Efficient Norm Emergence through Experiential Dynamic Punishment

Samhar Mahmoud and **Nathan Griffiths** and **Jeroen Keppens** and **Michael Luck** [1]

Abstract. Peer punishment has been an effective means to ensure that norms are complied with in a population of self-interested agents. However, current approaches to establishing norms have only considered *static* punishments, which do not vary with the magnitude or frequency of norm violation. Such static punishments are difficult to apply because it is difficult to identify an appropriate fixed penalty: one that is not too weak to disincentivise norm violations and not too strong to lead to significant deleterious effects on the system as a whole (such as those incurred by losing the benefits of a member of the population). This paper addresses this concern by developing an adaptive punishment technique that tailors penalty to norm violation. An experimental evaluation of the approach demonstrates its value compared to static punishment. In particular, the results show that our dynamic punishment technique is capable of achieving norm emergence, even when starting with an amount of punishment that is too low to achieve emergence in the traditional static approach.

1 Introduction

Norms can be seen as a very effective means of governing the behaviour of different members of decentralised open systems, such as peer-to-peer (P2P) file sharing systems, in which the cooperation of the members is vital in maintaining the benefits of the whole system. However, the self-interest of some can lead to the temptation to profit from the activities of others without reciprocating. For example, in P2P file sharing, free riding peers download files from others without sharing files, thus gaining the benefit of the files without wasting their own bandwidth. Such agents can avoid punishment due to the absence of a central authority to penalise such behaviour.

In this context, the enforcement of *norms*, rather than just the norms themselves, as a potential means of ensuring cooperative behaviour has been proposed (e.g., [1, 15]); here, agents punish others for behaviour that violates a norm. As Axelrod has shown, such punishment must apply not just to such norm violations, but also to the failure of others to penalise norm violations when observed [1]. Several different punishment schemes to address these concerns have been considered [5, 6, 7], but all are *static* in that they consist of applying the same fixed penalty regardless of the circumstances. As such, they rely on the system developer to find an appropriate such fixed penalty, yet the consequences of employing an inappropriate penalty are considerable. If it is too weak, punishment may be ineffective as a means of discouraging behaviour that does not support norm establishment. If it is excessive, it risks undercutting the benefits of cooperation. For example, in peer-to-peer file sharing systems, ostracising non-compliant individuals for extended periods of time may encourage norm compliance in the long run, but at the expense of losing the contributions of those individuals.

In response, this paper is concerned with how *dynamic* punishment may help a norm to emerge in a population of agents that starts out without any norm recognition. More specifically, the paper addresses the problem of determining the punishment most appropriate to the violation context, by providing an adaptive punishment technique. In order to evaluate this dynamic punishment approach, it is integrated with a variant of Axelrod's metanorm model [1], the effectiveness of which in achieving norm emergence has been considered elsewhere [10], and which is not the concern of this paper. Axelrod's model has previously only been applied in the context of static punishment, providing a clear baseline against which to assess the contributions here. While the notion of dynamic sanctioning, by which the level of punishment is modified in response to circumstances, has been suggested by Villatoro et al. [12], they consider only global modifications for a population as a whole rather than dynamism in relation to a particular agent's circumstances, as we do here.

The key contribution of the paper is a novel technique for dynamically adapting punishments to suit the circumstances of individual agents so that an overall system is more efficient. This is extensively evaluated with a series of experiments. The paper is structured as follows. Section 2 reviews the underlying metanorm model used as the basis for our work, while Section 3 distinguishes between static and dynamic punishment, and introduces our experiential dynamic punishment approach. Evaluation and experimental result analysis are presented in Section 4, before concluding in Section 5.

2 Metanorms and Punishment

In seeking to develop a model capable of supporting norm emergence in real world distributed systems, we integrate the dynamic punishment approach just described, with Mahmoud et al.'s modification [9] of the metanorm model originally introduced by Axelrod [1] as a means of studying norm emergence. In what follows, we first introduce the original model and the learning technique that agents adopt in this model. After that, the integration of the new punishment approach in the metanorm model.

