EJFAT: Towards Intelligent Compute Destination Load Balancing

Michael Goodrich, Carl Timmer, Vardan Gyurjyan, David Lawrence, Graham Heyes, Yatish Kumar, Stacey Sheldon

Thomas Jefferson National Accelerator Facility, Newport News, VA 23606 USA, and ESnet, 1 Cyclotron Road, Berkeley, CA 94720

E-mail: goodrich@jlab.org

Abstract. To handle increased data flow, Jefferson Lab (JLab) is partnering with ESnet for development of an AI/ML directed compute work Load Balancer (LB) of UDP streamed data. The LB is FPGA based featuring dynamically configurable, low latency and high throughput destination address switching. The LB provides integration of edge and core computing to support JLab experimental programs, the Electron-Ion Collider, as well as data centers of the future.

In the ESnet/JLab FPGA Accelerated Transport (EJFAT) initiative, the function of the LB Data Plane (DP) is to redirect data streams to selectable (but unknown to sender) destination hosts based on current worload and within that host to destination ports as a function of sub-stream id. This effects hierarchical scaling, first across compute machines for processing over a series of events and second, across ports so different data source sub-streams may be assigned to different processors for further parallelization.

The LB Control Plane (CP) programs the DP using compute farm telemetry to direct and balance workloads across a compute cluster as the operating conditions require.

While *Proportional/Integrative/Derivative* (PID) controllers are often seen in similar applications, here we investigate the feasibility of a *Reinforcement Learning* (RL) based schedule manager running in the CP to provide dynamic updates to the DP scheduling policy.

1. Introduction

A detailed description of EJFAT FPGA based LB operations is available in Reference [1]. As described in Reference [1], the EJFAT LB consists of two suites; (1) A DP consisting of an FPGA programmed with *Register Transfer Logic* (RTL) for network device support, and P4 [13] for packet processing and (2) a CP which monitors feedback from destination cluster compute nodes, and dynamically programs the FPGA based DP for the data event to destination node mappings in order to load balance computational work across a cluster of compute nodes.

The central problem of the LB CP is to reconfigure in a sufficiently timely manner the number of data events individually disbursed to cluster nodes over the course of completing one iteration cycle through a fixed number of round-robin slots where each slot is mapped to some node in the cluster, in order to balance work in the cluster and maximize the timeliness of the availability of cluster computed results. The round-robin slots are refereed to as the LB *calendar*, and one cycle through the calendar is one LB iteration.

The Q-Learning (QL) type of RL was chosen due to it's straightforward learning strategy. The QL engine *learns* on every iteration and uses the **Q** matrix as the predictive element; in this formulation it also functions as a learning momentum term and smoothing filter. For an accessible reference on RL/QL, see Reference [2]. The Q value for each cluster node is adjusted each iteration to affect learning according to the standard QL prescription:

$$Q[t+1, node] = Q[t, node] + \alpha \times (R[t, node] + \gamma \times max(E\{R[node]\}) - Q[t, node]) \tag{1}$$

Where α is the *learning rate* constant, γ is the *discount rate* constant, $\mathbf{R}$ is the *rewards* vector[1] from the cluster nodes sent periodically to the QL engine in the CP, and $max(E\{\mathbf{R}\})$ is the maximum expected future reward which in this analysis is the vector $\mathbf{1}$.

2. QL training simulation for Reinforcement Learning Based Load Balancing

Data event Reassembly Engines (RE) running in cluster nodes send the following *reward* (feedback) values to the EJFAT CP:

$$\text{Reward} = \%\text{RE input FIFO empty} \tag{2}$$

This value is on the interval [0,1] where the value 1 indicates that the RE FIFO is empty or at 0% capacity, whereas a value of 0 indicates that the RE FIFO is full or at 100% capacity. This value is typical of what might be sent to a PID controller running in the CP. For an accessible reference on PID, see Reference [3].

In practice, EJFAT CP control policy is to schedule cluster nodes in it's data event delivery disbursement strategy to maintain all nodes at a desired value of Reward = 1, or 0% FIFO occupancy signifying that nodes are experiencing minimum FIFO backlogs.

The EJFAT CP control input to the cluster nodes is effected by adjusting the fraction of schedule slots that each node occupies in the DP round-robin schedule which determines the rate in terms of *data events per cycle* for new work disbursed to each node. The schedule slot occupancy fractions for all nodes is referred to as the *Scheduling Density* (SD) for the cluster.

