Notation from Yamasaki and Ushio explained

November 5, 2024

Q Learning Lingo

- α is the *learning rate* or step size. According to Wikipedia, this variable determines to what extent newly acquired information overrides old information. A factor of 0 makes the agent learn nothing (exclusively exploiting prior knowledge). In contrast, a factor of 1 makes the agent consider only the most recent information (ignoring prior knowledge to explore possibilities).
- γ is the *discounted rate of reward* or discount factor. Its value determines the importance of future rewards. A factor of 0 will make the agent "myopic" (or short-sighted) by only considering current rewards, while a factor approaching 1 will make it strive for a long-term high reward.

Section 3 notation up to Eq. (13)

- SV_i - the local supervisor, where $i \in \{1, \dots, n\}$
  - I typically use the notation $\mathcal{S}_i$ for a *supervisor*, which is another name for a *controller*, occasionally represented by $\mathcal{C}_i$. In truth, a while ago, I reverted to using H for the global (control) decision function and h_i for the local (control) decision functions, where $i \in I$ and I is the index set of the local controllers (i.e., $I = \{1, \dots, n\}$).
- $G = (X, \Sigma, f, x_0)$ - the discrete-event system, where X is the finite set of states, Σ is the set of events, f is the state transition function and x_0 is the initial state
  - I typically use G for the uncontrolled system, but I use Q as the state set, call Σ the alphabet, and use $\delta : Q \times \Sigma \to Q$ as the (partial) transition function (recall that a complete function will have a transition of every element of Σ from every state in Q); then q_0 is the initial state.
- $\Sigma^c \subseteq \Sigma$ is the set of controllable events and $\Sigma^o \subseteq \Sigma$ is the set of observable events
  - It is more typical to use Σ_c and Σ_o to denote the controllable and observable event sets, respectively.
  - Similarly, to denote the controllable and observable sets of events for a specific controller/supervisor/agent $i \in I$, you'll see $\Sigma_{c,i}$ and $\Sigma_{o,i}$.
- They use ϵ to represent the empty string.
  - I prefer to use ε. It's easier to distinguish from set membership $\in$ depending on the font set used.
- $F_G(x)$ is used to denote the *active event set*. This is pretty antiquated vocabulary, although not unheard of depending on the research genealogy of the authors. This set simply denotes all the elements of Σ that are labels of outgoing transitions from x. As noted above, since δ may not be a complete function, not all elements in Σ are involved in a transition from every state. So $F_G(x)$ allows you to collect the events that "are defined" from state x. In modern terminology, we now call this the *feasible event set* and denote it by Γ in the centralized case, but by Γ_i for an observer $\mathcal{O}_i$, for $i \in I$.

- $M_i^e : \Sigma \to (\Sigma_i^o \cup \{\epsilon\})$ is the *natural projection* for supervisor/controller/agent i. In effect, this operator simply sets an event $\sigma \in \Sigma$ to ϵ if $\sigma \notin \Sigma_i^o$. The operator is extended to work on Σ^* and Σ_i^o. Think of it as an eraser that removes all events that an agent cannot observe.
  - I almost exclusively use π to represent natural projection and, thus, π_i for the projection for supervisor/controller/agent $i \in I$
- $\text{SV}_i = (S_i, \Sigma_i^o, g_i, x_0)$ is used to denote a local controller (supervisor, agent). [I think there are typos here as it seems silly to use S_i to represent the state set and then x_0 as the initial state.] Usually, this is what we're trying to produce as output. But for the reinforcement learning application, it seems as if they start with each controller's local view of G and then use the rewards/penalties to converge to a better solution.
  - The more standard notation is to invoke the observer $\mathcal{O}$, which follows the construction of the subset construction algorithm. I typically use $\mathcal{O}_i = (X_i, \Sigma_{o,i}, \delta_i, x_{0,i})$, where $X_i \subseteq 2^Q$ (the power set of states in Q).
- The paper talks about a *control pattern*, which simply refers to a function that defines, for each controller, the events that are enabled at each state. Note that by definition, any event in $\Sigma \setminus \Sigma_c$ is called *uncontrollable* and is permanently enabled (i.e., cannot be prevented from happening). In this paper, the local control patterns are combined via intersection to determine the overall (aka *global*) control pattern at that state.

