Rax: Deep Reinforcement Learning for Internet Congestion Control

Maximilian Bachl, Tanja Zseby, Joachim Fabini

Problem

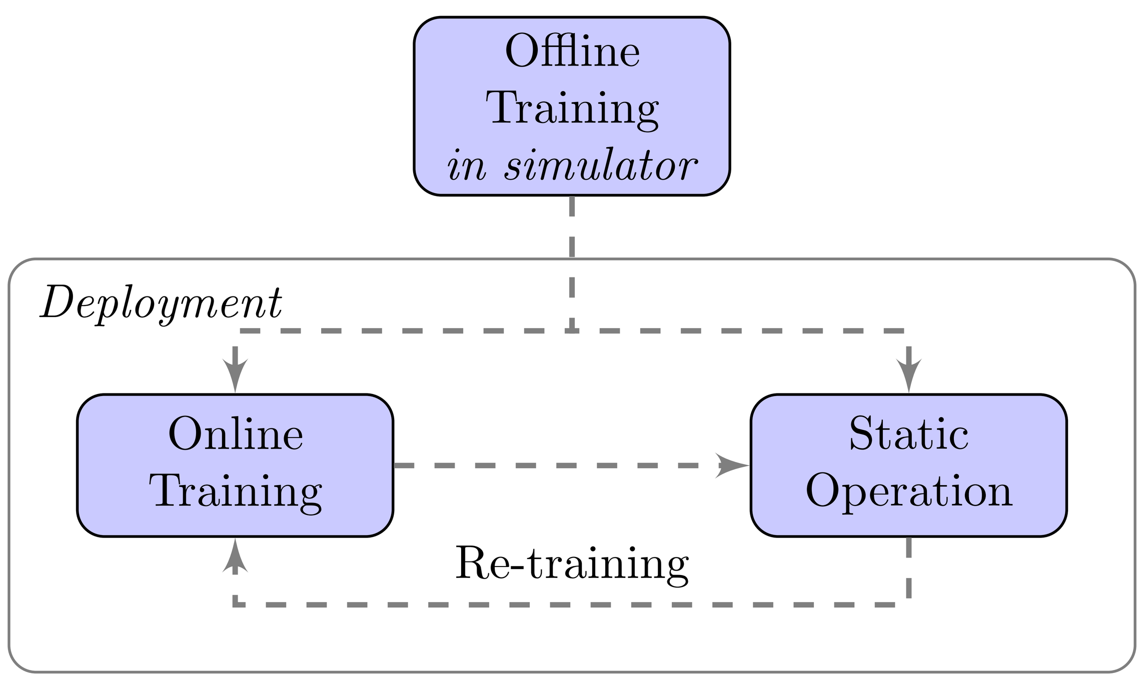
On the Internet you **don't know how fast** you can send:
Too slow ➜ waste of resources
Too fast ➜ "traffic jams" ➜ delay and lost packets
This is called *congestion control*.

