Learning to Sail Dynamic Networks: The MARLIN Reinforcement Learning Framework for Congestion Control in Tactical Environments

Raffaele Galliera[§,†], Mattia Zaccarini[‡], Alessandro Morelli[§]
Roberto Fronteddu[§], Filippo Poltronieri[‡], Niranjan Suri[⋆,§,†], Mauro Tortonesi[‡]

[†]The University of West Florida (UWF)
[‡]The University of Ferrara (UNIFE)
[§]Institute for Human and Machine Cognition (IHMC)
[⋆]US Army DEVCOM Army Research Laboratory (ARL)

Presenter: Raffaele Galliera (rgalliera@ihmc.org) - Paper #354

Overview

Background
- The Congestion Control Environment
- Challenges

Method

Experimental Setting

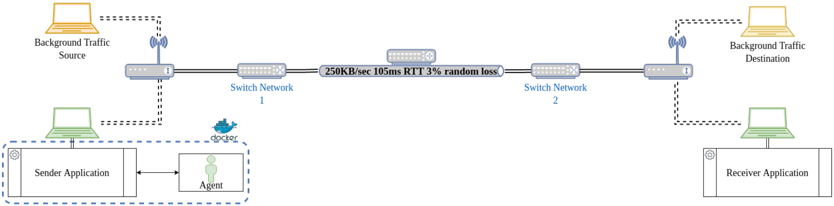
Results

Backup

Congestion Control

- The channel can saturate causing delays and re-transmissions
- Different heuristics are used (eg, **TCP Cubic**).
- **Congestion Window (CWND):** control over bytes allowed to be **in-flight**.
- Heavy congestion can take a link to an impracticable state.

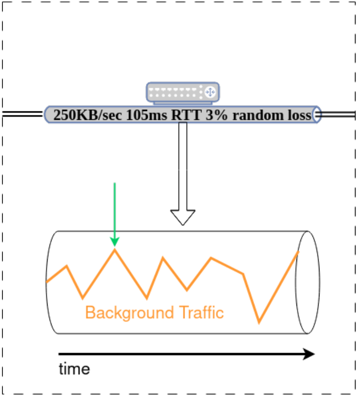
Starting Point



Physical Training network.

R. Galliera, A. Morelli, R. Fronteddu and N. Suri, “MARLIN: Soft Actor-Critic based Reinforcement Learning for Congestion Control in Real Networks”, NOMS 2023 IEEE/IFIP Network Operations and Management Symposium

The Agent's Perception

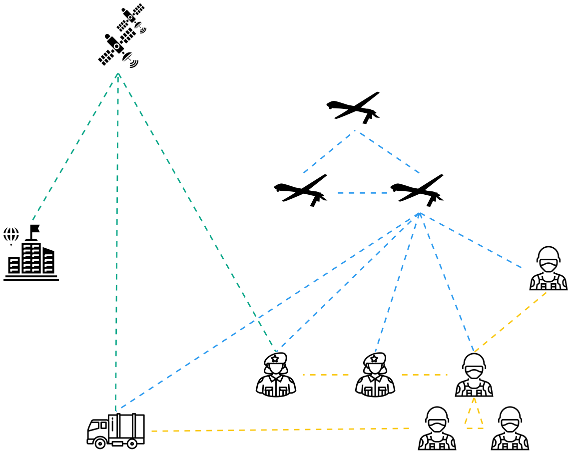


R. Galliera, A. Morelli, R. Fronteddu and N. Suri, "MARLIN: Soft Actor-Critic based Reinforcement Learning for Congestion Control in Real Networks", NOMS 2023 IEEE/IFIP Network Operations and Management Symposium

MARLIN Summary

- **Partnering protocol:** Mockets.
- **Non-blocking** communication.
- Third-party sources utilize the **shared link**
- **Learning algorithm:** Soft Actor-Critic (SAC).
- **Action space:**$[-1, 1]$ tweaking the CWND.
- **Action-Observation History**: 10 steps.

