction*** of law L_0. Also, notice that any further reduction of law L_1 is no longer useful. For example, the reduced law $L_2 = \{d_a^1, d_b^2\}$ of law L_1 allows all factories to dump on the third day and kill the fish. In other words, law L_1 satisfies a minimality property regarding usefulness. In general, we say that a law is ***minimal-useful*** if *it is useful but cannot be further reduced while keeping usefulness*. Considering the minimality of law captures the idea of setting minimal constraints on society, which is in line with the opinion that "the minimal state is the most extensive state that can be justified. Any state more extensive violates people's rights ..." (Nozick 1974).

Essentially, the usefulness of a law captures the *ability of the law to prevent* the undesirable outcomes. This is a typical focus in the literature on law design in multiagent systems. While reliable, the usefulness requirement excludes the possibility of *coordination* among agents as a means of prevention. In our example, even without a law, the three factories can still negotiate a dumping plan to avoid the death of the fish. On one hand, a negotiated plan could be more adaptive to the production of the factories than a law and thus bring higher efficiency. On the other hand, a law should be stable (Jefferson 1787; Raz 1979). In contrast, coordination achieved by negotiation is flexible and thus can better accommodate changes in production. Although appealing, coordination is not always realisable in multiagent settings. In our case, the three factories may not be able to reach an agreement on the dumping plan due to the lack of communication or the conflict of interest. In general, coordination is even harder to achieve in complex multiagent systems, particularly when different types of agents exist (*e.g.* a traffic

system including autonomous vehicles and human drivers, or a ranch including sheep, sheepdogs, herders, and wolves).

In this paper, we relax the usefulness requirement and consider laws that can simultaneously *accommodate coordination and the failure of coordination*. Informally, under such laws, prohibited outcomes may appear when the agents *all obey the law but do not coordinate*. Nevertheless, there is always *at least one* ***principal agent*** *who has a* ***safe action*** *that is lawful, and enables the prevention of prohibited outcomes without the need for coordination*. As an example, law L_2 satisfies the above property. Recall that law L_2 only bans a from dumping on the first day and b from dumping on the second day. Consequently, even if all factories comply with law L_2, without coordination, the death of the fish may still happen when they simultaneously dump on the third day. However, factory c has a safe action under law L_2 (*i.e.* dumping on the first or the second day) that can *solely* prevent the death of the fish as long as a and b obey law L_2. By this means, law L_2 leaves factory c a chance to dump on the third day while keeping the fish alive by negotiation with other factories. Meanwhile, if the negotiation fails, factory c can still prevent the death of the fish on its own. In other words, factory c is a principal agent under law L_2.

Observe that, under such laws, if a prohibited outcome finally appears, then either there is an agent whose action breaks the law, or the principal agent could have solely prevented it with a safe action but fails to do so. In the former case, we say that the agent who breaks the law bears ***legal responsibility***. In the latter case, we say that the principal agent bears ***counterfactual responsibility***. In our example, if the fish is killed because all three factories dump on the first day, then factory a breaks law L_2 and thus a is legally responsible; if it happens on the second day, then factory b breaks law L_2 and thus b is legally responsible; if it happens on the third day, then factory c, the principal agent under law L_2, fails to utilise her safe action to prevent it, and thus c is counterfactually responsible. In a word, under those laws, *if a prohibited outcome happens, then there is at least one agent either legally or counterfactually responsible*. That is, a responsible agent can always be identified for any prohibited outcome. In this sense, we refer to such laws ***responsibility gap-free*** (*abbr.* ***gap-free***).

To further clarify our terminology, *counterfactual responsibility* is usually regarded as a form of moral responsibility (Robb 2023). It captures the *principle of alternative possibilities* (Frankfurt 1969): *a person is morally responsible for what she has done only if she could have done otherwise*. In the recent literature on responsibility in multiagent systems, the component "could have done otherwise" is commonly interpreted as an agent's strategic ability to guarantee prevention irrespective of the behaviour of other agents (Naumov and Tao 2019; Yazdanpanah et al. 2019; Baier, Funke, and Majumdar 2021; Shi 2024). Note that the former discussion about the principal agent's safe action to prevent prohibited outcomes aligns with such an interpretation. In this sense, our use of the term *counterfactual responsibility* in this paper (*i.e.* principal agents who fail to prevent are counterfactually responsible) is consistent with its treatment in the existing literature. As for *responsibility gap*, also called *responsibility void*, it is one of the important topics discussed in the ethics literature, especially in the context of artificial intelligence (Matthias 2004; Braham and van Hees 2011; Duijf 2018; Braham and van Hees 2018; Burton et al. 2020; Gunkel 2020; Langer et al. 2021; Goetze 2022). Informally, responsibility gap captures the situation where an undesired outcome occurs but no agent can be held responsible, which is usually regarded as "unwanted" (Hiller, Israel, and Heitzig 2022). Correspondingly, the term *gap-freeness* in this paper precisely captures the *absence of responsibility gaps* as discussed in the literature and serves as a desired property of multiagent systems.

Back to the point, by the former analysis, law L_2 is gap-free. It sets fewer constraints on the agents' actions than the minimal-useful law L_1 and allows for potential coordination. However, law L_2 is not minimal yet in terms of gap-freeness. Let us consider the law $L_3 = \{d_a^1\}$, a reduction of law L_2. Since law L_3 bans factory a from dumping on the first day, as long as factory b dumps on that day (*i.e.* safe action), the fish would survive regardless of factory c's choice. The same applies to factory c. In other words, both factories b and c are principal agents under law L_3. In addition, under law L_3, if the fish is killed because all factories dump on the first day, then factory a is legally responsible; if it happens on the second or the third day, then factories b and c are both counterfactually responsible. This implies the gap-freeness of law L_3. On top of it, we say that law L_3 is a ***gap-free reduction*** of law L_2 because $L_3 \subsetneq L_2$. Moreover, observe that the law $L_4 = \varnothing$ is not gap-free because the death of the fish may happen, but no agent has a safe action that can individually prevent it under law L_4. This shows that any further reduction of law L_3 is not gap-free, and thus law L_3 satisfies a minimality property in terms of gap-freeness. In general, we say that a law is ***minimal-gap-free*** if *it is gap-free but cannot be further reduced while keeping gap-freeness*.

Contribution In this paper, we investigate the *design of useful laws and gap-free laws in multiagent systems*. In particular, we model multiagent systems as one-shot concurrent games and interpret the law design problems in a graph-theoretical perspective. Specifically, we first formalise the concepts of usefulness and gap-freeness in Section 2. Then, in Section 3, we establish the equivalence between the useful law design problem and the vertex cover problem in hypergraphs (*a.k.a.* hitting set problem) by providing polynomial-time reductions between them. After that, in Section 4, we show that the gap-free law design problem is at least as hard as the useful law design problem, and further present a method for solving the gap-free law design problem by reducing it to the vertex cover problem in hypergraphs. Note that the vertex cover problem is one of the most important problems in complexity theory, listed as one of the 21 NP-complete problems by Karp (1972). Extensive research has focused on its hardness and approximation (Chvatal 1979; Bar-Yehuda and Even 1985; Slavík 1996; Feige 1998).

