

## A MECHANISM DESIGN

Mechanism design prescribes a way for resolving compromise between self-interested agents (Nisan et al., 2007). For example, in the VCG mechanism (Clarke, 1971), all agents must reveal their incentives to a central coordinator, the *principal*. This mechanism achieves optimal group behavior by taxing each agent appropriately but then “burns” the collected payments, failing eliminate all the original inefficiency (Hartline and Roughgarden, 2008; Green and Laffont, 1979; Rothkopf, 2007), i.e., VCG is not strongly *budget-balanced*.

## B BAD NASH & FUTILE OPPONENT SHAPING

Here, we present a small two-player game where the Nash equilibrium results in poor outcomes for both agents individually and as a group. We then point out how although an opponent shaping approach would typically be able to manipulate players into avoiding such equilibria, it fails in this specific game. We seek a general algorithm for resolving multi-agent dilemmas and so we propose a new solution.

### B.1 BAD NASH

**Game 1** (Nash Paradox)  $\min_{x_1 \in [0,1]} f_1(x_1, x_2) = x_1^2 + \frac{1}{x_2 + \kappa}$ ,  $\min_{x_2 \in [0,1]} f_2(x_1, x_2) = x_2^2 + \frac{1}{x_1 + \kappa}$ .

The unique Nash equilibrium of this general-sum game is  $(x_1, x_2) = (0, 0)$  regardless of  $\kappa \in [0, 1]$ ; at Nash, each player sees a loss of  $\frac{1}{\kappa}$ . The minimal total loss solution is  $(x_1, x_2) = (\sqrt{1 - \kappa}, \sqrt{1 - \kappa})$  for  $\kappa < 1$  where each player sees a loss of  $2 - \kappa$ . The price of anarchy is  $\frac{1/\kappa}{2 - \kappa}$  which goes to  $\infty$  as  $\kappa \rightarrow 0$ . For  $\kappa < \text{golden ratio} - 1 \approx 0.618$ , Nash achieves maximum total loss among all possible strategy sets. While computing a Nash is an important technical problem, Game 1 proves that even if a Nash can be computed, it may be undesirable. Thus solving for Nash is orthogonal to this work.

### B.2 GRADIENT DESCENT WITHOUT DESCENT

Game 1 shows that the Nash equilibrium can give the worst outcome for all agents. It follows that agents learning with gradient descent in this game must observe their loss increase upon their final approach to Nash. Why stick to gradient descent then? In multi-agent games, the adjustment of another player’s strategy coupled with our own can increase our loss. Let  $f_i(t)$  be shorthand for  $f_i(\mathbf{x}(t))$  where  $\mathbf{x}(t)$  contains all strategies at time (iteration)  $t$ . Then a series expansion (see Eqn (7)) of agent  $i$ ’s loss around the current time step makes this concrete:

$$f_i(t + \Delta t) = f_i(t) + \Delta t \frac{df_i}{dt} + \frac{\Delta t^2}{2} \frac{d^2 f_i}{dt^2} + \mathcal{O}(\Delta t^3) \quad (7)$$

$$= f_i(t) + \Delta t \frac{\partial f_i}{\partial x_i} \frac{dx_i}{dt} + \frac{\Delta t^2}{2} \left[ \frac{\partial^2 f_i}{\partial x_i^2} \left( \frac{dx_i}{dt} \right)^2 + \frac{\partial f_i}{\partial x_i} \frac{d^2 x_i}{dt^2} + 2 \sum_{j \neq i} \frac{\partial^2 f_i}{\partial x_i \partial x_j} \frac{dx_i}{dt} \frac{dx_j}{dt} \right] \\ + h\left(\frac{dx_{j \neq i}}{dt}\right) + \mathcal{O}(\Delta t^3) \quad (8)$$

where  $\Delta t > 0$  is a small learning rate and  $h(\frac{dx_{j \neq i}}{dt})$  contains terms that agent  $i$  cannot manipulate (i.e.,  $h$  is constant w.r.t. agent  $i$ ’s update dynamics,  $\frac{dx_i}{dt}$ ). We show the full derivation of the series expansion in Section C for those interested.

### B.3 THE UPDATE IS NOT THE ONLY PROBLEM

In Eqn (7), other agents can affect  $f_i(t + \Delta t)$  through the bold terms and  $h(\frac{dx_{j \neq i}}{dt})$ . The bold terms indicate where agent  $i$ ’s update couples with other players’ updates (Schäfer and Anandkumar, 2019). To account for these terms, agent  $i$  must predict the other agents’ updates,  $\frac{dx_j}{dt}$ , and understand how their behaviors affect agent  $i$ ’s loss,  $\frac{d^2 f_i}{dx_i dx_j}$ . Recent methods, such as LOLA, LookAhead and Stable Opponent Shaping (Foerster et al., 2018; Letcher et al., 2018), model these terms. However, all these

methods converge to Nash in Game 1 because  $\frac{d^2 f_i}{dx_i dx_j} = 0$  as do all other mixed derivatives of agent  $i$ 's loss. In contrast, agent  $i$  can never mitigate increases in loss due to  $h$ . Incorporating more terms in the expansion generates higher level reasoning, but even the infinite expansion cannot avoid the *Nash paradox* in Game 1. If  $x_1$  knows  $x_2$ 's learning trajectory converges to 0,  $x_1$  is still incentivized to play 0. **The fault lies in the game, not the learning.**

## C TAYLOR SERIES EXPANSION

Here, we derive the Taylor series expansion given in Section B.2. The derivation is as follows:

$$\frac{df_i}{dt} = \sum_j \frac{\partial f_i}{\partial x_j} \frac{dx_j}{dt} = \frac{\partial f_i}{\partial x_i} \frac{dx_i}{dt} + \sum_{j \neq i} \frac{\partial f_i}{\partial x_j} \frac{dx_j}{dt} \quad (9)$$

$$\frac{d^2 f_i}{dt^2} = \frac{d}{dt} \left( \frac{df_i}{dt} \right) = \frac{d}{dt} \left( \sum_j \frac{\partial f_i}{\partial x_j} \frac{dx_j}{dt} \right) = \sum_j \frac{d}{dt} \left( \frac{\partial f_i}{\partial x_j} \frac{dx_j}{dt} \right) \quad (10)$$

$$= \sum_j \left[ \frac{d}{dt} \left( \frac{\partial f_i}{\partial x_j} \right) \frac{dx_j}{dt} + \frac{\partial f_i}{\partial x_j} \frac{d^2 x_j}{dt^2} \right] \quad (11)$$

$$= \sum_j \left[ \left( \sum_k \frac{\partial^2 f_i}{\partial x_j \partial x_k} \frac{dx_k}{dt} \right) \frac{dx_j}{dt} + \frac{\partial f_i}{\partial x_j} \frac{d^2 x_j}{dt^2} \right] \quad (12)$$

$$= 2 \sum_{j \neq i} \frac{\partial^2 f_i}{\partial x_j \partial x_i} \frac{dx_i}{dt} \frac{dx_j}{dt} + \frac{\partial^2 f_i}{\partial x_i^2} \left( \frac{dx_i}{dt} \right)^2 \quad (13)$$

$$+ \sum_{j \neq i} \sum_{k \neq i} \frac{\partial^2 f_i}{\partial x_j \partial x_k} \frac{dx_j}{dt} \frac{dx_k}{dt} + \sum_{j \neq i} \frac{\partial f_i}{\partial x_j} \frac{d^2 x_j}{dt^2} + \frac{\partial f_i}{\partial x_i} \frac{d^2 x_i}{dt^2} \quad (14)$$

$$h\left(\frac{dx_{j \neq i}}{dt}\right) = \Delta t \sum_{j \neq i} \frac{\partial f_i}{\partial x_j} \frac{dx_j}{dt} + \frac{\Delta t^2}{2} \left[ \sum_{j \neq i} \sum_{k \neq i} \frac{\partial^2 f_i}{\partial x_j \partial x_k} \frac{dx_j}{dt} \frac{dx_k}{dt} + \sum_{j \neq i} \frac{\partial f_i}{\partial x_j} \frac{d^2 x_j}{dt^2} \right] \quad (15)$$

$$f_i(t + \Delta t) = f_i(t) + \Delta t \frac{\partial f_i}{\partial x_i} \frac{dx_i}{dt} + \frac{\Delta t^2}{2} \left[ \frac{\partial^2 f_i}{\partial x_i^2} \left( \frac{dx_i}{dt} \right)^2 + \frac{\partial f_i}{\partial x_i} \frac{d^2 x_i}{dt^2} + 2 \sum_{j \neq i} \frac{\partial^2 f_i}{\partial x_i \partial x_j} \frac{dx_i}{dt} \frac{dx_j}{dt} \right] \quad (16)$$

$$+ h\left(\frac{dx_{j \neq i}}{dt}\right) + \mathcal{O}(\Delta t^3). \quad (17)$$

## D DERIVATION OF AN UPPER BOUND ON *Local* PRICE OF ANARCHY

**Definition 2** (Smooth Game). A game is  $(\lambda, \mu)$ -smooth (Roughgarden, 2015) if:

$$\sum_{i=1}^N f_i^A(x_i, x'_{-i}) \leq \lambda \sum_{i=1}^N f_i^A(x_i, x_{-i}) + \mu \sum_{i=1}^N f_i^A(x'_i, x'_{-i}) \quad (18)$$

for all  $x, x' \in \mathcal{X}$  where  $\lambda > 0$ ,  $\mu < 1$ , and  $\sum_i f_i^A(x)$  is assumed to be non-negative for any  $x \in \mathcal{X}$ .

The last condition is needed for the price of anarchy to be meaningful.

**Lemma 2** (Smooth Games Imply a Bound on Price of Anarchy). *The price of anarchy,  $\rho$ , the ratio of the worst case Nash total loss to the minimal total loss, is bounded above by a ratio of the coefficients of a smooth game (Roughgarden, 2015):*

$$\rho = \frac{\max_{\mathcal{X}^*} \sum_i f_i^A(x^*)}{\min_{\mathcal{X}} \sum_i f_i^A(x)} \geq 1 \quad (19)$$

$$\leq \inf_{\lambda > 0, \mu < 1} \left[ \frac{\lambda}{1 - \mu} \right]. \quad (20)$$

where  $x^*$  is an element of the set of Nash equilibria,  $\mathcal{X}^*$ .

