Evaluating Reinforcement Learning Methods for Bundle Routing Control

Gandhimathi Velusamy and Ricardo Lent

University of Houston

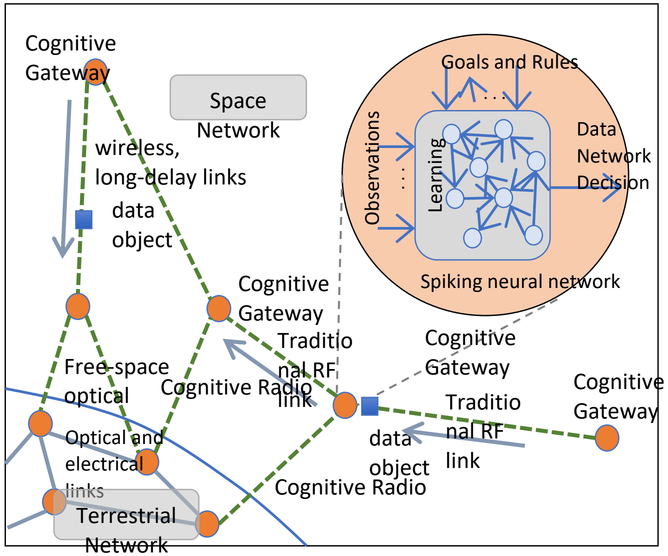
IEEE CCAA Workshop, June 26–27, 2019 –Cleveland, OH

Introduction

- Problem context:
  - Space networks: long propagation delays, frequent network partitions, large loss rates, limited capacity
  - Increasing demand for bandwidth, performance expectations, less operational burden
  - Routing: key performance driver that becomes harder to control centrally as the complexity of space networks increases
  - How to achieve optimal routing decisions onboard
- Reinforcement learning:
  - No pre-training, no need of policies
  - Learning from experience, exploration, exploitation
  - Context awareness and adaptation
- Neuromorphic computing
  - Low-power, parallel computing paradigm
  - Cognitive network controller for space gateways
  - Performance vs. regular RL techniques?

Space Network Routing

- Space gateways that search online the optimal routing point
  - As a continuous and distributed process
  - Optimal routing is a moving target
  - Different metrics (e.g., latency) are of interest (rewards)
- Evaluate the (experimental) performance of different policy and value RL iteration methods:
  - Need of a reference routing application
  - Identical assumptions (e.g., type of rewards) and testing conditions



RL-Based Routing Methods

<u>Q-Routing</u>

- Value iteration method
- Maintains Q-values for each possible next-hop or next-link decision (i.e., action)
- Reward is the inverse of cost expressed as the bundle delivery latency
- Cost r_n

$$Q_n(x,a) = \begin{cases} (1-\alpha)Q_{n-1}(x,a) + \alpha(r_n + \gamma \min_b Q(y_n, b)), \\ \qquad \text{where,} \quad x = x_n,\ a = a_n. \\ Q_{n-1}(x,a), \quad \text{otherwise.} \end{cases}$$

- Routing decision is Ɛ-greedy and seeks to minimize the average delivery latency

RL-Based Routing Methods

Double Q-Learning

- Uses two Q-Functions Q^A and Q^B (two vectors of Q values)
- The action selection is Ɛ-greedy using both functions
- Randomly updates one of the two vectors using reward R_a

if A is selected:

$$a^* = \text{argmin}_a Q^A(x, a)$$

$$Q^A(x, a) = (1 - \alpha)Q^A(x, a) + \alpha(R_a + \gamma Q^B(x, a^*))$$

else:

$$b^* = \text{argmin}_a Q^B(x, a)$$

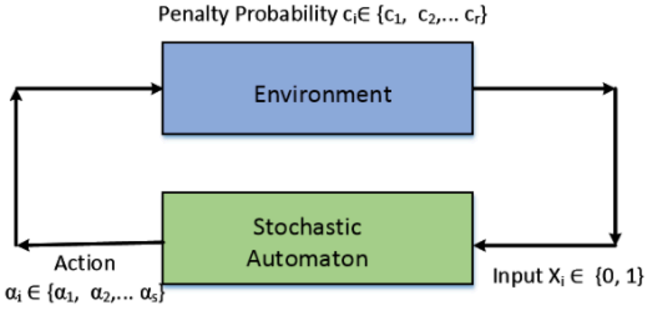
$$Q^B(x, a) = (1 - \alpha)Q^B(x, a) + \alpha(R_a + \gamma Q^A(x, b^*))$$

