Who Should I Trust? Cautiously Learning with Unreliable Experts Supplementary Material

Tamlin Love,[1*] Ritesh Ajoodha,[1] Benjamin Rosman[1]

[1]School of Computer Science and Applied Mathematics,
University of the Witwatersrand, Johannesburg, South Africa

[*]1438243@students.wits.ac.za

1 Pseudocode

Algorithm 1 outlines the standard process for solving SSDPs without any external information, as described in Section 2.1. of the paper. This can be compared to Algorithm 2, which outlines the CLUE process for solving SSDPs with the advice of multiple experts.

Algorithm 1 Single Stage Decision Problem

procedure STANDARD_SSDP($environment, N$) ▷ $N =$ number of trials
  for $t \in [0, ..., N-1]$ **do**
    $s_t \leftarrow$**sample_state**($environment$)
    $a_t \leftarrow$**select_action**(s_t) ▷ agent acts (e.g. Algorithms 4 - 7)
    $r_t \leftarrow$**execute_action**($a_t, environment$)
    learn(s_t, a_t, r_t) ▷ agent learns (e.g. Algorithm 3)

Algorithm 3 presents the action-value update rule as described in Equation 1 of the main paper, where $\alpha \in (0, 1]$ is the learning rate parameter.

Algorithms 4, 5, 6 and 7 outline how the unassisted Epsilon Greedy, Adaptive Greedy, Explore-then-Exploit (ETE) and Upper Confidence Bound (UCB) algorithms select actions

Algorithm 2 Cautiously Learning with Unreliable Experts

procedure CLUE(D, E, N) ▷ D = environment, E = panel of experts, N = number of trials
  for $t \in [0, ..., N-1]$ **do**
    $s_t \leftarrow$**sample_state**(D) ▷ environment selects state
    $a_t \leftarrow$**act**(s_t) ▷ Algorithm 8
    $r_t \leftarrow$**execute_action**(a_t, D) ▷ environment returns reward
    $advice_t \leftarrow$**advise**(E, s_t, a_t, r_t) ▷ e.g. Algorithm 10
    learn(s_t, a_t, r_t) ▷ e.g. Algorithm 3
    update($s_t, a_t, r_t, advice_t$) ▷ Algorithm 9

Algorithm 3 Baseline Approach for Solving SSDPs

procedure LEARN(s_t,a_t,r_t,t)
  $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t - Q(s_t, a_t))$

given a state s_t in trial t. In these algorithms, $\epsilon \in [0, 1]$ denotes the exploration parameter for Epsilon Greedy, $z \in \mathbb{R}$ denotes the utility threshold parameter for Adaptive Greedy, t_b denotes the time at which the ETE algorithm stops exploring and starts exploiting, $c \in \mathbb{R}$ denotes the UCB parameter that balances exploration and exploitation, and $N(s, a)$ is a function that denotes the number of times the state-action pair (s, a) has been observed.

Algorithm 4 Epsilon Greedy

procedure ACT(s_t,ϵ)
  $p \leftarrow$**random**()
  if $p < \epsilon$ **then**
    return random $a \in A$ ▷ agent "explores"
  else
    return $\arg\max_a Q(s_t, a)$ ▷ agent "exploits"

Algorithm 8 outlines the process by which a CLUE agent selects an action in trial t for state s_t, given the threshold parameter T and the reliability estimate $\mathbb{E}[\rho^{(e)}]$ of each expert e in E. This augments an unassisted action-selection algorithm (e.g. Algorithms 4 - 7). The procedure for determining whether or not the agent is exploring is discussed in Section 3.2 of the main paper.

Algorithm 5 Adaptive Greedy

procedure ACT(s_t,z)
  $a_t^* \leftarrow \arg\max_a Q(s_t, a)$
  if $Q(s_t, a_t^*) > z$ **then**
    return a_t^* ▷ agent “exploits"
  else
    return random $a \in A$ ▷ agent “explores"

Algorithm 6 Explore-then-Exploit

procedure ACT(s_t,t,t_b)
  if $t \geq t_b$ **then**
    return $\arg\max_a Q(s_t, a)$ ▷ agent “exploits"
  else
    return random $a \in A$ ▷ agent “explores"

Algorithm 9 outlines the process by which a CLUE agent updates the reliability estimate of each expert, where $v_t^{(e)}$ denotes the advice given by expert e in trial t, V_t denotes the set of all advice given in trial t, and $\delta \in [0, 1]$ is a weight parameter for the recency-weighted moving average.