Assume the loss function gradients are Lipschitz as well. We say a loss function,  $f_i^A(x) = f_i^A(x_i, x_{-i})$ , has a  $\beta_i$ -Lipschitz gradient for all  $A$  if

$$\|\nabla_{x_i} f_i^A(x) - \nabla_{y_i} f_i^A(y)\| \leq \beta_i \|x - y\| \quad \forall x, y, A. \quad (21)$$

Note that this implies

$$\|\nabla_{x_i} f_i^A(x_i, z_{-i}) - \nabla_{y_i} f_i^A(y_i, z_{-i})\| \leq \beta_i \|x_i - y_i\| \quad \forall x_i, y_i, z_{-i}, A \quad (22)$$

as a special case.

The following lemmas are useful in deriving a local notion of smoothness.

**Lemma 3.** *If  $f_i^A(x_i, x_{-i}) = g_i(x)$  has a  $\beta_i$ -Lipschitz gradient, then*

$$\left| \|\nabla_{x_i} g_i(x)\| - \|\nabla_{y_i} g_i(y)\| \right| \leq \beta_i \|x - y\| \quad \forall x, y. \quad (23)$$

*Proof.* The proof proceeds in two main steps. First,

$$\|\nabla_{y_i} g_i(y)\| = \|\nabla_{x_i} g_i(x) + \nabla_{y_i} g_i(y) - \nabla_{x_i} g_i(x)\| \quad (24)$$

$$\leq \|\nabla_{x_i} g_i(x)\| + \|\nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\| \text{ by triangle inequality} \quad (25)$$

$$\leq \|\nabla_{x_i} g_i(x)\| + \beta_i \|x - y\| \text{ by Lipschitz gradient} \quad (26)$$

which implies  $\|\nabla_{y_i} g_i(y)\| - \|\nabla_{x_i} g_i(x)\| \leq \beta_i \|x - y\|$ . And vice versa,

$$\|\nabla_{x_i} g_i(x)\| = \|\nabla_{y_i} g_i(y) + \nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\| \quad (27)$$

$$\leq \|\nabla_{y_i} g_i(y)\| + \|\nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\| \text{ by triangle inequality} \quad (28)$$

$$\leq \|\nabla_{y_i} g_i(y)\| + \beta_i \|x - y\| \text{ by Lipschitz gradient} \quad (29)$$

which implies  $\|\nabla_{x_i} g_i(x)\| - \|\nabla_{y_i} g_i(y)\| \leq \beta_i \|x - y\|$ . The two implications together prove the lemma.  $\square$

**Lemma 4.** *If  $f_i^A(x_i, x_{-i}) = g_i(x)$  has a  $\beta_i$ -Lipschitz gradient, then*

$$\left| \|\nabla_{x_i} g_i(x)\|^2 - \|\nabla_{y_i} g_i(y)\|^2 \right| \leq 3\beta_i^2 \|x - y\|^2 + 2\beta_i \|\nabla_{x_i} g_i(x)\| \|x - y\| \quad \forall x, y. \quad (30)$$

*Proof.* The proof proceeds similarly to before. First,

$$\|\nabla_{y_i} g_i(y)\|^2 = \|\nabla_{x_i} g_i(x) + \nabla_{y_i} g_i(y) - \nabla_{x_i} g_i(x)\|^2 \quad (31)$$

$$\leq \left( \|\nabla_{x_i} g_i(x)\| + \|\nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\| \right)^2 \text{ by triangle inequality} \quad (32)$$

$$= \|\nabla_{x_i} g_i(x)\|^2 + \|\nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\|^2 + 2\|\nabla_{x_i} g_i(x)\| \|\nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\| \quad (33)$$

$$\leq \|\nabla_{x_i} g_i(x)\|^2 + \beta_i^2 \|x - y\|^2 + 2\beta_i \|\nabla_{x_i} g_i(x)\| \|x - y\| \text{ by Lipschitz gradient and Lemma 3} \quad (34)$$

which implies  $\|\nabla_{y_i} g_i(y)\|^2 - \|\nabla_{x_i} g_i(x)\|^2 \leq \beta_i^2 \|x - y\|^2 + 2\beta_i \|\nabla_{x_i} g_i(x)\| \|x - y\|$ . And vice versa,

$$\|\nabla_{x_i} g_i(x)\|^2 = \|\nabla_{y_i} g_i(y) + \nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\|^2 \quad (35)$$

$$\leq \left( \|\nabla_{y_i} g_i(y)\| + \|\nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\| \right)^2 \text{ by triangle inequality} \quad (36)$$

$$= \|\nabla_{y_i} g_i(y)\|^2 + \|\nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\|^2 + 2\|\nabla_{y_i} g_i(y)\| \|\nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\| \quad (37)$$

$$\leq \|\nabla_{y_i} g_i(y)\|^2 + \beta_i^2 \|x - y\|^2 + 2\beta_i \|\nabla_{y_i} g_i(y)\| \|x - y\| \text{ by Lipschitz gradient and Lemma 3} \quad (38)$$

which implies  $\|\nabla_{x_i} g_i(x)\|^2 - \|\nabla_{y_i} g_i(y)\|^2 \leq \beta_i^2 \|x - y\|^2 + 2\beta_i \|\nabla_{y_i} g_i(y)\| \|x - y\|$ . The two implications together imply

$$\left| \|\nabla_{x_i} g_i(x)\|^2 - \|\nabla_{y_i} g_i(y)\|^2 \right| \leq \beta_i^2 \|x - y\|^2 + 2\beta_i \max\{\|\nabla_{x_i} g_i(x)\|, \|\nabla_{y_i} g_i(y)\|\} \|x - y\| \quad (39)$$

$$\leq \beta_i^2 \|x - y\|^2 + 2\beta_i \max\{\|\nabla_{x_i} g_i(x)\|, \|\nabla_{x_i} g_i(x)\| + \beta_i \|x - y\|\} \|x - y\| \quad (40)$$

$$= 3\beta_i^2 \|x - y\|^2 + 2\beta_i \|\nabla_{x_i} g_i(x)\| \|x - y\| \quad (41)$$

where the last inequality follows from Lemma 3  $\square$

**Lemma 5.** *If  $f_i^A(x_i, x_{-i}) = g_i(x)$  has a  $\beta_i$ -Lipschitz gradient, then there exists a  $\Delta t > 0$  sufficiently small s.t.*

$$\langle \nabla_{x_i} g_i(x), \nabla_{x'_i} g_i(x') \rangle \geq \|\nabla_{x_i} g_i(x)\|^2 - \delta_i \Delta t - \gamma_i \Delta t^2 \geq 0 \quad (42)$$

where  $x'_i = x_i - \Delta t \nabla_{x_i} g_i(x)$  for each  $i$ ,  $\Delta t > 0$ ,  $\delta_i = \beta_i \|\nabla_{x_i} g_i(x)\| \zeta$ ,  $\gamma_i = 2\beta_i^2 \zeta^2$ , and  $\zeta = \sqrt{\sum_j \|\nabla_{x_j} g_j(x)\|^2}$ .

*Proof.* We begin with the assumption of a Lipschitz gradient which trivially implies the following:

$$\|\nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\| \leq \beta_i \|x - y\| \quad \forall x, y \quad (43)$$

$$\implies \|\nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y)\|^2 \leq \beta_i^2 \|x - y\|^2 \quad \forall x, y. \quad (44)$$

This, in turn, is equivalent to

$$\langle \nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y), \nabla_{x_i} g_i(x) - \nabla_{y_i} g_i(y) \rangle \leq \beta_i^2 \|x - y\|^2 \quad \forall x, y \quad (45)$$

$$= \|\nabla_{x_i} g_i(x)\|^2 + \|\nabla_{y_i} g_i(y)\|^2 - 2\langle \nabla_{x_i} g_i(x), \nabla_{y_i} g_i(y) \rangle \leq \beta_i^2 \|x - y\|^2 \quad \forall x, y. \quad (46)$$

Rearranging terms gives

$$\langle \nabla_{x_i} g_i(x), \nabla_{y_i} g_i(y) \rangle \geq \frac{1}{2} \left[ \|\nabla_{x_i} g_i(x)\|^2 + \|\nabla_{y_i} g_i(y)\|^2 - \beta_i^2 \|x - y\|^2 \right] \quad \forall x, y. \quad (47)$$

Now let  $y_i = x'_i = x_i - \Delta t \nabla_{x_i} g_i(x)$  for each  $i$ . Lemma 4 implies

$$\begin{aligned} \|\nabla_{x'_i} g_i(x')\|^2 &\geq \|\nabla_{x_i} g_i(x)\|^2 - 3\beta_i^2 \|x - y\|^2 - 2\beta_i \|\nabla_{x_i} g_i(x)\| \|x - y\| \\ &= \|\nabla_{x_i} g_i(x)\|^2 - 3\beta_i^2 \Delta t^2 \underbrace{\sum_j \|\nabla_{x_j} g_j(x)\|^2}_{\zeta^2} - 2\beta_i \Delta t \|\nabla_{x_i} g_i(x)\| \underbrace{\sqrt{\sum_j \|\nabla_{x_j} g_j(x)\|^2}}_{\zeta}. \end{aligned} \quad (48)$$

(49)

