Joint Power Allocation and Network Slicing in an Open RAN System

Mojdeh Karbalaee Motalleb
School of ECE, College of Engineering, University of Tehran, Iran
Email: {mojdeh.karbalaee}@ut.ac.ir,

***Abstract*—**
***Index Terms*—**

I. Questions

1) **Why deep Q learning is better than Q learning?**
Deep RL uses a Deep Neural Network to approximate Q(s,a). Non-Deep RL defines Q(s,a) using a tabular function.
Popular Reinforcement Learning algorithms use functions Q(s,a) or V(s) to estimate the Return (sum of discounted rewards). The function can be defined by a tabular mapping of discrete inputs and outputs. However, this is limiting for continuous states or an infinite/large number of states. A more generalized approach is necessary for large number of states.
Function approximation is used for a large state space. A popular function approximation method is Neural Networks. You can make a Deep Neural Network by adding many hidden layers.
Thus, Deep Reinforcement Learning uses Function Approximation, as opposed to tabular functions. Specifically DRL uses Deep Neural Networks to approximate Q or V (or even A).
When the environment gets complicated, the knowledge space can become huge and it no longer becomes feasible to store all (state, action) pairs. If you think about it in raw terms, even a slightly different state is still a distinct state (e.g. different position of the enemy coming through the same corridor). You could use something that can generalize the knowledge instead of storing and looking up every little distinct state.
So, what you can do is create a neural network, that e.g. predicts the reward for an input (state, action) (or pick the best action given a state, however you like to look at it) So, what you effectively have is a NN that predicts the Q value, based on the input (state, action). This is way more tractable than storing every possible value like we did in the table above.
$Q = neural - network.predict(state, action)$
2) **Find QoS that is required for different applications**
   - **Bandwidth and throughput**: Bandwidth is the available capacity of connection between two terminals as the most popular term for that is (bps). Throughput slightly differs from bandwidth as it stands for effective bandwidth that is provided by network

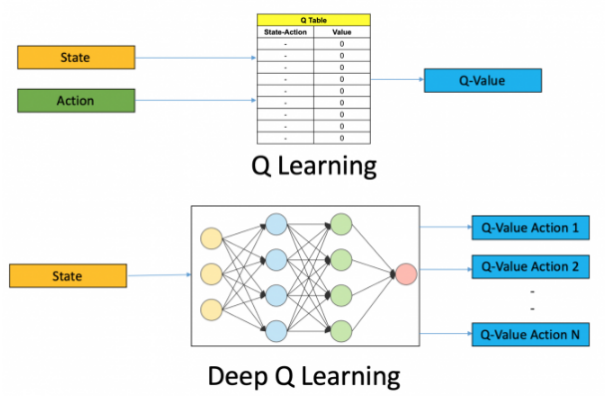

Fig. 1: deep Q learning structure [1]

Application	Bandwidth	Sensitivity to:		
		Delay	Jitter	Loss
VOIP	Low	High	High	Med
Video Conferencing	High	High	High	Med
Streaming Video	High	Med	Med	Med
Streaming Audio	Low	Med	Med	Med
Client/Server Transactions	Med	Med	Low	High
Email	Low	Low	Low	High
File Transfer	Med	Low	Low	High

Fig. 2: deep Q learning structure [2]

- **Delay or latency:** It specifies the time it takes for a packet to leave source until reaching the destination. Applications and network devices can cause delay.
- **Jitter (delay variation):** Jitter is an interval between subsequent packets. It is occurred by network congestion, route alternation and etc.
- **Loss:** It is amount of packets out of all that are not received at destination. The success of QOS depends on this factor.
- **Reliability:** Some applications are sensitive to packet loss such as real-time applications. Thus there must be some mechanism either in application or network to minimize the packet loss, such as forward error correction (FEC).

3) **Where do we use out of order delivery?**
out-of-order delivery is the delivery of data packets in a different order from which they were sent. Out-of-order delivery can be caused by packets following multiple paths through a network, or via parallel processing paths within network equipment that are not designed to ensure that packet ordering is preserved. One of the functions of TCP is to prevent the out-of-order delivery of data, either by reassembling packets into order or forcing retries of out-of-order packets.
Some applications running on your network are sensitive to delay. These applications commonly use the UDP protocol as opposed to the TCP protocol. The key difference between TCP and UDP as it relates to time sensitivity is that TCP will retransmit packets that are lost in transit while UDP does not. For a file transfer from one PC to the next, TCP should be used because if any packets are lost, malformed or arrive out of order, the TCP protocol can retransmit and reorder the packets to recreate the file on the destination PC.