RL-Based Routing Methods

<u>Actor-Critic Reinforcement Learning (Learning Automata)</u>

- Routing decisions are random with distribution $P = [p_j]$, j=1..N
- Uses the normalized reward (or costs) of R_l

$$\beta = \frac{R_l - x_1}{x_2 - x_1}$$



- x1, x2 are the min/max Q-values (for costs)
- The probability distribution of action *l* is updated with:

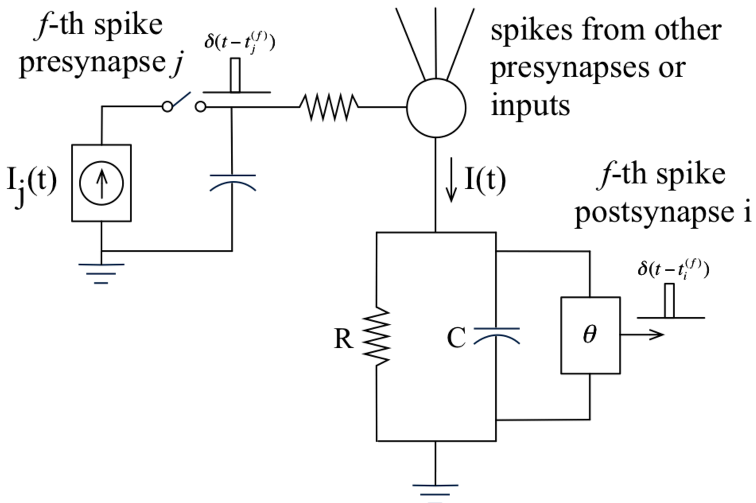
$$p_l \leftarrow p_l + a(1-\beta)(1-p_l) - b\beta p_l$$

$$p_j \leftarrow p_j - (1-\beta)ap_j + b\beta\left(\frac{1}{r-1} - p_j\right) \quad ; \forall j \neq l.$$

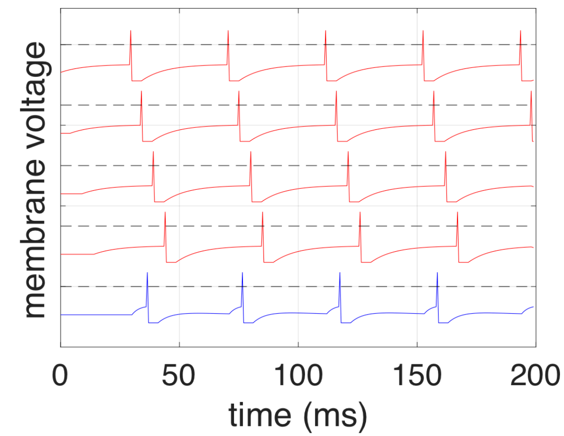
Neuromorphic Computing

- Computation with biologically realistic neuron models
- Commonly known as 3rd generation neural networks
- Neurons have an analog-digital behavior:
  - May accept external stimuli and inputs from other neurons
  - Once their membrane potential reaches a certain level, they emit a spike
  - Spike travels to other neurons through synapses
  - Effect on the post-synapse depends on the type of pre-synapse (excitatory or inhibitory) and the synapse strength (weight)
- Software vs. hardware implementations

Leaky-Integrate-and-Fire Neuron Model



$$\tau \frac{d}{dt} u(t) = -u(t) + RI(t)$$



R: Resistance
C: Capacitance
τ = RC: Time constant
I(t): Input current
u(t): Neural membrane potential

