Networks & Asynchronous Information

UF UNIVERSITY of FLORIDA

Duke UNIVERSITY

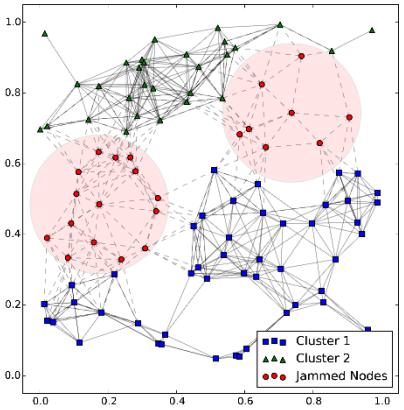
TEXAS The University of Texas at Austin

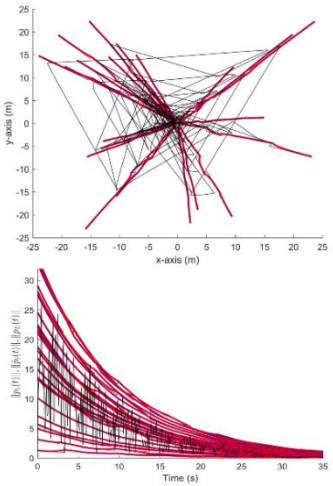
UC SANTA CRUZ

Network Systems

- Design and analysis challenges for both controlling agents within a network (stochastic time-varying and random graph models) and controlling agents over a network
- Determining conditions under which random communication graphs attain required connectivity properties and positioning agents to achieve network objectives (e.g., jamming adversarial networks)
- Develop models where the control system can adapt in real time as service degrades
- Develop control techniques that allow a system to adapt its operation and use of network resources based on QoS that the network is able to provide

RT3 will develop analysis, design, and synthesis methods for agents *within* a network and *over* a network to generalize existing graph theory-based methods and improve the interface and adaptability between controls and communications

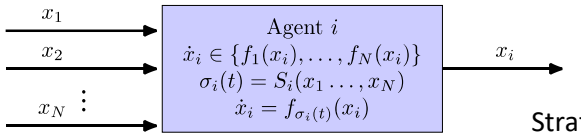




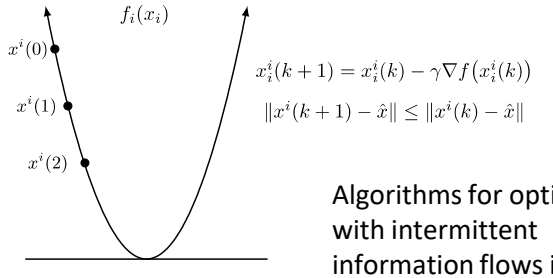
Asynchronous Information

- Investigate stability analysis methods for hybrid dynamic systems incorporating delays due to asynchronous communications and computations
- Explore the utilization of delay bounds to design asynchrony-tolerant algorithms for distributed data processing
- Explore methods to automatically delegate computations based on a problem's structure
- Refine switching algorithms for online optimization and computation to render desired sets invariant
- Experimentally validate these approaches to networked hybrid systems through implementation on mobile autonomous agents

RT4 will develop tools for the design of networked hybrid systems resilient to delayed and asynchronous information, producing novel generalizations of the hybrid analysis methods and new insights into shared computations even beyond cloud-based collaboration



Strategies to use exogenous information to make an endogenous switch



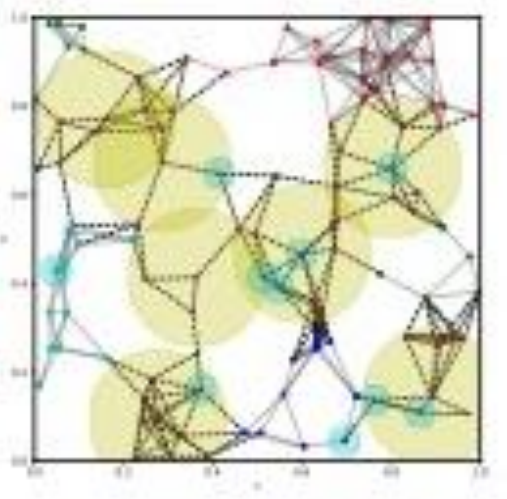
Algorithms for optimizing with intermittent information flows in networks