Tactical Networking Environments



Challenges

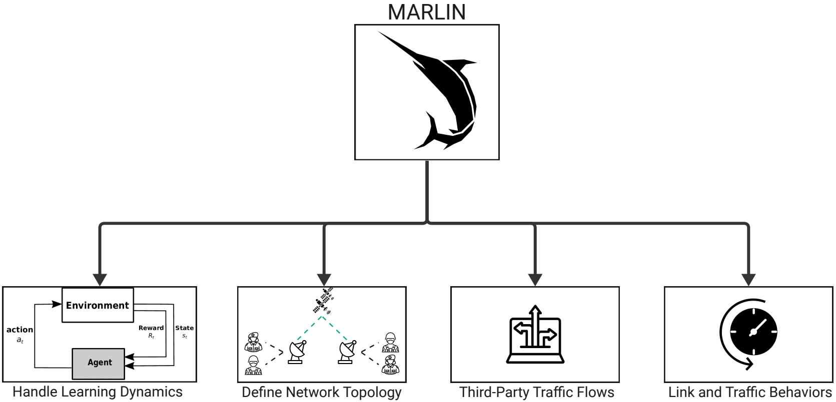
- **Designing networking scenarios** might become unfeasible.
- **Reproducibility**.
- **Reinforcement signals** in tactical networks.

Objective

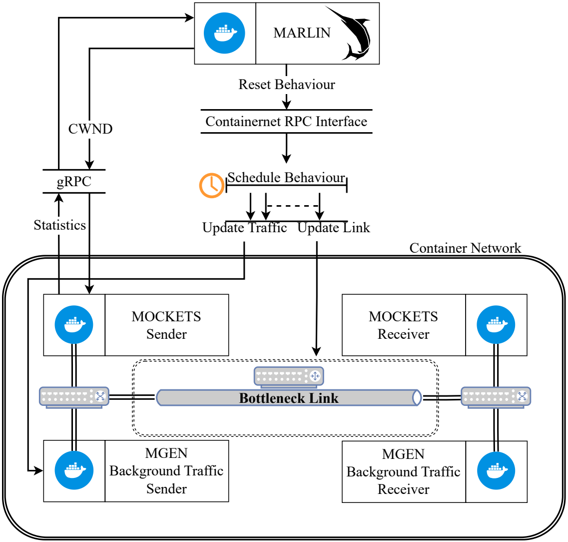
Design a **flexible framework** to train agents in **dynamic** and **unreliable networking** scenarios.

Overview

Extending the MARLIN framework



Scheduling Link Behaviors

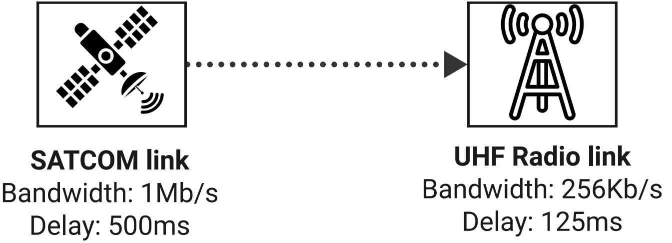


Reward

$$r_t = -\frac{target_t[1 + retr * (1 - loss_c)]}{target_t + acked_t^{cumulative}} \tag{1}$$

Overview

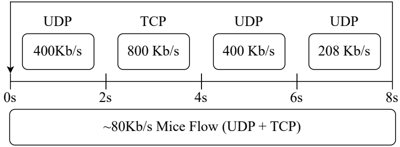
Link Transitions



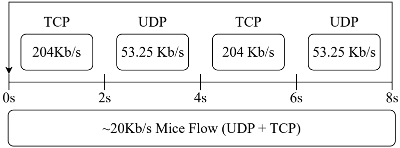
Major training details:

- 500K steps
- Fixed random packet loss (0% SATCOM, 3% UHF Radio)
- Link switch after 10 Seconds from the episode start

Third-Party Traffic Behaviors



(a) Traffic during **SATCOM** link



(b) Traffic during **UHF** link

Flows generated with Multi-Generator (MGEN) Network Test Tool - U.S. Naval Research Laboratory

