Finite-horizon Equilibria for Neuro-symbolic Concurrent Stochastic Games

(Supplementary material)

Rui Yan [*1] **Gabriel Santos** [*1] **Xiaoming Duan**[2] **David Parker**[3] **Marta Kwiatkowska**[1]

[1]Department of Computer Science, University of Oxford, Oxford, UK
[2]Department of Automation, Shanghai Jiao Tong University, Shanghai, China
[3]School of Computer Science, University of Birmingham, Birmingham, UK

A PROOFS OF MAIN RESULTS

To prove Lemmas 6 and 8, we introduce the following example.

Example 1. *Consider a two-stage two-agent game with deterministic transitions in Fig. 6, in which each agent has two actions: $\{U, D\}$ for agent 1 and $\{L, R\}$ for agent 2. Non-leaf and leaf nodes, containing the node numbers, are marked with circles and rectangles, respectively. For clarity, several histories reaching stage 2 are not displayed here. Edges are labelled with the associated joint actions. The payoff vectors below leaf nodes are the terminal rewards, while the payoff vectors below non-leaf nodes denote the unique equilibrium payoffs (expected accumulated rewards) from these nodes to the leaf nodes, where ϕ is negative. The immediate rewards along the edges are assumed to be zero.*

By GBI, there are three NEs at node 4: $\mu^{4(1)} = \{(1,0),(1,0)\}$, $\mu^{4(2)} = \{(1/5,4/5),(1,0)\}$ and $\mu^{4(3)} = \{(0,1),(0,1)\}$, and the respective equilibrium payoffs are $V^{4(1)} = (0,8)$, $V^{4(2)} = (0,8/5)$ and $V^{4(3)} = (5,2)$. The NE and the equilibrium payoff at the initial node 1 depend on which NE is considered at node 4. If $V^{4(1)}$ or $V^{4(2)}$ is selected, then there is a unique NE at node 1: $\mu^{1(1)} = \{(1,0),(1,0)\}$ with equilibrium payoff $(1, 1+\phi)$. If $V^{4(3)}$ is chosen, then there is a unique NE at node 1: $\mu^{1(2)} = \{(0,1),(1,0)\}$ with equilibrium payoff $(5,2)$.

Proof of Lemma 6. We consider the game in Example 1. Given $\phi < 0$, the SW-SPNE and SW-SPCE starting at node 1 are the same and unique with social welfare $5 + 2 = 7$, in which the strategy at node 4 is $\mu^{4(3)}$. However, the SW-SPNE and SW-SPCE for the subgame starting at node 4 are both $\mu^{4(1)}$ instead of $\mu^{4(3)}$, which completes the proof.

Proof of Proposition 7. It is well known in game theory that, for a normal-formal game, (mixed-strategy) NEs always exist [Nash, 1951] and all NEs are fully characterized by the set of feasible solutions of a nonlinear program with compact constraints [Osborne et al., 2004]. This implies that the SWNEs, which are NEs maximising social welfare, always exist as well. Since every NE is a CE and all CEs are fully characterized by the set of feasible solutions of a linear program with compact constraints [Aumann, 1974], then SWCEs always exist, which completes the proof.

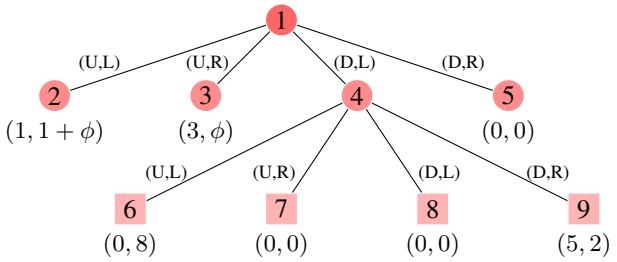

Figure 6: A two-stage game tree with two agents with $\phi < 0$.

Proof of Lemma 8. We consider Example 1 again. Since $\mu^{4(1)}$ has the maximum social welfare, then Generalized BI via SWE feeds $V^{4(1)}$ to node 1 for both the case of SWNE and SWCE, thus leading to node 1's social welfare $W^{\mu}_{0,s} = 2 + \phi$. However, node 1's social welfare $W^{\mu^*}_{0,s}$ under both SW-SPNE and SW-SPCE μ^* is 7. Thus, if ϕ is negative enough, the difference $W^{\mu^*}_{0,s} - W^{\mu}_{0,s} = 5 - \phi$ is positive and unbounded.

Proof of Theorem 9. The conclusions (i) and (ii) are straightforward by the encoding procedure. The sets of feasible solutions to (4) and (5) are not empty, as (mixed-strategy) NEs of a normal-form game always exist [Nash, 1951], and thus so do CEs. Additionally, they are compact by noting the constraints (1), (2) and (3). Then, the conclusions (iii) and (iv) follow from the continuity of the objective function.