UF UNIVERSITY of FLORIDA · Duke UNIVERSITY · TEXAS The University of Texas at Austin · UC SANTA CRUZ

Major initiatives this year:

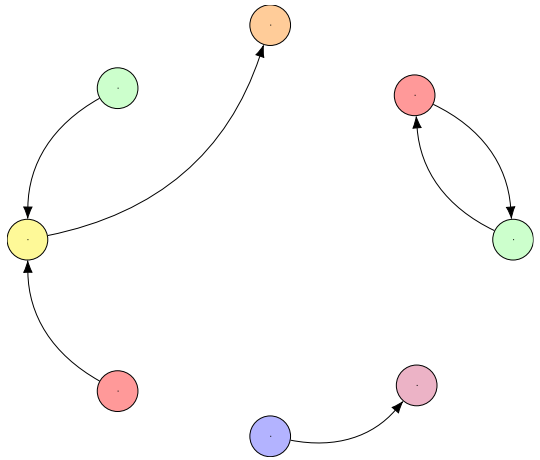
- Developing techniques to optimize dynamic spectrum usage in mixed cooperative/ competitive scenarios

- Developing techniques to infer, attack, and protect vulnerable portions of networks and networked agents

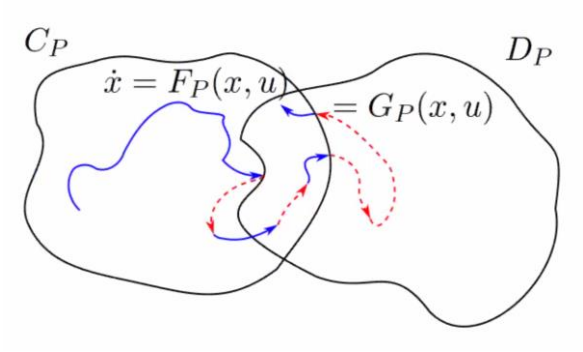


Major initiatives this year:

- Developing techniques to quantify the flow of information in systems of distributed agents, allowing us to assess robustness or vulnerabilities to attacks

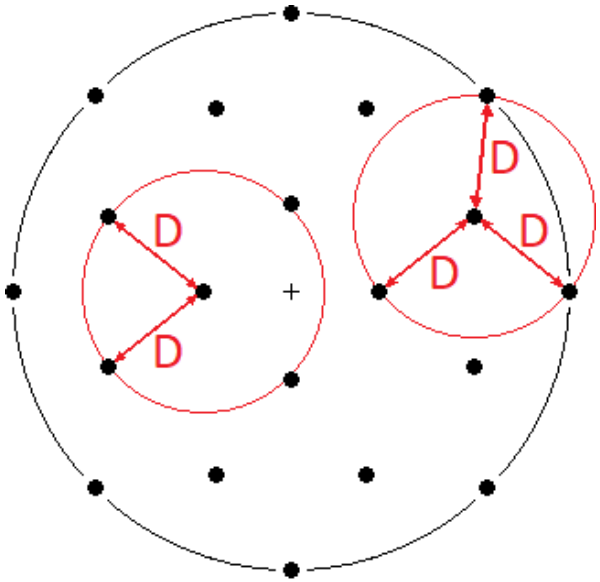


- Developing hybrid continuous/discrete models for control fusing diverse physics/communication events