$$I(t) = i_e(t) + \sum_f \sum_j \sum_k w_{jk} \, i_{jk}^{(f)}(t)$$

External stimulus

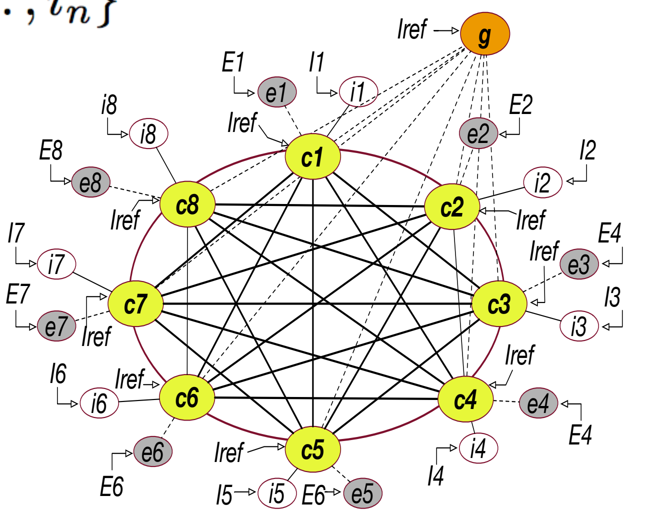
Presynapse *j* is characterized by a weight w_{jk} >= 0
<u>excitatory or inhibitory</u>

Spikes from the *k*-th presynapse from *j* arrive as unit-impulses at firing times *f*:

Cognitive Network Controller

- It consists of the recurrent SNN: $G = (V, W)$
- SNN encodes Q-value related information
- Vertices: $V = \{c_1, \ldots, c_n, g, e_1, \ldots, e_n, i_1, \ldots, i_n\}$

c_i	core neuron, one per **action** available
g	single (global) inhibitory neuron
e_i	excitatory neuron (optional)
i_i	inhibitory neuron (optional)



8-action CNC

- W is the set of synapses:

$$w(c_i, c_j),\ i, j = 1, \ldots, n$$
$$w^{\kappa}(e_i, c_i),\ w^{\kappa}(i_i, c_i),\ i, k = 1, \ldots, n$$
$$w^{k}(g, c_i),\ i, k = 1, \ldots, n$$

$$|V| = 3n+1 \qquad |W| = n(n+2)$$

Action Decision

- Exploitation uses Ɛ-greedy
- The timing of spikes determines the action selection:
  - Each core neuron represents each possible action
  - All core neurons receive constant stimuli
  - The first spike ”bootstraps” the SNN
  - The core neuron producing the faster second spike indicates decision *i**:

$$c_{i*} = \underset{c_i}{\operatorname{argmin}}\ t^{(2)}(c_i) \quad ; i = 1, \ldots, n$$

  - where $t^{(f)}(x)$ is the time to fire of the *f*-th spike of neuron *x*

Learning Step

- Each action *i* is associated to performance cost *C* (reward = *1/C*)
- The average observed cost is $G \leftarrow \alpha C + (1-\alpha)G$
- Weight updates are proportional to $\delta = G - C$

$$
\begin{aligned}
w(c_j, c_i) &\leftarrow w(c_j, c_i) + \eta\delta \quad ; j = 1, \ldots, n; i \neq j \\
w^k(g, c_i) &\leftarrow w^k(g, c_i) - \eta\delta \quad ; k = 1, \ldots, n
\end{aligned}
$$

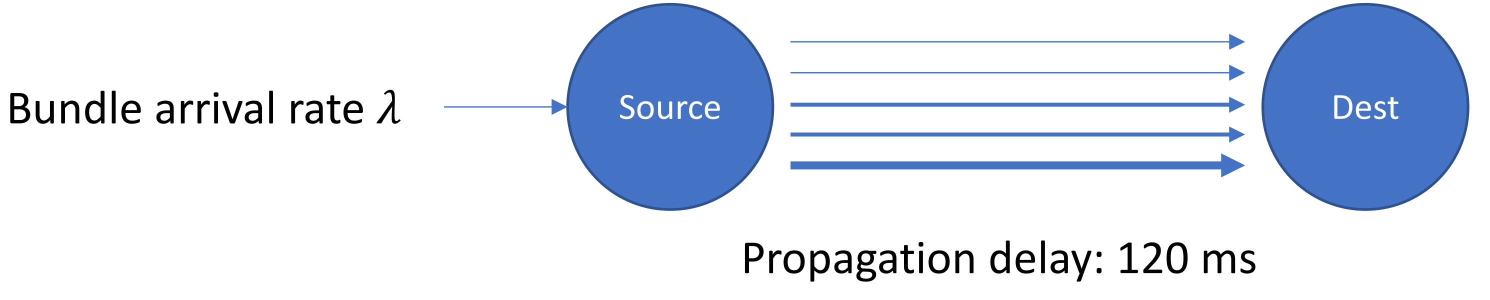
- η>0 is the learning rate
- A normalization phase keeps weights within their effective range
- Learning complexity O(n)