2.1 The Metanorm Model

Axelrod's metanorm model [1] aims to simulate distributed systems in which a community of self-interested agents is encouraged, without being forced to do so by a central authority, to adhere to a behavioural constraint, or *norm*, that benefits the community but not the individual agent adhering to the norm. The simulation of this

[1] Department of Informatics, King's College London, London WC2R 2LS, UK, email: samhar.mahmoud@kcl.ac.uk and Department of Computer Science, University of Warwick, Coventry CV4 7AL, UK.

model provides a means to test the conditions under which the norm governs the behaviour of individual agents, leading to convergence.

Inspired by Axelrod's model, our simulation focusses only on the essential features of the problem. In the simulation, the agents play a game iteratively; in each iteration, they make a number of binary decisions. First, each agent decides whether to comply with the norm or to defect. Defection brings a reward for the defecting agent called *temptation*, and a penalty to all other agents called *hurt*, but each defector risks being observed by the other agents and punished as a result. These other agents thus decide whether to punish agents that were observed defecting, with a low penalty for the punisher known as an *enforcement cost*, and a high penalty for the punished agent known as a *punishment cost*. Agents that do not punish those observed defecting risk being observed themselves, and potentially incur metapunishment. Thus, each agent decides whether to metapunish agents observed to spare defecting agents. Again, metapunishment brings a high penalty for the punished agent and a low penalty for the punisher, the punishment and enforcement costs respectively.

The behaviour of agents in each round of the game is random, but governed by two variables: *boldness*, and *vengefulness*, which are initialised at the start of any run with the use of a uniform distribution function. In each round, agents are given a fixed number of opportunities to defect, in which boldness determines the probability that an agent defects. Conversely, vengefulness is the probability that an agent punishes or metapunishes another agent. Thus, the boldness and vengefulness of an agent are said to comprise that agent's *policies*. After several rounds of the game, each agent's rewards and penalties are tallied, and successful and unsuccessful policies are identified. By comparing themselves to other agents on this basis, the policies of poorly performing agents are revised such that features of successful policies are more likely to be retained.

A final remark is that agents are situated in this simulation according to a specific topological structure, which limits the observability of agents' actions such that agents are only able to observe their neighbours. Moreover, this implies that agent are only able to punish and metapunish their neighbours.

2.2 Dynamic Policy Adaptation

To be able to learn from experience, agents change their policies at the end of each round of the simulation. This is accomplished through the use of a reinforcement learning technique embedded in each agent; it is a form of q-learning [14], in which agents keep track of different scores that determine the utility gained or lost from taking the different actions available to the agent.

Based on the comparison of these different scores, agents adapt their policies using a dynamic policy adaptation approach (see [10] for details) in order to enhance their utility. However, agents adapt their policies in line with their own experiences, so that agents that score badly change their policies considerably more than agents that score much better, in a manner similar to the WOLF-PHC [3] reinforcement learning technique. In this adaptive policy review mechanism, in which agents change their policy proportionally to the utility gained or lost, policy changes are thus greater when the utility lost from taking a certain action is high than when it is low.

3 Dynamic Punishment

3.1 Dynamic and Static Punishment

Consider again the example of peer-to-peer (P2P) file sharing mentioned in the introduction, in which agents are able to download files from each other. Here, there is a norm obliging agents participating in the system to upload the files they have downloaded in order to share them with others, and to maintain the availability of these files on the network. However, since uploading consumes bandwidth, and in the absence of an appropriate punishment, self interested agents could choose not to share (upload) files they have downloaded in order to preserve bandwidth for their own use. In the case of frequent occurrences of such selfish behaviour, the efficiency of the entire P2P network can be threatened. In response to this problem, de Pinninck et al. [4] suggest that the punishment for such behaviour should be *blocking*, by which all other agents cease interacting with an agent that is observed not to share files after downloading them.