Algorithm 10 outlines the process by which each expert $e \in E$ decides whether or not to offer advice to the agent and whether or not the advice is optimal, as discussed in Section 4 of the paper.

2 Additional Experiments

2.1 Experts with Degrading Performance

In this set of experiments, we examine how CLUE performs with experts whose reliability degrades over time. The initial values of ρ_{true} are identical to those in Section 4.1 of the main paper, but are multiplied by 0.9999 each trial to simulate slow degradation in performance. Results across $10,000$ trials are averaged over 100 random environments, where $|V_S| = 7$ and $|V_A| = 3$, and are plotted in Figure 1.

Algorithm 7 Upper Confidence Bound

1: **procedure** ACT(s_t,t,c,N)

2: **return** $\arg\max_a Q(s_t, a) + c\sqrt{\frac{2ln(t)}{N(s_t,a)}}$

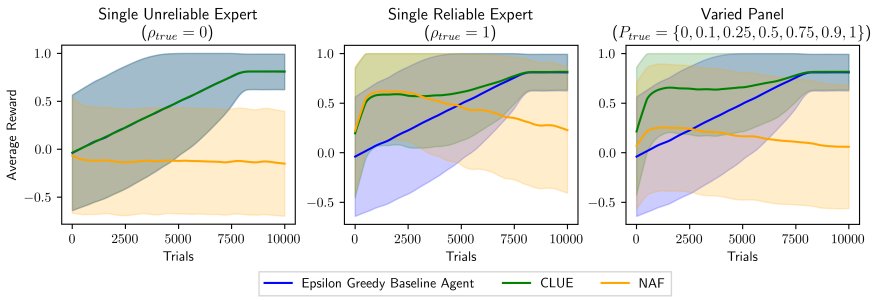

Figure 1: Panel comparisons with experts with degrading performance.

Results for $\rho_{true} = 0$ are identical to those in the *Single Unreliable Expert* case in Section 4.1 of the main paper, as there is no room for further degradation in performance. For the other panels, CLUE is able to benefit initially from the correct advice it receives, and as experts degrade learns to no longer trust them, retaining the improvement it receives from initial, correct advice while not suffering in performance due to later, incorrect advice. This demonstrates that CLUE is robust not only with consistent experts, but also with degrading experts.

2.2 Alternate Approach to Expert Simulation

In this set of experiments, we consider a different approach to simulating experts. In this approach, each expert only observes a subset of state variables in the influence diagram that represents the environment, and must advise an action based on this partial observation. Thus an expert with 0 hidden variables is reliable and an expert with $|V_S|$ hidden variables always advises the action most likely to be optimal given no observation of the state. A comparison of the performances of CLUE and NAF in an environment where $|V_S| = 7$ and $|V_A| = 3$, averaged over 10 runs, is given in Figure 2. For the sake of clarity, the shaded areas representing one

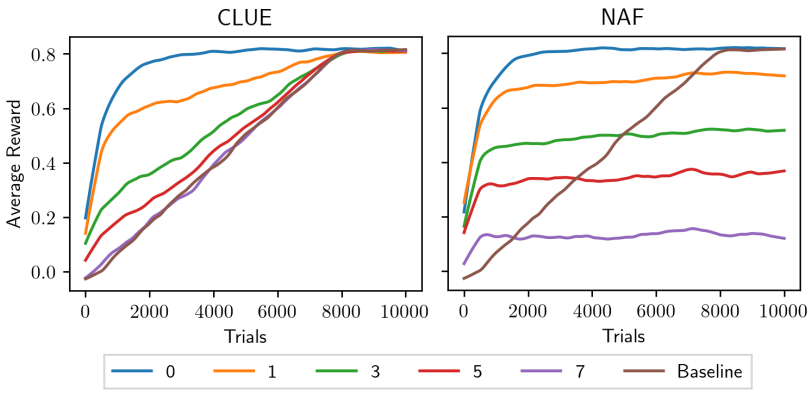

Figure 2: A comparison of agent performance advised by a single expert with varying numbers of hidden state variables. The legend denotes the number of hidden state variables. The Epsilon Greedy Baseline Agent is also given for comparison.

standard deviation are removed.