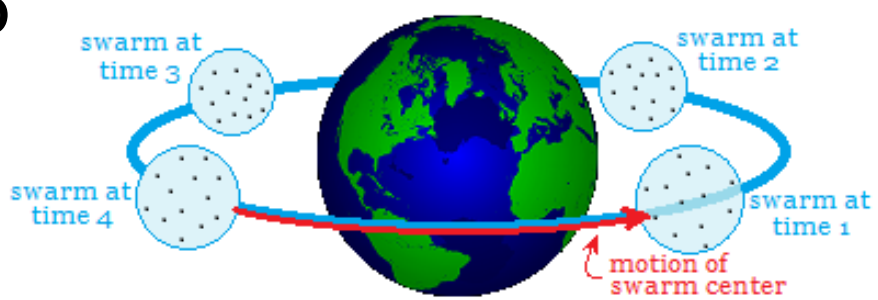


Major initiatives this year:

- Developing distributed space architectures that can provide assured operation of high-value assets in contested environments



- Developing Swarm Shield: A system of networked space assets to obfuscate or disaggregate high-valued assets for mission assurance



UF UNIVERSITY of FLORIDA · Duke UNIVERSITY · TEXAS The University of Texas at Austin · UC SANTA CRUZ

DARPA SC2 Challenge

Spectrum Collaboration and Competition

UF UNIVERSITY of FLORIDA

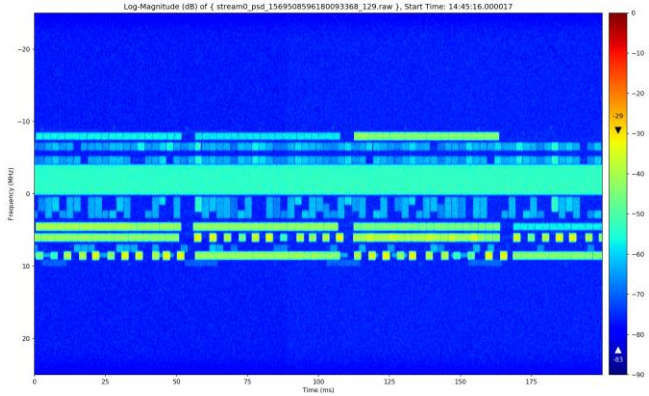
Duke UNIVERSITY

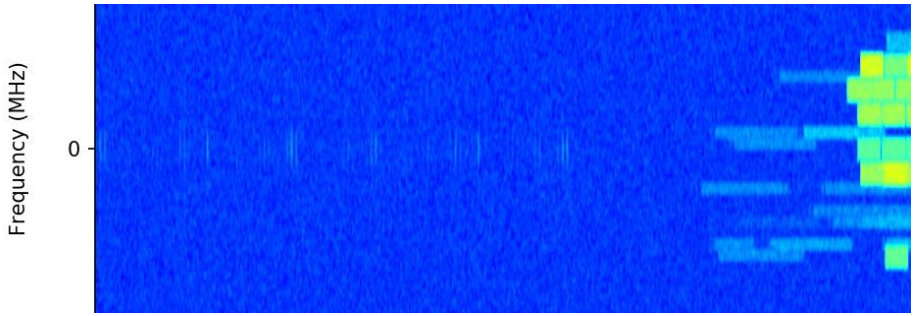
TEXAS The University of Texas at Austin

UC SANTA CRUZ

SC2 Problem

- Dynamic spectrum access environment with 5 teams/networks of radios communicating in same frequency band
- Each team scores points by delivering traffic flows achieving certain QoS mandates (throughput, latency, hold time, etc.)
- Other impairments: jammers, active and passive incumbents that must be protected





UF UNIVERSITY of FLORIDA · Duke UNIVERSITY · TEXAS The University of Texas at Austin · UC SANTA CRUZ

SC2 Problem

- **A mixed cooperative/competitive game**
- Team's match score $= \begin{cases} \text{min score} & \text{if min score} \leq \text{scoring threshold} \\ \text{achieved score} & \text{if min score} > \text{scoring threshold} \end{cases}$

where min score = minimum among all 5 team score

SC2 Problem

- Adapt strategy in presence of **rich but incomplete information**:
  - No online scoring information, other than teams' estimates
  - Teams use CIL to report frequency use, radio locations, performance (score) estimates
    - Some CIL veracity checks on spectral use, scores
    - Teams do not have to report their true scores when their scores are above the threshold
  - Incumbents report channel usage, interference received and threshold, threshold violations
  - Spectrum sensing to estimate peer channel usage and detect jamming and active incumbents