Idea

Define utility function, e.g. high throughput, low packet loss

Use **reinforcement learning** to find optimum behavior

Train offline in simulator, online or a combination of both

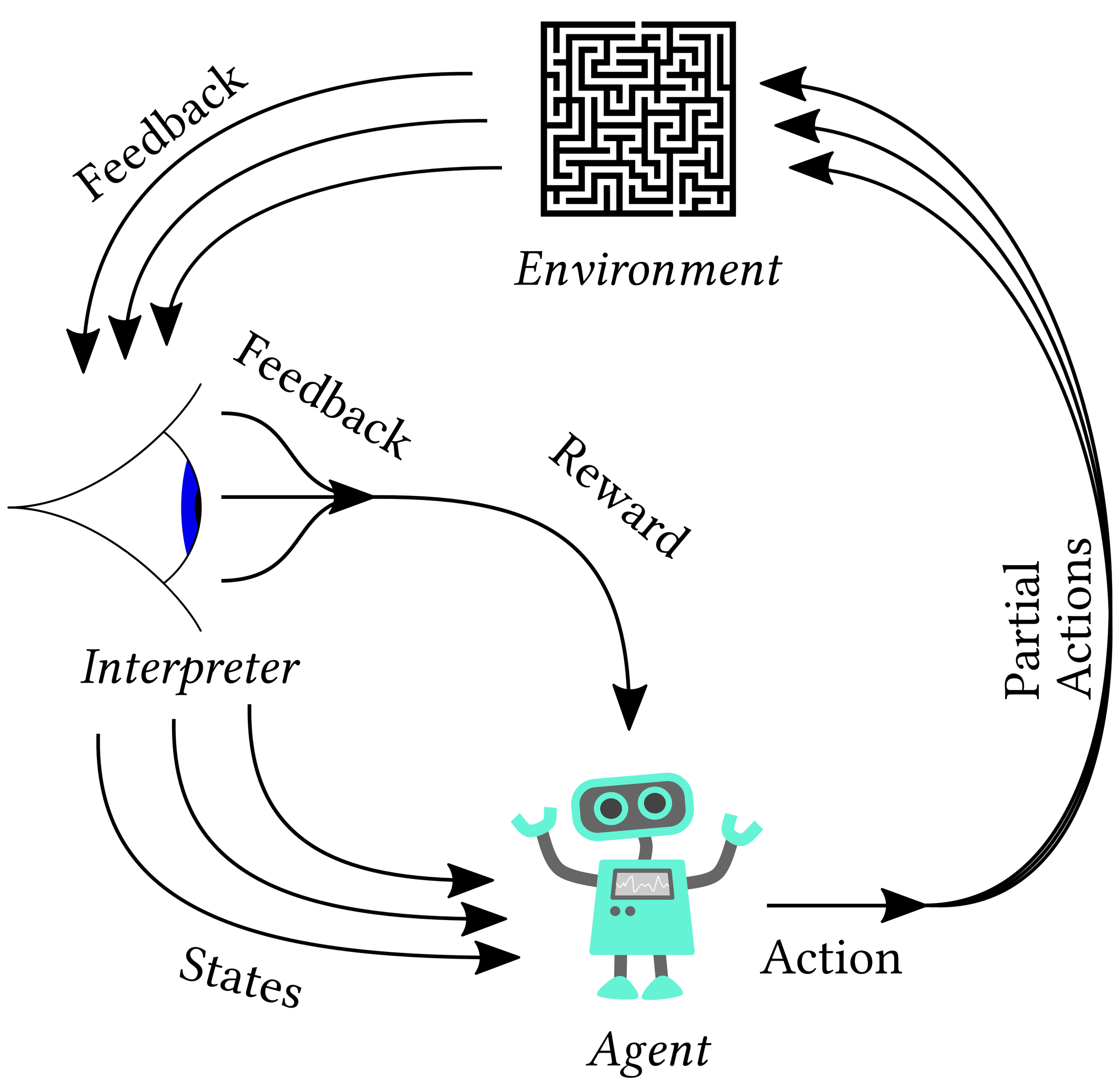


Method

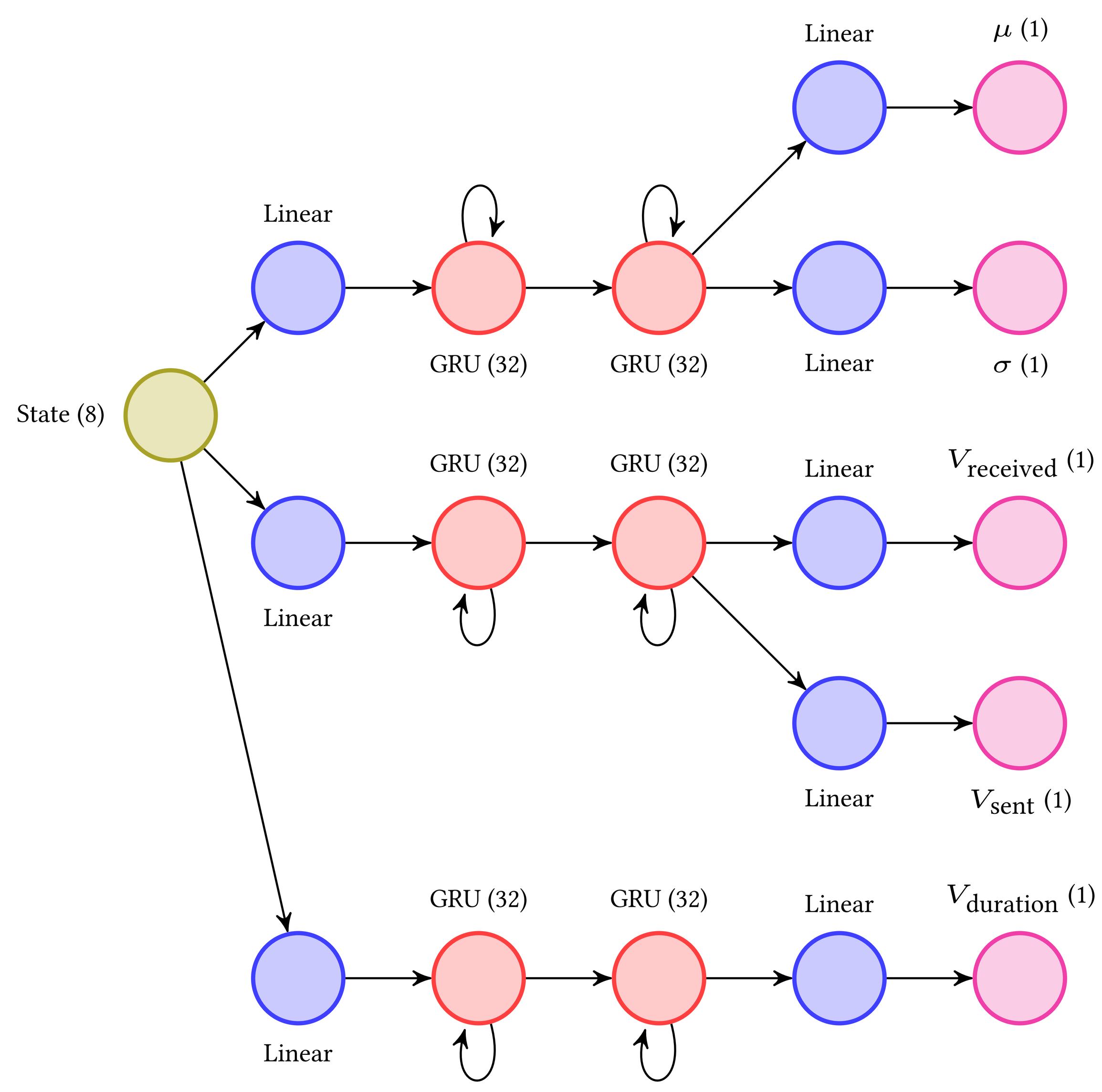
Use *Asynchronous Advantage Actor Critic* framework (Mnih et al., 2016)

Problem:
• **Rewards** come **delayed** (delay in the network)
• **Actions** and **rewards are not atomic**
• **Number** of actions and rewards **not constant per unit of time** (the more packets are sent, the more reward is coming back)

So: modify existing framework to support partial actions and specific characteristics of congestion control



Neural network



We use a **two layer neural network** with **GRUs instead of LSTMs** because LSTMs have an internal state which prevents online learning (state takes huge values after long training episodes).

Separate network for the actor and the critic

Critic network **outputs several values** which are assembled to form the reward function.