Novelty By reducing the law design problems to the vertex cover problem, we make it possible to *tackle the computational intractability in the law design problems using approximation techniques*. This is distinct from the literature

on law design (*i.e.* norm synthesis) in multiagent systems. In fact, there has been a rich amount of studies about *normative system*, where laws/norms are used to regulate the multiagent systems. They mainly use first-order logic (Shoham and Tennenholtz 1995; Fitoussi and Tennenholtz 2000) or modal logic (van der Hoek, Roberts, and Wooldridge 2007; Ågotnes et al. 2009) to describe the concerned properties of a system and use deontic logic to capture laws/norms (Alechina, Dastani, and Logan 2018). In this way, they are capable of modelling more complex systems than one-shot concurrent games. Meanwhile, they study both offline design (*i.e.* design-time norm synthesis) (Fitoussi and Tennenholtz 2000; Ågotnes, van der Hoek, and Wooldridge 2012) and online design (*i.e.* run-time norm synthesis) (Morales, López-Sánchez, and Esteva 2011; Morales et al. 2013, 2014; Riad, Ghanadbashi, and Golpayegani 2022), considering both static and dynamic norms that may evolve (Alechina et al. 2022; Riad, Ghanadbashi, and Golpayegani 2022). Depending on the difference in object and formalisation, the computational complexity of their problems ranged from NP-complete (Shoham and Tennenholtz 1995) to beyond EXPTIME (Perelli 2019; Galimullin and Kuijer 2024). Although few of them try to address the complexity by heuristics (Christelis and Rovatsos 2009) or by degenerating their problems into optimisation problems (Ågotnes and Wooldridge 2010; Wu et al. 2022), *none have tackled the intractability by considering inapproximability in addition to approximation, as done in this paper*. As a by-product, our attempt at gap-free law design offers an applicable way to *address the responsibility-gap concern* in the literature.

2 Formalisation

In this section, we first define a (one-shot current) game to model multiagent systems. Then, we formalise the concepts of law, usefulness, and gap-freeness as discussed in the previous section. After that, we formally define hypergraphs that are used later to solve the law design problems.

In the rest of this paper, by $|S|$ we denote the size of a set S; by $\prod F$ and $\bigcup F$ we denote, respectively, the Cartesian product and the union of all sets in a family F. For an indexed set $\alpha = \{\alpha_i\}_{i\in I}$, by $\mathcal{S}(\alpha) = \{\alpha_i \mid i \in I\}$ we denote the support set of α that forgets the order and multiplicity.

Definition 1. *A **game** is a tuple $(\mathcal{A}, \Delta, \mathbb{P})$ such that*

1. *$\mathcal{A}$ is a nonempty finite set of agents;*
2. *$\Delta = \{\Delta_a\}_{a\in\mathcal{A}}$ is a family of sets of **actions**, where Δ_a is a finite set of actions available to agent a;*
3. *$\mathbb{P} \subseteq \prod \Delta$ is the **prohibition**.*

In a game $(\mathcal{A}, \Delta, \mathbb{P})$, each profile $\delta \in \prod \Delta$ represents an *outcome*. Instead of utilities of agents[1], we consider a set $\mathbb{P}$ of prohibited outcomes. The purpose of designing laws is to avoid the game ending in the prohibited outcomes. For instance, in the factory example, all factories dumping on the first day represents an outcome where the fish is killed, and the set of prohibited outcomes consists of the three cases where all agents dump on the same day during each three-day cycle. A law is established to avoid the death of the fish, that is, to prevent any of the prohibited outcomes.

Technically, item 2 of Definition 1 allows the action set Δ_a to be empty for any agent $a \in \mathcal{A}$. This is made for mathematical convenience. Moreover, a profile $\delta \in \prod \Delta$ is essentially a set indexed by every agent $a \in \mathcal{A}$ (*i.e.* $\delta = \{\delta_a\}_{a\in\mathcal{A}}$). Correspondingly,

$$\mathcal{S}(\delta) = \{\delta_a \mid a \in \mathcal{A}\} \tag{1}$$

is the set of actions taken by the agents under a profile δ. Also, the action sets of different agents are not necessarily identical or disjoint. This allows greater flexibility in modelling multiagent systems that simultaneously include multiple agents of the same type who are likely to share the same action space, and agents of different types whose available actions usually differ. More importantly, the laws that set bans on actions, as discussed in Section 1 and formally defined below, will be *agent-independent*. In other words, it is impossible to forbid one agent from using an action while permitting another agent to use the same action. In this sense, *fairness* is embedded in our formalisation.

Definition 2. *A law in a game $(\mathcal{A}, \Delta, \mathbb{P})$ is an arbitrary set $L \subseteq \bigcup \Delta$ of actions.*

For two laws L, L' in the same game, we say that L is a ***reduction*** of L' if $L \subseteq L'$. Note that, as stated in Section 1, agents can break a law. However, in a hypothetical situation where all agents obey the law, the agents indeed play a "subgame" where only lawful actions are available. We call such a "subgame" *law-imposed game* and formalise it below.

Definition 3. *For a law L in a game $(\mathcal{A}, \Delta, \mathbb{P})$, the **law-imposed** game is the game $(\mathcal{A}, \Delta^L, \mathbb{P}^L)$ where*

1. *$\Delta^L = \{\Delta^L_a\}_{a\in\mathcal{A}}$ such that $\Delta^L_a = \Delta_a \setminus L$ is the set of **lawful action**s of agent a;*
2. *$\mathbb{P}^L = \mathbb{P} \cap \prod \Delta^L$ is the **law-imposed prohibition**.*

Informally, for a given game and a given law, in the law-imposed game, the actions are the *lawful actions*, the profiles are the *lawful profiles*, and the prohibition consists of the lawful profiles prohibited in the original game.

2.1 Usefulness of Law

Let us now formalise usefulness. Recall that, as discussed in Section 1, a law is useful if the prohibited outcomes never appear when every agent obeys the law. That means none of the lawful profiles are prohibited, which is formalised below.

Definition 4. *A law L in a game $(\mathcal{A}, \Delta, \mathbb{P})$ is called **useful** if $\mathbb{P}^L = \varnothing$ in the law-imposed game.*

The next lemma characterises when a law is useful: every prohibited profile consists of at least one banned action by the law. See its formal proof in Appendix A.1 of the extended version (Shi and Naumov 2025).

Lemma 1. *A law L in a game $(\mathcal{A}, \Delta, \mathbb{P})$ is useful if and only if $L \cap \mathcal{S}(\delta) \neq \varnothing$ for each profile $\delta \in \mathbb{P}$.*

Next, we consider the minimality of a useful law. Our discussion about it in Section 1 can be formally captured below.