Notation from Eq. (14) and onwards in Section 3

- $\mathcal{P}_i^1(s_i, \pi_i, \sigma)$ is the probability that supervisor/agent i *observes* $\sigma \in \Sigma_{o,i} \cap \pi_i$ when it chooses the control pattern π_i at state s_i. It is computed in Eq. (16)
- $\mathcal{P}_i^2(s_i, \sigma, s_i')$ is the probability that supervisor/agent i *takes* a transition from state s_i to state s_i' via event $\sigma \in \Sigma_{o,i}$.
- $\mathcal{P}_i(s_i, \pi_i, s_i')$ is then the combination of these two probabilities, defined by Eq. (15), summed over agent i's observable events in $\pi_i \cap \Sigma_{o,i}$

Section 4 Algorithm notation

- $T_i(s_i, \sigma)$ is a learning parameter, which seems to be tied to T_i^* in Eq. (20), where it is referred to as a discounted expected total reward. The parameter itself is initialized according to Eq. (21). The authors note that this is the reward when agent i observes an event $\sigma \in \Sigma_{o,i}$ at state s_i
- R_i^1 is another learning parameter, which is introduced as part of Eq. (18). It is defined as the *expectation of a reward* when supervisor i selects control pattern π_i. The authors say that this corresponds to the cost of disabling controllable events that are not included in the control pattern (since it represents events that are enabled at a given state). This learning parameter uses β.
- η_i is the final learning parameter. After reading the authors' previous paper on a centralized controller and reinforcement learning, it seems that $\eta_i(s, \sigma)$ is simply a number between 0 and 1 that is assigned to each observable event $\sigma \in \Sigma_{o,i}$ such that for all feasible and observable events at state $s_i \in S_i$,

$$\sum_{\sigma \in F(s) \cap \Sigma_{o,i}} \eta_i(s_i, \sigma) = 1.$$

  This is "computed" on the automaton that defines SV_i. For example, in SV_1 in state 14, there are two feasible and observable events: $m2$ and $c2$. So one possible assignment could be $\eta_1(14, m2) = 0.5$ and

$\eta_1(14, c2) = 0.5$ because their sum is 1.0. Similarly, you could (randomly) define $\eta_1(14, m2) = 0.65$ and $\eta_1(14, c2) = 0.35$. And none of these individual values can be 0, since they are part of the feasible alphabet at state 14. The values assigned to the η_i parameter are used as a starting point to determine the probabilities of an event occurring, which is defined by Eq. (16) over the control patterns, not the feasible events. This learning parameter uses δ.

Values used in the Cat and Mouse Example

- All Q values were initialized to 0.
- $\alpha = 0.1$ (According to Wikipedia, this is a standard value)
- $\beta = 0.1$
- $\delta = 0.1$
- $\gamma = 0.9$