4) **What is the best method for online learning?! Game theory vs RL vs DRL vs Deep learning?**
In Reinforcement Learning (RL) it is common to imagine an underlying Markov Decision Process (MDP). Then the goal of RL is to learn a good policy for the MDP, which is often only partially specified. MDPs can have different objectives such as total, average, or discounted reward, where discounted reward is the most common assumption for RL. There are well-studied extensions of MDPs to two-player (i.e., game) settings.
Game theory is quite involved in the context of Multi-agent Reinforcement learning (MARL).
RL: A single agent is trained to solve a Markov decision problem (MDPS). GT: Two agents are trained to solve Games. A multi-agent Reinforcement learning (MARL) can be used to solve for stochastic games.
If you are interested in the single-agent application of RL in deep learning, then you do not need to go for any GT course. For two or more agents you may need to know the game-theoretic techniques.
So a big differences is that mostly RL is a single agent decision and GT is a multiple agent decision.
Then there are minor differences related to the most common uses. RL uses decision through time, GT only equilibria points. RL considers infinite state space, GT considers finite ones. RL is a learning problem (you have to learn the model of the world) while GT is a planning problem (you already know your game matrix). Of course there are exception to all these minor differences.
For instance it is easy to prove that two agents using Q-Learning in a competitive game will converge to the Nash equilibria of a game. The reinforcement learning techniques enable a single agent to learn optimal behavior through trial-and-error interactions

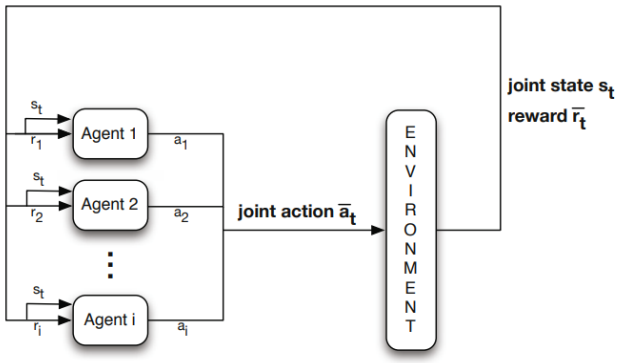

Fig. 3: Multiple agents acting in the same environment [3]

with its environment. Various RL techniques have been developed which allow an agent to optimize its behavior in a wide range of circumstances. However, when multiple learners simultaneously apply reinforcement learning in a shared environment, the traditional approaches often fail. When, in addition to multiple agents, we assume a dynamic environment which requires multiple sequential decisions, the problem becomes even more complex. Now agents do not only have to coordinate, they also have to take into account the current state of their environment. This problem is further complicated by the fact that agents typically have only limited information about the system.
Game theory is generally defined as the mathematics of conflicts and are being utilized in various fields like economy, psychology, AI, sociology etc. In Game theory w.r.t RL, the policy is the strategy, mapping all possible state of actions, in respect to one of the players of the game. The types of games in Multi-Agent RL (MARL) are: Static Games : players are independent and make simultaneous decision Stage Games : the rules depend on specific stages Repeated Games : when a game is played in sequence An extension of the single agent Markov decision process (MDP) to the multi-agent case can be defined by Markov Games. In a Markov Game, joint actions are the result of multiple agents choosing an action independently. While in normal form games the challenges for reinforcement learners originate mainly from the interactions between the agents, in Markov games they face the additional challenge of an environment with state transitions. This means that the agents typically need to combine coordination methods or equilibrium solvers used in repeated games with MDP approaches from single-agent RL.

5) **Find best delay formula which is the nearest to reality (M/M/1 M/D/1 M/G/1)** In the LTE system, the latency can be divided into two major parts: (1) user plane (U-plane) latency and (2) control plane (C-plane) latency. The U-plane latency is measured by one directional transmit time of a packet to become

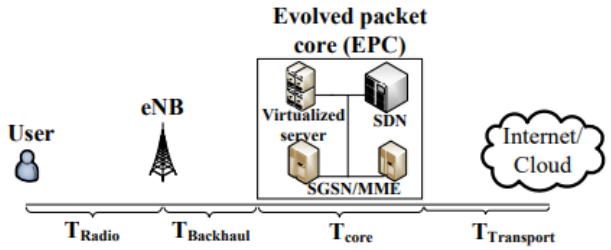

Fig. 4: Latency contribution in E2E delay of a packet transmission [4]

available in the IP layer between evolved UMTS terrestrial radio access network (E-UTRAN) edge/UE and UE/E-UTRAN node. On the other hand, C-plane latency can be defined as the transition time of a UE to switch from idle state to active state. At the idle state, an UE is not connected with radio resource control (RRC). After the RRC connection is being setup, the UE switches from idle state into connected state and then enters into active state after moving into dedicated mode. Since the application performance is dependent mainly on the U-plane latency, U-plane is the main focus of interest for low latency communication. In the U-plane, the delay of a packet transmission in a cellular network can be contributed by the RAN, backhaul, core network, and data center/Internet.