[1] We do not yet consider the incentives of agents in the law design problem, and thus get rid of the utility functions in our model.

Definition 5. *A law L in a game is **minimal-useful** if L is useful and no law $L' \subsetneq L$ is useful in the same game.*

Motivated by the pursuit of minimum constraints on society, if a law is useful but not minimal, then we would reduce it to leave as much freedom (*i.e.* lawful actions) as possible. This is formally captured as the minimality of useful reduction in the definition below.

Definition 6. *For a useful law L in a game,*

1. *a **useful reduction** of L is a useful law $L' \subseteq L$;*
2. *a useful reduction L' of L is called **minimum** if there is no useful reduction L'' of L such that $|L''| < |L'|$.*

Note that the approach of reducing an existing law, rather than crafting a new one from scratch, aligns with the view that laws should be stable (Jefferson 1787; Raz 1979), as it ensures that *actions previously permitted remain permitted*. This is indeed a type of *Pareto optimisation*. Also, it is not hard to deduce from Lemma 1 that any further reduction of a non-useful law is also non-useful, which is why we consider only the reduction of useful laws here. Moreover, observe that the law which bans all actions (*i.e.* $L = \bigcup \Delta$) is useful by Lemma 1, and thus crafting a new useful law from scratch is equivalent to getting a useful reduction of the law $L = \bigcup \Delta$. In this sense, *minimising an existing useful law encompasses the task of minimum useful law design*.

Notably, the law $L = \bigcup \Delta$ may not be reducible while preserving usefulness. For example, consider a matching-pennies game where two agents have the same action space $\{head, tail\}$ and the outcomes $(head, head)$ and $(tail, tail)$ are prohibited. In this case, the set $\{head, tail\}$ is the only useful law by Lemma 1 and thus minimal-useful. In other words, some games may admit no useful law that permits any lawful actions. However, such situations are uncommon in real-world scenarios, where every agent usually has access to a *default action* that can avoid undesirable outcomes. As a result, a law that permits only default actions will be useful, and any useful reduction of such a law continues to allow those default actions. In the worst-case scenario where no suitable default action exists, a useful initial law can still be constructed by distinguishing between different agents' actions (*i.e.* by making their action sets disjoint), albeit at the cost of symmetry. Then, a law that restricts each agent to a single action leading to a fixed non-prohibited outcome (*e.g.* banning one agent from choosing $head$ and the other from choosing $tail$ in the matching-pennies game) is always useful. This approach mirrors how traffic lights prevent collisions at intersections: when east-west traffic is permitted to proceed, north-south traffic is banned, and vice versa.

2.2 Gap-Freeness of Law

In this subsection, we formalise gap-freeness. As discussed in Section 1, a key feature of gap-freeness is the presence of a principal agent who has a safe action that can individually prevent the prohibited outcomes. We first formalise a "safe action" in our one-shot games without incorporating a law, namely an action that guarantees non-prohibited outcomes.

Definition 7. *For a game $(\mathcal{A}, \Delta, \mathbb{P})$ and an agent $a \in \mathcal{A}$, an action $d \in \bigcup \Delta$ is called a **safe action of agent** a if $d \in \Delta_a$ and $\delta_a \neq d$ for each profile $\delta \in \mathbb{P}$.*

Informally, action d is available to agent a, but she does not choose d in any prohibited outcome. In other words, every potential outcome when agent a chooses action d is not prohibited. As a result, agent a can prevent the prohibited outcomes by choosing action d. Also note that a safe action d of agent a is not necessarily safe for another agent b, even if $d \in \Delta_a \cap \Delta_b$. This arises from the asymmetry across agents in the prohibition as specified in item 3 of Definition 1.

However, as noted in Section 1, with a law in place, a principal agent needs only a *lawful* safe action that prevents the prohibited outcomes when others act lawfully. That is essentially a *safe action in the law-imposed game*. Next, we capture the legal responsibility of an agent breaking a law and the counterfactual responsibility of a principal agent failing to prevent. In particular, we say that an agent is responsible if a prohibited outcome happens and she is responsible either legally or counterfactually, as defined below.

Definition 8. *In a prohibited profile $\delta \in \mathbb{P}$ of a game $(\mathcal{A}, \Delta, \mathbb{P})$, an agent $a \in \mathcal{A}$ is **responsible** under law L if*

1. *(legally) $\delta_a \in L$, **or***
2. *(counterfactually) $\mathcal{S}(\delta) \cap L = \varnothing$ and a safe action of agent a exists in the law-imposed game $(\mathcal{A}, \Delta^L, \mathbb{P}^L)$.*

Informally, in a prohibited outcome δ, agent a is legally responsible if her action breaks the law (*i.e.* $\delta_a \in L$); if no agent breaks the law (*i.e.* $\mathcal{S}(\delta) \cap L = \varnothing$) but agent a is a principal agent under the law (*i.e.* has a safe action in the law-imposed game), then a is counterfactually responsible.

The next lemma characterises when an action is a safe action of an agent under a law: the law should make the action lawful and make each prohibited outcome where the agent takes the action unlawful. See Appendix A.2 for its proof.

Lemma 2. *An action $d \in \bigcup \Delta$ is a safe action of an agent $a \in \mathcal{A}$ in the law-imposed game $(\mathcal{A}, \Delta^L, \mathbb{P}^L)$ if and only if $d \in \Delta_a^L$ and $L \cap \mathcal{S}(\delta) \neq \varnothing$ for each $\delta \in \mathbb{P}$ such that $\delta_a = d$.*

The next definition formalises the gap-freeness property.

Definition 9. *A law L in a game $(\mathcal{A}, \Delta, \mathbb{P})$ is **gap-free** if there is at least one responsible agent in each profile $\delta \in \mathbb{P}$.*

Observe that, by Lemma 1 and statement (1), if a law is useful, then there is at least one legally responsible agent in each prohibited profile. Thus, *a useful law is also gap-free*. Conversely, if a law is not useful, then there is at least one prohibited profile where no agent is legally responsible. To make such a law gap-free, there should be a principal agent who bears counterfactual responsibility in those lawful but prohibited profiles. This intuition is formally captured by the next lemma. See Appendix A.3 for its formal proof.

Lemma 3. *A law L in a game $(\mathcal{A}, \Delta, \mathbb{P})$ is gap-free if and only if L is useful or there is an agent $a \in \mathcal{A}$ and a safe action of agent a in the law-imposed game $(\mathcal{A}, \Delta^L, \mathbb{P}^L)$.*

Now, we formalise the minimality of gap-freeness and gap-free reduction in a manner analogous to Section 2.1.