Testing Scenario

- **Objective:** 600KB payload transfer
- Varying UHF radio link random packet loss (0-3%)
- 100 testing episodes for each packet loss value
- Metrics used:
  - Transfer time (s)
  - Retransmissions
  - RTT Transition Impact

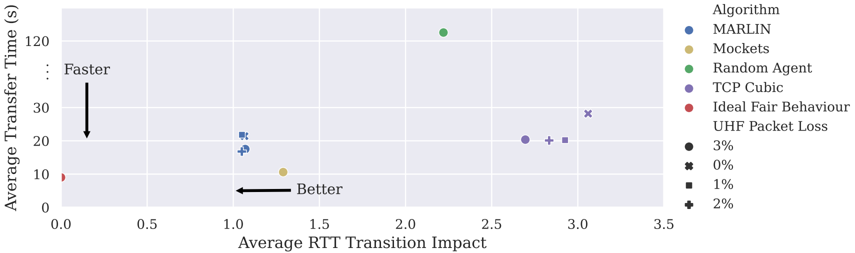
RTT Transition Impact (RTI)

$$RTI = \ln\left(\frac{\sum_{i=1}^{m} \frac{rtt_{i,max}}{rtt_{i,nom}}}{m}\right) \tag{2}$$

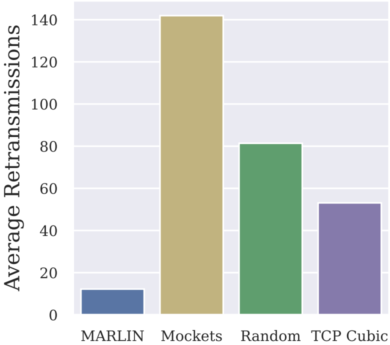
- m link transitions
- $rtt_{i,max}$ maximum rtt detected during link i
- $rtt_{i,nom}$ nominal rtt value during link i

Overview

Average Transfer Time - RTI



Retransmissions



During experiments with 3% random packet loss set on the UHF link.

Conclusion

- A **containerized RL environment** to train agents for CC.
- **Centralized control** of the entire training environment.
- **Programmable link behaviour**.
- **Retransmission-sensitive** rewards.
- **Competitive** trained policies.

GitHub Repository

Overview

Hyperparameters

Hyperparameter	Value
Training steps	5×10^5
History length	10
Training episode length	200
Learning rate	3×10^{-4}
Buffer size	2.5×10^5
Warm-up (learning starts)	1×10^4 steps
Batch size	512
Tau	5×10^{-3}
Gamma	0.99
Training Frequency	$1/episode$
Gradient Steps	-1 (same as episode length)
Entropy regularization coefficient	"auto" (Learned)
MLP policy hidden layers	$[400, 300]$

Table: Hyperparameters used in our experiment.

State

	Feature	Description		Statistic
1	Current cwnd	Current cwnd	1	Last
2	KBs Sent	Amount of KB sent *	2	Mean
3	New KBs sent	Amount of KB acked *	3	STD
4	Acked KBs	Amount of KB acked *	4	Min
5	Packets sent	Packets sent *	5	Max
6	Retransmissions	Number of packets retransmitted *	6	EMA
7	Instantaneous Throughput	Throughput *	7	Difference from Previous
8	Instantaneous Goodput	Goodput *		
9	Unacked KBs	Amount of KBs in flight		
10	Last RTT	Last rtt detected *		
12	Min RTT	Min rtt *		
12	Max RTT	Max rtt *		
13	SRTT	Smoothed rtt *		
14	VAR RTT	rtt variance *		
		* During the last rtt timeframe		

Every feature has 7 nested statistics with a 10 observations history.

Implementation Stack

- **Reinforcement Learning framework:** Stable Baselines 3.
- **Partnering Protocol:** Mockets.
- **Protocol-Agent Communication:** gRPC.
- **Network Emulation:** Containernet + RPYC.
- **Third-Party traffic:** MGEN