Reinforcement Learning & PyCIGAR Architecture

- **Reinforcement Learning:**
  - Branch of Artificial Intelligence which has demonstrated the ability to find optimal control policies in systems with complex dynamical interactions via intelligent simulation (i.e., “smart” trial and error)

- **PyCIGAR:**
  - Python-based simulation framework used for training and evaluation of RL agents
  - Brings together:
    - Dynamic models of smart inverter functions (based on IEEE 1547)
    - Distribution system power flow tools (OpenDSS & custom solvers)
    - Open source reinforcement learning library (RLLib: [https://docs.ray.io/en/master/rllib.html](https://docs.ray.io/en/master/rllib.html))
Markov Decision Processes (MDPs)

- **MDPs** provides a mathematical framework for modeling decision making in situations where outcomes are partly random and partly under the control of a decision maker.
- Consists of the 4-tuple: \((S, A, P_a, R_a)\)
- \(S\) is the set of possible states (i.e., the *state space*)
- \(A\) is the set of admissible actions
- \(P_a\) is the probability that action \(a\) in state \(s\) at time \(t\) will lead to \(s'\) at \(t+1\), or
  \[ P_a(s, s') = \Pr(s_{t+1} = s' | s_t = s, a_t = a) \]
- \(R_a(s, s')\) is the reward received after applying \(a\) at state \(s\) to transition to \(s'\)
- Goal of the optimization is to maximize the cumulative reward over a time horizon
- Can also be solved by dynamic programming or Monte Carlo techniques
• At time $t$ the agent selects action $a_t$ from a policy $\pi$ and applies that action to the environment
• Action causes a transition to a new state $s_t$ which has an associated reward $R_t$
• $(s_t, a_t, R_t)$ are used to update the policy $\pi$
Reinforcement Learning Agent Training

Approximated by neural networks

Mapping from $s_t$ to $\text{sum}(R_t)$

Approximated by neural networks
Reinforcement Learning Training Loop

Agent
(Proximal Policy Optimization - PPO)

Environment
OpenDSS
- PV with smart inverter functions (Volt-VAR/Volt-Watt) - CIGAR
- Battery Storage - SPADES
- EV/EV charging - TBD

states & rewards

actions

e.g. PV and battery setpoints

Voltage and power flow timeseries & attack mitigated?
Examples of Reinforcement Learning

Machine Learning Improves Google Data Center Cooling Efficiency

Reinforcement Learning Beats “Go” Expert

These algorithms enable decision-making in high dimensional uncertain problems
PyCIGAR - Smart Inverter Models

- PyCIGAR smart inverter module replicates the dynamic behavior of the Volt-VAR and Volt-Watt control
- $H_M(z), H_O(z)$ are low pass filters
- $f^q(\hat{v})$: inverter Volt-VAR curve
- $f^p(\hat{v})$: inverter Volt-Watt curve
- Volt-Watt precedence: available reactive power determined by excess inverter capacity
• RLlib is a scalable open-source library for reinforcement learning
• Provides application support
  – Multi-agent/hierarchical
• Abstractions of multiple RL algorithms:
  – PPO, DDPG, A3C, etc.
• Supports distributed training
• Highly customizable
PyCIGAR Environment

*RB Controller: Rule-based controller
*RL Controller: Reinforcement learning controller