Here CLUE outperforms the Baseline Agent when the number of hidden state variables is less than $|V_S|$, as in these cases the expert is more likely to advise the optimal action than any other action, and performs on par with the Baseline Agent when the amount of information that can be gained from the expert is minimal. NAF, on the other hand, only converges to the optimal policy when the expert is reliable, performing poorly otherwise. These results further demonstrate the fact that CLUE benefits from advice from reliable experts and is robust to advice from unreliable experts.

2.3 Initial Reliability Estimates

In this set of experiments, we investigate the effect of varying the α_0 and β_0 parameters which determine the prior reliability distribution (see Section 3.1 of the main paper). Recall that α_0 and β_0 can be thought of as prior counts of the expert giving incorrect and correct advice respectively. Thus $\alpha_0 > \beta_0$ biases ρ towards 0, and $\alpha_0 < \beta_0$ towards 1, with $\alpha_0 + \beta_0$ determining the strength of that prior against trial data. To measure their effect, we plot the difference

between the average total regret of the Baseline Agent and the average total regret of CLUE (both using Epsilon Greedy unassisted action selection) for a number of different α_0 and β_0 values, summed over $10,000$ trials and averaged over 100 runs for an environment with $|V_S| = 7$ and $|V_A| = 3$, where the total regret is equal to

$$R_{Agent} = \sum_{0 \leq t < N} r(s_t, \pi^*(s_t)) - r(s_t, a_t), \tag{1}$$

and the difference in regret is

$$R_{CLUE} - R_{Baseline}. \tag{2}$$

Therefore, a value of 0 indicates performance equal to the Baseline Agent. Minimising regret is equivalent to maximising reward, and thus a lower difference in regret indicates better performance. This process was repeated for each of the panels described in Section 4.1 of the main paper, and plotted in Figure 3. The average total regret obtained by the Baseline Agent was 3505.0, and the average total regret obtained by NAF was 411.1 for the single reliable expert (difference: -3093.9), 9449.7 for the single unreliable expert (difference: 5944.7), and 4994.5 for the varied panel (difference: 1489.5).

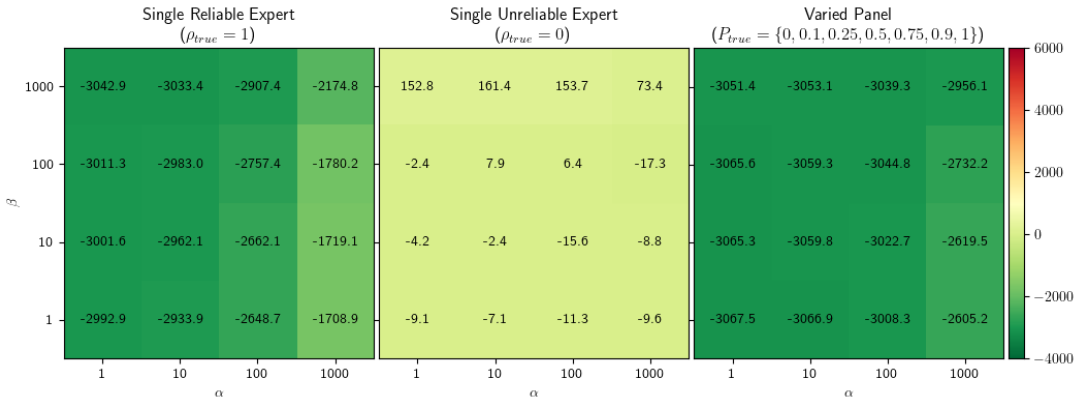

Figure 3: The total difference in regret between the Baseline Agent and CLUE for each value of α_0 and β_0 and for each panel. Lower values indicate better performance of CLUE.

For the single reliable expert, the best performance occurs when the parameters heavily bias

the estimate towards 1. However, all tested values result in improved performance over the Baseline Agent, and the gain in performance between $\alpha_0 = 1, \beta_0 = 1$ and $\alpha_0 = 1, \beta_0 = 1000$ is minimal. Results for the single unreliable expert are less varied, with performance close to the Baseline Agent for all tested values. Performance is only degraded when β_0 is large.

The best performance for the varied panel occurs when both parameters are low, as the variety in ρ_{true} means that no single strong prior can bias the reliability distributions correctly for all experts at a time. Across all panels, the performance with a relatively uninformative prior is close to, if not equal to, the best performance, making it a reasonable choice in the absence of strong belief about an expert's reliability. These results also demonstrate that CLUE is robust to the choice of prior, except where $\alpha_0 + \beta_0$ approaches the order of magnitude of the total number of trials.