Definition 10. *A law L in a game is **minimal-gap-free** if L is gap-free and no law $L' \subsetneq L$ is gap-free in the same game.*

Definition 11. *For a gap-free law L in a game,*

1. *a **gap-free reduction** of L is a gap-free law $L' \subseteq L$;*

2. *a gap-free reduction L' of L is called* ***minimum*** *if there is no gap-free reduction L'' of L such that $|L''| < |L'|$.*

Recall that the law $\bigcup \Delta$ is useful and thus gap-free by Lemma 3. Hence, designing a gap-free law is equivalent to finding a gap-free reduction of the law $\bigcup \Delta$. Accordingly, *minimising an existing gap-free law encompasses the task of minimum gap-free law design*.

2.3 Hypergraphs with Fixed Rank

Now, let us introduce the hypergraphs that are used later to resolve the law design problems. Unlike standard graphs where each edge connects exactly two vertices, a hypergraph allows each edge (*a.k.a.* hyperedge) to connect any positive number of vertices. In particular, we consider hypergraphs where each edge contains *at most* k vertices for any fixed parameter $k \geq 1$, as formalised in the next definition.

Definition 12. *For any integer $k \geq 1$, a* ***rank-k hypergraph*** *(abbr.* ***k-graph****) is a tuple (V, E) such that V is a finite set of vertices and E is a set of (hyper)edges where $e \subseteq V$ and $1 \leq |e| \leq k$ for each edge $e \in E$.*

Note that an edge is a set of vertices. A vertex is said to *cover* an edge if the edge includes the vertex. A ***vertex cover*** of a k-graph is a set of vertices that *intersects* with every edge in the k-graph, as formalised below.

Definition 13. *For a k-graph (V, E),*

1. *a set C is called a* ***vertex cover*** *if $C \subseteq V$ and $C \cap e \neq \varnothing$ for each edge $e \in E$;*
2. *a vertex cover C is called* ***minimal*** *if there is no vertex cover $C' \subsetneq C$;*
3. *a vertex cover C is called* ***minimum*** *if there is no vertex cover C' such that $|C'| < |C|$.*

We consider the following problems about vertex cover, which are collectively referred to as $\mathbf{VC}$ problems:

- $\mathbf{IsVC}$: to verify if a set is a vertex cover of a k-graph.
- $\mathbf{IsMiniVC}$: to verify if a set is a minimal vertex cover of a k-graph.
- $\mathbf{MinVC}$: to find a minimum vertex cover of a k-graph.

These problems are extensively explored in the literature. In particular, $\mathbf{IsVC}$ and $\mathbf{IsMiniVC}$ can both be solved efficiently (*i.e.* in polynomial time). In contrast, $\mathbf{MinVC}$ is NP-hard (Garey and Johnson 1979), which means no efficient algorithm is believed to exist for this problem. As a compromise, *approximation algorithms* were developed to efficiently find *good enough, though not necessarily optimal*, solutions for those hard problems. An algorithm approximates $\mathbf{MinVC}$ within *factor* t if, for any k-graph, the size of the vertex cover it finds is at most t times the size of the minimum vertex cover. It is shown that $\mathbf{MinVC}$ can be *effectively approximated within factor* k, using greedy algorithms (Bar-Yehuda and Even 1981; Hall and Hochbaum 1986) or linear programming relaxation (Hochbaum 1982). However, it is hard to do better than this (Holmerin 2002; Dinur et al. 2005). In what follows, we will rely on the following algorithms and theorem.

Algorithms. *While solving the law design problems, the following efficient* $\mathbf{VC}$ *algorithms serve as* ***gadgets****:*

- `IsVC`*: an algorithm for* $\mathbf{IsVC}$.
- `IsMiniVC`*: an algorithm for* $\mathbf{IsMiniVC}$.
- `AppMinVC`*: a k-approximation algorithm for* $\mathbf{MinVC}$.

Theorem 1 (Khot and Regev, 2008)**.** $\mathbf{MinVC}$ *is NP-hard to approximate within factor $k-\varepsilon$ for any $\varepsilon > 0$ when $k \geq 2$.*[2]

3 Useful Law Design

To continue the discussion in Section 2.1, we consider the next three problems about useful law design, which are collectively referred to as $\mathbf{UL}$ problems:

- $\mathbf{IsUL}$: to verify if a set is a useful law in a game.
- $\mathbf{IsMiniUL}$: to verify if a set is a minimal-useful law in a game.
- $\mathbf{MinUR}$: to find a minimum useful reduction of a useful law in a game.

In particular, we establish an equivalence between the $\mathbf{VC}$ problems and the $\mathbf{UL}$ problems by providing two-way polynomial-time reductions. On top of this, we illustrate a way to solve the $\mathbf{UL}$ problems using the $\mathbf{VC}$ algorithms.

3.1 Reducing Vertex Cover to Useful Law

In this subsection, we show that any instance of a $\mathbf{VC}$ problem can be reduced to an instance of a $\mathbf{UL}$ problem in polynomial time. This demonstrates that the $\mathbf{UL}$ problems are *at least as hard as* the $\mathbf{VC}$ problems. The polynomial-time reduction is formally captured below.

Definition 14. *For any k-graph (V, E), let $\mathcal{G}_{(k,V,E)}$ be the game $(\mathcal{A}_{(k,V,E)}, \Delta_{(k,V,E)}, \mathbb{P}_{(k,V,E)})$ such that*

1. *$\mathcal{A}_{(k,V,E)} = [k]$ where $[k] = \{i \in \mathbb{N} \mid 1 \leq i \leq k\}$;*
2. *$\Delta_{(k,V,E)} = \{\Delta_i\}_{i \in [k]}$ where $\Delta_i = V$ for each $i \in [k]$;*
3. *$\mathbb{P}_{(k,V,E)} = \{\delta^e = \{\delta^e_i\}_{i \in [k]} \mid e \in E\}$ where δ^e_i is the $((i \bmod |e|)+1)^{th}$ item in any predefined order of set e.*

Note that, in the above definition, $k \geq 1$ and set V is finite by Definition 12. Thus, $\mathcal{G}_{(k,V,E)}$ is well-defined by Definition 1. Informally, $\mathcal{G}_{(k,V,E)}$ is a game of k agents with the same action space V. Each vertex in set V is an action. Each edge $e \in E$ corresponds to a prohibited profile δ^e where every vertex in edge e is taken as an action δ^e_i of some agent i. Then, by Lemma 1 and item 1 of Definition 13, it is easy to verify the next theorem. See Appendix B.1 for its proof.

Theorem 2. *A set C is a vertex cover of a k-graph (V, E) if and only if C is a useful law in the game $\mathcal{G}_{(k,V,E)}$.*

Note that, due to the consistency of minimality in item 2 of Definition 13 and Definition 5, Theorem 2 indeed implies reductions from $\mathbf{IsVC}$ and $\mathbf{IsMiniVC}$ to $\mathbf{IsUL}$ and $\mathbf{IsMiniUL}$, respectively. On the other hand, the set $V = \bigcup \Delta_{(k,V,E)}$ is a useful law in the game $\mathcal{G}_{(k,V,E)}$ by Definition 14 and Lemma 1. Then, every useful law in the game $\mathcal{G}_{(k,V,E)}$ is a useful reduction of law V by item 1 of Definition 6. Thus, the next corollary follows from Theorem 2.