Non-RL Routing Methods

- Shortest path (Dijkstra’s and Bellman ford algorithms)
  - Require information exchange in their distributed versions
  - May not be adequate for highly dynamic networks
- Random, Round Robin
  - Routing decisions consists in the simple random selection or the rotation of link selections
  - Easy to implement with very low complexity (very fast)
  - Starvation free, uniform resource allocation

Evaluation

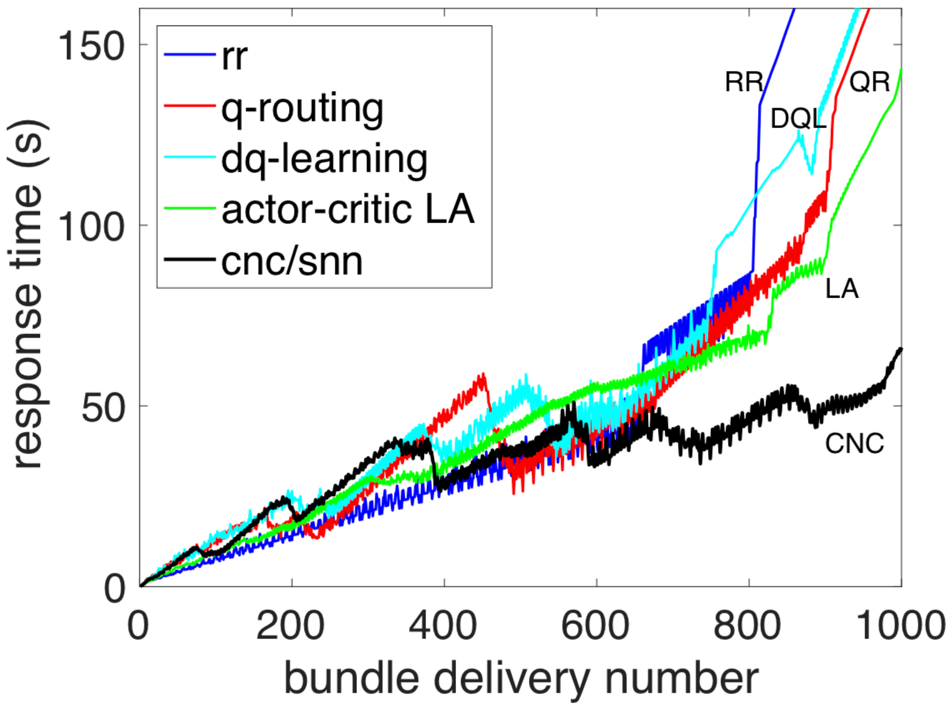
- Linux VMs connected via 5 (physical) links: 2x512 Kbps, 2x1Mbps, 1x2Mbps
- Gateway and routing methods implemented in Python
- Simulated neuromorphic processor
- Emulated wireless links
  - Emulated with point-to-point Ethernet (nominal rate: 1 Gbps)
  - NetEm creates (symmetric) link impairments
  - Non-preemptive, FCFS buffers