The original algorithm

Algorithm 1 Original decentralized reinforcement learning algorithm

Initialize T_i, R_i^1, η_i, and Q_i for all SV_i, where $i \in \{1, \dots, n\}$
▷ *The true system state s is observed a state estimate s_i for each of the $i\{1, \dots, n\}$* ◁
for each episode **do**
  $s \leftarrow x_0$
  for each SV_i **do**
    ▷ *We want to initialize each supervisor's initial state to be the state estimate of x_0, the initial state of G. We have no notation for this right now.* ◁
    Initialize a state $s_i \leftarrow x_0$
    $atMarkedState \leftarrow$ `False`
    Select a control pattern $\pi_i(s_i) \in \Pi_i(s_i)$ based on the Q_i values by SV_i.
  Apply global control pattern $\pi(s) \leftarrow \cap_{i \in I} \pi_i(s_i)$ to the system (aka DES) G
  Choose an event $\sigma \in \Gamma(s) \cap \pi(s)$ to occur in G (Guided by the use of Eq. (9))
  ▷ *Update the current state of G* ◁
  $s \leftarrow \delta(s, \sigma)$
  for all SV_i such that $\sigma \in \Sigma_{o,i}$ and not $atMarkedState$ **do**
    Observe the occurrence of event $\sigma \in \Sigma_{o,i}$
    Acquire rewards r_i^1 and r_i^2.
    In SV_i, make a transition $g_i(s_i, \sigma) = s_i'$.
    Update $T_i(s_i, \sigma)$, $R_i^1(s_i, \pi_i)$ and $\eta_i(s_i, \sigma)$ using Eqs. X, Y and Z, respectively.
    Update Q_i values by Eq. XX.
    $s_i \leftarrow s_i'$
    if s_i is a marked state **then**
      $atMarkedState \leftarrow$ `True`

Algorithm 2 Decentralized Reinforcement Learning algorithm

procedure A METHOD(G,$\mathcal{O}_1, \ldots, \mathcal{O}_n$) ▷ $\mathcal{O}_i = (X_i, \Sigma_{o,i}, \delta_i, x_{i,0})$

▷ *The set of control patterns (typically called a control action) at a state $x_i \in X_i$ is the product of (i) $\Sigma_{uc,i}$ intersected with the feasible events at x_i, that is $\Gamma_i(x_i)$ and (ii) the elements of the power set of $\Sigma_{c,i}$ intersected with $\Gamma_i(x_i)$.*

▷ *$\mathsf{h}_i(x_i)$ is the local control action (aka the control pattern) for $\mathcal{O}_i$ at state x_i. What is it initialized to? All events in $\Gamma_i(x_i) \cup (\Sigma_{uc,i} \cap \Gamma_i(x_i)$?*

    for $i \in I$ **do**
        for $x_i \in X_i$ **do**
            ▷ $T_i(x_i, \sigma_i), \eta_i(x_i, \sigma_i)$ ◁
            $T_i \leftarrow 0$
            $\eta_i \leftarrow$
            ▷ $R_i^1(x_i, \mathsf{h}_i(x_i))$ ◁
            $R_i^1 \leftarrow 0$
            $Q_i(x_i, \mathsf{h}_i(x_i)) \leftarrow 0$
    ▷ *End of initialization (Lines 1 and 2 of the original paper* ◁
    ▷ *Lines below form the basis of an "episode" or epoch* ◁
    $atMarkedState \leftarrow$ `False`
    while not $atMarkedState$ **do**
        $q \leftarrow q_0$ ▷ *initialize current state of Q*
        for $i \in I$ **do**
            $x_i \leftarrow x_{i,0}$ ▷ *initialize current state of each $\mathcal{O}_i$*
            Choose the "best" $\mathsf{h}_i(x_i)$ according to current Q_i values for $\mathcal{O}_i$
        Apply $\mathsf{H}(q) \leftarrow \cap_{i \in I} \mathsf{h}_i(x_i)$ to G
        Choose $\sigma \in (\Gamma(q) \cap \mathsf{H}(q))$ guided by Eq. (9)
        $q \leftarrow \delta(q, \sigma)$ ▷ *Update current state of G*
        for $i \in I$ such that $\sigma \in \Sigma_{o,i}$ **do**
            Observe $\sigma_i \in \Gamma_i(x_i)$
            Acquire rewards r_i^1 and r_i^2
            Update $T_i(x_i, \sigma_i)$
            Update $\eta_i(x_i, \sigma_i)$
            Update $R_i^1(x_i, \mathsf{h}_i(x_i))$
            Update Q_i according to Equation 24
            $x_i \leftarrow \delta_i(x_i, \sigma_i)$
            if $x_i \in Marked(X_i)$ **then**
                $atMarkedState \leftarrow$ `True`