[2] Khot and Regev (2008) consider a subset of k-graphs where each edge has exactly k vertices. The theorem holds under the Unique Game Conjecture (Khot 2002), which we also adopt.

Corollary 1. *A set C is a vertex cover of a k-graph (V, E) if and only if C is a useful reduction of law V in game $\mathcal{G}_{(k,V,E)}$.*

Note that, due to the consistency of minimality in item 3 of Definition 13 and item 2 of Definition 6, Corollary 1 implies a reduction from $\mathbf{MinVC}$ to $\mathbf{MinUR}$. This, together with Theorem 1, further implies an *inapproximability result* of the $\mathbf{MinUR}$ problem as stated in the next theorem. See Appendix B.2 for its proof.

Theorem 3. $\mathbf{MinUR}$ *in a game $(\mathcal{A}, \Delta, \mathbb{P})$ is NP-hard to approximate within factor $|\mathcal{A}|-\varepsilon$ for any $\varepsilon > 0$ when $|\mathcal{A}| \geq 2$.*

3.2 Reducing Useful Law to Vertex Cover

In this subsection, we show that any instance of a $\mathbf{UL}$ problem can be polynomially reduced to an instance of a $\mathbf{VC}$ problem. This demonstrates that the $\mathbf{UL}$ problems are *no harder than* the $\mathbf{VC}$ problems and suggests how $\mathbf{VC}$ algorithms can be used to solve the $\mathbf{UL}$ problems.

Technically, for any game $(\mathcal{A}, \Delta, \mathbb{P})$, consider the $|\mathcal{A}|$-graph $(\bigcup \Delta, \mathcal{S}(\mathbb{P}))$ where $\mathcal{S}(\mathbb{P}) = \{\mathcal{S}(\delta) \mid \delta \in \mathbb{P}\}$.[3] It is not hard to get the next theorem by Lemma 1 and item 1 of Definition 13. See Appendix B.4 for its formal proof.

Theorem 4. *A set L is a useful law in a game $(\mathcal{A}, \Delta, \mathbb{P})$ if and only if L is a vertex cover of the graph $(\bigcup \Delta, \mathcal{S}(\mathbb{P}))$.*

Opposite to Theorem 2, Theorem 4 implies reductions from $\mathbf{IsUL}$ and $\mathbf{IsMiniUL}$ to $\mathbf{IsVC}$ and $\mathbf{IsMiniVC}$, respectively. The corresponding algorithms `IsUL` and `IsMiniUL` call `IsVC` and `IsMiniVC`, respectively, as shown in Algorithm 1 of Appendix B.6.

Next, we consider the $\mathbf{MinUR}$ problem. Note that, by Theorem 4, reducing a useful law corresponds to finding a smaller vertex cover within an existing one. Meanwhile, the smaller vertex cover intersects every edge at the vertices in the original cover. Given this observation, the smaller vertex cover can be regarded as a vertex cover in a subgraph induced by the original cover, which is formalised as follows.

Definition 15. *For a vertex cover C of a k-graph (V, E), the induced subgraph is the k-graph (C, E^C) such that $E^C = \{C \cap e \mid e \in E\}$.*

The subgraph (C, E^C) is well-defined because $C \cap e \neq \varnothing$ and $|C \cap e| \leq |e| \leq k$ for each edge $e \in E$ as C is a vertex cover in the k-graph (V, E). Moreover, observe that a vertex cover of the subgraph (C, E^C) is also a vertex cover of the graph (V, E). Then, Theorem 4 implies the next theorem. See Appendix B.5 for its formal proof.

Theorem 5. *For a useful law L in a game $(\mathcal{A}, \Delta, \mathbb{P})$, law L' is a useful reduction of L if and only if L' is a vertex cover of the induced subgraph $(L, \mathcal{S}(\mathbb{P})^L)$.*

In contrast with Corollary 1, Theorem 5 implies a reduction from $\mathbf{MinUR}$ to $\mathbf{MinVC}$ and an algorithm `AppMinUR` that calls `AppMinVC` (see Algorithm 1 of Appendix B.6). Note that the graph $(L, \mathcal{S}(\mathbb{P})^L)$ in Theorem 5 is an $|\mathcal{A}|$-graph, on which `AppMinVC` is an $|\mathcal{A}|$-approximation. This makes `AppMinUR` an $|\mathcal{A}|$-approximation of $\mathbf{MinUR}$, where $|\mathcal{A}|$ is the number of agents in the input game (see Appendix B.7 for a discussion). It means `AppMinUR` achieves a nearly optimal approximation factor by Theorem 3.

[3] $(\bigcup \Delta, \mathcal{S}(\mathbb{P}))$ is an $|\mathcal{A}|$-graph by Lemma 7 in Appendix B.3.

4 Gap-Free Law Design

Following Section 2.2, in this section, we consider the next three problems about gap-free law design, which are collectively referred to as $\mathbf{GFL}$ problems:

- $\mathbf{IsGFL}$: to verify if a set is a gap-free law in a game.
- $\mathbf{IsMiniGFL}$: to verify if a set is a minimal-gap-free law in a game.
- $\mathbf{MinGFR}$: to find a minimum gap-free reduction of a gap-free law in a game.

We first show the hardness of the $\mathbf{GFL}$ problems by reductions from the $\mathbf{UL}$ problems. Then, we provide a way to solve the $\mathbf{GFL}$ problems again using the $\mathbf{VC}$ algorithms.

4.1 Reducing Useful Law to Gap-Free Law

In this subsection, we show that any instance of a $\mathbf{UL}$ problem can be polynomially reduced to an instance of a $\mathbf{GFL}$ problem. Specifically, for any game $(\mathcal{A}, \Delta, \mathbb{P})$, we construct a game $(\bar{\mathcal{A}}, \bar{\Delta}, \bar{\mathbb{P}})$ as illstrated in the next definition.