Everything is adaptive

- PHY: Acquisition, Modulation, Coding, TX Power, RX Gain
- LL: **Channels and Time Slots/Channel**, Mapping of SRCs to Time Slots
- NET: Supported flows, admission control granularity down to individual files/bursts
- Other: Channels to jam

Spectrum Access Decisions

- Decision engine determines **which flows are transmitted** and **in which time/frequency slot (pocket) they will be transmitted** with goal of maximizing our team's match score
- Spectrum access action = **Pocket Schedule**
- Action space is huge!
  - 40 channels x 10 time slots = 400 pockets
  - As many as 100+ flows
  - 100^{400} possible pocket schedules!

Inputs to Decision Engine

- Set of specified QoS mandates for our team's flows
- Estimated number of achieved mandates and total mandates for our network
- Information on throughput per pocket expected between each SRC-DST pair
- Peer networks' IDs (identified based on CIL message characteristics)
- Channels used by our network and by peer networks
- Estimated channel occupancies from our spectrum sensor (PSD measurements)
- Computed SINRs from our interference map (GPS and voxel info from CIL messages)
- Estimated achieved and total mandates from competitor networks (Performance info from CIL messages)

Decision Engine Design

- No “magic” machine-learning black box that can solve spectrum access problem
- Apply age-old engineering approach of “divide and conquer”: Break problem down into smaller pieces:

1. Channel selection
  - Determines target set of channels *C* to be used by our network
  - ML and expert system approaches

2. Admission control
  - |*C*| determines number of pockets available
  - Estimates number of pockets needed to support each flow
  - Iterative process to determine set of flows to admit in order to maximize points scored

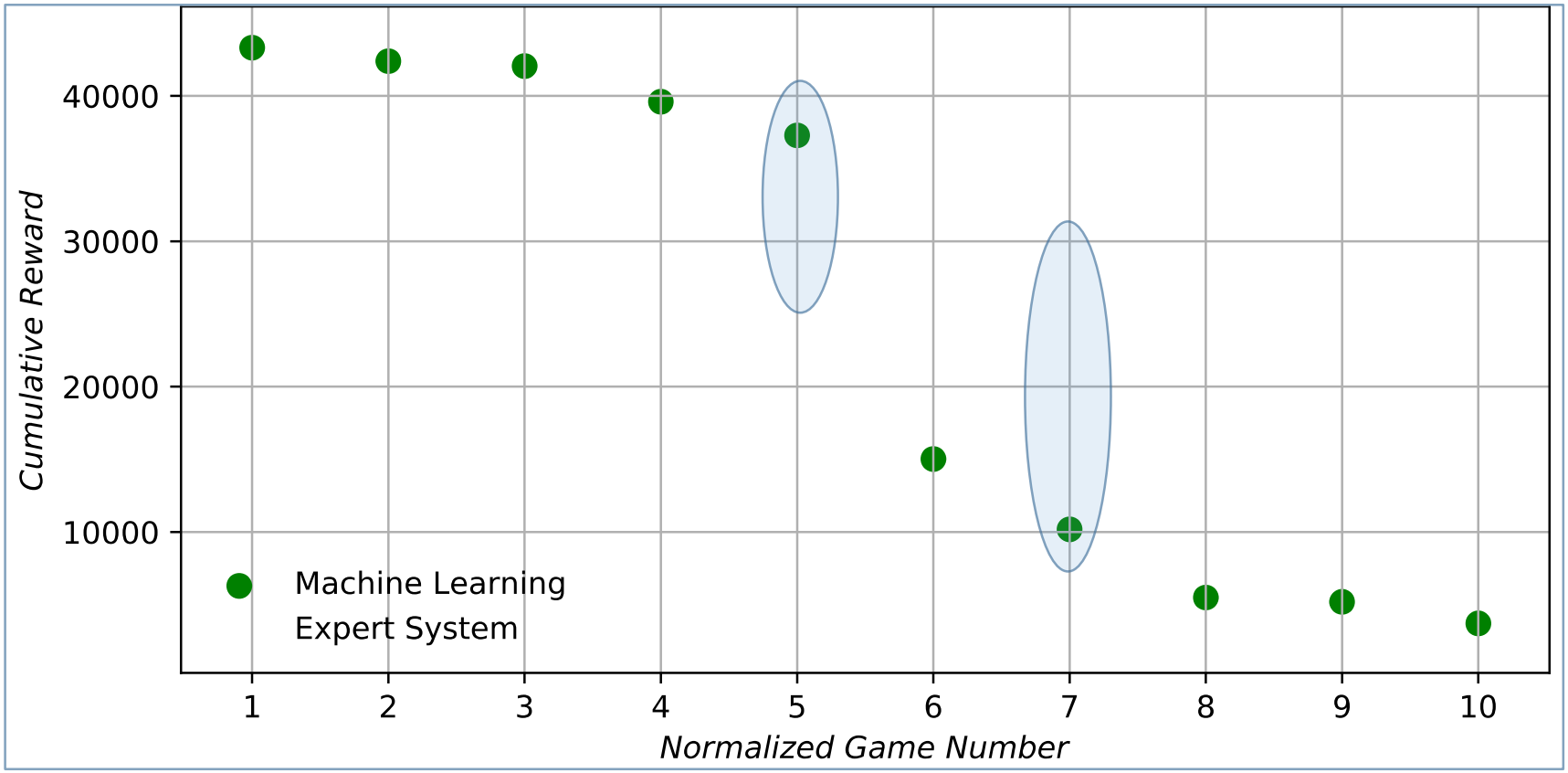
3. Pocket schedule assignment
  - Linear program to allocate number of pockets to satisfy latency requirements of all admitted flows
  - Greedy algorithm to assign pockets in each frame to satisfy mandates of all admitted flows
  - Maps to channels in *C* based on worst-case SINR over links of SRC-DST pairs in above assignments