If punishment is static, it can be pre-specified at design time. In the P2P example, such static punishment involves identifying agents that violate the norm, and blocking them for the specified duration, on each occurrence of a violation. While this fixed blocking period is determined by the system designer, the challenge is to determine the most appropriate blocking period. For example, if the blocking period is fixed at 30 minutes for each violation, this may be just the right penalty needed to regulate the behaviour of some agents, but others may be persuaded to avoid violation (and upload files as demanded by the norm) with a shorter blocking period of only 10 minutes. Such a situation suggests that the network is losing the potentially valuable participation of some agents for a period of 20 minutes. Conversely, there may be still more agents for whom 30 minutes is insufficient to convince them to cease violations, for example due to the gain from violation exceeding the loss incurred through being blocked for 30 minutes. It is in this sense that determining an appropriate blocking period can be crucial to the performance of the overall system, especially those that rely on the participation of members for their functionality and effectiveness (as with P2P networks). However, determining such an appropriate blocking period can be difficult, if not impossible, simply because there is no single fixed period that is effective when dealing with different types of agents.

In consequence, dynamic punishment can play an important role in overall system performance by adapting punishment values in line with available information about the violating agent, so that punishment is increased when the current value is found to be ineffective, and decreased when it is excessive. Returning to our example, if the agent does not take the opportunity to share files at the first opportunity after the blocking period has ended, then it can be blocked again, this time for a longer period, with the aim of bringing about compliant behaviour. This can continue until the agent begins to comply (at least occasionally), when the duration of blocking for for subsequent infrequent violations may now be reduced.

Clearly, as the example above suggests, the appropriate punishment at any moment should be determined by an agent's prior behaviour: an agent that has a history of violation should be punished more than one with a history of compliance, in order to bring about a change to ingrained behaviour. Now, since agents themselves must apply punishments, they must maintain a record of their prior interactions (involving requests for files and decisions about whether or not to share these files) with others and build up a repository of experience to determine the level of punishment to apply.

3.2 Recording Experience of Violation

Two things must be recorded by each agent involved in any such interaction in which one agent requests a file and the other decides whether to share it (cooperating, and thus complying with the norm) or not (and defecting, and thus violating the norm): the identity of the other agent involved in the interaction; and the type of action taken by this other agent, whether cooperating or defecting. Each agent can thus build up a repository of of records of interactions in this way over time, providing a store of information on which to base punishment decisions.

A limitless repository, however, can cause problems. Since our aim is to encourage norm-compliant behaviour in individual agents, and since the mechanism we propose seeks to amplify punishments in the case of repeat violations, clearly a key target is to modify behaviour of such repeat offenders. Yet if we consider only the average prior behaviour of such repeat offenders over a long period, any recent adjustments towards compliance may be vastly outweighed in the repository by the long history of violations, bringing further increased punishment rather than the reduction in punishment that is warranted. In order to address this, therefore, we need some means to determine punishments in light of more recent interactions between agents rather than much older interactions that do not reflect current reality: agents that start to act in support of societal norms should not be punished severely just because they had previously behaved badly. A *window* over the repository, with a particular *window size*, can thus be used to limit the interactions that are considered to a specific period of recent time, allowing agents to forget old violations and adapt their punishments to changes in the behaviour of others much more quickly and effectively. In this way, the prior defections of an agent that defected regularly in the past, but that has recently begun to cooperate, will be weighted much less in comparison to more recent compliance when determining the punishment value.

We define the *memory*, M_i, of an agent i, to be a set of cells, each containing the identifier $agID$ of the other agent, j, involved in an interaction and the action act taken by that agent in the interaction:

$$M_i = \{m_1, m_2, .., m_n\}$$

where n is the window size, and m_j is the jth cell:

$$m_j = \langle agID, act \rangle$$

3.3 Dynamic Punishment

Given this notion of memory, and within the available window of data, we can specify two useful measures from agent i's perspective (from its memory): the number of previous instances of defection of agent j (nd_j), and the number of previous instances of compliance of j (nc_j). In turn, this gives the *defection proportion*, dp_j, as follows:

$$dp_j = \frac{nd_j}{nd_j + nc_j}$$

This alone is not enough to determine the level of punishment, since the absolute number of defections is also relevant. An agent that violates a norm ten times merits a greater sanction that one that violates it just once. Moreover, an agent that violates a norm once in ten instances merits a lower sanction than one that violates it norms ten times from 100, since this indicates persistent and repeated offence. We reflect these concerns in what we call the local defection view of agent i on agent j, as follows.

$$LocalView : AGENT \times AGENT \rightarrow \mathbb{R}$$
$$\forall ag_i, ag_j \in AGENT : LocalView(ag_i, ag_j) = dp_j \times nd_j$$

where ag_i is the punishing agent; ag_j is the defecting agent; dp_j is the defection proportion of agent j in agent i's memory; and nd_j is the number of defections of agent j in agent i's memory.