Definition 16. *For a game $(\mathcal{A}, \Delta, \mathbb{P})$, an agent $\gamma \notin \mathcal{A}$, and two distinct actions $p, n \notin \bigcup \Delta$, let the game $(\bar{\mathcal{A}}, \bar{\Delta}, \bar{\mathbb{P}})$ be:*

1. $\bar{\mathcal{A}} = \mathcal{A} \cup \{\gamma\}$;
2. $\bar{\Delta} = \{\bar{\Delta}_a\}_{a \in \bar{\mathcal{A}}}$ *where* $\bar{\Delta}_a = \begin{cases} \Delta_a \cup \{n\}, & \text{if } a \in \mathcal{A}; \\ \{p, n\}, & \text{if } a = \gamma; \end{cases}$
3. $\bar{\mathbb{P}} = \bar{\mathbb{P}}_1 \cup \bar{\mathbb{P}}_2 \cup \bar{\mathbb{P}}_3$ *where*
   - $\bar{\mathbb{P}}_1 = \{\bar{\delta} \mid \bar{\delta}_\gamma = p, (\exists \delta \in \mathbb{P}, \forall a \in \mathcal{A}, \bar{\delta}_a = \delta_a)\}$;
   - $\bar{\mathbb{P}}_2 = \{\bar{\delta} \mid \exists a \in \mathcal{A}\,(\bar{\delta}_a \in \Delta_a, \forall b \in \bar{\mathcal{A}} \setminus \{a\}, \bar{\delta}_b = n)\}$;
   - $\bar{\mathbb{P}}_3 = \{\bar{\delta} \mid \forall a \in \bar{\mathcal{A}}, \bar{\delta}_a = n\}$.

Observe that the new agent γ takes action n in each profile in sets $\bar{\mathbb{P}}_2, \bar{\mathbb{P}}_3$. Meanwhile, by Lemma 1 and the definition of $\bar{\mathbb{P}}_1$, a useful law L in the game $(\mathcal{A}, \Delta, \mathbb{P})$ intersects with each profile in set $\bar{\mathbb{P}}_1$. Then, law L makes p a safe action of agent γ in the game $(\bar{\mathcal{A}}, \bar{\Delta}^L, \bar{\mathbb{P}}^L)$ by Lemma 2. Thus, law L is a gap-free law in the game $(\bar{\mathcal{A}}, \bar{\Delta}, \bar{\mathbb{P}})$ by Lemma 3. Theorem 6 below formalises the above observation and establishes its converse as well. See Appendix C.1 for its formal proof.

Theorem 6. *A set $L \subseteq \bigcup \Delta$ is a useful law in a game $(\mathcal{A}, \Delta, \mathbb{P})$ if and only if L is a gap-free law in the game $(\bar{\mathcal{A}}, \bar{\Delta}, \bar{\mathbb{P}})$.*

Note that, due to the consistency of minimality in Definition 5 and Definition 10, Theorem 6 indeed implies reductions from $\mathbf{IsUL}$ and $\mathbf{IsMiniUL}$ to $\mathbf{IsGFL}$ and $\mathbf{IsMiniGFL}$, respectively. Moreover, when considering a reduction L' of a useful law L in the game $(\mathcal{A}, \Delta, \mathbb{P})$, it is guaranteed that $L' \subseteq L \subseteq \bigcup \Delta$. Then, by item 1 of Definition 6 and item 1 of Definition 11, Theorem 6 further implies the next corollary.

Corollary 2. *For a useful law L in a game $(\mathcal{A}, \Delta, \mathbb{P})$, a set L' is a useful reduction of L in the game $(\mathcal{A}, \Delta, \mathbb{P})$ if and only if L' is a gap-free reduction of L in the game $(\bar{\mathcal{A}}, \bar{\Delta}, \bar{\mathbb{P}})$.*

Given the consistency of minimality in item 2 of Definition 6 and item 2 of Definition 11, Corollary 2 implies a reduction from $\mathbf{MinUR}$ to $\mathbf{MinGFR}$. This, together with Theorem 3 and the extra agent γ in Definition 16, further implies an *inapproximability result* of $\mathbf{MinGFR}$ as stated in the next theorem. See Appendix C.2 for its proof.

Theorem 7. $\mathbf{MinGFR}$ *in a game* $(\mathcal{A}, \Delta, \mathbb{P})$ *is NP-hard to approximate within factor* $|\mathcal{A}| - 1 - \varepsilon$ *for any* $\varepsilon > 0$ *when* $|\mathcal{A}| \geq 3$.

4.2 Reducing Gap-Free Law to Vertex Cover

In this subsection, we reduce the $\mathbf{GFL}$ problems to the $\mathbf{VC}$ problems. By this means, we show a way to address the $\mathbf{GFL}$ problems using the $\mathbf{VC}$ algorithms.

Recall Lemma 3 that the gap-freeness of a law corresponds to the usefulness of the law and the existence of a safe action in the law-imposed game. Section 3.2 demonstrated how the $\mathbf{UL}$ problems can be addressed using the $\mathbf{VC}$ algorithms. Now, we discuss how a "safe action in a law-imposed game" is captured in the $\mathbf{VC}$ context. Note that not every action can become a safe action in a law-imposed game. If it can, we say the action is ***safable***.

Definition 17. *For a game* $(\mathcal{A}, \Delta, \mathbb{P})$*, an action* $d \in \bigcup \Delta$ *is* ***safable*** *if there is a law* L *and an agent* a *such that* d *is a safe action of agent* a *in the law-imposed game* $(\mathcal{A}, \Delta^L, \mathbb{P}^L)$.

The next lemma characterises when an action is safable: every agent taking it does not lead to a prohibited outcome. See Appendix C.3 for its formal proof.

Lemma 4. *For a game* $(\mathcal{A}, \Delta, \mathbb{P})$*, an action* $d \in \bigcup \Delta$ *is safable if and only if* $\mathcal{S}(\delta) \neq \{d\}$ *for each profile* $\delta \in \mathbb{P}$.

Next, let us recall Lemma 2, which characterises when an action d is a safe action of agent a in a law-imposed game $(\mathcal{A}, \Delta^L, \mathbb{P}^L)$. Note that the second half of Lemma 2 implies that $L \subseteq \bigcup \Delta \setminus \{d\}$ and $L \cap (\mathcal{S}(\delta) \setminus \{d\}) \neq \varnothing$ for each profile $\delta \in \mathbb{P}$ such that $\delta_a = d$. This, by item 1 of Definition 13, implies that L is a vertex cover in the graph defined below.

Definition 18. *For a game* $(\mathcal{A}, \Delta, \mathbb{P})$*, an agent* $a \in \mathcal{A}$*, and a safable action* $d \in \Delta_a$*, let the* $|\mathcal{A}|$*-graph* $\mathcal{H}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})}$ *be the pair* $(\mathcal{V}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})}, \mathcal{E}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})})$ *where*

1. $\mathcal{V}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})} = (\bigcup \Delta) \setminus \{d\}$;
2. $\mathcal{E}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})} = \{\mathcal{S}(\delta) \setminus \{d\} \mid \delta \in \mathbb{P}, \delta_a = d\}$.

Note that the $|\mathcal{A}|$-graph $\mathcal{H}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})}$ is well-defined because $1 \leq |\mathcal{S}(\delta) \setminus \{d\}| < |\mathcal{S}(\delta)| \leq |\mathcal{A}|$ by Lemma 4 and statement (1). Following the above observation, the next lemma bridges "a safe action in a law-imposed game" with the $\mathbf{VC}$ problems. See Appendix C.4 for its formal proof.