$$T = T_{Radio} + T_{Backhaul} + T_{Core} + T_{Transport} \quad (1)$$

- TRadio
  is the packet transmission time between eNB and UEs and is mainly due to physical layer communication. It is contributed by eNBs, UEs and environment. It consists of time to transmit, processing time at eNB/UE, retransmissions, and propagation delay. Processing delay at the eNB involves channel coding, rate matching, scrambling, cyclic redundancy check (CRC) attachment, precoding, modulation mapper, layer mapper, resource element mapper, and OFDM signal generation. On the other hand, uplink processing at UE involves CRC attachment, code block segmentation, code block concatenation, channel coding, rate matching, data and control multiplexing, and channel interleaver. Propagation delay depends on obstacles (i.e. building, trees, hills etc.) on the way of propagation and the total distance traveled by the RF signal;
- TBackhaul
  is the time for building connections between eNB and the core network (i.e. EPC). Generally, the core network and eNB are connected by copper wires or microwave or optical fibers. In general, microwave involves lower latency while optic fibers come with comparatively higher latency. However, spectrum limitation may curb the capacity of microwave ;
- TCore
  is the processing time taken by the core network. It is contributed by various core network entities such as mobility management entity (MME), serving GPRS support node (SGSN), and SDN/NFV. The processing steps of core network includes NAS security, EPS bearer control, idle state mobility handling, mobility anchoring, UE IP address allocation, and packet filtering;
- TTransport
  is the delay to data communication between the core network and Internet/cloud. Generally, distance between the core network and the server, bandwidth, and communication protocol affect this latency

$$T_{Radio} = t_Q + t_{FA} + t_{tx} + t_{bsp} + t_{mpt} \quad (2)$$

- tQ is the queuing delay which depends on the number of users that will be multiplexed on same resources;
- tFA is the delay due to frame alignment which depends on the frame structure and duplexing modes (i.e., frequency division duplexing (FDD) and time division duplexing (TDD));
- ttx is the time for transmission processing, and payload transmission which uses at least one TTI depending on radio channel condition, payload size, available resources, transmission errors and retransmission;
- tbsp is the processing delay at the base station;
- tmpt is the processing delay of user terminal. Both the base station and user terminal delay depend on the capabilities of base station and user terminal (i.e., UE), respectively.

Assume the packet arrival of UEs follows a Poisson process with arrival rate λ. It is assumed that there are load balancers in each layer for each slice to divide the incoming traffic to VNFs equally [5]–[7]. Suppose the baseband processing of each VNF is depicted as an M/M/1 processing queue. Each packet is processed by one of the VNFs of a slice. So, the mean delay of slice, modeled as M/M/1 queue, is formulated as follow, respectively

$$d = \frac{1}{\mu - \alpha/M} \quad (3)$$

where $1/\mu$ is the mean service time. Besides, α is the arrival rate which is divided by load balancer before arriving to the VNFs. The arrival rate of each VNF in each layer of the slice s is α/M. In addition, d_{tr} is the transmission delay on the wireless link. The arrival data rate of wireless link is equal to the arrival data rate of load balancers for each slice [5]. Moreover, it is assumed that the service time of transmission queue has an exponential distribution with mean $1/(R_{tot})$ and can be modeled as a M/M/1 queue [5]–[8].

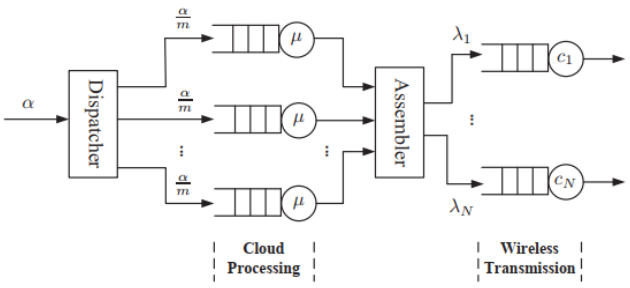

Fig. 5: delay of a packet [5]

Therefore, the mean delay of the transmission layer is

$$d_{tr} = \frac{1}{R_{tot} - \alpha}; \tag{4}$$

where, R_{tot} is the total achievable rate of each slice that is mapped to specific service. Mean delay of each slice is obtained as below.

$$D_s = d + d_{tr} \tag{5}$$

6) **Assume a system model for the paper**

Suppose there are S slices Serving V services. Each Service $v \in \{1, 2, ..., V\}$ consists of U_v UEs that require certain service. Each slice $s \in \{1, 2, ..., S\}$ contains VNFs. There are processing layer in the system, represented with a VNF.

