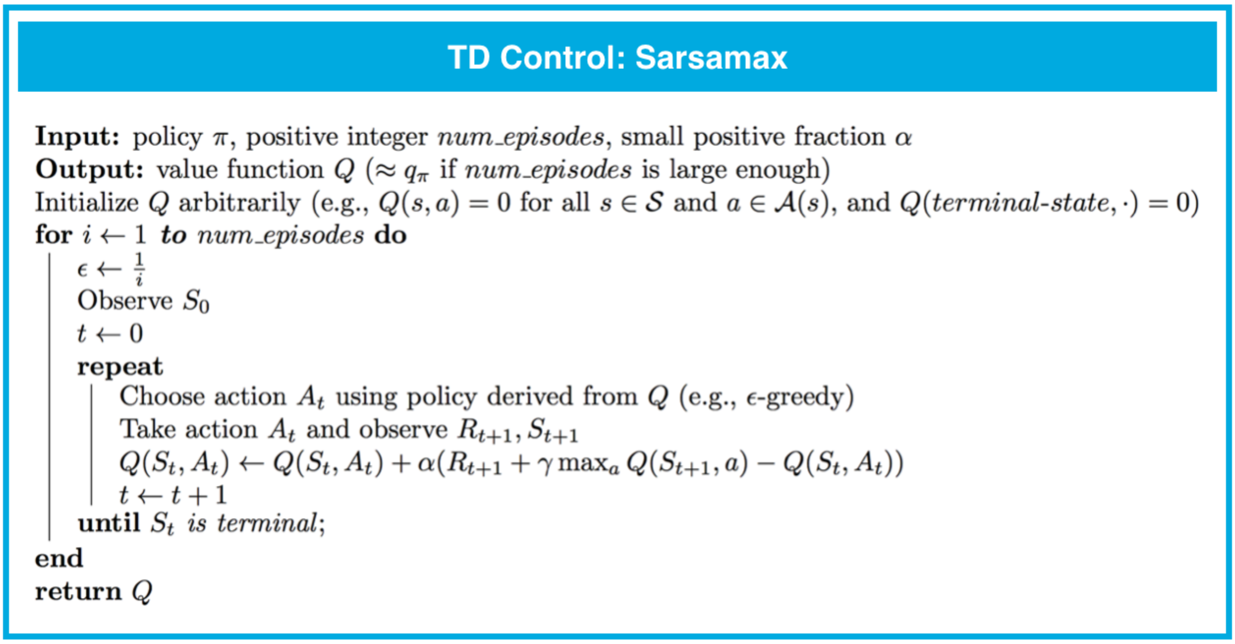
Implementation: Sarsamax

The pseudocode for Sarsamax (or *Q*-learning) can be found below.



TD Control: Sarsamax

```
Input: policy $\pi$, positive integer $num\_episodes$, small positive fraction $\alpha$
Output: value function $Q$ ($\approx q_\pi$ if $num\_episodes$ is large enough)
Initialize $Q$ arbitrarily (e.g., $Q(s,a) = 0$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$, and $Q(terminal\text{-}state, \cdot) = 0$)
for $i \leftarrow 1$ to $num\_episodes$ do
    $\epsilon \leftarrow \frac{1}{i}$
    Observe $S_0$
    $t \leftarrow 0$
    repeat
        Choose action $A_t$ using policy derived from $Q$ (e.g., $\epsilon$-greedy)
        Take action $A_t$ and observe $R_{t+1}, S_{t+1}$
        $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$
        $t \leftarrow t + 1$
    until $S_t$ is terminal;
end
return $Q$
```

Sarsamax is **guaranteed to converge** under the same conditions that guarantee convergence of Sarsa.

Please use the next concept to complete **Part 3: TD Control: Q-learning** of `Temporal_Difference.ipynb`. Remember to save your work!

If you'd like to reference the pseudocode while working on the notebook, you are encouraged to open this sheet in a new window.

Feel free to check your solution by looking at the corresponding section in `Temporal_Difference_Solution.ipynb`.

NEXT