Then

$$\langle \nabla_{x_i} g_i(x), \nabla_{x'_i} g_i(x') \rangle \geq \frac{1}{2} \left[ \|\nabla_{x_i} g_i(x)\|^2 + \|\nabla_{x'_i} g_i(x')\|^2 - \Delta t^2 \beta_i^2 \sum_j \|\nabla_{x_j} g_j(x)\|^2 \right] \quad (50)$$

$$\geq \frac{1}{2} \left[ 2\|\nabla_{x_i} g_i(x)\|^2 - 2\beta_i \Delta t \|\nabla_{x_i} g_i(x)\| \zeta - 3\Delta t^2 \beta_i^2 \zeta^2 - \Delta t^2 \beta_i^2 \zeta^2 \right] \quad (51)$$

$$= \|\nabla_{x_i} g_i(x)\|^2 - \Delta t \beta_i \|\nabla_{x_i} g_i(x)\| \zeta - 2\Delta t^2 \beta_i^2 \zeta^2 \quad (52)$$

$$= \|\nabla_{x_i} g_i(x)\|^2 - \delta_i \Delta t - \gamma_i \Delta t^2 \quad (53)$$

where  $\delta_i = \beta_i \|\nabla_{x_i} g_i(x)\| \zeta$  and  $\gamma_i = 2\beta_i^2 \zeta^2$ . Note that  $\|\nabla_{x_i} g_i(x)\|^2 \geq 0$  and if  $\|\nabla_{x_i} g_i(x)\|^2 = 0$ , then  $\langle \nabla_{x_i} g_i(x), \nabla_{x'_i} g_i(x') \rangle = 0$ .  $\square$

**Lemma 6.** *If  $f_i^A(x_i, x_{-i}) = g_i(x)$  has a  $\beta_i$ -Lipschitz gradient, then*

$$f_i^A(x_i, x'_{-i}) \geq f_i^A(x'_i, x'_{-i}) + \langle \nabla_{x'_i} f_i^A(x'_i, x'_{-i}), x_i - x'_i \rangle - \frac{\beta_i}{2} \|x_i - x'_i\|^2 \quad (54)$$

where  $x'_i = x_i - \Delta t \nabla_{x_i} g_i(x)$  for each  $i$  and  $\Delta t > 0$ .

*Proof.* Let  $f_i^A(x_i, x'_{-i}) = h_i(x_i)$ . We begin with the assumption of a Lipschitz gradient which implies the following:

$$\|\nabla_{x_i} h_i(x_i) - \nabla_{y_i} h_i(y_i)\| \leq \beta_i \|x_i - y_i\| \quad \forall x, y \quad (55)$$

$$\implies |h_i(x_i) - h_i(y_i) - \langle \nabla_{y_i} h_i(y_i), x_i - y_i \rangle| \leq \frac{\beta_i}{2} \|x_i - y_i\|^2 \quad \forall x, y. \quad (56)$$

This then implies

$$h_i(x_i) = h_i(y_i) + \langle \nabla_{y_i} h_i(y_i), x_i - y_i \rangle + \kappa_i \|x_i - y_i\|^2 \quad \forall x, y \text{ where } \kappa_i \in [-\frac{\beta_i}{2}, \frac{\beta_i}{2}] \quad (57)$$

Rewriting with  $f_i^A$  for clarity, letting  $y_i = x'_i = x_i - \Delta t \nabla_{x_i} g_i(x)$  for each  $i$ , and selecting the lower bound gives

$$f_i^A(x_i, x'_{-i}) \geq f_i^A(x'_i, x'_{-i}) + \langle \nabla_{x'_i} f_i^A(x'_i, x'_{-i}), x_i - x'_i \rangle - \frac{\beta_i}{2} \|x_i - x'_i\|^2. \quad (58)$$

□

**Lemma 7.** *If every  $f_i^A(x_i, x_{-i}) = g_i(x)$  has a  $\beta_i$ -Lipschitz gradient, then by Lemmas 5 and 6 there exists a  $\Delta t$  such that*

$$\sum_{i=1}^N f_i^A(x_i, x'_{-i}) \geq \sum_{i=1}^N f_i^A(x'_i, x'_{-i}) + \underbrace{a_i}_{\geq 0} \quad (59)$$

where  $x'_i = x_i - \Delta t \nabla_{x_i} f_i^A(x)$  and  $a_i = \|\nabla_{x_i} f_i^A(x_i, x_{-i})\|^2 - \delta_i \Delta t - \gamma_i \Delta t^2$  for each  $i$ .

*Proof.* Consider simultaneous gradient descent dynamics. Let  $x'_i = x_i - \Delta t \nabla_{x_i} f_i^A(x)$ . Then by Lemmas 5 and 6 we find

$$f_i^A(x_i, x'_{-i}) \geq f_i^A(x'_i, x'_{-i}) + \langle \nabla_{x'_i} f_i^A(x'_i, x'_{-i}), x_i - x'_i \rangle - \frac{\beta_i}{2} \|x_i - x'_i\|^2 \quad (60)$$

$$= f_i^A(x'_i, x'_{-i}) + \Delta t \langle \nabla_{x'_i} f_i^A(x'_i, x'_{-i}), \nabla_{x_i} f_i^A(x_i, x_{-i}) \rangle - \frac{\beta_i}{2} \|x_i - x'_i\|^2 \quad (61)$$

$$= f_i^A(x'_i, x'_{-i}) + \Delta t \langle \nabla_{x'_i} f_i^A(x'_i, x'_{-i}), \nabla_{x_i} f_i^A(x_i, x_{-i}) \rangle - \frac{\beta_i}{2} \Delta t^2 \|\nabla_{x_i} f_i^A(x_i, x_{-i})\|^2 \quad (62)$$

$$\geq f_i^A(x'_i, x'_{-i}) + \underbrace{\|\nabla_{x_i} f_i^A(x_i, x_{-i})\|^2 \Delta t - \xi_i \Delta t^2 - \gamma_i \Delta t^3}_{a_i} \quad (63)$$

where  $\xi_i = \delta_i + \frac{\beta_i}{2} \|\nabla_{x_i} f_i^A(x_i, x_{-i})\|^2$ . The parameters  $\xi_i$  and  $\gamma_i$  are bounded, therefore, there exists a  $\Delta t > 0$  small enough such that  $a_i \geq 0$ . □

**Theorem 8** (Local Smoothness). *Given  $n$  losses,  $f_i^A(x)$ ,  $i \in \{1, \dots, n\}$ , with  $\beta_i$ -Lipschitz gradients there exists a  $\Delta t > 0$  sufficiently small such that the game defined by these losses is smooth only if*

$$\sum_{i=1}^N a_i \leq \lambda \sum_{i=1}^N f_i^A(x_i, x_{-i}) + (\mu - 1) \sum_{i=1}^N f_i^A(x'_i, x'_{-i}) \quad \forall x_i \quad (64)$$

where  $x'_i = x_i - \Delta t \nabla_{x_i} f_i^A(x)$  and  $a_i = \|\nabla_{x_i} f_i^A(x_i, x_{-i})\|^2 \Delta t - \xi_i \Delta t^2 - \gamma_i \Delta t^3 \geq 0$ . Note this is a necessary, not sufficient condition for a game to be globally smooth.

*Proof.* Plugging Lemma 7 into the original definition of smoothness for  $x'_i = x_i - \Delta t \nabla_{x_i} f_i^A(x)$  and  $\Delta t$  sufficiently small gives

$$\sum_{i=1}^N f_i^A(x'_i, x'_{-i}) + a_i \leq \sum_{i=1}^N f_i^A(x_i, x'_{-i}) \leq \lambda \sum_{i=1}^N f_i^A(x_i, x_{-i}) + \mu \sum_{i=1}^N f_i^A(x'_i, x'_{-i}). \quad (65)$$

Rearranging the outer terms of the inequalities gives

$$\sum_{i=1}^N a_i \leq \lambda \sum_{i=1}^N f_i^A(x_i, x_{-i}) + (\mu - 1) \sum_{i=1}^N f_i^A(x'_i, x'_{-i}). \quad (66)$$

□

Note this is different than the definition of local smoothness in (Roughgarden and Schoppmann, 2015).

**Theorem 9.** Given  $n$  losses,  $f_i^A(x)$ ,  $i \in \{1, \dots, n\}$ , with  $\beta_i$ -Lipschitz gradients there exists a  $\Delta t > 0$  sufficiently small such that the **utilitarian** local price of anarchy of the game (to  $\mathcal{O}(\Delta t^2)$ ) is upper bounded by

$$\rho \leq \max_i \left\{ 1 + \Delta t \operatorname{ReLU} \left( \frac{d}{dt} \log(f_i^A(x)) + \frac{\|\nabla_{x_i} f_i^A(x)\|^2}{f_i^A(x_i, x_{-i}) \bar{\mu}} \right) \right\} \quad (67)$$

where  $i$  indexes each agent and  $\bar{\mu}$  is a user defined positive scalar.

*Proof.* To ease exposition, let  $b_i = f_i^A(x_i, x_{-i})$  and  $c_i = f_i^A(x'_i, x'_{-i})$  so that local smoothness becomes

$$\sum_{i=1}^N a_i \leq \lambda \sum_{i=1}^N b_i + (\mu - 1) \sum_{i=1}^N c_i. \quad (68)$$

If each agent  $i$  ensures local *individual* smoothness is satisfied, i.e.,

$$a_i \leq \lambda_i b_i + (\mu_i - 1) c_i, \quad (69)$$

then this is sufficient to satisfy local smoothness

$$\sum_{i=1}^N a_i \leq \max_i \{\lambda_i\} \sum_{i=1}^N b_i + (\max_i \{\mu_i\} - 1) \sum_{i=1}^N c_i. \quad (70)$$

Rearranging inequality [69] and letting  $\hat{\mu}_i = 1 - \mu_i$ ,  $\hat{a}_i = a_i/b_i$ , and  $\hat{c}_i = c_i/b_i$  gives

$$\lambda_i \geq \frac{a_i}{b_i} - (\mu_i - 1) \frac{c_i}{b_i} \quad (71)$$

$$\lambda_i \geq \hat{a}_i + \hat{\mu}_i \hat{c}_i. \quad (72)$$

Let each agent  $i$  attempt to measure the local price of anarchy given the losses it observes on its trajectory and call this measure  $\rho_i$ . Then

$$\rho_i = \inf_{\lambda_i, \hat{\mu}_i} \left[ \frac{\lambda_i}{\hat{\mu}_i} \right] \quad (73)$$

$$\text{s.t.} \quad (74)$$

$$\lambda_i \geq \hat{a}_i + \hat{c}_i \hat{\mu}_i \quad (75)$$

$$\lambda_i \geq \hat{\mu}_i \quad (76)$$

$$\hat{\mu}_i > 0 \quad (77)$$

$$\hat{\mu}_i \leq \bar{\mu} \quad (78)$$

where constraint [75] ensures local individual smoothness, constraint [76] encodes that price of anarchy  $\geq 1$  by definition, and constraint [77] is required by the original conditions on  $\mu$  for smoothness. Note that including an additional constraint for  $\lambda_i > 0$  would be redundant and so is omitted. Constraint [78] is optional and included to encode a prior by the agents on the smoothness parameters.