2.4 Expert Parameters

In this set of experiments, we investigate the effect of varying the number of interactions between the agent and each expert. Recall from Section 4 of the main paper that the number of interactions is determined by the values of μ and γ, with lower values of both resulting in more interactions and higher values of both resulting in fewer interactions. Similar to the previous section, we plot the difference in average total regret between the Baseline Agent and CLUE, and between the Baseline Agent and NAF, for varying values of μ and γ, as depicted in Figure 4. The environment and number of trials and runs is identical to Section 2.3, and $\alpha_0 = 1$, $\beta_0 = 1$, as in Section 4.1 of the main paper.

With NAF, more advice results in better performance when the advice is always correct, but worse performance when the advice is sometimes incorrect. CLUE on the other hand is robust to the presence of incorrect advice; more correct advice results in better performance, but more incorrect advice has no adverse effect on performance.

Figure 4: The total difference in regret between the Baseline Agent and CLUE and Baseline Agent and NAF for each value of μ and γ and for each panel. Lower values indicate better performance.

2.5 Adversarial Advice

To illustrate how CLUE can benefit from adversarial advice (advice from an expert that is consistently wrong), we compare the average reward obtained by the agent advised by a single reliable expert ($\rho_{true} = 1$) and a single unreliable expert ($\rho_{true} = 0$), in an environment where $|V_S| = 10$ and $|V_A| = 1$, and thus $|A| = 2$, averaged over 100 runs. Both the Baseline and CLUE use Epsilon Greedy unassisted action selection. Results are plotted in Figure 5.

For both experts, CLUE shows a similar improvement in performance over the Baseline Agent. This improvement is possible in the case of the single unreliable expert because, having estimated the reliability of the expert to be low, the suboptimal action advised by the expert is deemed to have a low probability of being optimal, thus improving the probability of the other action being optimal.

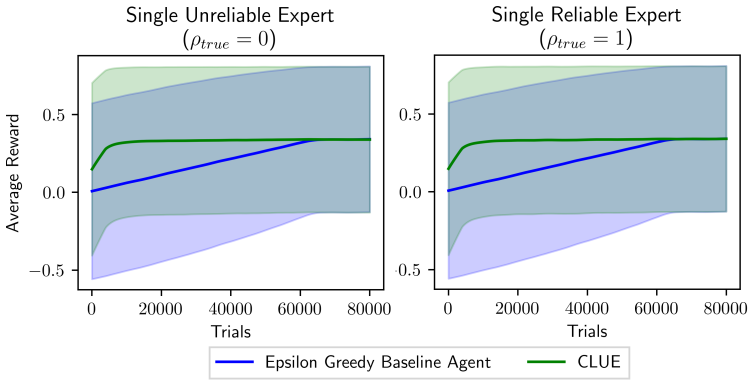

Figure 5: A comparison of the performance of CLUE with a single reliable expert ($\rho_{true} = 1$) and a single unreliable expert ($\rho_{true} = 0$) for an environment where $|A| = 2$.

3 Theoretical Analysis

In this section, we show the conditions for which CLUE, when exploring, will have a higher probability of selecting the optimal action for a given state than some default unassisted exploration strategy, when acting in an environment where $|A| = 2$ and being advised by a single expert that operates under the following assumption

Assumption 1. *Assume an expert has a true reliability $\rho_{true} \in [0, 1]$ such that, when giving advice, it advises the optimal action with probability ρ_{true} and otherwise advises some suboptimal action with probability $\frac{1-\rho_{true}}{|A|-1}$.*

To do this, we define a function $W(a)$ which represents the probability of selecting a given action when exploring. For example, if selecting actions with uniform random probability, $W(a) = \frac{1}{|A|}\ \forall a \in A$. We show the values of $\mathbb{E}[\rho]$ (the agent's estimate of the reliability, which may or may not be accurate) for which the probability of selecting the optimal action a^* is greater than or equal to $W(a^*)$, given W and ρ_{true}.

To aid in the proof of this theorem (Theorem 1), we first prove Lemma 1, which shows the conditions for which a CLUE agent is guaranteed to identify a given action as optimal.