Each node in the cluster is specified as having a mean *event processing rate* (EPR) which is the number of data events it can process per cycle, and a FIFO size in terms of data events awaiting processing.

A simulation intended to demonstrate the training and performance of a QL engine running in the CP acting as the QL agent to select scheduling actions was constructed. Two simulations were studied; the first for a compute cluster of five nodes with symmetric properties in terms of FIFO capacities and EPRs, and the second with asymmetric properties as listed in Table (1).

Each iteration of the simulation is the effect of one full round-robin (or cycle through the LB calendar) disbursement of data events to the cluster of total *batch size* (Bs) for processing equal to the total EPR of the cluster (*EPR*-Table (1)) as

$$Bs = \sum_{nodes} EPR[node] - src \tag{3}$$

where *src* is a small margin of *spare event processing rate capacity* (SR) reserved to handle *process noise* (PN) in the system. It was experimentally determined that one unit of SR was adequate for this experiment.

[1] By *vector* we mean the collection of similar values over all cluster nodes.

Process noise is modeled as a discrete probability mass function (PMF) over deviations in the number of data events dequeued from the RE FIFOs each iteration based on each node's stated EPR in Table (1). A PMF vector of

$$PMF = [4\%, 10\%, 34\%, 4\%, 34\%, 10\%, 4\%] \tag{4}$$

for the deviations in event counts

$$[-3, -2, -1, 0, 1, 2, 3] \tag{5}$$

was sampled for process noise on each iteration for every node.

3. Discussion of Results

For statistics, results from 10,000 iterations was used; for display purposes, only the first 100 iterations is shown for clarity. In all figures, the following color scheme was used: black for node 1, blue for node 2, green for node 3, orange for node 4, magenta for node 5. Table (1) gives the expected asymptotic **SD** for each node based on their fractional **EPR** per cycle (FEPR). In the figures below, the left column is the symmetric case, and the right column the asymmetric case. Figure (1) shows the progression of Schedule Densities for all nodes starting from a uniform distribution over all nodes for both simulations for the first few iterations for clarity and Figure (2) shows their asymptotic behavior.

SD is calculated from **Q** as[2]

$$\mathbf{SD}_i = \mathbf{Q}_i \div \sum_{nodes} Q_i[node] \tag{6}$$

Each data line from lowest SD to highest follows in the same order implied by the values, which are the red lines in the figures, in the *FEPR*-Table (1) data line in the asymmetric simulation, whereas all lines are noisy about the common FEPR value in the symmetric case. Statistics for these values are shown in Table (2), which indicates that in both cases that **SD** is close to the expected **EPR** values. We note here the divergence of the measured **SD** in the asymmetric case from the theoretical values in Table (1) that however still resulted in the expected values for disbursement counts (*Db*-Table (2)) compared to **EPR**.

SD is used to determined the number of events distributed to each node each iteration s.t.

$$\mathbf{Db}_i = Bs \times \mathbf{SD}_i \tag{7}$$

where the actuals are shown in Figure (3) and the statistics for which are given as *Db*-Table (2) which should be compared to the values listed as *EPR*-Table (1). Here we observe that the difference between Db and EPR determine the allocation of *spare rate capacity* by the dynamics of the scheduling decision and is given as *SR*-Table (2) and shows in the symmetric case the highly desirable pattern of uniformity across nodes allowing each node equal opportunity to work off any FIFO backlogs that might form due to process noise. This issue appears to be the key success factor in inhibiting FIFO growth and is explained below. The resulting allocation in the asymmetric case is graduated across nodes and scales roughly as the nodes EPR for reasons that are currently not understood.

[2] The matrix notation like $\mathbf{X}_i$ means the i^{th} row in the matrix corresponding to that iteration in the simulation, i.e., it is a vector.

Disbursement pattern (**Db**) together with **EPR** and **PN** the process noise matrix, combine to determine RE FIFO size (the matrix **Fs**) s.t.

$$\mathbf{Fs}_{i+1} = \mathbf{Fs}_i + \mathbf{Db}_i - (\mathbf{EPR}_i + \mathbf{PN}_i) \tag{8}$$

each cycle and which is depicted in Figure (4) and the statistics for which are given as *Fs*-Table (2) which should be compared to the FIFO capacities for each node in Table (1). It should be noted that Equation (8) implies a random walk structure for FIFO size processes indicating that deployed systems *must* have some reserve processing capacity to remove data events from the FIFOs that have temporarily increased since a random walk unchecked will tend to excurse to ever increasing bounds proportional to $\sqrt{i}$ as the system progresses through increasing iterations.