Recall that  $\hat{a}_i$  and  $\hat{c}_i$  are both non-negative;  $\hat{c}_i$  controls the slope of constraint [75]. We can solve this optimization in closed form for the four distinct cases outlined in Figure 6.

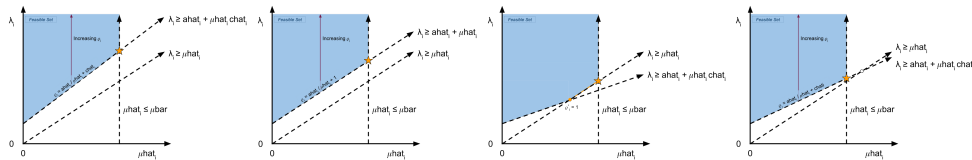


Figure 6: From left to right: a)  $\hat{c}_i > 1$ , b)  $\hat{c}_i = 1$ , c)  $\hat{c}_i < 1$  and  $\hat{c}_i + \frac{\hat{a}_i}{\bar{\mu}} \leq 1$ , d)  $\hat{c}_i < 1$  and  $\hat{c}_i + \frac{\hat{a}_i}{\bar{\mu}} > 1$ .

Figure 6 shows  $\hat{\mu}$  always leads to minimal  $\rho_i$  at  $\bar{\mu}$ , therefore  $\max_i \{\mu_i\} = \max_i \{1 - \hat{\mu}_i\} = 1 - \bar{\mu}$ . And so  $\rho \leq \frac{\max_i \{\lambda_i\}}{\bar{\mu}} = \max_i \{\rho_i\} = \max(1, \frac{\max_i \{\hat{a}_i + \bar{\mu} \hat{c}_i\}}{\bar{\mu}}) = \max(1, \max_i \{\frac{\hat{a}_i}{\bar{\mu}} + \hat{c}_i\})$ . Assuming  $\bar{\mu}$  is

large allows us to approximate with  $\max(1, \max_i \{\hat{c}_i\})$ , so the local price of anarchy is determined by the largest increase in loss over all the agents; if all losses are decreasing, the local price of anarchy is 1.

In summary, if  $\hat{c}_i < 1$  and  $\bar{\mu} \geq \frac{\hat{a}_i}{1-\hat{c}_i}$  (the intersection points of constraints [75] and [76]), then  $\rho_i = 1$ . The latter inequality,  $\frac{\hat{a}_i}{1-\hat{c}_i} \leq \bar{\mu}$ , can be rewritten as  $\hat{c}_i \leq 1 - \frac{\hat{a}_i}{\bar{\mu}}$ . Alternatively, if  $\hat{c}_i = 1$  and  $\bar{\mu} \rightarrow \infty$  (i.e., constraint [78] is omitted),  $\rho_i$  also equals 1. In all other cases,  $\rho_i = \frac{\hat{a}_i}{\bar{\mu}_i} + \hat{c}_i$ . If we assume  $\hat{a}_i > 0$  (i.e.,  $\|\nabla_{x_i} f_i^A(x)\| > 0$ ), we can reduce the cases above to

$$\begin{cases} \rho_i = 1, & \text{if } \hat{c}_i \leq 1 - \frac{\hat{a}_i}{\bar{\mu}} \\ \rho_i = \hat{c}_i + \frac{\hat{a}_i}{\bar{\mu}}, & \text{else.} \end{cases} \quad (79)$$

Let  $\epsilon_i = \frac{\hat{a}_i}{\bar{\mu}} > 0$ , then the two cases can be rewritten succinctly as

$$\rho_i = \max(1, \hat{c}_i + \epsilon_i). \quad (80)$$

If we expand  $\hat{c}_i$  as a series we find

$$\hat{c}_i = \frac{f_i^A(x')}{f_i^A(x)} \quad (81)$$

$$= \frac{f_i^A(x) + \frac{df_i^A(x)}{dt} \Delta t}{f_i^A(x)} + \mathcal{O}(\Delta t^2) \quad (82)$$

$$= 1 + \frac{\frac{df_i^A(x)}{dt}}{f_i^A(x)} \Delta t + \mathcal{O}(\Delta t^2). \quad (83)$$

Therefore, to  $\mathcal{O}(\Delta t^2)$ ,

$$\rho_i = \max(1, 1 + \left[ \frac{\frac{df_i^A(x)}{dt}}{f_i^A(x)} + \overbrace{\frac{\|\nabla_{x_i} f_i^A(x)\|^2}{f_i^A(x_i, x_{-i}) \bar{\mu}}}^{\epsilon_i} \right] \Delta t) \quad (84)$$

$$= 1 + \Delta t \max(0, \frac{\frac{df_i^A(x)}{dt}}{f_i^A(x)} + \frac{\|\nabla_{x_i} f_i^A(x)\|^2}{f_i^A(x_i, x_{-i}) \bar{\mu}}) \quad (85)$$

$$= 1 + \Delta t \text{ReLU}\left(\frac{d}{dt} \log(f_i^A(x)) + \frac{\|\nabla_{x_i} f_i^A(x)\|^2}{f_i^A(x_i, x_{-i}) \bar{\mu}}\right) \quad (86)$$

$$= 1 + \Delta t \text{ReLU}\left(\frac{d}{dt} \log(f_i^A(x))\right) \text{ as } \bar{\mu} \rightarrow \infty. \quad (87)$$

□

The following lemma establishes that the proposed bound may be tight in some games although we do not conjecture that this bound is at all tight in general.

**Lemma 10.** *The local  $\rho$  bound with  $\mu \rightarrow \infty$  in Eqn [67] is tight for some games.*

*Proof.* Consider the two player game with loss functions  $f_1(x_1) = x_1 - \kappa x_2$  and  $f_2(x_2) = x_2 - \kappa x_1$  for players 1 and 2 respectively with  $\kappa > 1$ . Assume the player strategies are constrained to the line segment  $x_1(\tau) = x_1 - \tau \Delta t$  and  $x_2(\tau) = x_2 - \tau \Delta t$  with  $\tau \in [0, 1]$ . Also, let  $x_1 = x_2$  and recall each player is assumed to run gradient descent

Then  $\frac{df_1}{dt} = \frac{\partial f_1}{\partial x_1} \frac{dx_1}{dt} + \frac{\partial f_1}{\partial x_2} \frac{dx_2}{dt} = \kappa - 1 > 0$ . Similarly,  $\frac{df_2}{dt} = \kappa - 1$ . Given  $x_1 = x_2$ , the price of anarchy bound simplifies to  $1 + \Delta t \text{ReLU} \frac{d}{dt} \log(f_i^A(x)) = 1 + \Delta t \text{ReLU} \frac{d/dt f_i^A(x)}{f_i^A(x)} = 1 + \Delta t \frac{\kappa-1}{f_i(x)}$ .

Also,  $f_1(x(\tau)) = x_1 - \tau \Delta t - \kappa(x_2 - \tau \Delta t) = x_1 - \kappa x_2 - \tau \Delta t(1 - \kappa) = f_1(x) + \tau \Delta t(\kappa - 1)$ . Likewise,  $f_2(x(\tau)) = f_2(x) + \tau \Delta t(\kappa - 1)$ . By inspection, the Nash occurs where  $x_1$  and  $x_2$  are minimal along the segment at  $\tau = 1$ , so  $x_1^* = x_1 - \Delta t$  and  $x_2^* = x_2 - \Delta t$ . The values at Nash are

$f_1(x^*) = f_1(x) + \Delta t(\kappa - 1)$  and  $f_2(x(\tau)) = f_2(x) + \Delta t(\kappa - 1)$ . In contrast, optimal group loss,  $\min_{x_1, x_2} (1 - \kappa)(x_1(\tau) + x_2(\tau))$ , occurs at  $\tau = 0$  and with values of  $f_1(x)$  and  $f_2(x)$ . This implies the true price of anarchy is  $1 + \Delta t \frac{2(\kappa-1)}{f_1(x)+f_2(x)}$ . Given  $x_1 = x_2$ , the true price of anarchy simplifies to  $1 + \Delta t \frac{\kappa-1}{f_i(x)}$  which is the same as the upper bound.  $\square$

The goal of this work is to derive an approximate proxy that can be both easily estimated and optimized. The bound we derive relies on first order information. It would be interesting to tighten the bound with second order information or by computing the price of anarchy for an appropriate polymatrix approximation to the game.

#### D.1 ACCOMMODATING NEGATIVE LOSS FUNCTIONS

In experiments, we replace the second term,  $\epsilon_i$ , with a constant hyperparameter  $\epsilon$ :

$$\rho_i = 1 + \Delta t \text{ReLU}\left(\frac{d}{dt} \log(f_i^A(x)) + \epsilon\right). \quad (88)$$

The log term appears due to price of anarchy being defined as the worst case Nash total loss divided by the minimal total loss. Although we have not defined an alternative price of anarchy, it is reasonable to believe one which defines the price of anarchy additively might drop the log term, leading to minimizing the following:

$$\hat{c}_i = f_i^A(x') - f_i^A(x) \quad (89)$$

$$= f_i^A(x) + \frac{df_i^A(x)}{dt} \Delta t - f_i^A(x) + \mathcal{O}(\Delta t^2) \quad (90)$$

$$= \frac{df_i^A(x)}{dt} \Delta t + \mathcal{O}(\Delta t^2) \quad (91)$$

so that

$$\rho_i = \Delta t \text{ReLU}\left(\frac{d}{dt} f_i^A(x) + \tilde{\epsilon}\right) \quad (92)$$

where  $\tilde{\epsilon}_i \approx \frac{\|\nabla_{x_i} f_i^A(x)\|^2}{\mu}$  is replaced in experiments with a constant hyperparameter,  $\tilde{\epsilon}$  as before. This objective is appealing as it does not require losses to be positive.