Now, while we are interested in modifying punishments to suit the circumstances, we need an initial *punishment unit* (pu) as a basis for such modification. In this way, an applied punishment can be determined by multiplying the defection proportion with the absolute number of defections and the punishment unit. Punishment can thus be seen as a function that takes two agents and returns the punishment value applied by the first agent to the second.

$$ExpPunish : AGENT \times AGENT \rightarrow \mathbb{R}$$
$$\forall ag_i, ag_j \in AGENT :$$
$$ExpPunish(ag_i, ag_j) = LocalView(ag_i, ag_j) \times pu$$

As indicated above, non-compliance with both norms and metanorms (not punishing a norm defector) are considered to be defections. As a result, each agent must make two punishment decisions: whether to punish a norm defector and whether to metapunish an agent that does not itself punish a defector.

The metapunishment decision is slightly different to the punishment decision, and calculated by replacing the number of defections with the number of previously unpunished (or *spared*) defectors (nds), and the number of times of compliance with the number of previously punished defectors (ndp), giving a *sparing proportion*, sp_j:

$$sp_j = \frac{nds_j}{nds_j + ndp_j}$$

4 Evaluation

Based on the model just described, we carried out several experiments to understand the impact and potential of dynamic punishment Before introducing the results obtained through experiments with this model, however, it is important to explain the specific meaning of norm emergence here, and how it can be observed in the results. To be precise, norm emergence is achieved when the population of agents has low boldness and high vengefulness, with the former indicating that agents will not defect, and the latter indicating that agents will defend the norm by punishing its defectors. In this context, the higher the vengefulness, the more stable the norm is. Importantly, all of our experiments are undertaken over a scale-free network (generated with Barabasi's algorithm [2]) since this has been shown to be the most challenging [13, 11] in which to engineer emergence due to the complexities arising from the distribution of connections between nodes. In particular, in scale-free networks, there are *hubs* with very many connections to other nodes, and *outliers* with very few. Our experiments were run over 1,000,000 time steps, with 1,000 agents for 1,000 runs, and with a *temptation* value of 3, a *hurt* value of -1 and an *enforcement* cost of -2.

4.1 Original Model Baseline

The results obtained from the initial modification of Axelrod's original model, before integrating dynamic punishment, are shown in Figure 1, indicating that the model is successful to some degree in achieving norm emergence in an agent population over a scale-free network. While the approach is able to regulate agent behaviour with regard to norm defection (which can be seen from the near 0 average boldness of the population), the level of average vengefulness achieved is between 0.4 and 0.7, suggesting that norm emergence is good but not optimal; an agent can still defect without punishment.

The punishments that agents apply to each other are thus sufficient to learn that high boldness is harmful and should be reduced. However, while metapunishments are sufficient to prevent vengefulness