For FIFO behavior we note that in both cases FIFOs are uniformly near zero as the distribution of SR across nodes as indicated by Table (2) inhibits FIFO occupancy growth. In both cases, FIFO occupancies fluctuate rapidly due to process noise where the SR allocation necessarily lags, and in some cases temporary increases in FIFO levels persist for some number of subsequent cycles. Linear trend lines fitted to these FIFO processes show slopes on the order of 10^{-5} with mixed algebraic signs.

Quantization Error/Spare Capacity Table (2) shows the allocation of spare capacity by the scheduler where in general it has no foreknowledge as to its proper placement. Since SD values which are real numbers must be transformed into integer counts to populate the integral number of LB schedule slots, any assignment inaccuracy or bias, particularly due to quantization errors, together with process noise is prone to lead to over allocation of events above a nodes processing capacity and lead to permanent or long term elevations in FIFO counts. A good example of this phenomena is seen in Figure (5) where a hypothetical small over-allocation of events in certain cycles (e.g., cycle 15) causes the FIFO level to increase to higher levels which persist until an under-allocation cycle occurs (e.g., cycle 30) where the red line is EPR, the circled black line is Db and the stair-stepped black is Fs.

Table 1. Simulation Parameters - Symmetric/Asymmetric.

	N1	N2	N3	N4	N5		N1	N2	N3	N4	N5
EPR	10	10	10	10	10	EPR	8	9	10	11	12
FEPR	0.2	0.2	0.2	0.2	0.2	FEPR	0.16	0.18	0.20	0.22	0.24
FIFO	30	30	30	30	30	FIFO	24	27	30	33	36

Table 2. Simulation Results - Symmetric/Asymmetric.

	Node1	Node2	Node3	Node4	Node5
Mean-SD	0.2	0.2	0.2	0.2	0.2
StdDev-SD	0.0017	0.0017	0.0017	0.0017	0.0016
Mean-Db	9.8	9.8	9.8	9.8	9.8
StdDev-Db	0.49	0.49	0.48	0.49	0.49
Mean-Fs	3.0	2.86	3.1	3.1	3.2
StdDev-Fs	3.02	2.77	3.05	3.03	3.16
SR	0.2	0.2	0.2	0.2	0.2

	Node1	Node2	Node3	Node4	Node5
Mean-SD	0.19	0.19	0.19	0.21	0.22
StdDev-SD	0.003	0.002	0.003	0.003	0.002
Mean-Db	7.8	8.8	9.8	10.8	11.7
StdDev-Db	1.76	1.68	0.82	0.42	0.45
Mean-Fs	3.0	2.9	3.5	3.0	2.6
StdDev-Fs	2.38	2.47	3.48	3.21	2.96
SR	0.16	0.17	0.18	0.22	0.27