##### D.1.1 MULTIPLICATIVE VS ADDITIVE PRICE OF ANARCHY

In §2.5 we proposed an alternative gradient direction to the one derived in Eqn (5). This was a pragmatic change to make D3C amenable to games with negative loss, but may have appeared theoretically unappealing to the reader. Here, we show that the price of anarchy, as a multiplicative ratio, is already a somewhat arbitrary and non-robust choice.

Specifically, the price of anarchy of a game is not invariant to a global offset to the loss functions. Let the original price of anarchy of a game be  $\frac{a}{b}$ . Consider adding a constant  $c$  to each of the  $n$  losses in the game; note this does not change the locations of the Nash equilibrium or the total loss minimizer. However, the new price of anarchy becomes  $\frac{a+nc}{b+nc} \rightarrow 1$  as  $c \rightarrow \infty$ . On the other hand, let  $c \rightarrow -b/n$  from the right. Then the new price of anarchy approaches infinity. In summary, the price of anarchy, as defined multiplicatively, can be made arbitrarily large or small by adding a constant to each loss function in the game.

By removing the log term from the gradient,  $\nabla_{A_i} \rho_i$ , we effectively removed this effect. Lastly, the most important and general part of gradient direction,  $\nabla_{A_i} \rho_i$ , is the the Improve-Stay, Suffer-Shift component which is retained in  $\tilde{\nabla}_{A_i} \rho_i$ .

##### D.1.2 WHY MINIMIZE $\frac{d}{dt} f_i^A(x)$ W.R.T. $A_i$ ? WHY NOT $\frac{d}{dt} f_i(x)$ ?

The local price of anarchy is defined using the time derivative of the transformed loss. Instead, can agents minimize the time derivative of their original loss w.r.t.  $A_i$ ? Note the dependence on  $A_i$  appears in the time derivative terms through the update dynamics, e.g.  $\frac{dx_i}{dt} = \frac{dx_i}{dt}(A)$ .



In our loss mixing model, agent  $i$  can influence the update of agent  $j$  directly through  $A_i$ . This occurs because the transformed losses are computed using  $A^\top$  and so  $A_{ij}$  is used to re-mix agent  $j$ 's loss. This allows agent  $i$  to affect the  $h(\frac{dx_{j \neq i}}{dt})$  terms mentioned back in §B.2 and §B.3, circumventing the issues originally discussed in those sections.

However, we conducted experiments on the prisoner's dilemma using this approach, and although minimizing  $\frac{d}{dt} f_i(x)$  w.r.t.  $A_i$  worked for the 2-player variant, it failed to minimize the price of anarchy for 3, 5, or 10 players. Therefore, we discontinued its use in further experiments.

## D.2 EGALITARIAN PRICE OF ANARCHY

If the objective of interest is *egalitarian* rather than *utilitarian*, then a game is  $(\lambda, \mu)$ -smooth instead if:

$$\sum_{i=1}^N f_i^A(x_i, x'_{-i}) \leq \lambda \max_i f_i^A(x_i, x_{-i}) + \mu \max_i f_i^A(x'_i, x'_{-i}) \quad (93)$$

for all  $x, x' \in \mathcal{X}$  where  $\lambda > 0$ ,  $\mu < 1$ , and  $\max_i f_i^A(x)$  is assumed to be non-negative for any  $x \in \mathcal{X}$ .

The price of anarchy,  $\rho_e$ , gives the ratio of the worst case Nash max-loss to the minimal max-loss:

$$\rho = \frac{\max_{\mathcal{X}^*} \max_i f_i^A(x^*)}{\min_{\mathcal{X}} \max_i f_i^A(x)} \geq 1 \quad (94)$$

$$\leq \inf_{\lambda > 0, \mu < 1} \left[ \frac{\lambda}{1 - \mu} \right] \quad (95)$$

where  $x^*$  is an element of the set of Nash equilibria,  $\mathcal{X}^*$ .

**Theorem 11.** *Given  $n$  losses,  $f_i^A(x)$ ,  $i \in \{1, \dots, n\}$ , with  $\beta_i$ -Lipschitz gradients there exists a  $\Delta t > 0$  sufficiently small such that the local **egalitarian** price of anarchy of the game (to  $\mathcal{O}(\Delta t^2)$ ) is upper bounded by*

$$\rho_e \leq 1 + \Delta t \text{ReLU}\left(\frac{d}{dt} \log(\max_i \{f_i^A(x)\}) + \sum_{i=1}^N \frac{\|\nabla_{x_i} f_i^A(x)\|^2}{f_i^A(x_i, x_{-i}) \bar{\mu}}\right). \quad (96)$$

where  $i$  indexes each agent and  $\bar{\mu}$  is a user defined positive scalar.

*Proof.* By Lemma 7

$$\sum_{i=1}^N f_i^A(x'_i, x'_{-i}) + a_i \leq \sum_{i=1}^N f_i^A(x_i, x'_{-i}) \leq \lambda \max_i f_i^A(x_i, x_{-i}) + \mu \max_i f_i^A(x'_i, x'_{-i}). \quad (97)$$

Rearranging the outer terms of the inequalities gives

$$\sum_{i=1}^N a_i \leq \lambda \max_i f_i^A(x_i, x_{-i}) + \mu \max_i f_i^A(x'_i, x'_{-i}) - \sum_{i=1}^N f_i^A(x'_i, x'_{-i}) \quad (98)$$

$$\leq \lambda \max_i f_i^A(x_i, x_{-i}) + (\mu - 1) \max_i f_i^A(x'_i, x'_{-i}) \quad (99)$$

$$\implies a \leq \lambda b + (\mu - 1)c. \quad (100)$$

where  $a = \sum_{i=1}^N a_i$ ,  $b = \max_i f_i^A(x_i, x_{-i})$ , and  $c = \max_i f_i^A(x'_i, x'_{-i})$ . The proof proceeds as before in the *utilitarian* case except the price of anarchy does not decompose into a max over agent-centric estimates.  $\square$

## E DESCRIPTION OF GAMES IN EXPERIMENTS

We describe the traffic network and prisoner's dilemma games in detail here. We point the reader to (Eccles et al., 2019a) for further details of Coins and (Hughes et al., 2018) for Cleanup.

### E.1 GENERATING NETWORKS THAT EXHIBIT BRAESS’S PARADOX

In order to randomly generate a traffic network exhibiting Braess’s paradox, it is sufficient to guarantee two properties. One is that the shortcut route is a strictly dominant path (shorter commute time). This ensures all agents take the shortcut in the Nash equilibrium. The other is that there exists a joint strategy avoiding the shortcut with lower total commute time than all agents taking the shortcut. We assume there are four drivers.

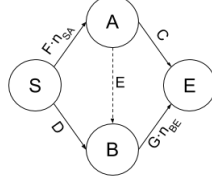


Figure 7: A theoretical traffic network with congestion parameters,  $F$  and  $G$ , and constant commute time parameters  $C$ ,  $D$ , and  $E$ .

The shortcut, SABE, is a strictly dominant (strictly shorter commute) if

$$Fn_{sa} + Gn_{be} + E < Fn_{sa} + C \quad (101)$$

$$Fn_{sa} + Gn_{be} + E < Gn_{be} + D \quad (102)$$

$$\implies E < \min\{C - Gn_{be}, D - Fn_{sa}\} \quad (103)$$

$$\implies G < \frac{C}{n_{be}} \text{ which is ensured if } C > 4G \quad (104)$$

$$\implies F < \frac{C}{n_{sa}} \text{ which is ensured if } D > 4F. \quad (105)$$

And there exists a pure joint strategy with at least  $\Delta$  less total commute time if

$$\tau_{Nash} = 4(4(F + G) + E) \quad (106)$$

$$\tau_{Opt} = \arg \min_{n_{sa} \in \{1,2,3\}, n_{be}=4-n_{sa}} \{n_{sa}(Fn_{sa} + C) + n_{be}(Gn_{be} + D)\} \quad (107)$$

$$\tau_{Nash} > \tau_{Opt} + \Delta \implies E > \frac{\tau_{Opt} + \Delta}{4} - 4(F + G). \quad (108)$$

So we can randomly generate a Braess network with Algorithm 4.

---

**Algorithm 4** gen\_braess

---

```

fail ← True
while fail do
   $F \sim \{1, \dots, 20\}$ 
   $G \sim \{1, \dots, 20\}$ 
   $C \sim \{4G + 10, \dots, 4G + 20\}$  ▷ 10 is an arbitrary buffer
   $D \sim \{4F + 10, \dots, 4F + 20\}$  ▷ 20 is an arbitrary upper limit
   $\tau_{Opt} \leftarrow \arg \min_{n_{sa} \in \{1,2,3\}, n_{be}=4-n_{sa}} \{n_{sa}(Fn_{sa} + C) + n_{be}(Gn_{be} + D)\}$ 
   $E_{min} = \max\{\frac{\tau_{Opt} + \Delta}{4} - 4(F + G), 0\}$ 
   $E_{max} = \min\{C - 4G, D - 4F\}$ 
  if  $E_{min} < E_{max}$  then
    fail ← False
     $E \sim \{E_{min}, \dots, E_{max}\}$ 
  end if
end while
Output:  $C, D, E, F, G$ 