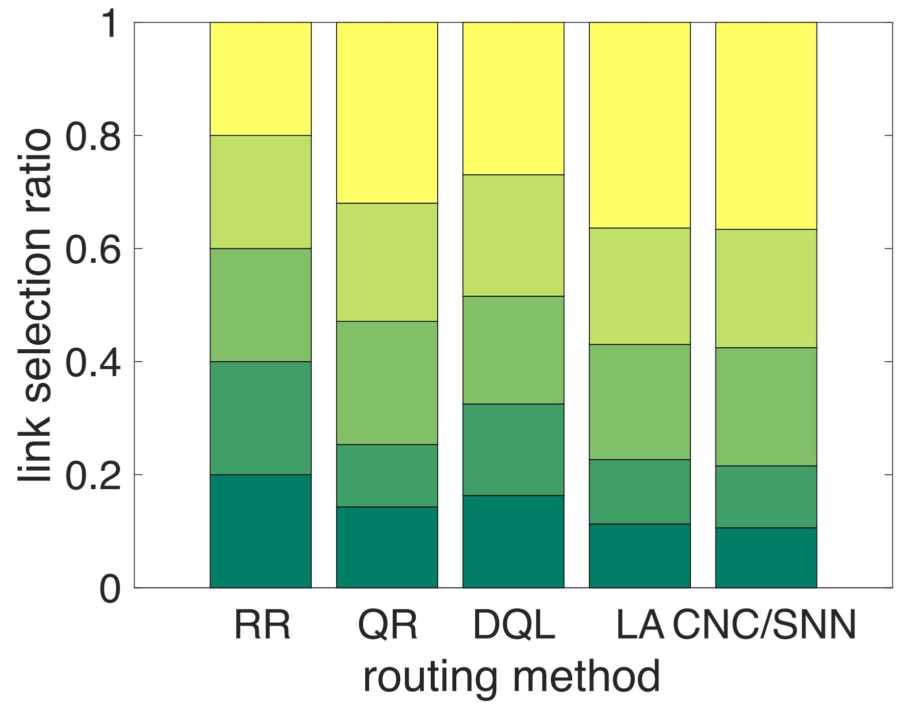


Experiment 1: Constant Link Parameters

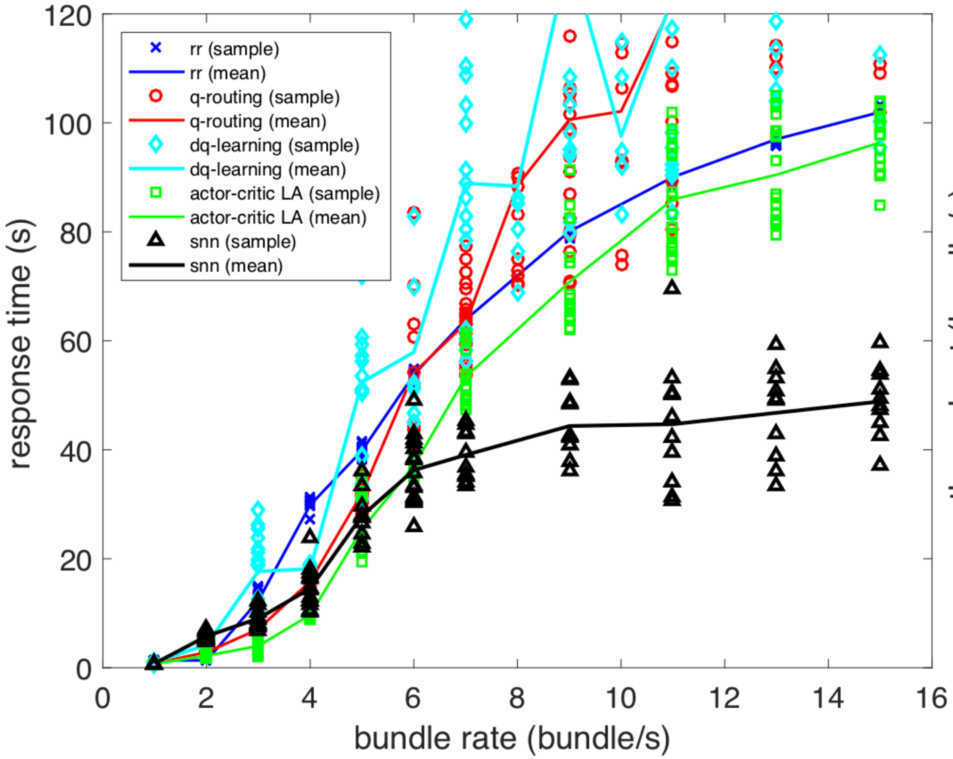
- Links' rate constant for the duration of each experiment
- Traffic consists of 100-Kb bundles sent at a rate λ
- Results with random bundle sizes are very similar
- Parameter λ was chosen relatively large compared to the system capacity
  - This tends to saturate buffers yielding increasing delay over time
  - Suboptimal routing decisions lead to worse response times (e.g., by sending bundles to already saturated buffers)