Proof of Theorem 10. In Algorithm 2, step 1 returns a feasible solution to the nonlinear program (4) or (5) (depending on the equilibrium type T). Since the variables of

[*]Equal Contributions.

Accepted for the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022).

the nonlinear program P (step 5) are independent of the frozen variables due to the game tree structure and the history selection (or region construction), the pair (μ, V) in steps 7 and 8 is still a feasible solution to (4) or (5). The conclusions follow from the coordinate descent optimization with constraints [Wright, 2015].

B FURTHER DETAILS FOR ALGORITHMS

B.1 APPROXIMATION ALGORITHMS

FSI is described in Sec. 4 and is summarised as Algorithm 2. In Fig. 7, we give an illustration of the approach: FSI freezes all variables related to the red histories and optimizes over the blue history, where each node contains the current equilibrium payoff.

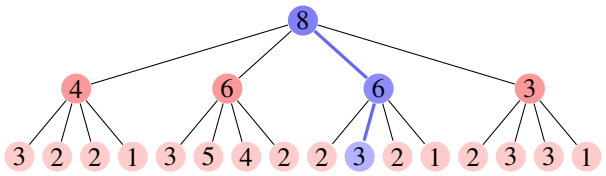

Figure 7: An example for Frozen Subgame Improvement.

We also suggest an alternative approach for the selection of histories in FSI, shown in Algorithm 3. It returns a history by starting from the initial state s, moving to the successor with the maximum social welfare indicated by the current equilibrium payoff V and perturbed by ϵ (if there are multiple such successors, we select one randomly), and iterating until the stage $K-1$, where $\mathsf{UNIFORM}(\cdot)$ is a uniform sampling function.

Algorithm 3 Finding a History by Maximum Social Welfare

Input: histories H_s, distribution μ, equilibrium payoff V, exploration rate $\epsilon \in [0,1]$
Output: a history $h \in H_s^{K-1}$

$h \leftarrow s$
repeat
  $h' \leftarrow \arg\max_{h'' \in \mathrm{Succ}(h)} \sum_{i \in N} V_i^{h''}$
  if $\mathsf{UNIFORM}([0,1]) > \epsilon$ **then**
    $h \leftarrow h'$
  else
    $h \leftarrow \mathsf{UNIFORM}(\mathrm{Succ}(h))$
  end if
until $h \in H_s^{K-1}$
return h

C FURTHER DETAILS FOR CASE STUDIES

C.1 AUTOMATED PARKING

The formal details of the NS-CSG model for the automated parking case study are as follows. There are two players (vehicles) $\{\mathsf{Ag}_i\}_{i \in N}$ for $N = \{1,2\}$ and two parking slots $M = \{1,2\}$ in a 5×4 grid C. The coordinate of the cell in the ith row and jth column is denoted by (i,j). Thus, $C = \{(i,j) \mid i \in [5], j \in [4]\}$. The coordinates of two parking slots are $y_1 = (2,4)$ and $y_2 = (5,1)$. Fig. 3 shows the grid. Vehicles are forbidden to enter the red cells and have to follow the traffic rules indicated by black arrows.

The environment state is $s_E = (x_1, x_2)$, where $x_i \in C$ is vehicle i's coordinate. Each agent $i \in N$ is as follows:

- a state of agent Ag_i is $s_i = (loc_i, (x_1, x_2))$, where the local state loc_i is dummy, and the coordinates $x_k \in C$ ($k \in N$) of two vehicles constitute the percept;
- actions include four directions $\mathrm{U} = (0,1)$, $\mathrm{D} = (0,-1)$, $\mathrm{L} = (-1,0)$, and $\mathrm{R} = (1,0)$. We assume that Ag_1 is twice as fast as Ag_2, i.e., $A_2 = \{\mathrm{U}, \mathrm{D}, \mathrm{L}, \mathrm{R}\}$ and $A_1 = A_2 \times A_2 \setminus \{\mathrm{UD}, \mathrm{DU}, \mathrm{LR}, \mathrm{RL}\}$;
- the available action function is such that $a_i \in \Delta_i(s_i)$ iff taking action a_i at s_i does not break the traffic rules or enter a red cell;
- observation function obs_i computes the cells where two vehicles are, i.e., $obs_i(s_1, s_2, s_E) = (x_1, x_2)$;
- the local transition function δ_i is dummy.

For $\alpha = (a_1, a_2) \in A_1 \times A_2$, $\delta_E(s_E, \alpha) = (x_1', x_2')$ where $x_i' = x_i + a_i$ for all $i \in N$. The two vehicles start from $x_1^0 = (3,1)$ and $x_2^0 = (2,2)$.