Lemma 5. *For a law* L *in a game* $(\mathcal{A}, \Delta, \mathbb{P})$ *and an agent* $a \in \mathcal{A}$*, an action* $d \in \bigcup \Delta$ *is a safe action of agent* a *in the law-imposed game* $(\mathcal{A}, \Delta^L, \mathbb{P}^L)$ *if and only if* $d \in \Delta^L_a$*,* d *is safable in the game* $(\mathcal{A}, \Delta, \mathbb{P})$*, and* L *is a vertex cover in the graph* $\mathcal{H}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})}$.

Observe that Lemma 5, Theorem 4, and Lemma 3 imply Theorem 8 below, which further implies a Cook reduction from $\mathbf{IsGFL}$ to $\mathbf{IsVC}$ and a corresponding algorithm `IsGFL` that *iteratively* calls `IsVC` for polynomial times, as illustrated in Algorithm 2 of Appendix C.8.

Theorem 8. *A law* L *in a game* $(\mathcal{A}, \Delta, \mathbb{P})$ *is gap-free if and only if at least one of the following statements is true:*

1. L *is a vertex cover in the graph* $(\bigcup \Delta, \mathcal{S}(\mathbb{P}))$;
2. *there is an agent* $a \in \mathcal{A}$ *and a safable action* $d \in \Delta^L_a$ *such that* L *is a vertex cover in the graph* $\mathcal{H}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})}$.

Note that, by Definition 10 and item 1 of Definition 11, a gap-free law is not minimal if there is a "strict" gap-free reduction. By Lemma 3, such a reduction retains the gap-freeness in three possible approaches: maintaining usefulness, maintaining a safe action under the original law, or introducing a new safe action. Moreover, we can use an induced subgraph in Definition 15 to capture a reduction of an existing law. Following the above hints, we can get the next two theorems. See Appendices C.5 and C.6 for their proofs.

Theorem 9. *A gap-free law* L *in a game* $(\mathcal{A}, \Delta, \mathbb{P})$ *is minimal if and only if all of the following statements are true:*

1. *if* L *is a vertex cover of the graph* $(\bigcup \Delta, \mathcal{S}(\mathbb{P}))$*, then* L *is a minimal vertex cover in this graph;*
2. *for each agent* $a \in \mathcal{A}$ *and each safable action* $d \in \Delta^L_a$*, if* L *is a vertex cover in the graph* $\mathcal{H}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})}$*, then* L *is a minimal vertex cover in this graph;*
3. *for each agent* $a \in \mathcal{A}$ *and each safable action* $d \in \Delta_a \cap L$*, the set* $L \setminus \{d\}$ *is not a vertex cover in graph* $\mathcal{H}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})}$.

Theorem 10. *For a gap-free law* L *in a game* $(\mathcal{A}, \Delta, \mathbb{P})$*, a law* L' *is a gap-free reduction of* L *if and only if at least one of the following statements is true:*

1. L *is a vertex cover of the graph* $(\bigcup \Delta, \mathcal{S}(\mathbb{P}))$ *and* L' *is a vertex cover in the subgraph* $(L, \mathcal{S}(\mathbb{P})^L)$;
2. *there is an agent* $a \in \mathcal{A}$ *and a safable action* $d \in \Delta^L_a$ *such that* L *is a vertex cover in graph* $\mathcal{H}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})}$ *and* L' *is a vertex cover in the subgraph* $(L, (\mathcal{E}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})})^L)$;
3. *there is an* $a \in \mathcal{A}$ *and a safable action* $d \in \Delta_a \cap L$ *such that* $L \setminus \{d\}$ *is a vertex cover in graph* $\mathcal{H}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})}$ *and* L' *is a vertex cover in the subgraph* $(L \setminus \{d\}, (\mathcal{E}^{a,d}_{(\mathcal{A},\Delta,\mathbb{P})})^{L \setminus \{d\}})$.

The above two theorems imply Cook reductions from $\mathbf{IsMiniGFL}$ and $\mathbf{MinGFR}$ to the $\mathbf{VC}$ problems. The corresponding algorithms `IsMiniGFL` and `AppMinGFR` call the $\mathbf{VC}$ algorithms polynomial times (see Algorithm 2 of Appendix C.8). Note that all graphs in Theorem 10 are $|\mathcal{A}|$-graphs, on which `AppMinVC` is an $|\mathcal{A}|$-approximation. This makes `AppMinGFR` an $|\mathcal{A}|$-approximation of $\mathbf{MinGFR}$, where $|\mathcal{A}|$ is the number of agents in the input game (see Appendix C.7 for a discussion).

5 Conclusion

For the usefulness and gap-freeness of law, we studied the corresponding law design problems by relating them to vertex cover problems in hypergraphs. We proved that the task of minimising a law while keeping its usefulness or gap-freeness is NP-hard even to approximate. We also proposed reductions from law design problems to vertex cover problems, which imply law-design algorithms that make polynomial-time calls to the vertex cover algorithms.

As an ending discussion, note that we don't consider the weight of actions in law minimisation. However, this can be done with no extra effort because the vertex cover algorithms in the literature apply to weighted vertices.

Acknowledgements

The research is funded by the China Scholarship Council (CSC No.202206070014).

References

Ågotnes, T.; Van Der Hoek, W.; Rodríguez-Aguilar, J. A.; Sierra, C.; and Wooldridge, M. 2009. A temporal logic of normative systems. In *Towards Mathematical Philosophy: Papers from the Studia Logica conference Trends in Logic IV*, 69–106. Springer.

Ågotnes, T.; van der Hoek, W.; and Wooldridge, M. 2012. Conservative social laws. In *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI-12)*, 49–54. IOS Press.

Ågotnes, T.; and Wooldridge, M. 2010. Optimal social laws. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-10)*, 667–674.

Alechina, N.; Dastani, M.; and Logan, B. 2018. Norm specification and verification in multi-agent systems. *Journal of Applied Logics*, 5(2): 457.

Alechina, N.; Giacomo, G. D.; Logan, B.; and Perelli, G. 2022. Automatic synthesis of dynamic norms for multi-agent systems. In *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning (KR-22)*, 12–21.

Baier, C.; Funke, F.; and Majumdar, R. 2021. A game-theoretic account of responsibility allocation. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI-21)*, 1773–1779.

Bar-Yehuda, R.; and Even, S. 1981. A linear-time approximation algorithm for the weighted vertex cover problem. *Journal of Algorithms*, 2(2): 198–203.

Bar-Yehuda, R.; and Even, S. 1985. A local-ratio theorem for approximating the weighted vertex cover problem. In *North-Holland Mathematics Studies*, volume 109, 27–45. Elsevier.

Braham, M.; and van Hees, M. 2011. Responsibility voids. *The Philosophical Quarterly*, 61(242): 6–15.