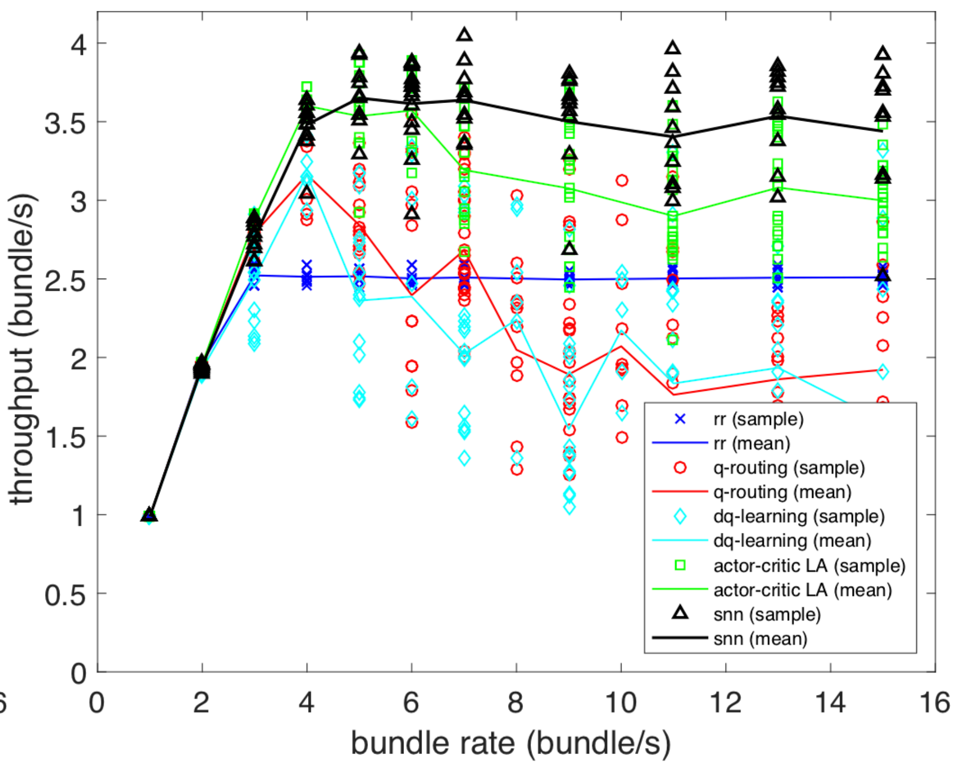
Typical Observations of Routing Performance (Single Run)





