Self-Competence Learning for Bandwidth Slicing and Confidential Computing Requirement

Yen-Hung Chen

1Department of Information Management, National Taipei University of Nursing and Health Sciences, Taiwan, R.O.C.

Corresponding author: en-Hung Chen (e-mail: pplong@gmail.com).

This work was supported by the Ministry of Science and Technology of R.O.C. for financially supporting this research under contract no. MOST 110-2221-E-227-001.

ABSTRACT This work proposes a Self-Competence Learning bandwidth slicing method (SCL) to determine the quantity and position of each slice’s spectrum resources considering the channel diversity and varying bandwidth requirements. SCL obtains high throughput in the following aspects: (1) SCL provides artificial intelligence (AI) based functionality to determine the quantity and position of the spectrum resource, (2) SCL can be trained solely by self-competence without any assistance or labeling by manual operations, and (3) SCL provides confidentiality without compromising on training performance in collaborative learning environment. Simulation results demonstrate SCL improves the 10% overall throughput and saving up 25% bandwidth resource for AI model merging.

INDEX TERMS channel allocation, confidential computing, federal learning, network slicing.

I. INTRODUCTION

Network bandwidth slicing enables the 5G/B5G service providers to accommodate heterogeneous use cases over a common underlying physical telecommunication infrastructure, such as the enhanced mobile broadband (eMBB), ultra-reliable and low latency communications (URLLC), and massive machine type communications (mMTC) [1]. In order to maximize the spectrum efficiency (SE) and guarantee superior quality of experience (QoE), the service providers have to intelligently allocate sufficient spectrum to each slice according to the channel diversity and varying bandwidth requirement of the service types in different slices. This is called inter-slicing resource management problem [2], which is a maximum-matching problem based on bipartite graph theory to allocate slices to channels [3-4], of which the problem statement:

Given a bipartite graph \( G = (S, C; E, R) \) with \( I \) slice vertices (\( S \)), \( J \) channel vertices (\( C \)), and each edge \( e \in E \) has a nonnegative modulation coding rate \( w_{i,j} \in MCS \) where \( i \in S \) and \( j \in C \). The target is to find the matching \( M \) that assigns each channel \( j \) to slice \( m \), in order to maximize the value function, \( e.g., \) overall throughput \( (M) \), under the constrain that the allocated bandwidth to each slice \( i \) will not exceed its requirements \( r_i \in R \).

Various works provide valuable knowledge on building bandwidth slices, however, there exist two issues. First, the conventional spectrum resource allocation method consists of two phases: first determine the quantity of bandwidth resource each slice would get, and then determine the position of the spectrum. Qi et al [2] apply deep reinforcement learning to decide the quantity of bandwidth to each slice according to the network status. Li et al [5] consider the user mobility and traffic patterns and adopt long/short-term memory to predict the possible demanding quantity of bandwidth of each slice. Hua et al [6] use a generative adversarial network to stabilize artificial intelligence (AI) training procedures. These works reach a great achievement, however, they merely determine the quantity of spectrum without indicating the position of the spectrum.

Second, the AI-based method should cooperate existed or traditional methods, instead of simply replacing them. That is, besides the general models [2-6] to maximize the throughput, there are also economic model [7] to maximize the profit of service provider, robustness model [8] to achieve high availability of telecommunication network, and confidential...
computing model [9] to provide confidentiality of private data without compromising on training performance. Replacing the existed methodology embedded in the current system may be costly and impractical, and the proposed AI-based method should be capable of cooperative learning and consider the privacy issues of confidential computing requirements.

This study proposed a Self-Competence Learning bandwidth slicing method (SCL) that applies one-step functionality, instead of two-phase allocation, to determine the quantity and position of spectrum resources to each slice considering the channel diversity and varying bandwidth requirements. SCL differs from previous works [2-8] in several aspects. First, SCL is the first work, to our best knowledge, that applies AI-based functionality to simultaneously determine the quantity and position of the spectrum resource. Second, SCL is trained solely by self-competence reinforcement learning without any assistance by manual operations: building multiple AI agents of a single neural network structure, starting from random strategy making, competing, and learning from each other. Third, SCL enables service providers to collaboratively train a shared model without knowing each other’s local data in a confidential computing environment. The detailed design regarding the End-to-End Function Design and the Self-Competence Training Procedure is provided in the following subsection.

II. THE SCL DESIGN

SCL applies one-step allocation to simultaneously determine the quantity and position of spectrum resources to each slice considering the channel diversity and varying bandwidth requirements. The one-step allocation of SCL adopts a self-competence function design to transform the INPUT directly to the OUTPUT by AI-based methodology. On the other hand, SCL adopts a self-competence learning procedure by creating multiple AIs to compete and learn each other to maximize the overall throughput without any assistance by manual operations. Self-competence learning is also compatible with collaboratively training without knowing each other’s local data in a confidential computing environment. The detailed design regarding the End-to-End Function Design and the Self-Competence Training Procedure is provided in the following subsection.