```

---

The expected commute times for this Braess network can be computed exactly given stochastic commuting policies. Consider a network with four drivers and let  $x_{ij}$  specify the probability of driver  $i$  taking route  $j$  through the network. Then let

$$\mathbf{x} = \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ \vdots \\ x_{41} \\ x_{42} \\ x_{43} \end{bmatrix}, \mathbf{b} = \begin{bmatrix} C \\ D \\ E \end{bmatrix}, M = \begin{bmatrix} F & 0 & F \\ 0 & G & G \\ F & G & F+G \end{bmatrix}, \mathbf{b}_r = \begin{bmatrix} \mathbf{b} \\ \mathbf{b} \\ \mathbf{b} \end{bmatrix}, M_r = \begin{bmatrix} M \\ M \\ M \end{bmatrix}, I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (109)$$

and let

$$S = [I \quad I \quad I \quad I], A_i = \begin{bmatrix} \mathbb{1}(i==1)I & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbb{1}(i==2)I & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbb{1}(i==3)I & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbb{1}(i==4)I \end{bmatrix}. \quad (110)$$

Then  $\tau_r = M_r S \mathbf{x} + \mathbf{b}_r$  gives commute time for each path replicated for four agents:

$$\tau_r = M_r S \mathbf{x} + \mathbf{b}_r \quad (111)$$

$$= \begin{bmatrix} \text{top route time for player 1} \\ \text{bottom route time for player 1} \\ \text{shortcut time for player 1} \\ \vdots \\ \text{top route time for player 4} \\ \text{bottom route time for player 4} \\ \text{shortcut time for player 4} \end{bmatrix}. \quad (112)$$

The expected commute time for agent 1 is just the inner product of the first 3 entries of this vector with agent 1's policy. We use the matrix  $A_i$  to effectively select the appropriate commute times from  $\tau_r$ . Continuing, let

$$Q_i = A_i^\top M_r S \quad (113)$$

$$d_i = A_i^\top \mathbf{b}_r = A_i \mathbf{b}_r \quad (114)$$

$$C_i = \text{Cov}(\mathbf{x}_i) = \text{diag}(\mathbf{x}_i) - \mathbf{x}_i \mathbf{x}_i^\top \quad (115)$$

$$C = \text{Cov}(\mathbf{x}) = \text{block\_diag}(C_i). \quad (116)$$

We can now write agent  $i$ 's loss as

$$l_i(\mathbf{x}) = (A_i \mathbf{x})^\top \tau_r \quad (117)$$

$$= \mathbf{x}^\top Q_i \mathbf{x} + d_i^\top \mathbf{x} \quad (118)$$

$$\mathbb{E}[l_i(\mathbf{x})] = \mathbb{E}[\mathbf{x}^\top Q_i \mathbf{x}] + d_i^\top \bar{\mathbf{x}} \quad (119)$$

$$= \text{Tr}(Q_i C) + \mathbf{x}^\top Q_i \mathbf{x} + d_i^\top \bar{\mathbf{x}} \quad (120)$$

$$= \text{Tr}(M C_i) + \mathbf{x}^\top Q_i \mathbf{x} + d_i^\top \bar{\mathbf{x}} \quad (121)$$

which is easily amenable to analysis and makes the fact that the loss is quadratic, readily apparent.

## E.2 A REFORMULATION OF THE PRISONER'S DILEMMA

In an  $n$ -player prisoner's dilemma, each player must decide to defect or cooperate with each of the other players creating a combinatorial action space of size  $2^{n-1}$ . This requires a payoff tensor with  $2^{n(n-1)}$  entries. Instead of generalizing prisoner's dilemma (Rapoport et al., 1965) to  $n$  players using  $n$ th order tensors, we translate it to a game with convex loss functions. Figure 8 shows how we can accomplish this. Generalizing this to  $n$  players, we say that for all  $i, j, k$  distinct, 1) player  $i$  wants to defect against player  $j$ , 2) player  $i$  wants player  $j$  to defect against player  $k$ , and 3) player  $i$  wants

		(0,1/2)		(1/2,1/2)
	Player 2	Cooperate		
		1/4 / 1/4	1/2 / 1/2	
		$f_1(x) = x_1^2 + (x_2 - 1)^2$		
		$f_2(x) = x_2^2 + (x_1 - 1)^2$		
	Defect	1 / 1	1/4 / 1/4	
		(0,0)		(1/2,0)
		Defect	Cooperate	
		*Nash*	Player 1	

Figure 8: A reformulation of the prisoner’s dilemma using convex loss functions instead of a normal form payoff table.

player  $j$  to cooperate with itself. In other words, each player desires a free-for-all with the exception that no one attacks it. See §E.2 for more details.

We can define the vector of loss functions succinctly with

$$\mathbf{f}(\mathbf{x}) = \left( \begin{bmatrix} \mathbf{x}^\top \\ \mathbf{x}^\top \\ \mathbf{x}^\top \end{bmatrix} - C \right)^2 \quad (\text{elementwise}) \quad (122)$$

where  $\mathbf{x} = [x_{ij}]$  is a column vector ( $i \in [1, n], j \in [1, n-1]$ , values flattened in major-row order) containing the player strategies and  $C$  is an  $n \times n(n-1)$  matrix with entries that either equal 0 or  $c \in \mathbb{R}^+$ .

More specifically,  $C$  is a circulant matrix with column order reversed. For example, the matrix  $C$  associated with the three player game is

$$C = \begin{bmatrix} 0 & 0 & c & 0 & 0 & c \\ 0 & c & 0 & 0 & c & 0 \\ c & 0 & 0 & c & 0 & 0 \end{bmatrix} \quad (123)$$

where  $c > 0$ . Setting  $x_{ij} = 0$  encodes that player  $i$  has defected against its  $j$ th opponent. In the first row of  $C$  above, the first two entries can be read as player 1 is incentivized to defect against players 2 and 3. The next two entries state that player 1 receives a penalty if player 2 doesn’t cooperate, but wants player 2 to defect against player 3. The final two entries state that player 1 receives a penalty if player 3 doesn’t cooperate, but wants player 3 to defect against player 2. The matrix,  $C$ , can be constructed for  $n$ -player games with numpy (Oliphant, 2006) as

```
row = numpy.array(( [0] * (n-1) + [c] ) * (n-1) ) [::-1]
C = scipy.linalg.circulant(row) [:n, ::-1]
```

Note that this matrix is of size  $n \times n(n-1)$  or  $\mathcal{O}(n^3)$  entries.

The minimal total loss for this problem is  $(n-1)^2 c^2$  and occurs at  $x_{ij} = \frac{c}{n}$ :

$$f_{\text{total}} = \mathbf{1}^\top \tilde{\mathbf{f}}(\mathbf{x}) = \sum_{i=1}^n \sum_{j=1}^{n-1} (n-1)x_{ij}^2 + (x_{ij} - c)^2 \quad (124)$$

$$\frac{\partial f_{\text{total}}}{\partial x_{ij}} = 2(n-1)x_{ij} + 2(x_{ij} - c) = 0 \quad (125)$$

$$\implies x_{ij} = \frac{c}{n} \quad (126)$$

$$\implies f_{\text{total}} = n(n-1) \left[ \frac{(n-1)c^2}{n^2} + \frac{(n-1)^2 c^2}{n^2} \right] = (n-1)^2 c^2. \quad (127)$$

Nash occurs at the origin. This can be quickly derived by leveraging variational inequality theory (Facchinei and Pang, 2007; Nagurney and Zhang, 2012) and noticing that the Jacobian of gradient

descent dynamics is  $2\mathbf{I}$ , hence strongly monotone. Strongly monotone variational inequalities have unique a Nash equilibrium coinciding with the strategy set at which the gradients are all zero (assuming this point lies in  $\mathcal{X}$ ). The total loss at Nash ( $x_{ij} = 0$ ) is  $n(n-1)c$  by inspection.

### E.2.1 PRISONER’S DILEMMA VARIANT EXPERIMENTS

Figure 9 shows that D3C with a randomly initialized strategy successfully minimizes the price of anarchy. In contrast, gradient descent learners provably converge to Nash at the origin with  $\rho = \frac{n}{c(n-1)}$ . The price of anarchy grows unbounded as  $c \rightarrow 0$ . We set  $n = 10$  and  $c = 1$  in experiments for a mild  $\rho = \frac{10}{9}$ .

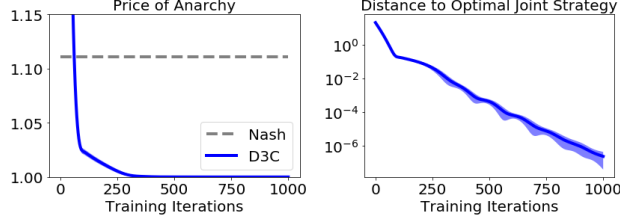


Figure 9: **Prisoner’s Dilemma** Convergence to  $\rho = 1$  (left) and the unique optimal joint strategy (right) over 1000 runs. The shaded region captures  $\pm 1$  standard deviation around the mean (too small to see on left). Gradient descent (not shown) provably converges to Nash.

Figure 10 highlights a single training run. Both agents are initialized to minimize their original loss, but then learn over training to minimize the mean of the two player losses.

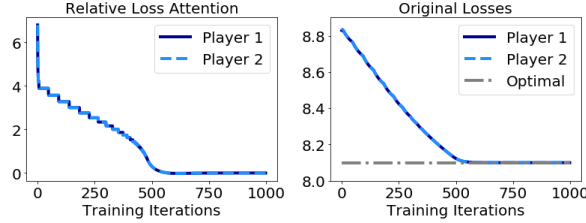


Figure 10: **Prisoner’s Dilemma** Single run: relative loss attention measured as  $\ln(\frac{A_{ii}}{A_{j \neq i}})$  (left) and player losses,  $f_i$ , (right).