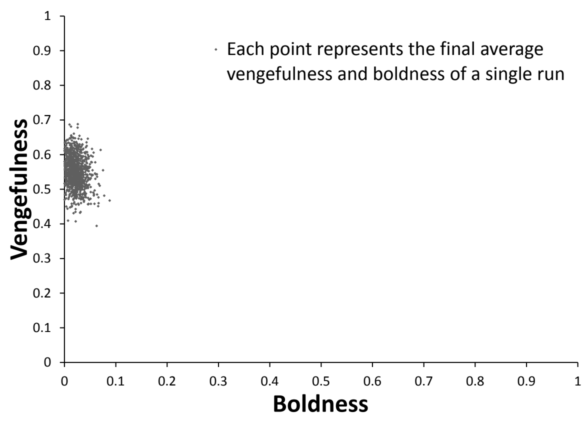

Figure 1. Baseline metanorms results

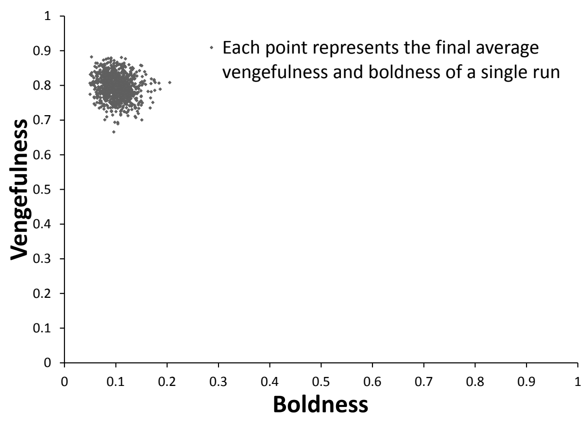

Figure 2. Experiential dynamic punishment results

from dropping below the midrange level, they do not cause agents to increase vengefulness adequately. We will not dwell further on these results, since a more detailed analysis of this situation is provided elsewhere [8], but the important point to note is that the static punishment unit used here is -9 which, as will be shown later, is unnecessary and excessive.

4.2 Dynamic Punishment Experiments

By introducing dynamic punishment as described in this paper, however, we are able to improve the results as shown in Figure 2, in which the level of vengefulness has increased to be between 0.7 and 0.9. This is much more stable than previously, with high probability of punishing defectors and metapunishing those agents that do not punish defectors. The improvement can be explained by the flexibility brought about by dynamic punishment, since if a previous metapunishment does not succeed in changing behaviour, the metapunishment cost can be increased. Consequently, agents are encouraged to increase their vengefulness and punish norm defectors in the future. A more detailed analysis of the effect of this dynamic approach to punishment follows.

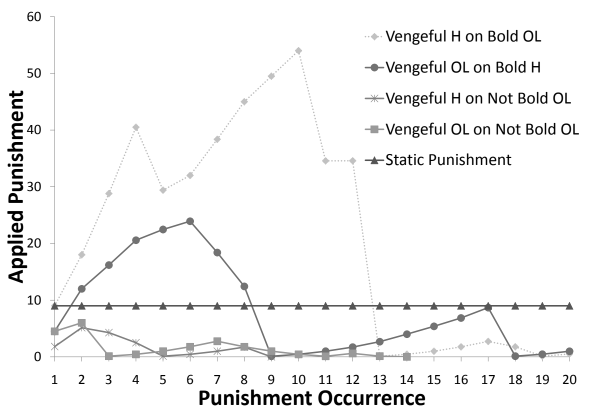

Figure 3. Punishment value and number: $pu = -9$ (H: Hub; OL: Outlier)

4.3 Punishment and Metapunishment Costs

While we can see that results improve through dynamic punishment, punishment and metapunishment values change in different ways depending on the situation. Figure 3 illustrates how two vengeful agents, a hub and an outlier (that punish regularly), adapt punishment when applying to hubs and outliers that have low boldness and high boldness. In addition, and for comparison, the figure includes a representation the static punishment of 9 that is used in the original model (shown as the horizontal line). Note that the x-axis represents the number of occurrences of punishment, rather than time-steps, so that line each indicates merely the trend of punishment applied by one agent to another. While the figure is limited to some illustrative cases due to space constraints, this is adequate to illuminate the dynamic punishment approach.

It can be seen from Figure 3 that bold agents incur high punishments regardless of whether the punishing agent is a vengeful hub or a vengeful outlier. However, in the case of the bold hub, the maximum value of punishment is much less than in the case of the bold outlier (about 23 compared to 54). This is because, since a hub is more exposed (and has very many connections), it will be punished by many other agents, and punishment that is applied to a hub thus need not be as high as for an outlier. Conversely, an outlier has few connections, requiring the punishment of a single agent to be sufficiently high to convince it to stop defecting. In addition, it can be seen that the high punishment does not persist for many occurrences since it drops to a very low level after very few occurrences, because bold agents respond positively to the punishment and the high level ceases to be required. In this way, occasional stricter punishments bring about a relatively quick response when compared to the static approach, but demand a cumulatively lower amount of punishment, suggesting that dynamic punishment can be more efficient.

Interestingly, Figure 3 also shows that when a vengeful agent interacts with a low boldness agent, both the vengeful hub and the vengeful outlier always apply less punishment than in the static approach. This is because a low boldness outlier rarely defects and little is needed to prevent it defecting. Overall, it seems clear that the punishment cost here is much less than in the static approach, especially over an extended period (the duration of the simulation).