There are two reward structures. The first one is plain time minimizing: $r_i^A(s,\alpha) = 0$; if $x_1 = x_2$, then $r_i^S(s) = -20$; if $x_1 \neq x_2$ and $x_i = y_j$ for some $j \in M$, then $r_i^S(s) = 0$; $r_i^S(s) = -1$ otherwise. The second one is time minimizing with bonus, in which we add a bonus of 5.5 to agent 2 at a designated cell (in yellow): $r_2^S(s) = 5.5 - 1 = 4.5$ if $x_2 = (1,2)$ when $k \leq 1$.

This example was modelled using the PRISM-games modelling language, since the simplicity of the perception mechanism lets it be reduced to a discrete-state CSG.

C.2 TWO-AGENT AIRCRAFT COLLISION AVOIDANCE SCENARIO

In the VCAS[2] system (Figure 1) there are two aircraft (ownship and intruder, denoted by Ag_i for $i \in \{\mathrm{own}, \mathrm{int}\}$), each of which is equipped with an NN-controlled collision avoidance system called VCAS. Each second, VCAS issues an advisory (ad_i) from which, together with the current trust in the previous advisory (tr_i), the pilot needs to make a decision about accelerations, aiming at avoiding a near mid-air

collision (NMAC), a region where two aircraft are separated by less than 100 ft vertically and 500 ft horizontally.

The environment state $s_E = (h, \dot{h}_{\text{own}}, \dot{h}_{\text{int}}, t)$ records the altitude h of the intruder relative to the ownship (ft), the vertical climb rate $\dot{h}_{\text{own}}$ of the ownship (ft/sec), the vertical climb rate $\dot{h}_{\text{int}}$ of the intruder (ft/sec), and the time t until loss of horizontal separation of the two aircraft (sec).

Each aircraft is endowed with a perception function implemented via a feed-forward NN $f_{ad_i} : \mathbb{R}^4 \to \mathbb{R}^9$ with four inputs, seven hidden layers of 45 nodes and nine outputs representing the score of each possible advisory. There are nine NNs $F = \{f_i : \mathbb{R}^4 \to \mathbb{R}^9 \mid i \in [9]\}$, each of which corresponds to an advisory.

Each advisory will provide two non-zero acceleration actions for the agent to select from, except that the agent is also allowed to adopt zero acceleration. The trust in the previous advisory and previous advisory (percept) are stored in a state of the agent $s_i = (tr_i, ad_i)$. There are four trust levels $\{4, 3, 2, 1\}$ and nine possible advisories [Akintunde et al., 2020b]. The current advisory is computed from the previous advisory ad_i and environment state s_E using the observation function obs_i. The trust level is increased probabilistically if the current advisory is compliant with the executed action, and decreased otherwise.

Formally, each agent Ag_i for $i \in \{\text{own}, \text{int}\}$ and the environment E are defined as follows:

- $s_i = (tr_i, ad_i)$ is a state of the agent Ag_i with local state $tr_i \in [4]$ and percept $ad_i \in [9]$;
- the set of environment states is $S_E = [-3000, 3000] \times [-2500, 2500] \times [-2500, 2500] \times [0, 40]$, with $s_E = (h, \dot{h}_{\text{own}}, \dot{h}_{\text{int}}, t)$ as above;
- $A_i = \{0, \pm 3.0, \pm 7.33, \pm 9.33, \pm 9.7, \pm 11.7\}$, where $a_i \in A_i$ is an acceleration $\ddot{h}_i$;
- the available action function Δ_i returns two non-zero acceleration actions [Akintunde et al., 2020a] shown in Table 3 given a state of the agent, plus zero acceleration;
- observation function obs_i, implemented via F, is given by $ad_i' = obs_i(ad_i, s_E)$, where $obs_{\text{own}}(ad_{\text{own}}, s_E) = \text{argmax}(f_{ad_{\text{own}}}(h, \dot{h}_{\text{own}}, \dot{h}_{\text{int}}, t))$ and $obs_{\text{int}}(ad_{\text{int}}, s_E) = \text{argmax}(f_{ad_{\text{int}}}(-h, \dot{h}_{\text{int}}, \dot{h}_{\text{own}}, t))$;
- the local transition function δ_i computes a trust level according to the current trust level tr_i, the updated advisory ad_i' and the executed action a_i: if a_i is compliant with ad_i' (i.e., a_i is non-zero), when $tr_i \leq 3$, then $tr_i' = tr_i + 1$ with probability $1 - \epsilon_i$ and $tr_i' = tr_i$ with probability ϵ_i, and when $tr_i = 4$, then $tr_i' = tr_i$; otherwise, when $tr_i \geq 2$, then $tr_i' = tr_i - 1$ with probability $1 - \epsilon_i$ and $tr_i' = tr_i$ with probability ϵ_i, and when $tr_i = 1$, then $tr_i' = tr_i$, where $\epsilon_i \in [0, 1]$.
- the environment transition function $\delta_E(s_E, \alpha)$ is defined as: $h' = h - \Delta t(\dot{h}_{\text{own}} - \dot{h}_{\text{int}}) - 0.5\Delta t^2(\ddot{h}_{\text{own}} - \ddot{h}_{\text{int}})$, $\dot{h}_{\text{own}}' = \dot{h}_{\text{own}} + \ddot{h}_{\text{own}}\Delta t$, $\dot{h}_{\text{int}}' = \dot{h}_{\text{int}} + \ddot{h}_{\text{int}}\Delta t$ and $t' = t - \Delta t$, where $\Delta t = 1$ is the time step.