Channel Selection

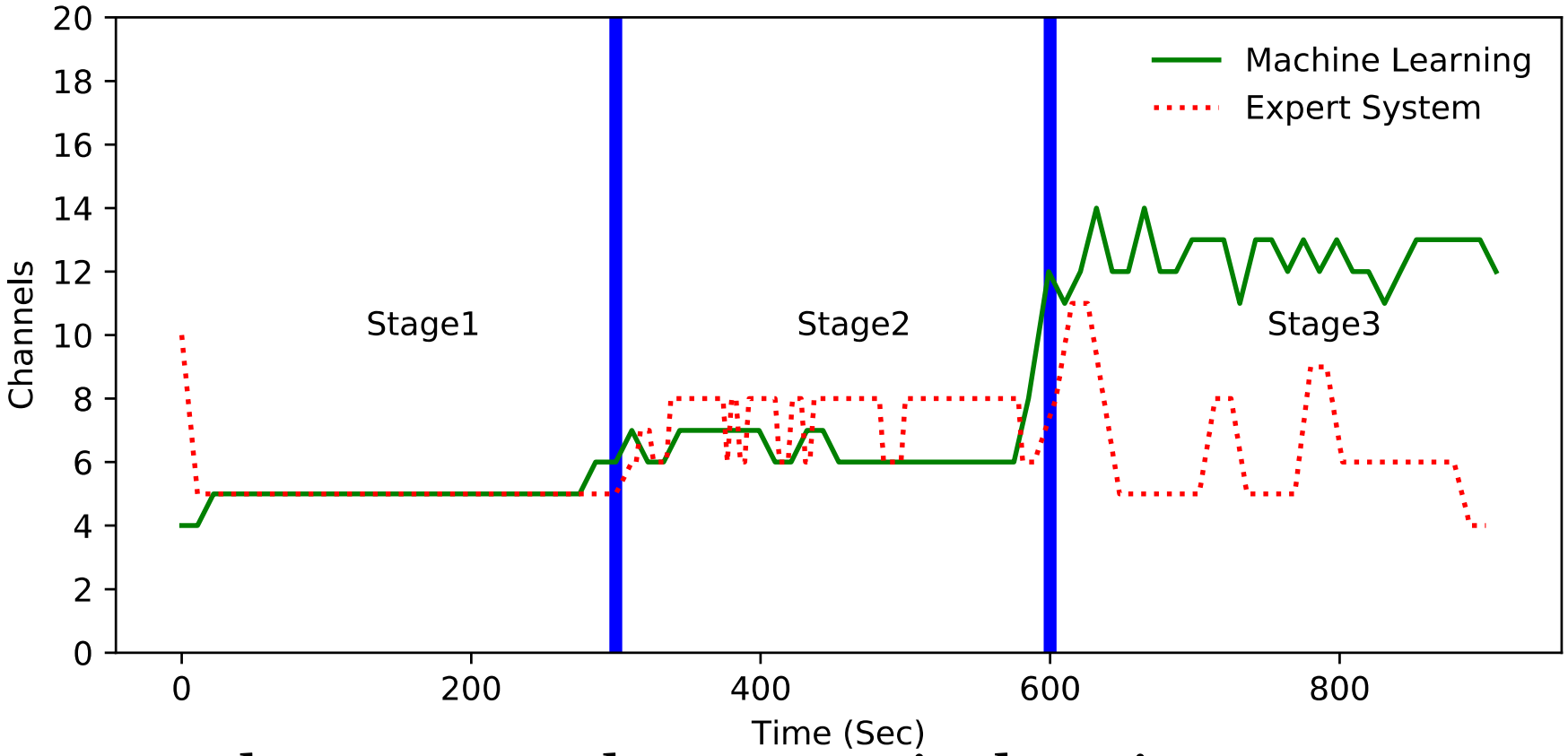
- Compare performance of two approaches to choosing number of channels to use:
  - Switched System/Controls/Expert System: continually adapt number of channels with different strategies in different operating regimes
  - Reinforcement Learning: Train different agents to select number of channels to use for each stage of each scenario and each individual team

Pre-competition Comparisons

- Compared in 3-team Alleys of Austin mobile networking scenario:

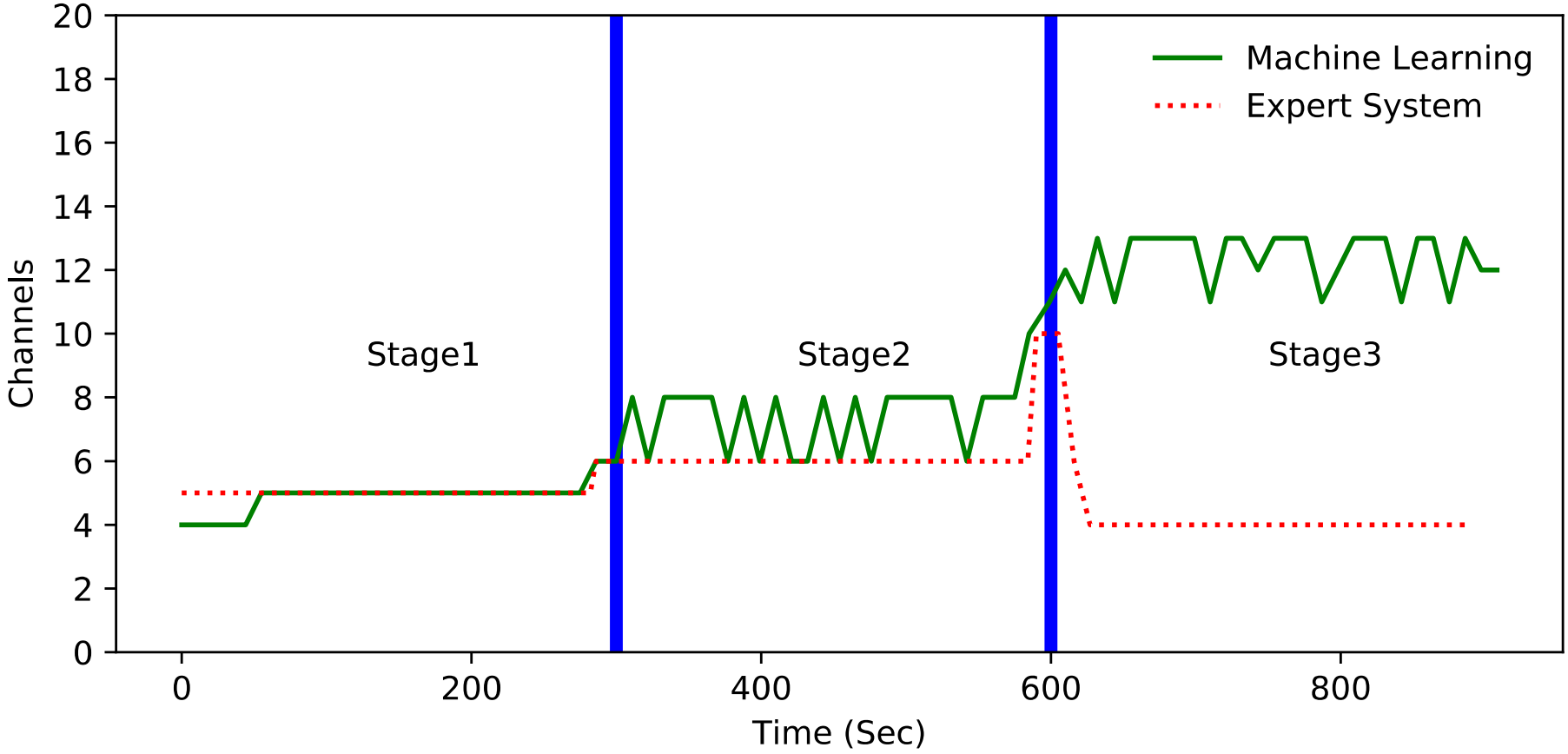


Match 5: RL > ES



- ML gets better score by aggressively using more channels in stage 3 when playing with teams D & E

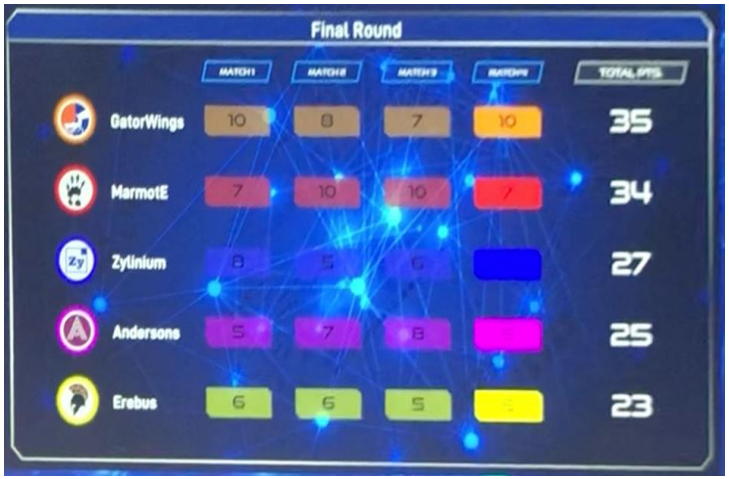
Match 7: ES > RL



- ES less aggressive in using channels in stage 3 leads to better score when playing with teams D & G

Competition Results

Final Results



UF UNIVERSITY of FLORIDA

Duke UNIVERSITY

TEXAS The University of Texas at Austin

UC SANTA CRUZ

Discussion

- No “magic” machine-learning black box that can solve spectrum access problem
- Developed algorithms to achieve robust communication and spectrum dominance in highly contested environments

- RL does show potential, but ES was safer and proved successful
  - Matches too short for online version of SARSA
  - Not enough training and validation data for ML
  - Need a less resource-intensive simulation environment to train ML algorithms