Assume we have M VNFs for processing data. Each VNF belongs to one or more slices. So, in the s^{th} slice, there are M_s VNFs. The VNFs have the computational capacity that is equal to μ.

Each VNF requires physical resources that contain memory, storage and CPU. Let the required resources for VNF f in slice s is represented by a tuple as

$$\bar{\Omega}_s^f = \{\Omega_{M,s}^f, \Omega_{S,s}^f, \Omega_{C,s}^f\}, \tag{6}$$

where $\bar{\Omega}_s^f \in \mathbb{C}^3$ and $\Omega_{M,s}^f, \Omega_{S,s}^f, \Omega_{C,s}^f$ indicate the amount of required memory, storage, and CPU, respectively. Moreover, the total amount of required memory, storage and CPU of all VNFs of a slice is defined as

$$\bar{\Omega}_{\mathfrak{z},s}^{tot} = \sum_{f=1}^{M_s} \bar{\Omega}_{\mathfrak{z},s}^f \quad \mathfrak{z} \in \{M, S, C\}. \tag{7}$$

Also, there are D_c data centers (DC), serving the VNFs. Each DC contains several servers that supply VNF requirements. The amount of memory, storage and CPU is denoted by τ_{M_j}, τ_{S_j} and τ_{C_j} for the j^{th} DC, respectively

$$\tau_j = \{\tau_{M_j}, \tau_{S_j}, \tau_{C_j}\},$$

In this system model, the assignment of physical DC resources to VNFs is considered. Let $y_{s,d}$ be a binary variable indicating whether the d^{th} DC is connected to the VNFs of s^{th} slice or not. Assume the power consumption of baseband processing at each DC d that is connected to VNFs of a slice s is depicted as $\phi_{s,d}$. So the total power of the system for all active DCs that are connected to slices can be represented as

$$\phi_{tot} = \sum_{s=1}^{S} \sum_{d=1}^{D_c} y_{s,d}(t)\phi_{s,d}.$$

In addition, in each time the deploying a new VNF on a VM needs more power.

$$\phi_{diff} = \sum_{s=1}^{S} \sum_{d=1}^{D_c} [y_{s,d}(t) - y_{s,d}(t-1)]^+ \phi_{s,d}^{new}.$$

Also, a cost function for the placement of VNFs into DCs is defined as

$$\psi_{tot} = \phi_{tot} + \phi_{diff} \tag{8}$$

where, ν is a design variable to value between the first term of (8) which is the total power consumption of physical resources and the second term that is shown the amount of admitted slices to have physical resources.

$$\min_{\boldsymbol{y}} \quad \psi_{tot}(\boldsymbol{Y}) \tag{9a}$$

$$\text{s. t.} \quad \sum_{d=1}^{D_c} \sum_{v=1}^{V} y_{s,d}(t) \geq 1 \forall s, \tag{9b}$$

$$\sum_{s=1}^{S} y_{s,d}(t)\bar{\Omega}_{\mathfrak{z},s}^{tot} \leq \tau_{\mathfrak{z}_d} \forall d, , \forall \mathfrak{z} \in \mathcal{E}; \tag{9c}$$

REFERENCES

[1] (2020) Deep reinforcement learning: Guide to deep q-learning. [Online]. Available: https://www.mlq.ai/deep-reinforcement-learning-q-learning/

[2] A. Gholamhosseinian, A. Khalifeh, and N. Zare Hajibagher, "Qos for multimedia applications with emphasize on video conferencing," 2011.

[3] (2012) Game theory and multi-agent reinforcement learning.

[4] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A survey on low latency towards 5g: Ran, core network and caching solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3098–3130, 2018.

[5] J. Tang, W. P. Tay, T. Q. Quek, and B. Liang, "System cost minimization in cloud ran with limited fronthaul capacity," *IEEE Transactions on Wireless Communications*, vol. 16, no. 5, pp. 3371–3384, 2017.

[6] P. Luong, C. Despins, F. Gagnon, and L.-N. Tran, "A novel energy-efficient resource allocation approach in limited fronthaul virtualized c-rans," in *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*. IEEE, 2018, pp. 1–6.

[7] P. Luong, F. Gagnon, C. Despins, and L.-N. Tran, "Joint virtual computing and radio resource allocation in limited fronthaul green c-rans," *IEEE Transactions on Wireless Communications*, vol. 17, no. 4, pp. 2602–2617, 2018.

[8] K. Guo, M. Sheng, J. Tang, T. Q. Quek, and Z. Qiu, "Exploiting hybrid clustering and computation provisioning for green c-ran," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 4063–4076, 2016.