Since punishment counters the tendency to defect, it needs to outweigh the impact of the temptation value (the reward from defecting). Importantly, an agent gains only one instance of this temptation

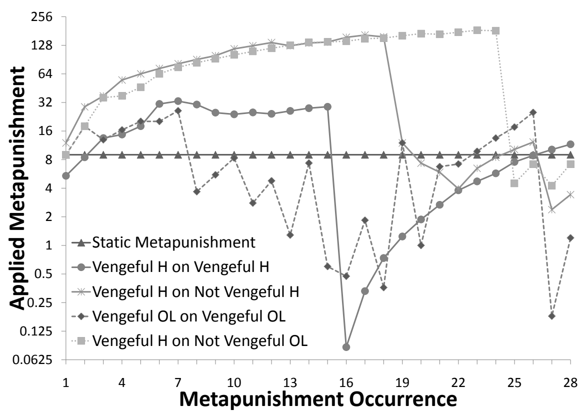

Figure 4. Dynamic metapunishment: $pu = -9$

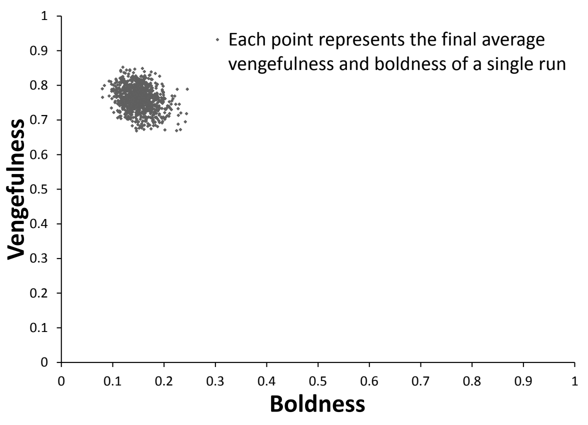

Figure 5. Impact of punishment unit when $pu = -1$

value as reward, but can incur multiple punishments (from each agent to which it is connected) in response. In contrast, because metapunishment instead counters the enforcement cost, instances of which are incurred for each agent that should be punished or metapunished, this can lead to much higher metapunishment, as shown in Figure 4. Enforcement costs cause agents to decrease their vengefulness because they lose utility, but metapunishment seeks to balance with the threat of losing even more utility by not punishing. The results in Figure 4 show that in our experiments, the metapunishment cost peaks at a level of 180, even though it is less than 9 (the fixed static metapunishment) in the majority of cases.

In fact, the different lines shown on the graph can be divided into two main categories: strongly vengeful agents metapunishing weakly vengeful agents, and highly vengeful agents metapunishing other highly vengeful agents. With regard to the first category of weakly vengeful agents, it is clear that very high metapunishments are needed to deal with agents that have low vengefulness. This is because such agents are less likely to impose punishments, especially given the enforcement costs that they are required to pay by increasing their vengefulness and punishing more agents as a result. However, such high metapunishment does not persist for long: it increases until around the 20th occurrence in the case of a hub, and until the 25th occurrence for an outlier. Then, metapunishment drops and remains below the traditional level of the static metapunishment.

With regard to the second category of highly vengeful agents, metapunishment only rises to a peak that is much lower than with weakly vengeful agents. Moreover, within this, a high level of metapunishment persists much longer for the vengeful hub than for the vengeful outlier. This is because hubs have many more connections than outliers, so that they are responsible for very many punishments and incur high enforcement costs. In consequence, metapunishment is needed to be much higher in order to counter this and to prevent the vengeful hub from decreasing its vengefulness to too low a level. While the same is true for vengeful outliers, it is for a much shorter period and for a lower cost, since outliers have far fewer connections.