For a uniform random exploration strategy ($W(a) = \frac{1}{|A|}\ \forall a \in A$), the implication of Theorem 1 is that there will be improved performance provided that the estimate $\mathbb{E}[\rho]$ is on the same side of $\frac{1}{2}$ as the true reliability ρ_{true}. For another exploration strategy, the improvement may increase or decrease depending on the probability of selecting a^* under that strategy.

Lemma 1. *Suppose an environment with $|A| = 2$ and a panel consisting of a single expert. Let $\mathbb{E}[\rho]$ denote the agent's estimate of the reliability of the expert. For any given state the expert has advised for, the optimal action a^* will be identified as such by the agent if one of the following holds*

- *The expert advised a^* and $\mathbb{E}[\rho] > \frac{1}{2}$*
- *The expert did not advise a^* and $\mathbb{E}[\rho] < \frac{1}{2}$*

If $\mathbb{E}[\rho] = \frac{1}{2}$, the agent is equally likely to identify

Proof. From Equation 3 of the main paper, we have that

$$P(a_j = a^*|V_t) = \frac{\prod_{e \in E_t} P(v_t^{(e)}|a_j = a^*)}{\sum_{k=0}^{|A|} \prod_{e \in E_t} P(v_t^{(e)}|a_k = a^*)}, \tag{3}$$

which, given that $|E_t| = 1$ and $|A| = 2$, reduces to

$$P(a_j = a^*|V_t) = \frac{P(v_t^{(e)}|a_j = a^*)}{P(v_t^{(e)}|a_0 = a^*) + P(v_t^{(e)}|a_1 = a^*)}. \tag{4}$$

Without loss of generality, let a_0 denote the optimal action for s_t. Substituting in Equation 4 of the main paper, Equation 4 is equal to

$$\begin{aligned} P(a_j = a^*|V_t) &= \frac{P(v_t^{(e)}|a_j = a^*)}{\mathbb{E}[\rho] + 1 - \mathbb{E}[\rho]} \\ &= P(v_t^{(e)}|a_j = a^*), \end{aligned} \tag{5}$$

which is equal to $\mathbb{E}[\rho]$ if the expert advised a_j and $1 - \mathbb{E}[\rho]$ otherwise. Let a_{best} denote the action that maximises $P(a_j = a^*|V_t)$.

We consider 2 cases.

Case 1: The expert has advised a_0. Thus,

$$P(a_0 = a^*|V_t) = \mathbb{E}[\rho]$$

$$P(a_1 = a^*|V_t) = 1 - \mathbb{E}[\rho].$$

$P(a_0 = a^*|V_t) > P(a_1 = a^*|V_t)$ is therefore only true when $\mathbb{E}[\rho] > \frac{1}{2}$, and thus the agent will identify a_0 as the optimal action if $\mathbb{E}[\rho] > \frac{1}{2}$. If $\mathbb{E}[\rho] = \frac{1}{2}$, the agent will do so with probability $\frac{1}{2}$.

Case 2: The expert has advised a_1. Thus,

$$P(a_0 = a^*|V_t) = 1 - \mathbb{E}[\rho]$$

$$P(a_1 = a^*|V_t) = \mathbb{E}[\rho].$$

$P(a_0 = a^*|V_t) > P(a_1 = a^*|V_t)$ is therefore only true when $\mathbb{E}[\rho] < \frac{1}{2}$, and thus the agent will identify a_0 as the optimal action if $\mathbb{E}[\rho] < \frac{1}{2}$. If $\mathbb{E}[\rho] = \frac{1}{2}$, the agent will do so with probability $\frac{1}{2}$. □

Theorem 1. *Suppose an environment with $|A| = 2$ and a panel consisting of a single expert. Let $W(a)$ denote the probability of selecting action a when exploring unassisted. Then the probability of a CLUE agent selecting the optimal action a^* when exploring is greater than or equal to $W(a^*)$ if one of the following holds*

- $\mathbb{E}[\rho] = \frac{1}{2}$ *and* $W(a^*) \leq \frac{1}{2}$
- $\mathbb{E}[\rho] < \frac{1}{2}$ *and* $W(a^*) \leq 1 - \rho_{true}$
- $\mathbb{E}[\rho] > \frac{1}{2}$ *and* $W(a^*) \leq \rho_{true}$

Proof. Let $P(a)$ denote the probability of selecting action a. Let $a^{(e)}$ denote the action advised by the expert. From the decision-making process described in Section 3.2 of the main paper,

we have that

$$P(a^*) = \mathbb{E}[\rho] + (1 - \mathbb{E}[\rho])W(a^*) \quad \text{if } a_{best} = a^* \tag{6}$$

$$P(a^*) = (1 - \mathbb{E}[\rho])W(a^*) \quad \text{if } a_{best} \neq a^* \tag{7}$$

We consider three cases.