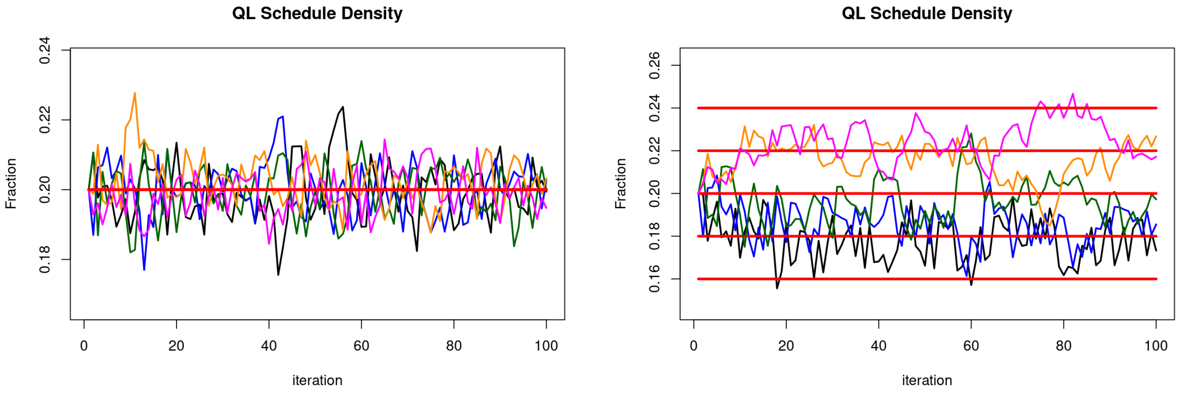

Figure 1. Initial QL Schedule Density

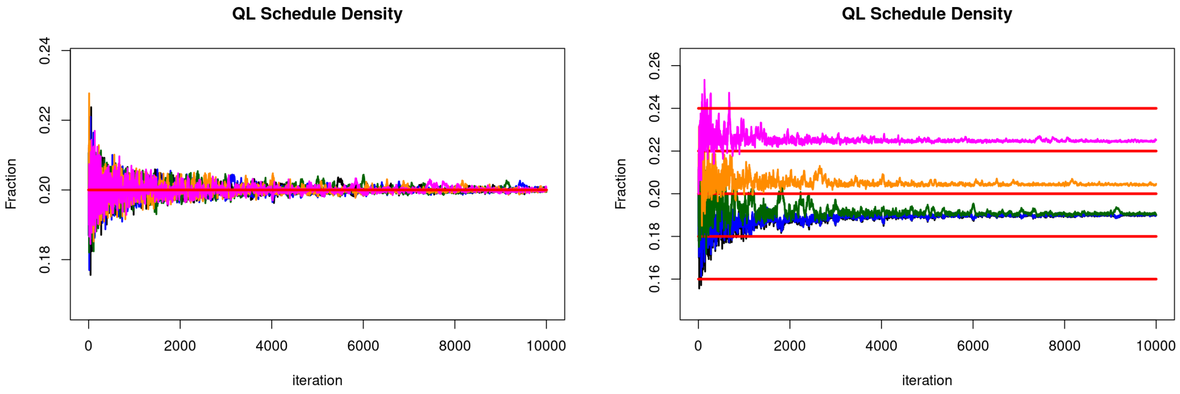

Figure 2. Asymptotic QL Schedule Density

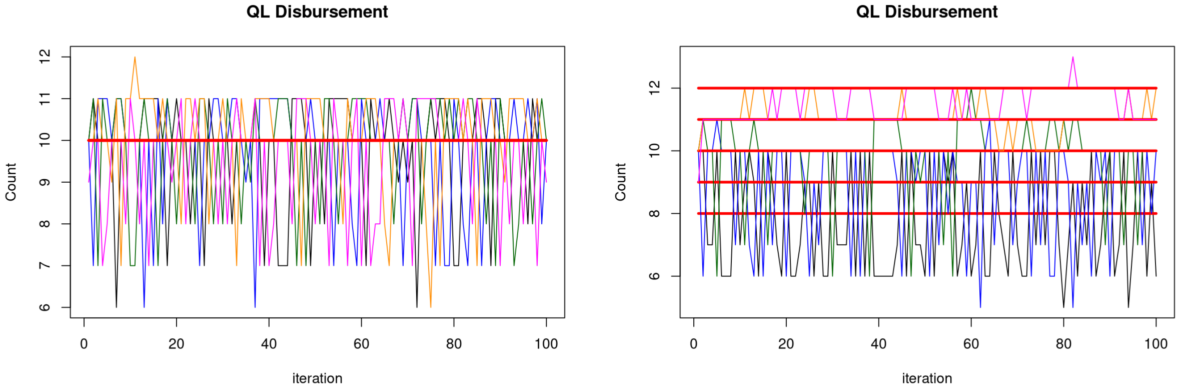

Figure 3. QL Disbursement

.