A. ABBREVIATIONS AND ACRONYMS

The one-step allocation of SCL adopts end-to-end function consisting of four basic components including INPUT, Forward Function (FF), Reward Value (RV), and OUTPUT. Fig.1 shows three steps of SCL to transform the INPUT to the OUTPUT.

First step is that, as shown on the left side of Fig.1, according to the inter-slicing resource management problem statement described in the introduction section, we have the INPUT including the network status $MCS$ and the bandwidth requirements $R$:

$$MCS = \begin{bmatrix} w_{1,1} & \cdots & w_{1,J} \\ \vdots & \ddots & \vdots \\ w_{I,1} & \cdots & w_{I,J} \end{bmatrix}, \quad R = [r_1, \ldots, r_i, \ldots, r_J].$$

![FIGURE 1. SCL One Step End-to-End Function Design](image)
Second, as shown in the middle of Fig. 1, SLR applies Forward Function (FF) to transfer the INPUT, i.e., MCS and R, to the Reward Value (RV). The Reward Value $v_{ij} \in RV$ is the expected throughput of each possible slice-channel matching denoted by

$$RV = \begin{bmatrix} v_{1,1} & \ldots & v_{1,j} \\ \vdots & \ddots & \vdots \\ v_{l,1} & \ldots & v_{l,j} \end{bmatrix}$$

where $v_{ij}$ is the expected throughput while assigning $j$-th channel to $i$-th slice. The FF consists of full-connected, convolution, and pooling operations. As shown in Fig. 1, the input/output provides a point of input/output with an identity activation function, e.g., min, max, category index. The kernel applies a parameterized weight to the input or output. The hidden cell is a kernel that attenuates the input and forms a foundation to generate the output. The convolution or pooling cell is to aggregate and extract the features from the inputs. The parameters in the FF will be adjusted by the loss function, gradient operation, and corresponding self-competence strategies described in the next subsection.

SLC then applies argmax() operation to find the matching $m_j \in M$ in order to maximize the value function, e.g., overall throughput $\theta(M)$. That is, SLR explores the index $m_j$ owning the highest reward value along each channel $j$ by

$$m_j = \text{argmax}(v_{ij}, v_{ij}, \ldots, v_{ij})$$

The $m_j$ means channel $j$ is assigned to slice $m_j$ with highest expected throughput located on channel $j$. Then SLR finds all channel-slice matching $m_j \in M$ denoted by

$$M = [m_1, \ldots, m_j, \ldots, m_l].$$

Since each slice $m_j$ owns the highest expected throughput for channel $j$, the SLR infers the matching $m_j \in M$ provides maximized throughput and treats it as the OUTPUT.

Here using a simple example to demonstrate how SCL transforms the INPUT, i.e., MCS and R, to RV. At first, SCL flattens and combines MCS and R into a one-dimension input array $IA = [w_{i,1,1}, \ldots, w_{i,j,s}, n_1, \ldots, n_l]$. Then, given the FF using only one full-connected operation, SCL applies fully-connected operation aggregates information from IA that matters the most and then generates RV by (2):

$$RV = \text{Reshape}(IA \cdot W^{Fc} + B^{Fc}, I, J)$$

where $W^{Fc} = \begin{bmatrix} d_{1,1} & \ldots & d_{L \times J} \\ \vdots & \ddots & \vdots \\ d_{L \times J+1,1} & \ldots & d_{L \times J+1 \times J} \end{bmatrix}$ is the weight vector, $B^{Fc} = \begin{bmatrix} b_{1,1} & \ldots & b_{1,J} \\ \vdots & \ddots & \vdots \\ b_{L \times J+1,1} & \ldots & b_{L \times J+1 \times J} \end{bmatrix}$ is the bias vector.

The Equation (2) first executes the dot operation ($IA \cdot W^{Fc} + B^{Fc}$) to acquire a one-dimension array, and then applies Reshape function to reshape the one-dimension array to two-dimension $I \times J$ array. In addition, $W^{Fc}$ and $B^{Fc}$ will be iteratively adjusted in the training procedure to ensure the channel-slice matching $m_j \in M$ can achieve the highest throughput.

The design of Forward Function can be adjusted to balance the computation complexity and performance. The more complex the operation design, the higher throughput it achieves. However, the training time and computation time are getting higher for a deeper network.

In addition, the FF design of SCL can cooperate with the previous works [2-8] by setting the bandwidth requirements to the quantity scheduled for each slide.