4.4 Impact of Punishment Unit

In the previous experiments, an initial punishment unit of -9 was used, resulting in very high punishment. This is because, when a bold agent engages in a sequence of defections, the effect of the this high initial value multiplies the relatively high initial punishment unit. In seeking to understand the effect of this punishment value, we undertook several experiments to vary the initial value. While experiments were conducted with all integer values between -9 and -1, we show in Figure 5 only the results of using a punishment unit of -1 due to space constraints. However, all other experiments with these other values give very similar results. From the figure, it seems clear that even a punishment unit of -1 is sufficient to achieve norm emergence, since the average level of boldness is still low and the average level of vengefulness is high.

However, a deeper analysis reveals that establishment of the norm takes slightly longer with the use of lower punishment units. This is due to the cumulative effect of dynamic metapunishment requiring a longer period to overcome the high cumulative enforcement cost, especially in the case of hubs. In addition, smaller punishment units ensure that much more appropriate levels of punishment are applied. For example, while a punishment of -4 can be sufficient to disincentivise an agent from defecting, such an actual level of punishment is unlikely to arise when a high punishment unit, such as -9, is used since it takes too long to reduce to that level. Moreover, only the dynamic approach is able to achieve norm emergence when a low punishment unit is used. For example, the static punishment model does not succeed in achieving norm emergence if the static punishment cost is -1.

As can be observed from Figure 6, which shows the change in value of punishments over their different occurrences when a unit of -1 is used, it is clear that in the vast majority of cases, the punishment applied does not exceed the fixed static punishment used in the original model. In fact, punishment exceeds 9 (but only going as high as 12) only in the difficult case of bold outliers, which have few connections and require much more effort to regulate their behaviour.

In terms of metapunishment, as shown in Figure 7, the same trend is observed when using a punishment unit of -1, but with some important differences. First, the maximum metapunishment here is 60, as opposed to 180 in the case of punishment. This is because the value here is much lower, and this brings a much lower multiplier into the equation. However, it requires about 140 metapunishment occurrences to regulate a weakly vengeful agent, rather than 25 occurrences previously (not shown). In total, much less metapunishment can be used to regulate agent behaviour than with higher pun-

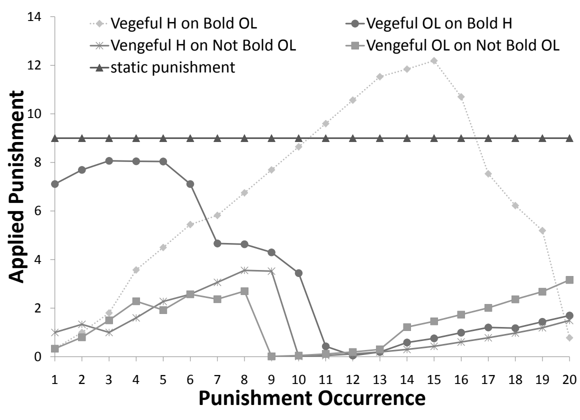

Figure 6. Dynamic punishment with $pu = -1$

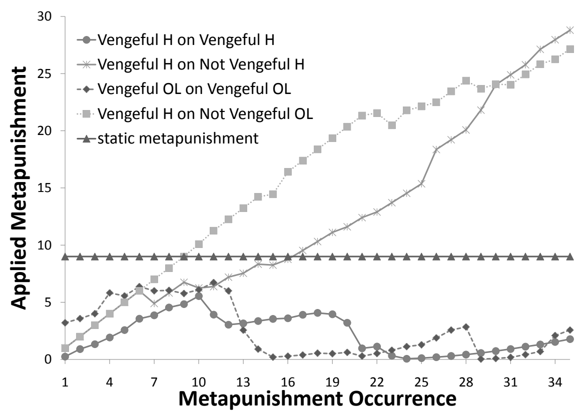

Figure 7. Dynamic metapunishment with $pu = -1$

ishment units, and with static punishment.

4.5 Agreement of Punishment

Based on the analysis of the results obtained from using the dynamic punishment approach with a variation of basic punishment units, there is a very interesting characteristic observed in relation to the agreement of punishment value. The agreement of punishment value means that all agents that decide to punish a defector are going to punish it with exactly the same amount. So for example, if an agent has three different neighbours and the agent has defected twice already. According to the model, all the neighbours would have observed the previous defections and would have recorded them in their memory regardless if they have punished the defecting agent in return of these defections or not. Because all agents are following the same technique in calculating their punishment, all of them will decide on using the same punishment and it make it look like all of them have come to a punishment agreement on what punishment should be used. This can be seen in-line with some interesting recent thread of research in which agents cooperate with each other to arrange on a common punishment value.