### E.2.2 COOPERATION ROBUST TO MAVERICKS

**Proposition 12.** *In heterogeneous populations containing both D3C agents and selfish (gradient descent) agents, D3C agents end up with strictly lower loss when playing the proposed reformulation of the prisoner’s dilemma.*

*Proof.* Note that player  $i$  controls variables  $x_{ij}$  and suffers loss  $f_i(x)$ . Assume some subset of the players defect and play some fixed strategy. Let this subset be the players 1 through  $m$  w.l.o.g. because the player losses are symmetric. The remaining player (non-defector) losses can be rewritten as

$$f_{i>m}(x) = f(x|_{C_{\{i>m, j>m(n-1)\}}}) + \mathcal{K} \quad (128)$$

where  $\mathcal{K}$  is some vector-valued constant independent of these non-defectors’ strategies. Due to the structure of  $C$ , the losses that remain simply represent a  $(n-m)$ -player prisoner’s dilemma. To see this, consider player 1 defecting in a 3-player prisoner’s dilemma, i.e., consider the  $C_{\{i>1, j>2\}}$  submatrix. The loss functions for players 2 and 3 depend in exactly the same way on the variables  $x_{21}$  and  $x_{32}$ , i.e.,  $(x_{21}-0)^2 + (x_{32}-0)^2 + \dots$ , therefore, they will both agree on setting  $x_{21} = x_{32} = 0$ . The game that remains is exactly the 2-player prisoner’s dilemma between players 2 and 3. So

assuming these players run our proposed algorithm (D3C), they will converge to minimizing total loss of this subgame.

Of particular interest is the case where the defectors naively play fixed selfish strategies, i.e.,  $x_{ij} = 0$ . In this case, cooperating agents not only achieve lower subgroup loss, but also lower individual loss.

Recall that the loss for each player when all defect (naive selfish play implies  $x_{ij} = 0$ ) is  $n - 1$ . If only a subset of players defect and the remaining cooperate, the defectors achieve losses greater than  $n - 1$ —this can be seen from the fact that  $x_{ij} = 0$  is a strict Nash. Therefore, if we show that a cooperator’s loss is less than  $n - 1$ , we prove that cooperators outperform defectors.

Each defector adds 1 to the loss of a cooperator and the loss due to the cooperators’ subgame prisoner’s dilemma is  $\frac{(n-m-1)^2}{n-m}$  (Eqn (127)). Therefore, the loss of a cooperator is  $m + \frac{(n-m-1)^2}{n-m}$ . The loss of a defector is always greater:

$$\underbrace{(n-1)}_{\text{defector}} - \underbrace{m - \frac{(n-m-1)^2}{n-m}}_{\text{cooperator}} = (n-m-1) - \frac{(n-m-1)^2}{n-m} \quad (129)$$

$$= \frac{(n-m)(n-m-1) - (n-m-1)^2}{n-m} = \frac{n-m-1}{n-m} > 0. \quad (130)$$

□

## F ADDITIONAL EXPERIMENTS

We present additional results on three RL experiments, one small game as another counterargument to welfare-maximization, and a negative result for local  $\rho$ -minimization (which D3C is an instance of).

### F.1 TRUST-YOUR-BROTHER

In this game, a predator chases two prey around a table. The predator is a bot with a hard-coded policy to move towards the nearest prey unless it is already adjacent to a prey, in which case it stays put. If the prey are equidistant to the predator, the predator flips a coin and moves according to the coin flip. The prey receive 0 reward if they chose not to move and  $-0.01$  if they attempted to move. They additionally receive  $-1$  if the predator is adjacent to them after moving.

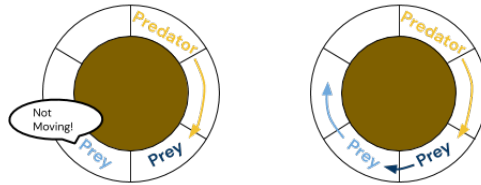


Figure 11: **Trust-Your-Brother** A bot chases agents around a table. The predator’s prey can only escape if the other prey simultaneously moves out of the way. Selfish (left), cooperative (right).

The prey employ linear softmax policies (no bias term) and train via REINFORCE (Williams, 1992). Both prey receive the same 2-d observation vector. The first feature specifies the counter-clockwise distance to the predator minus the clockwise distance for the blue prey. The second feature specifies the same for the green prey. Episodes last 5 steps and there are 6 grid cells in the ring around the table as shown in Figure 11.

Figure 12 shows D3C approaches maximal total return over training; this is achieved by the agents compromising on their original reward incentives and paying more attention to those of the other agent during training.

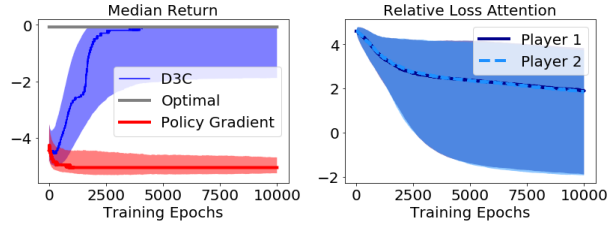


Figure 12: **Trust-Your-Brother** Median return achieved during training for agents trained with policy gradient vs policy gradient augmented with D3C (left); relative reward attention measured as  $\ln\left(\frac{A_{ii}}{A_{j \neq i}}\right)$  where a positive value corresponds to selfish attention and a negative value to other-regarding (right). The shaded region captures  $\pm 1$  standard deviation around the mean from 1000 runs.

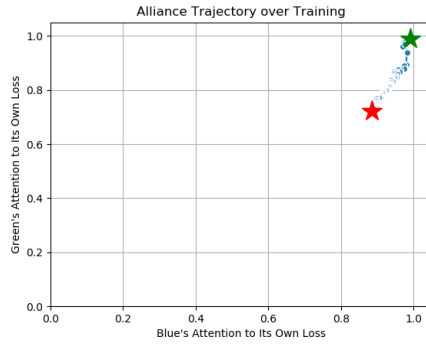


Figure 13: Agents are initialized to attend to their own losses. The trajectory here shows the agents compromising and adjusting to a mixture of losses (start at green, end at red star).

## F.2 LIO COMPARISON

Yang et al. (2020) propose an algorithm LIO (Learning to Incentivize Others) that equips agents with “gifting” policies represented as neural networks. At each time step, each agent observes the environment and actions of all other agents to determine how much reward to gift to the other agents. The parameters of these networks are adjusted to maximize the original environment reward (without gifts) minus some penalty regularizer for gifting meant to approximately maintain *budget-balance*. In order to perform this maximization, each agent requires access to every other agents action-policy, gifting-policy, and return making this approach difficult to scale and decentralize.

Yang et al. (2020) demonstrate LIO’s ability to maximize welfare and achieve division of labor on a restricted version of the Cleanup game with high apple re-spawn rates and where agents are constrained to facing in one direction (compare Figure 3 of (Yang et al., 2020) with Figure 1A of (Hughes et al., 2018)). While Yang et al. (2020) show AC failing to achieve maximal welfare, we found the opposite result using A2C (Espeholt et al., 2018) in Figure 14. In Figure 14, we also see that D3C is able to achieve near optimality. LIO appears to be approach maximal welfare as well in Figure 6C, therefore, this environment setting does not appear to differentiate the two approaches.

## F.3 HARVESTPATCH

McKee et al. (2020) introduce HarvestPatch as a common-pool resource game where apples spawn in predefined patches throughout a map. Agents must abstain from over-farming patches to the point of extinction by distributing their apple consumption as a group evenly across patches.

Figure 15 compares D3C against direct welfare maximization (Cooperation) and individual agent RL (A2C) on HarvestPatch.

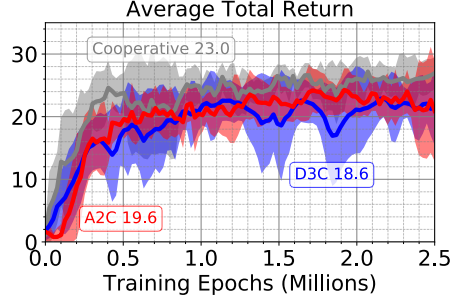


Figure 14: **Mini-Cleanup** Comparison against the mini Cleanup environment described in (Yang et al., 2020). In LIO, each agent requires access to every other agent’s policy which makes implementing it within our decentralized codebase intractable. We suggest comparing the asymptotes of this plot with that of Figure 6C in (Yang et al., 2020).

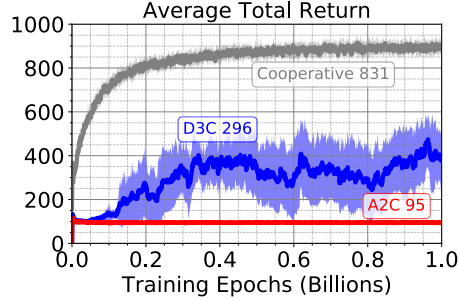


Figure 15: **HarvestPatch** Comparison against the HarvestPatch environment described in (McKee et al., 2020). D3C is able to increase welfare over the baseline approach of A2C at a slow rate.

#### F.4 IMPLICIT INEQUITY AVERSION

Welfare optimization can lead to poor outcomes as well, creating great inequity (Bertsimas et al., 2011; 2012; Gemici et al., 2018). We show that our approach generalizes beyond the goal of minimizing group loss to other interesting settings. **Game 2** (Efficient but Unfair):  $\min_{x_1 \in \mathbb{R}} x_1^2, \quad \min_{x_2 \in \mathbb{R}} x_2^2 - \frac{11}{10} x_1^2$ .

The minimal total loss solution of Game 2 is  $(x_1, x_2) = (\pm\infty, 0)$  where  $x_1$  achieves infinite loss and  $y$  achieves negative infinite loss. On the other hand, the Nash equilibrium is  $(x_1, x_2) = (0, 0)$  with a loss of zero for both agents. This hypothetical game may also arise if a loss is mis-specified. For example,  $x_1$ ’s true loss may have been  $2x_1^2$  implying no inequity issue with total loss minimization in the original game. The inequity of the cooperative solution to Game 2 may be undesirable. D3C converges to losses of 1.079 and  $-1.162$  for  $x_1$  and  $x_2$  respectively (sum is  $-0.083$ ) with  $x_1$  shifting its relative loss attention to  $\frac{A_{11}}{A_{12}} \approx \frac{11}{10}$  effectively halting training.