Experiment 1: Average Results

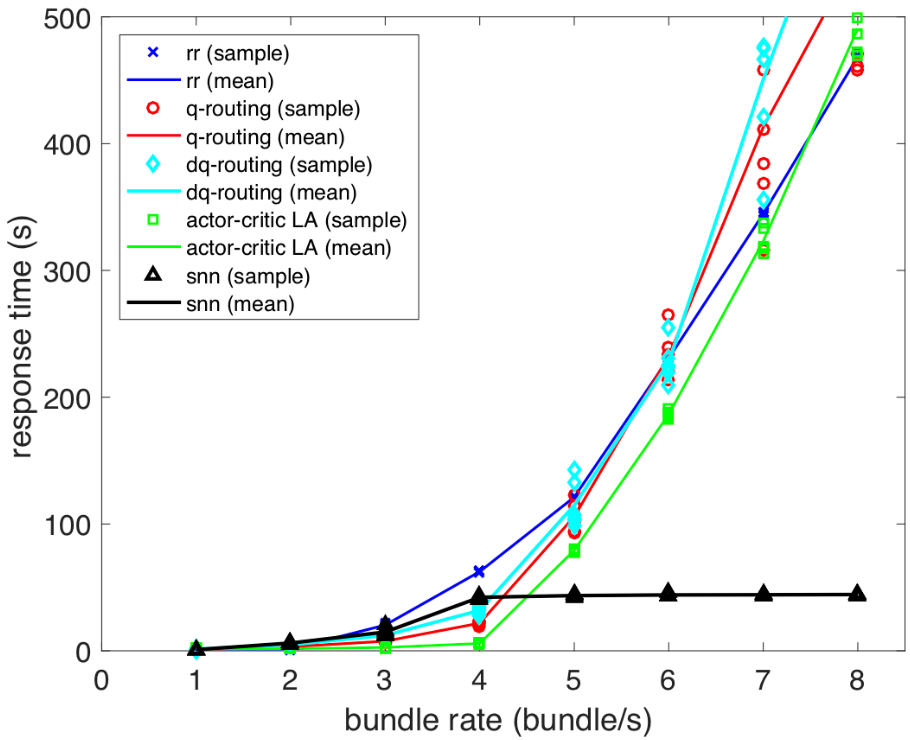


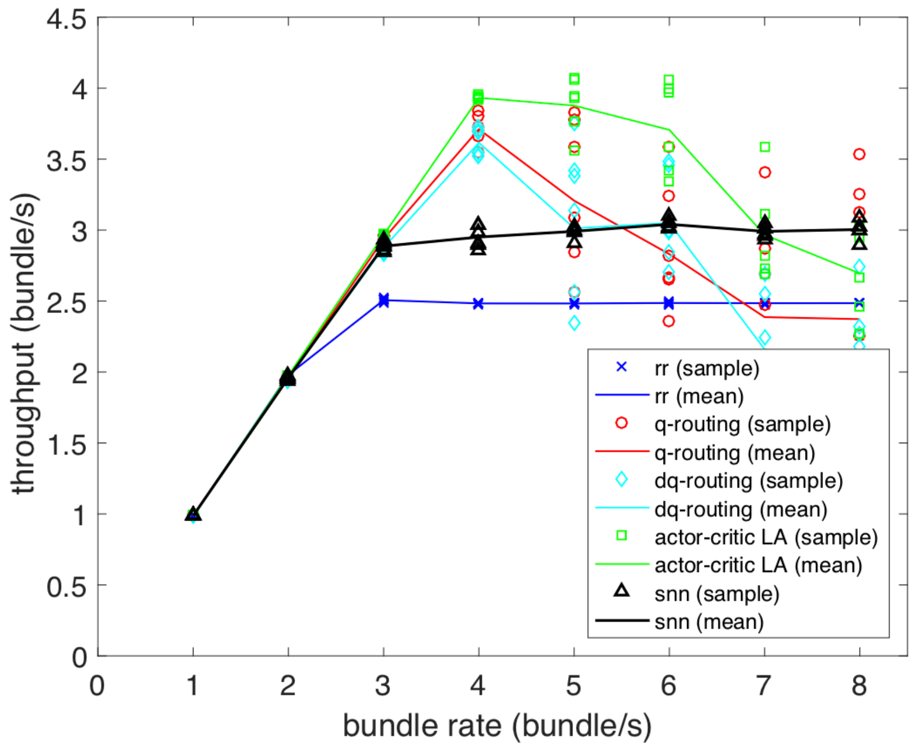


Experiment 2: Variable Link Parameters

- Same traffic parameters used in the previous experiment
- Link impairments vary over time:
  - Changes every 120 s
  - Swap transmission rates of a 512-Kbps and a 1-Mbps link
  - Each experiment lasts for 900 s
  - Total bundles sent per experiment: 900 λ
- Performance differences between the CNC and the other RL methods becomes more evident

Experiment 2: Average Results





Final Remarks

- Cognitive Network Controller:
  - Application of neuromorphic computing to space network routing
  - Autonomous and onboard decision making
  - Networking concept possibly useful for other DTN frameworks
- Observations of the relative performance of the CNC:
  - Reference test application, reproducible experiments
  - Lower bundle delivery latency under mid-high demand than related RL routing techniques
  - Slightly worse performance under low demand, possibly because of the time-complexity of the simulated neuromorphic processor
- We expect to further develop this work in the near future:
  - Parameter selection
  - Realistic testbed
  - Deep space assumptions
  - Hardware neuromorphic processor

Q&A

Acknowledgement: This work was supported by an Early Career Faculty grant from NASA's Space Technology Research Grants Program