5 Conclusions

Punishment has been shown to be effective in encouraging norm establishment among groups of self interested agents. In this context, the classic view of punishment has adopted static punishments so that agents violating norms always incur the same degree of penalty. However, determining an appropriate level of punishment is difficult, with punishments being excessive in some cases. Importantly, such punishments can themselves decrease the utility of the overall system, by reducing the number of participants (as in P2P networks, for example), bringing counterproductive effects.

In response, in this paper, we described a mechanism for determining punishment values dynamically, using the prior experience of agents with those they are interacting, in order to specify the appropriate level. Through simulation experiments and results, the paper has shown that our technique helps to achieve norm emergence, but at a much lower cost to the system in terms of the punishment incurred, bringing benefits the society or system as a whole, and improving efficiency in ways that suggest potential value in real-world cases.

REFERENCES

[1] R. Axelrod, ‘An evolutionary approach to norms’, *The American Political Science Review*, **80**(4), 1095–1111, (1986).

[2] A. L. Barabasi and R. Albert, ‘Emergence of Scaling in Random Networks’, *Science*, **286**(5439), 509–512, (1999).

[3] M. Bowling and M. Veloso, ‘Rational and convergent learning in stochastic games’, in *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pp. 1021–1026, (2001).

[4] A. P. de Pinninck, C. Sierra, and M. Schorlemmer, ‘Distributed Norm Enforcement: Ostracism in Open MultiAgent Systems’, in *Computable Models of the Law*, volume 4884 of *LNCS*, 275–290, Springer, (2008).

[5] A. Perreau de Pinninck, C. Sierra, and M. Schorlemmer, ‘Friends no more: Norm enforcement in multi-agent systems’, in *Proceedings of the 6th International Conference on Autonomous Agents and Multi-Agent Systems*, eds., E. H. Durfee and M. Yokoo, pp. 640–642, (2007).

[6] D. Grossi, H. Aldewereld, and F. Dignum, ‘Ubi lex, ibi poena: Designing norm enforcement in e-institutions’, in *In Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems II*, pp. 107–120. Springer, (2006).

[7] K. Jaffe and L. Zaballa, ‘Co-operative punishment cements social cohesion’, *Journal of Artificial Societies and Social Simulation*, **13**, 3, (2010).

[8] S. Mahmoud, N. Griffiths, J. Keppens, and M. Luck, ‘An analysis of norm emergence in axelrod’s model’, in *Proc. 5th International Workshop on Normative Multi-Agent Systems*, (2010).

[9] S. Mahmoud, N. Griffiths, J. Keppens, and M. Luck, ‘Norm establishment via metanorms in network topologies’, in *Proc. IAT Workshop on Coordination, Organizations, Institutions and Norms*, (2011).

[10] S. Mahmoud, N. Griffiths, J. Keppens, and M. Luck, ‘Overcoming omniscience in axelrod’s model’, in *Proc. IAT Workshop on Coordination, Organizations, Institutions and Norms*, (2011).

[11] O. Sen and S. Sen, ‘Effects of social network topology and options on norm emergence’, in *Coordination, Organizations, Institutions and Norms in Agent Systems V*, volume 6069 of *LNCS*, 211–222, (2010).

[12] D. Villatoro, G. Andrighetto, J. Sabater-Mir, and R. Conte, ‘Dynamic sanctioning for robust and cost-efficient norm compliance’, in *Proc. 22nd International Joint Conference on Artificial Intelligence*, (2011).

[13] D. Villatoro, S. Sen, and J. Sabater-Mir, ‘Topology and memory effect on convention emergence’, in *Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technologies*, pp. 233–240. IEEE, (2009).

[14] C. J. C. H. Watkins and P. Dayan, ‘Q-learning’, *Machine Learning*, **8**(3-4), 279–292, (1992).

[15] F. López y López and M. Luck, ‘Modelling norms for autonomous agents’, in *Proceedings of The Fourth Mexican Conference on Computer Science*, pp. 238–245, (2003).