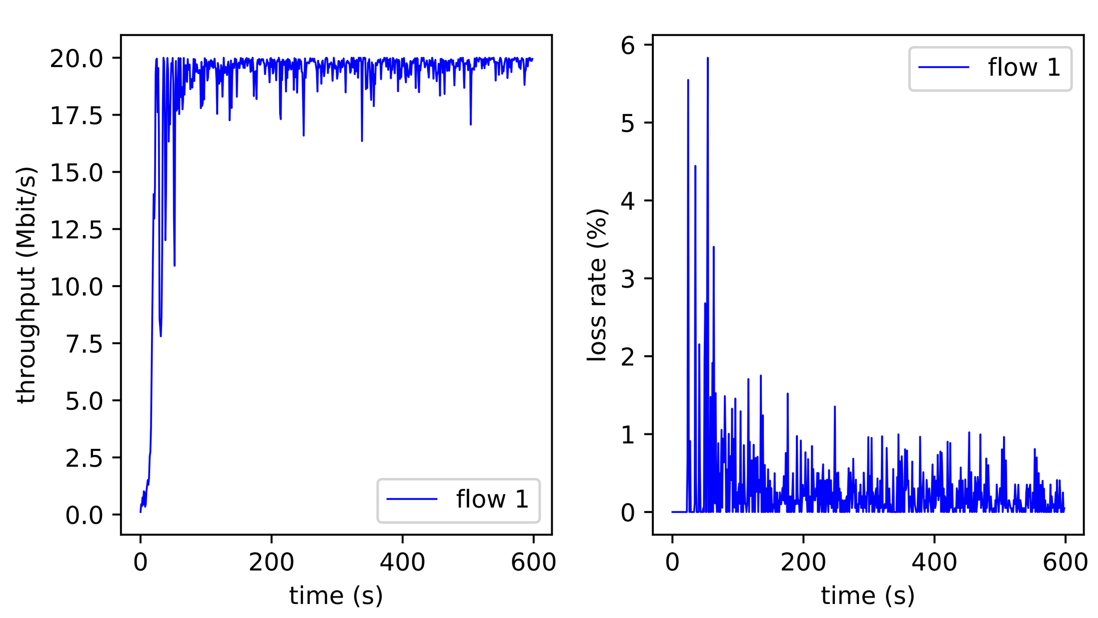
Utility function

Maximize the following utility function:
U = throughput - lost throughput

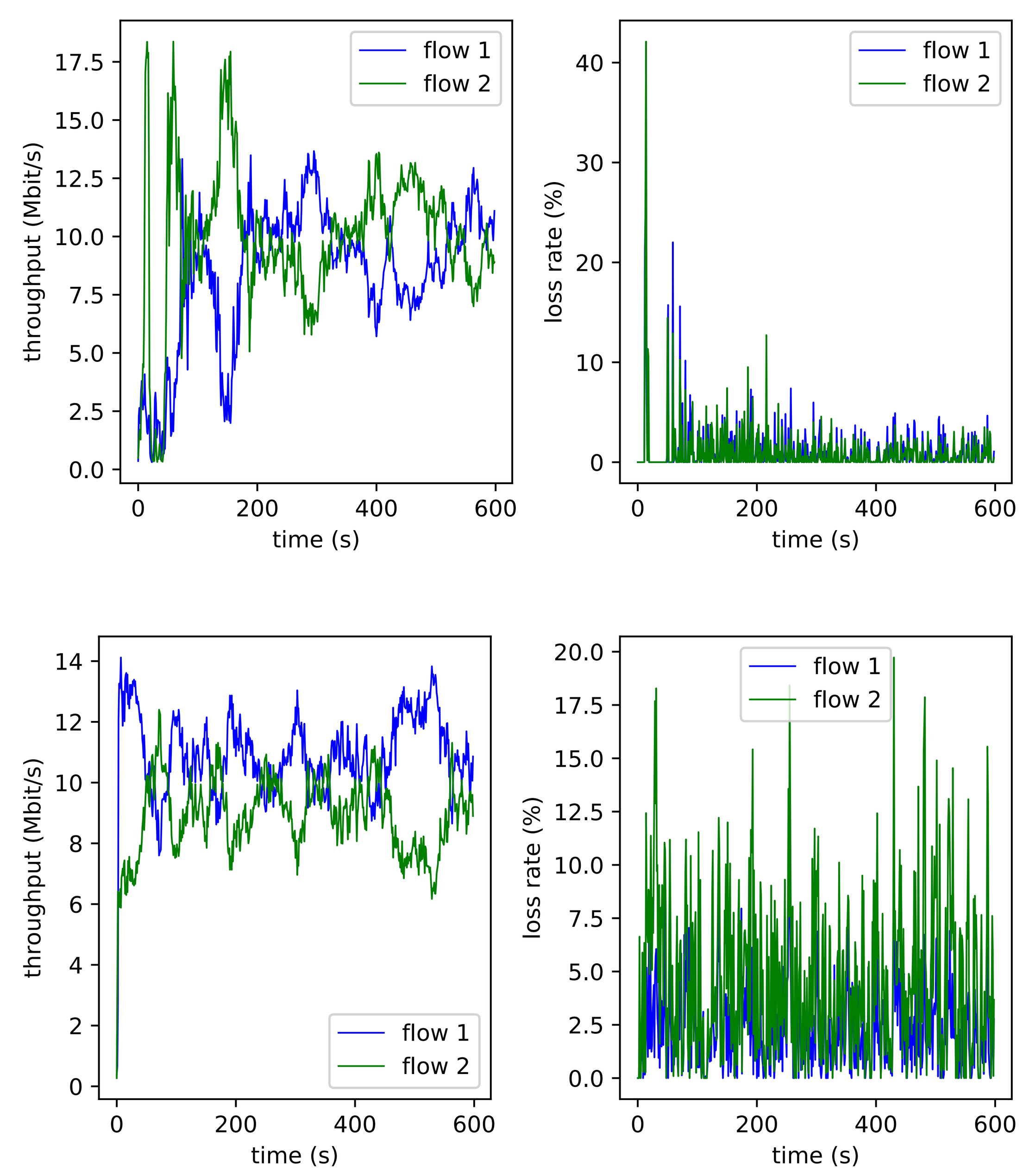
Utility function can be assembled from the values that are output by the critic network above:

$$U = \frac{R_{\text{received}}}{V_{\text{duration}}} - \alpha \frac{V_{\text{sent}} - V_{\text{received}}}{V_{\text{duration}}}$$