### B. Self-Competence Learning Design

The self-competence learning strategy of SCL is derived from the natural selection process. That is, as shown in Fig. 2, SCL adopts three phases to ensure multiple AIs compete/learn each other and then iteratively evolve from current generation of models into a new generation of the best model which can provide maximizes throughput.

**FIGURE 2.** SCL Self-Competence Training Procedure

For the first phase, SCL randomly generates multiple individual Forward Functions $f^k \in FF$, and then uses the same samples to compete and learn from each other:

Step 1: For the competing, each $f^k$ computes its own $RV^k$ and $M^k$ according the same input $R$ and MCS.
Step 2: SCL then pick up the $f_{\text{max}}$ with the highest throughput $\theta(M_{\text{max}})$. 

Step 3: SCL then asks the each $f_k$ except $f_{\text{max}}$ applies gradient function, e.g., Adam algorithm [10], based on the loss function denoted by (3):

$$
\text{loss}(RV^k, RV^{max}) = \text{mean}(|RV^k - RV^{max}|^2) \tag{3}
$$

where

$$
|RV^k - RV^{max}|^2 = \left[ (v_{11}^k - v_{11}^{max})^2 \ldots (v_{IJ}^k - v_{IJ}^{max})^2 \right].
$$

Therefore, the $\text{loss}(RV^k, RV^{max})$ is the average value of $(v_{ij}^k - v_{ij}^{max})^2$.

For the second phase, after sufficient competing and learning iterations of the first phase, SCL applies the elitism selection to allow the best FF from the current generation to carry over to the next generation, and to eliminate the remaining FF.

The third phase generates new FF and replaces the eliminated FF by mutation, cloning, and meiosis of currently chosen elites FF, as shown in Fig. 3. This phase attempt to change the heritable traits of the current FF population over generations to get better parameters and allocation strategies with partial reproduction are used.

For changing the heritable traits, SCL applies mutation to maintain genetic diversity from one generation of a population of chromosomes, i.e., the hidden kernel of SCL, to the next. As shown in the left part of Fig. 3, the mutation of SLR uses Gaussian distributed random value to the weight of the randomly chosen hidden kernel. For the partial reproduction operation, SCL combines the genetic information of two elite parents (FFs) to generate new offspring FF. The combination process is similar to meiosis which is a sexually-reproducing organism used to produce the gametes. As shown in the middle part of Fig. 3, SCL randomly chooses one hidden kernel from the elite FFs and builds a new generation of FF.

After the third phase of generating a new generation, SCL then goes to the first phase and then iteratively evolves into a new generation of the best FF.

In addition, the meiosis design of SLC is similar to and is capable to cooperate with the federal learning method such as AvgFed [9] methodology. Furthermore, the meiosis design can also use partial parameters, instead of all parameters, from other FF to save the bandwidth for model sharing and merging in a confidential computing environment.

III. PERFORMANCE EVALUATION

Simulations were performed to compare the proposed SCL algorithm with non-AI based method, i.e., Hungarian Algorithm based method (HA) [3,11], in terms of total throughput, since there is no AI-based method to determine the quantity and position of the spectrum simultaneously. We also evaluate the performance of SCL in a confidential computing environment especially for different parameters sharing strategies in order to save the bandwidth of model merging.

The simulator was extended from the ns2 VANET and OFDMA simulator [12-13]. The simulation environment was a 10MHz LTE system with three types of services (i.e., eMBB, URLLC, mMTC), with 300 UEs and three base stations (BSs) capable of sharing models in a confidential computing environment. The first BS applies Hungarian Algorithm to allocate bandwidth resources to each slice. The second one adopts SCL with 8 FFs for self-competence learning, 4 FFs owning lower throughput will be eliminated and replaced per generation. The third BS is equipped SCL
with only one FF to receive a model from the second BS for performance evaluation of SCL confidential computing. This means the third BS merely builds its own FF by acquiring the parameters of FF from the second BS without knowing the samples trained by the second BS. According to the received SNR, various MCSs were used, including QAM1/2, 16QAM1/2, 16QAM3/4, 256QAM1/2, 256QAM3/4.

A. Performance Comparison between HA and SCL
To investigate the effect of average requested bandwidth of each slice on total throughput, the aggregated data arrival rate of each slice was varied from 4 Mbps to 64 Mbps, and the number of slices are 10 in this scenario.

![Graph](image)

**FIGURE 4. SCL Self-Competence Training Procedure**

Fig. 4(a) presents the total throughput achieved using HA and SCL. SCL outperforms HA with about 9.3% improvement when the requested bandwidth increases for two main reasons. First, HA does not consider the requested bandwidth from each slice, and then some slice with the best channel quality will acquire the largest spectrum resources, even this slice does not have so much requested bandwidth and leave these oversupplying resources to be unused. Second, SCL, on the other hand, uses the channel quality and requested bandwidth from each slice as input, and applies self-competence methodology to ensure the AI iteratively evolves into the model that can provide the highest overall throughput.