Case 1: Let $\mathbb{E}[\rho] = \frac{1}{2}$. From Equations 6 and 7, we have that

$$P(a^*) = \frac{1}{2} + \frac{1}{2}W(a^*) \quad \text{if } a_{best} = a^*$$

$$P(a^*) = \frac{1}{2}W(a^*) \quad \text{if } a_{best} \neq a^*$$

From Theorem 1, $P(a_{best} = a^*) = \frac{1}{2} = P(a_{best} \neq a^*)$ and thus

$$\begin{aligned} P(a^*) &= \frac{1}{2}(\frac{1}{2} + \frac{1}{2}W(a^*)) + \frac{1}{2}(\frac{1}{2}W(a^*)) \\ &= \frac{1}{4} + \frac{1}{2}W(a^*), \end{aligned}$$

which is greater than or equal to $W(a^*)$ if and only if $W(a^*) \leq \frac{1}{2}$.

Case 2: Let $\mathbb{E}[\rho] < \frac{1}{2}$. From Equations 6 and 7, and from Theorem 1, we have that

$$P(a^*) = (1 - \mathbb{E}[\rho])W(a^*) \quad a^{(e)} = a^*$$

$$P(a^*) = \mathbb{E}[\rho] + (1 - \mathbb{E}[\rho])W(a^*) \quad a^{(e)} \neq a^*$$

From Assumption 1, $P(a^{(e)} = a^*) = \rho_{true}$ and $P(a^{(e)} \neq a^*) = 1 - \rho_{true}$. Therefore,

$$\begin{aligned} P(a^*) = P(a^{(e)} = a^*)(1 - \mathbb{E}[\rho])W(a^*) + \\ P(a^{(e)} \neq a^*)(\mathbb{E}[\rho] + (1 - \mathbb{E}[\rho])W(a^*)) \\ = \rho_{true}(1 - \mathbb{E}[\rho])W(a^*) + \\ (1 - \rho_{true})(\mathbb{E}[\rho] + (1 - \mathbb{E}[\rho])W(a^*)) \\ = \mathbb{E}[\rho](1 - \rho_{true}) + W(a^*)(1 - \mathbb{E}[\rho]), \end{aligned}$$

which is greater than or equal to $W(a^*)$ if and only if $\mathbb{E}[\rho](1-W(a^*)-\rho_{true}) \geq 0$. As $\mathbb{E}[\rho] \geq 0$, it is sufficient to prove that $1 - W(a^*) - \rho_{true} \geq 0$.

$$1 - W(a^*) - \rho_{true} \geq 0$$

$$-W(a^*) \geq \rho_{true} - 1$$

$$W(a^*) \leq 1 - \rho_{true}$$

Thus $P(a^*) \geq W(a^*)$ if and only if $W(a^*) \leq 1 - \rho_{true}$.

Case 3: Let $\mathbb{E}[\rho] > \frac{1}{2}$. From Equations 6 and 7, and from Theorem 1, we have that

$$P(a^*) = \mathbb{E}[\rho] + (1 - \mathbb{E}[\rho])W(a^*) \qquad a^{(e)} = a^*$$

$$P(a^*) = (1 - \mathbb{E}[\rho])W(a^*) \qquad a^{(e)} \neq a^*$$

From Assumption 1, $P(a^{(e)} = a^*) = \rho_{true}$ and $P(a^{(e)} \neq a^*) = 1 - \rho_{true}$. Therefore,

$$P(a^*) = P(a^{(e)} = a^*)(\mathbb{E}[\rho] + (1 - \mathbb{E}[\rho])W(a^*)) +$$

$$P(a^{(e)} \neq a^*)(1 - \mathbb{E}[\rho])W(a^*)$$

$$= \rho_{true}(\mathbb{E}[\rho] + (1 - \mathbb{E}[\rho])W(a^*)) + (1 - \rho_{true})(1 - \mathbb{E}[\rho]),$$

which is greater than or equal to $W(a^*)$ if and only if $\mathbb{E}[\rho](\rho_{true} - W(a^*)) \geq 0$. As $\mathbb{E}[\rho] \geq 0$, $P(a^*) \geq W(a^*)$ if and only if $W(a^*) \leq \rho_{true}$.