When computing the equilibria presented in Fig. 4, we use two reward structures, with the first given by $r^S_{\text{own}}(s) = r^S_{\text{int}}(s) = h$ if $k = t_{\text{init}} - t$, and 0 otherwise. For the zero-sum case, the reward for the intruder is negated. In both cases, action rewards are set to 0 for all state-action pairs, i. e., $r^A_{\text{own}}(s, \alpha) = r^A_{\text{int}}(s, \alpha) = 0, \forall s \in S, \alpha \in A$.

This case study was developed by extending the implementation available in [Michael E Akintunde and Lomuscio, 2020]. We first modified the original code in order to consider all actions recommended by the advisory system plus the action corresponding to zero acceleration. We later develop this model further by adding trust values to the states of the agents and the corresponding probabilistic updates as described in Section 2. In both cases, we build a game tree by considering all states the system could be in and translate that into a PRISM-games model.

We also consider another reward structure with additional preferences: (i) not only safety but also trust matters; and (ii) reducing fuel consumption is desired in addition to maintaining safety. More specifically, if $|h| \leq 200$, then $r^A_i(s, \alpha) = 0$ and $r^S_i(s) = |h|/h_{\max} + tr_i/4$; if $|h| > 200$, then $r^A_i(s, \alpha) = -|\ddot{h}_i|/\ddot{h}_{\max}$ and $r^S_i(s) = 0$ for $i \in \{\text{own}, \text{int}\}$, where $h_{\max}$ and $\dot{h}_{\max}$ are the maximal absolute values of all altitudes and accelerations in the generated game tree, respectively. The initial values are $h = 50$, $\dot{h}_{\text{own}} = -5$, $tr_{\text{own}} = 4$, $\dot{h}_{\text{int}} = 5$ and $tr_{\text{int}} = 4$.

References

Michael E. Akintunde, Elena Botoeva, Panagiotis Kouvaros, and Alessio Lomuscio. Verifying Strategic Abilities of Neural-symbolic Multi-agent Systems. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020)*, pages 22–32. IJCAI Organization, 9 2020a.

Michael E Akintunde, Elena Botoeva, Panagiotis Kouvaros, and Alessio Lomuscio. Formal verification of neural agents in non-deterministic environments. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020)*, pages 25–33. Springer, 2020b.

Robert J Aumann. Subjectivity and correlation in randomized strategies. *Journal of mathematical Economics*, 1(1): 67–96, 1974.

Panagiotis Kouvaros Michael E Akintunde, Elena Botoeva and Alessio Lomuscio. Venmas: Verification of neural-symbolic multi-agent systems, 2020. https://vas.doc.ic.ac.uk/software/neural/.

Label (ad_i)	Advisory	Description	Vertical Range (Min, Max) ft/min	Available Actions ft/s^2
1	COC	Clear of Conflict	$(-\infty, +\infty)$	-3, +3
2	DNC	Do Not Climb	$(-\infty, 0]$	-9.33, -7.33
3	DND	Do Not Descend	$[0, +\infty)$	+7.33, +9.33
4	DES1500	Descend at least 1500 ft/min	$(-\infty, -1500]$	-9.33, -7.33
5	CL1500	Climb at least 1500 ft/min	$[+1500, +\infty)$	+7.33, +9.33
6	SDES1500	Strengthen Descend to at least 1500 ft/min	$(-\infty, -1500]$	-11.7, -9.7
7	SCL1500	Strengthen Climb to at least 1500 ft/min	$[+1500, +\infty)$	+9.7, +11.7
8	SDES2500	Strengthen Descend to at least 2500 ft/min	$(-\infty, -2500]$	-11.7, -9.7
9	SCL2500	Strengthen Climb to at least 2500 ft/min	$[+2500, +\infty)$	+9.7, +11.7

Table 3: Two non-zero available actions given an advisory.

John Nash. Non-cooperative games. *Annals of mathematics*, pages 286–295, 1951.

Martin J Osborne et al. *An introduction to game theory*, volume 3. Oxford university press, New York, 2004.

Stephen J Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1):3–34, 2015.