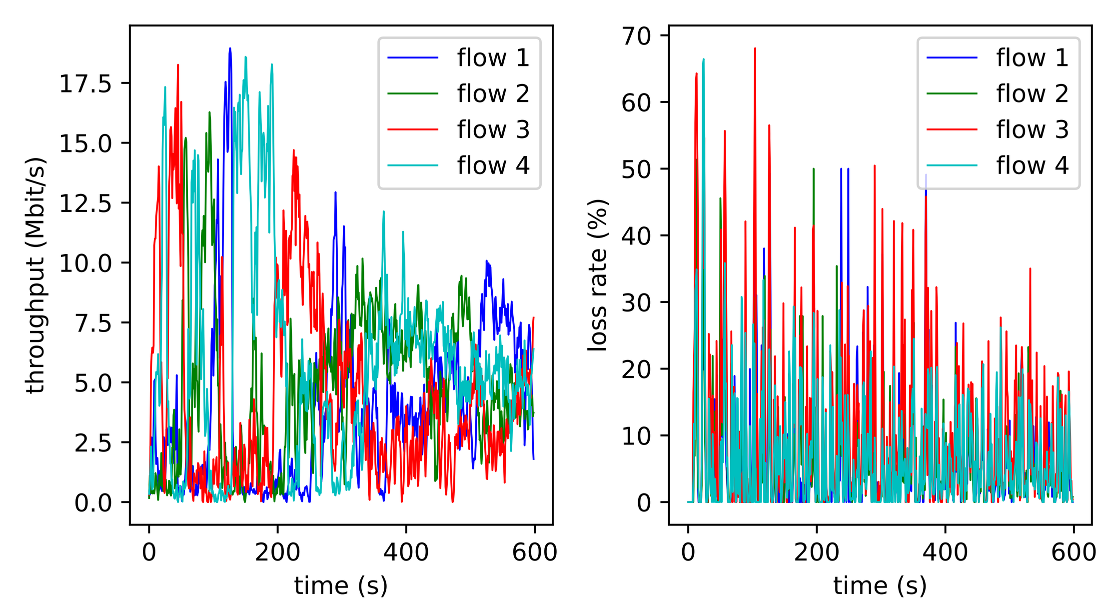
Evaluation



One sender learning to perform congestion control. Over time throughput is maximized and packet loss minimized.

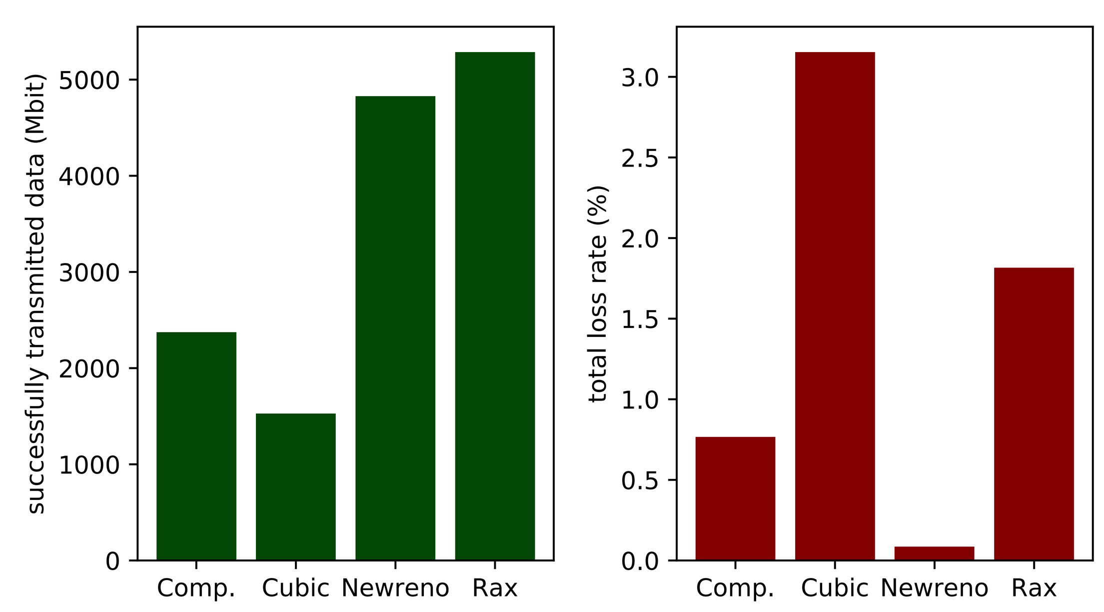


Two senders learning to perform congestion control. Over time throughput is maximized and packet loss minimized (top). Using a trained version (bottom) shows that it takes up available bandwidth very quickly.



The more senders there are, **the harder** the learning becomes.

Reasons:
• With **more senders**, each sender gets **fewer rewards**
• **Environment more unpredictable** (was it me or the others?)



Comparing Rax (two senders) to the traditional congestion control algorithms New Reno, Cubic and Compound

Conclusion

Possible to use **deep reinforcement learning** to **learn congestion control online**

Evaluation shows that after a few minutes of learning our method can achieve better performance than traditional methods.