Braham, M.; and van Hees, M. 2018. Voids or fragmentation: moral responsibility for collective outcomes. *The Economic Journal*, 128(612): F95–F113.

Burton, S.; Habli, I.; Lawton, T.; McDermid, J.; Morgan, P.; and Porter, Z. 2020. Mind the gaps: assuring the safety of autonomous systems from an engineering, ethical, and legal perspective. *Artificial Intelligence*, 279: 103201.

Christelis, G.; and Rovatsos, M. 2009. Automated norm synthesis in an agent-based planning environment. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-09)*, 161–168.

Chvatal, V. 1979. A greedy heuristic for the set-covering problem. *Mathematics of operations research*, 4(3): 233–235.

Dinur, I.; Guruswami, V.; Khot, S.; and Regev, O. 2005. A new multilayered PCP and the hardness of hypergraph vertex cover. *SIAM Journal on Computing*, 34(5): 1129–1146.

Duijf, H. 2018. Responsibility voids and cooperation. *Philosophy of the Social Sciences*, 48(4): 434–460.

Feige, U. 1998. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4): 634–652.

Fitoussi, D.; and Tennenholtz, M. 2000. Choosing social laws for multi-agent systems: Minimality and simplicity. *Artificial Intelligence*, 119(1-2): 61–101.

Frankfurt, H. G. 1969. Alternate possibilities and moral responsibility. *The Journal of Philosophy*, 66(23): 829–839.

Galimullin, R.; and Kuijer, L. B. 2024. Synthesizing social laws with ATL conditions: Extended Abstract. In *Proceedings of the 23rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-24)*, volume 2024, 2270–2272.

Garey, M. R.; and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco: W. H. Freeman.

Goetze, T. S. 2022. Mind the gap: autonomous systems, the responsibility gap, and moral entanglement. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, 390–400.

Gunkel, D. J. 2020. Mind the gap: responsible robotics and the problem of responsibility. *Ethics and Information Technology*, 22(4): 307–320.

Hall, N. G.; and Hochbaum, D. S. 1986. A fast approximation algorithm for the multicovering problem. *Discrete Applied Mathematics*, 15(1): 35–40.

Hayek, F. A. 1944. *The Road to Serfdom*. Chicago: University of Chicago Press.

Hiller, S.; Israel, J.; and Heitzig, J. 2022. An axiomatic approach to formalized responsibility ascription. In *Proceedings of the 24th International Conference on Principles and Practice of Multi-Agent Systems (PRIMA-22)*, 435–457. Springer.

Hochbaum, D. S. 1982. Approximation algorithms for the set covering and vertex cover problems. *SIAM Journal on Computing*, 11(3): 555–556.

Holmerin, J. 2002. Improved inapproximability results for vertex cover on k-uniform hypergraphs. In *International Colloquium on Automata, Languages, and Programming (ICALP-02)*, 1005–1016. Springer.

Jefferson, T. 1787. Letter to James Madison, 20 December 1787. Founders Online, National Archives, https://founders.archives.gov/documents/Jefferson/01-12-02-0454. Accessed: 2025-07-18.

Karp, R. M. 1972. Reducibility Among Combinatorial Problems. In Miller, R. E.; and Thatcher, J. W., eds., *Complexity of Computer Computations*, 85–103. Boston, MA: Springer.

Khot, S. 2002. On the power of unique 2-prover 1-round games. In *Proceedings of the 34th annual ACM symposium on Theory of Computing (STOC-02)*, 767–775.

Khot, S.; and Regev, O. 2008. Vertex cover might be hard to approximate to within $2-\varepsilon$. *Journal of Computer and System Sciences*, 74(3): 335–349.

Langer, M.; Oster, D.; Speith, T.; Hermanns, H.; Kästner, L.; Schmidt, E.; Sesing, A.; and Baum, K. 2021. What do we want from Explainable Artificial Intelligence (XAI)?– A stakeholder perspective on XAI and a conceptual model guiding interdisciplinary XAI research. *Artificial Intelligence*, 296: 103473.

Matthias, A. 2004. The responsibility gap: ascribing responsibility for the actions of learning automata. *Ethics and Information Technology*, 6: 175–183.

Morales, J.; López-Sánchez, M.; and Esteva, M. 2011. Using experience to generate new regulations. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI-11)*, 307–312.

Morales, J.; Lopez-Sanchez, M.; Rodriguez-Aguilar, J. A.; Wooldridge, M.; and Vasconcelos, W. 2014. Minimality and simplicity in the on-line automated synthesis of normative systems. In *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-14)*.

Morales, J.; Lopez-Sanchez, M.; Rodriguez-Aguilar, J. A.; Wooldridge, M. J.; and Vasconcelos, W. W. 2013. Automated synthesis of normative systems. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-14)*, volume 13, 483–490.

Naumov, P.; and Tao, J. 2019. Blameworthiness in strategic games. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI-19)*, 3011–3018.

Nozick, R. 1974. *Anarchy, State, and Utopia*. New York: Basic Books.

Perelli, G. 2019. Enforcing Equilibria in Multi-Agent Systems. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS-19)*, 13–17.

Raz, J. 1979. The rule of law and its virtue. In *The Authority of Law: Essays on Law and Morality*. Oxford University Press.

Riad, M.; Ghanadbashi, S.; and Golpayegani, F. 2022. Runtime norms synthesis in dynamic environments with changing objectives. In *Irish Conference on Artificial Intelligence and Cognitive Science*, 462–474. Springer.

Robb, D. 2023. Moral responsibility and the principle of alternative possibilities. In Zalta, E. N.; and Nodelman, U., eds., *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University.

Shi, Q. 2024. Responsibility in extensive form games. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, volume 38, 19920–19928.

Shi, Q.; and Naumov, P. 2025. A graph-theoretical perspective on law design for multiagent systems. *arXiv preprint arXiv:2511.06361*.

Shoham, Y.; and Tennenholtz, M. 1995. On social laws for artificial agent societies: off-line design. *Artificial intelligence*, 73(1-2): 231–252.

Slavík, P. 1996. A tight analysis of the greedy algorithm for set cover. In *Proceedings of the 28th Annual ACM symposium on Theory of Computing (STOC-96)*, 435–441.

van der Hoek, W.; Roberts, M.; and Wooldridge, M. 2007. Social laws in alternating time: Effectiveness, feasibility, and synthesis. *Synthese*, 156(1): 1–19.

Wu, J.; Cao, J.; Sun, H.; and Wang, C. 2022. A Bayesian optimal social law synthesizing mechanism for strategical agents. *Autonomous Agents and Multi-Agent Systems*, 36(2): 48.

Yazdanpanah, V.; Dastani, M.; Alechina, N.; Logan, B.; and Jamroga, W. 2019. Strategic responsibility under imperfect information. In *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-19)*, 592–600.