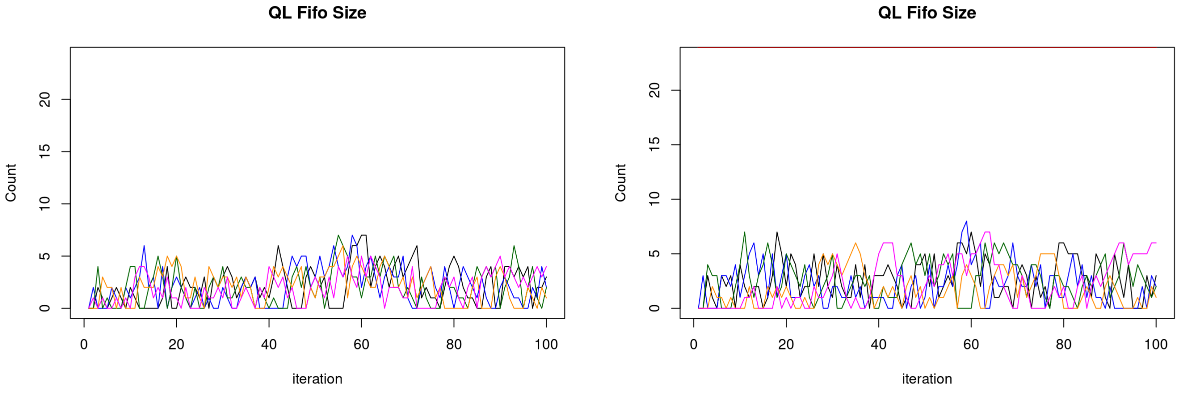

Figure 4. QL FIFO Size

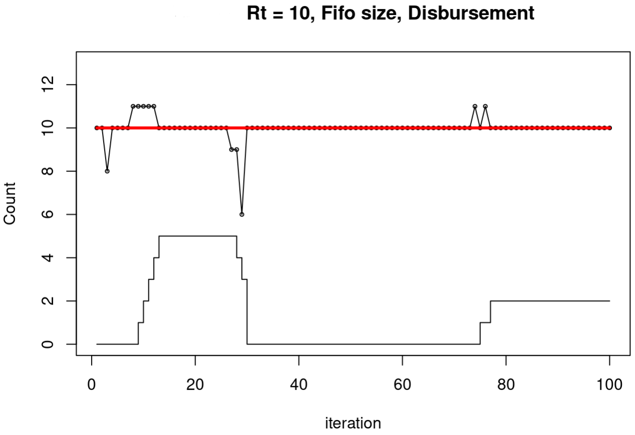

Figure 5. QL Disbursement / FIFO Size

4. Conclusions

Q Learning in this study appears do an adequate job of maintaining an acceptable though non-optimal schedule in both symmetric and asymmetric cases. Notable is allocation of spare rate capacity to keep the cluster balanced and free from excessive FIFO backlogs.

References

[1] "ESnet/JLab FPGA Accelerated Transport," IEEE Transactions on Nuclear Science, vol. 70, no. 6, Feb. 2023. DOI: 10.1109/TNS.2023.3243871.

[2] "Not a tutorial of a Bayesian implementation of a reinforcement learning model": Online: https://bruno.nicenboim.me/2021/11/29/bayesian-h-reinforcement-learning/#ref-ZhangEtAl2020.

[3] "PID Controller Design": [Online:] https://ctms.engin.umich.edu/CTMS/index.php?example=Introduction&section=ControlPID.

[4] "BES Computing and Data Requirements in the Exascale Age," [Online]. Available: https://science.osti.gov/-/media/ascr/pdf/programdocuments/docs/2017/DOE-ExascaleReport_BES_final.pdf. Accessed August 22, 2022

[5] "FES Exascale Requirements Review," [Online]. Available: https://science.osti.gov/-/media/ascr/pdf/programdocuments/docs/2017/DOE-ExascaleReport-FES-Final.pdf. Accessed August 22, 2022

[6] "HEP Exascale Requirements Review," [Online]. Available: https://science.osti.gov/-/media/ascr/pdf/programdocuments/docs/2017/DOE-ExascaleReport-HEP.pdf. Accessed August 22, 2022

[7] "Nuclear Physics Exascale Requirements Review," [Online]. Available: https://science.osti.gov/-/media/ascr/pdf/programdocuments/docs/2017/DOE-ExascaleReport-NP-Final.pdf. Accessed August 22, 2022

[8] National Academies of Sciences, "An Assessment of U.S.-Based Electron-Ion Collider Science.," The National Academies Press., [Online]. Available: https://doi.org/10.17226/25171. Accessed August 22, 2022

[9] ESnet, Energy Sciences Network, "Nuclear Physics Network Requirements Review," University of California, Publication Management System report number LBNL-2001281, May 8–9, 2019.

[10] Scientific Computing Plan for the ECCE Detector at the Electron Ion Collider [Online]. Available: https://arxiv.org/abs/2205.08607

[11] Streaming readout for next generation electron scattering experiment [Online]. Available: https://arxiv.org/abs/2202.03085 Accessed August 22, 2022

[12] Data Plane Development Kit [Online]. Available: https://www.dpdk.org Accessed August 22, 2022

[13] Programming Protocol-independent Packet Processors [Online]. Available: https://opennetworking.org/p4/ Accessed August 22, 2022