#### F.5 LIMITS OF A LOCAL UPDATE

We use a 2-player bilinear matrix game to highlight the limitations of a local  $\rho$ -minimization approach. Consider initializing  $A_{ij} = \frac{1}{2}$  so that the agents are purely cooperative. Even in this scenario, there are games where the agents minimizing local  $\rho$  will get stuck in local, suboptimal minima of the total loss landscape. Consider the following game transformed into an optimization problem via  $A_{ij} = \frac{1}{2}$ :

$$\min_{x_1} x_1^\top B_1 x_2 \quad \min_{x_2} x_1^\top B_2 x_2 \implies \min_{x_1} \min_{x_2} x_1^\top (B_1 + B_2) x_2 = x_1^\top C x_2 = f_C(x_1, x_2) \quad (131)$$

with  $x_1, x_2 \in \Delta^1$ . Let  $C = [a, b; c, d]$ . Then the Hessian of the cooperative objective  $f_C(x_1, x_2)$  has eigenvalues  $\pm|a - b - c + d|$ . This function is generally a saddle with possibly two local minima. For example, set  $a = d = 0$ ,  $b = -\frac{3}{4}$ , and  $c = -1$ . With random initializations, gradient descent will



converge to  $(p, q) = (1, 0)$   $\frac{3}{7}$  of the time with a value of  $b$ , else  $(p, q) = (1, 0)$  with a value of  $c$ , so we cannot expect local  $\rho$ -minimization to solve 2-player bilinear matrix games, in general, either.

## G AGENTS

### G.1 HYPERPARAMETERS

Game	$\eta_A$	$\delta$	$\nu$	$\tau_{\min}$	$\tau_{\max}$	$A_i^0$	$\epsilon$	$l$	$h$
Trust-Your-Brother	1.0	1.0	0.0	10	20	0.99	0.0	-5	5
Coins/Cleanup/HarvestPatch	$10^{-3}$	$10^{-1}$	$10^{-6}$	5	10	0.99	100.0	-5	5

Table 1: D3C hyperparameter settings for Algorithm 1

**Trust-Your-Brother:** The reinforcement learning algorithm,  $\mathbb{L}$ , used for D3C in Trust-Your-Brother is REINFORCE (Williams, 1992). Policy gradients are computed using batches of 10 episodes (full Monte Carlo returns, discount  $\gamma = 1$ ). Each batch of 10 episodes contains 5 episodes initialized with one prey closer to the predator, having only one grid space between itself and the predator. The other 5 episodes swap the prey so that each is attacked an equivalent number of times. Both prey always start in adjacent cells. The baseline subtracted from the returns is computed from linear value function. This value function is trained via temporal difference learning with a learning rate 0.1. The learning rate for REINFORCE is 0.1.

**Coins/Cleanup/HarvestPatch:** The reinforcement learning algorithm,  $\mathbb{L}$ , used for D3C in Coins, Cleanup, and HarvestPatch (§F.3) is A2C with V-trace (Espeholt et al., 2018).

Hyperparameter	Value
Entropy regularization	0.003
Baseline loss scaling	0.5
Unroll length	100
Discount ( $\gamma$ )	0.98
RMSProp learning rate	0.0004
RMSProp epsilon ( $\epsilon$ ) regularization parameter	$10^{-5}$
RMSProp momentum	0.0
RMSProp decay	0.99

Table 2: A2C hyperparameter settings for Coin, Cleanup, and HarvestPatch domains. No tuning or hyperparameter search was performed —these were default values used by our RL stack.

## H MISCELLANEOUS

### H.1 STEALING VS ALTRUISM

In our proposed mixing scheme, each agent  $i$  updates  $A_i \in \Delta^{n-1}$  and transformed losses are defined as  $\mathbf{f}^A = A^\top \mathbf{f}$ . This can be interpreted as each agent  $i$  deciding how to redistribute its losses over the other agents. In other words, agent  $i$  is deciding who to steal from (give loss equals steal reward).

Alternatively, we could define a scheme where each agent  $i$  updates  $A_i$ , however, the transformed losses are now defined as  $\mathbf{f}^A = A\mathbf{f}$  and the columns of  $A$  lie on the simplex. This scenario corresponds to agents taking on the losses of other agents. In other words, deciding which agents to help. In experiments on the prisoner’s dilemma, this approach did not make significant progress towards minimizing the price of anarchy so we discontinued its use in further experiments. In theory, this approach should be viable; it just requires that the information contained in agent  $j$ ’s loss is enough to accelerate descent of agent  $i$ ’s loss faster than the immediate loss (debt) that agent  $i$  takes on.

### H.1.1 TOWARDS A MARKET OF AGENTS

Expanding on this last perspective, when D3C agents, as defined in the main body, steal from other agents, they are exchanging immediate reward for information. The agent that is “stolen from” receives a loss signal that can then be used to derive policy update directions. The agent that is “stealing” receives immediate relief of loss, a form of payment. This exchange forms some of the components critical for a market economy of agents. The essential missing component is the negotiation phase where agents can choose to opt in or out of the exchange. In the current setting, the agent who steals is always able to force a transaction.

### H.2 RECIPROCITY IN COIN DOMAIN

To evaluate the extent to which there was a pattern of reciprocity in agents’ relative reward attention (i.e., the attention shifted synchronously), we conduct a permutation analysis. This permutation analysis estimates the probability that the level of synchrony we observe results from random chance.

We measure the synchrony between relative reward attention trajectories through co-integration (Murray, 1994). Co-integration allows us to estimate the synchrony between two timeseries. To do so, we take the discrete differences within each timeseries and then take the correlation of those two sequences of differences. If the timeseries are correlated, their movements should be correlated. This produces a set of co-integration coefficients ranging from 0.19 to 0.34 (see Figure 16, red).

To ensure that we are not overestimating the significance of these patterns, we employ a permutation analysis (Tibshirani and Efron, 1993). We resample the trajectories to calculate all possible values of co-integration coefficients (see Figure 16, blue). Comparing the real set against the full resampled set allows us to evaluate how extreme the real values are, under the assumption that there is no relationship between the two curves. The actual co-integration coefficients are the most extreme values across the full distribution of coefficients. To estimate the overall probability of this occurring, we evaluate the harmonic mean  $p$ -value (Wilson 2019). We find that the level of synchrony observed between the relative reward attention of co-learning agents significantly deviates from chance levels with  $p = 0.018$ .

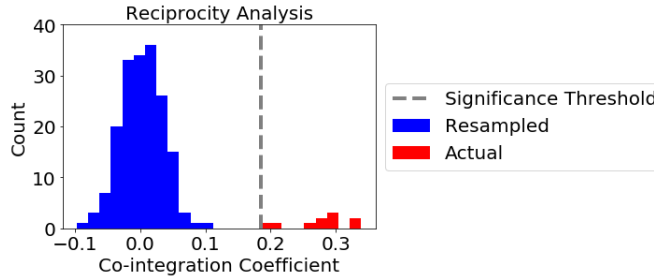


Figure 16: Histogram of co-integration coefficients for actual and resampled relative reward attention trajectories.

### H.3 CONVEX OPTIMIZATION VS SMOOTH 1-PLAYER GAMES

**Proposition 13.** *A convex loss function is not necessarily a smooth game where the players are interpreted as the elements of the variable to be minimized.*

*Proof.* Consider the following game:

$$\min_x (x + y)^2 \quad \min_y (x + y)^2. \quad (132)$$

Recall the definition of a smooth game (Definition 2) and let  $x = y = 0$  and  $x' = -y' = c$ . The game is not smooth for  $c > 0$  for any  $\lambda, \mu$  even though this is a convex optimization problem.  $\square$

#### H.4 GAMES WITH MIXING-AGNOSTIC UNIVERSALLY-STABLE NASH

Define the gradient map,  $F^A$ , and its Jacobian,  $J^A$ , for a game with loss vector  $\mathbf{f}$  concisely with

$$F^A(x) = [\langle A_i, \nabla_x \mathbf{f}(x) \rangle] \quad (133)$$

$$= \left[ \sum_j A_{ij} \frac{\partial f_j}{\partial x_i} \right] \quad (134)$$

$$J^A(x) = \left[ \sum_j A_{ij} \frac{\partial^2 f_j}{\partial x_i \partial x_k} \right] \quad (135)$$

$$= \left[ \sum_j A_{ij} H_{ik}^j \right] \quad (136)$$

where  $H^j$  is the Hessian of  $f_j(x)$ .

**Proposition 14.** *If each  $H^j$  is diagonally dominant, then  $J^A$  is diagonally dominant.*

*Proof.* We are given  $H_{ii}^j > \sum_{k \neq i} |H_{ik}^j|$ . Then

$$J_{ii}^A = \sum_j A_{ij} H_{ii}^j > \sum_j A_{ij} \sum_{k \neq i} |H_{ik}^j| \text{ by given \& } A_{ij} \geq 0 \quad (137)$$

$$= \sum_j \sum_{k \neq i} |A_{ij} H_{ik}^j| \text{ by } A_{ij} \geq 0 \quad (138)$$

$$= \sum_{k \neq i} \sum_j |A_{ij} H_{ik}^j| \text{ swap sums} \quad (139)$$

$$\geq \sum_{k \neq i} \left| \sum_j A_{ij} H_{ik}^j \right| \text{ by } \Delta\text{-inequality} \quad (140)$$

$$= \sum_{k \neq i} |J_{ik}^A|. \quad (141)$$

□

**Proposition 15.** *If each  $H^j$  is diagonally dominant and  $\mathcal{X}$  is unconstrained (i.e.,  $\mathbb{R}^d$  for some  $d$ ), then  $\mathbf{x}_A^*$  is the Nash equilibrium of the transformed game (i.e., with loss vector  $\mathbf{f}$  transformed by  $A$ ).*

*Proof.* Proposition 14 implies the dynamical system  $\dot{\mathbf{x}} = -F^A(\mathbf{x})$  is globally stable at  $\mathbf{x}_A^*$  for every fixed  $A$ . Proposition 14 also implies that each loss in the transformed game is convex. This is because  $J_{ii}^A$  is the Hessian of each loss  $i$  in the new game, and we showed these are positive. Moreover, the unique fixed point of an unconstrained game with convex losses is the solution to a suitably defined variational inequality:  $\text{VI}(F^A, \mathbb{R}^d)$ . This, in turn, implies that the fixed point is the Nash equilibrium of the game (Cavazzuti et al., 2002). □