□

4 Hardware and Software Specifications

Computations were performed using High Performance Computing infrastructure provided by the Mathematical Sciences Support unit at the University of the Witwatersrand.

Experiments were run on Ubuntu 18.04 machines with Intel(R) Core(TM) i9-10940X CPU @ 3.30GHz, with 125GiB of RAM.

All code was written and executed in python 3.8.6, with the following libraries:

- *numpy* version 1.19.2
- *matplotlib* version 3.3.2
- *statsmodels* version 0.12.0
- *networkx* version 2.5

Additionally, the implementations of graphical models, influence diagrams, factors and variable elimination are adapted from Poole and Mackworth [1].

References

[1] David L Poole and Alan K Mackworth. Python code for artificial intelligence: Foundations of computational agents. *Version 0.7*, 6, 2017.

Algorithm 8 Acting with Advice from a Panel of Potentially Unreliable Experts

procedure ACT_WITH_ADVICE(s_t,t,T,$\mathbb{E}[\rho^{(e)}]$)
  $a_{base} \leftarrow$**act**(s_t,...) ▷ unassisted action (e.g. Algorithms 4 - 7)
  if exploring **then**
    $E_t \leftarrow \{e | e$ advised for s_t in $\tau \in [0, ..., t-1]\}$
    if $|E_t| = \varnothing$ **then** ▷ no advice for s_t
      return a_{base} ▷ act unassisted
    else ▷ At least one expert has offered advice
      for $a \in A$ **do**
        $L_a \leftarrow 0$
        for $e \in E_t$ **do**
          if expert e advised (s_t, a) **then**
            $L_a \leftarrow L_a \times \mathbb{E}[\rho^{(e)}]$
          else
            $L_a \leftarrow L_a \times \frac{1-\mathbb{E}[\rho^{(e)}]}{|A|-1}$
    for $a \in A$ **do**
      $P(a = a_*) \leftarrow \frac{L_a}{\sum_{i=0}^{|A|} L_{a_i}}$
    $a_{best} \leftarrow \arg\max_a P(a = a_*)$
    if $P(a_{best} = a_*) < T$ **then**
      return a_{base} ▷ act unassisted
    else
      $q \leftarrow$**random**()
      if $q < P(a_{best} = a_*)$ **then**
        return a_{best} ▷ follow advice
      else
        return a_{base} ▷ act unassisted
  else
    return a_{base} ▷ act unassisted

Algorithm 9 Updating Unreliability Estimates

procedure UPDATE_ESTIMATES(s_t,a_t,r_t,V_t,δ)
  $E'_t \leftarrow \{e | e$ advised for s_t in $\tau \in [0, ..., t]\}$
  for $e \in E$ **do**
    if e advised some action in trial t **then**
      store_advice(e,s_t,$v_t^{(e)}$)
    if $e \in E'_t$ **then**
      $best\_reward \leftarrow \max_a EU(s_t, a)$
      $advice\_reward \leftarrow EU(s_t, v_t^{(e)})$
      if $advice\_reward \geq best\_reward$ **then**
        $x \leftarrow 1$
      else
        $x \leftarrow 0$
      $\mathbb{E}[\rho_{t+1}^{(e)}] \leftarrow (1 - \delta)\mathbb{E}[\rho_t^{(e)}] + \delta x$

Algorithm 10 Expert Advice Process

procedure ADVISE(s_t, a_t, r_t, E)
  $advice \leftarrow []$
  for $e \in E$ **do**
    $a_t^* \leftarrow$**get_optimal_action**(s_t)
    $t' \leftarrow$**last_advice_trial**()
    if $t - t' \geq \mu^{(e)}$ **then**
      if $\sum_{t' < i \leq t} \frac{EU(s_t, a_i^*) - EU(s_t, a_i)}{t - t'} \geq \gamma^{(e)}$ **then**
        $p \leftarrow$**random**()
        if $p < \rho_{true}^{(e)}$ **then**
          $advice[e] \leftarrow a_t^*$
        else
          $advice[e] \leftarrow$random $a \in A \setminus \{a_t^*\}$
  return $advice$