The effects of the number of slices on the total throughput were also investigated. The number of connections was varied from 3 to 10 with the same overall data arrival rates, i.e., 280 Mbps. Fig. 4(b) reveals that SCL outperforms HA with about 7.3 to 9.3% improvement along with the increasing number of slices. Under the same overall data rate, the larger the number of slices, the smaller the bandwidth requested by each slice and the fewer the spectrum resource area of each slice. A slice with fewer spectrum resources provides SCL with a higher probability to place the slice on high-quality channels.

A. Performance Evaluation for Confidential Computing
To evaluate the meiosis design of SLC is capable to cooperate with the federal learning such as AvgFed [9] methodology and, furthermore, and can save bandwidth for model sharing/merging in confidential computing environment. The simulation environment settings are 10 slices with overall 280 Mbps aggregated data arrival rate.

Fig 5 demonstrates that using AvgFed methodology needs 8 model exchange times to achieve 234 Mbps overall throughput, which is close to the performance of second BS and ensure that the SCL model is capable of model sharing for confidential computing environment.

We furthermore conduct an experiment that the second BS randomly selects partial parameters (from 25% to 75% parameters of the second BS’s FF) and shares them with the third BS. This experiment attempts to observe that SCL only required partial parameters of sharing to achieve the similar performance of complete parameter sharing. Fig 5 reveals that using only 75% parameters (AvgFed 75%) can achieve 231 Mbps after 8 exchange times and 234 Mbps at 9th exchange time. This means SCL meiosis design of SLC can achieve similar performance of full model sharing of FedAvg by saving 25% bandwidth of model sharing/merging procedure.

![Graph](image)

**FIGURE 5. Effects of different AvgFed compression rate on overall throughput**

IV. CONCLUSION
This study introduces a Self-Competence Learning bandwidth slicing method (SCL) that applies one-step functionality, instead of two-phase allocation, to determine the quantity and position of spectrum resources to each slice considering the channel diversity and varying bandwidth requirements. The simulation results reveal that SCL provides 10% improvement of overall throughput than the previous works. SCL also achieves similar performance by saving up to 25% bandwidth resource for model sharing and merging procedures in a confidential computing environment.
REFERENCES

[1] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, “Network slicing in 5G: Survey and challenges,” *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 94–100, May 2017.

[2] C. Qi, Y. Hua, R. Li, Z. Zhao, and H. Zhang, “Deep Reinforcement Learning With Discrete Normalized Advantage Functions for Resource Management in Network Slicing,” *IEEE Commun. Lett.*, vol. 23, no. 8, pp. 1337–1341, Aug. 2019.

[3] E. J. dos Santos, R. D. Souza, J. L. Rebelatto, and H. Alves, “Network Slicing for URLLC and eMBB With Max-Matching Diversity Channel Allocation,” *IEEE Commun. Lett.*, vol. 24, no. 3, pp. 658–661, Mar. 2020.

[4] Y. Hua, R. Li, Z. Zhao, X. Chen, and H. Zhang, “GAN-Powered Deep Distributional Reinforcement Learning for Resource Management in Network Slicing,” *IEEE J. Select. Areas Commun.*, vol. 38, no. 2, pp. 334–349, Feb. 2020.

[5] R. Li, C. Wang, Z. Zhao, R. Guo, and H. Zhang, “The LSTM-Based Advantage Actor-Critic Learning for Resource Management in Network Slicing With User Mobility,” *IEEE Commun. Lett.*, vol. 24, no. 9, pp. 2005–2009, Sep. 2020, doi: 10.1109/LCOMM.2020.3001227.

[6] R. Su, D. Zhang, R. Venkatesan, Z. Gong, C. Li, F. Ding, F. Jiang, and Z. Zhu, “Resource Allocation for Network Slicing in 5G Telecommunication Networks: A Survey of Principles and Models,” *IEEE Network*, vol. 33, no. 6, pp. 172–179, Nov. 2019.

[7] R. Su, D. Zhang, R. Venkatesan, Z. Gong, C. Li, F. Ding, F. Jiang, and Z. Zhu, “Resource Allocation for Network Slicing in 5G Telecommunication Networks: A Survey of Principles and Models,” *IEEE Network*, vol. 33, no. 6, pp. 172–179, Nov. 2019, doi: 10.1109/MNET.2019.1900024.

[8] Y.-H. Chen, Y.-C. Lai, P.-T. Jan, T.-Y. Tsai, “A Spatiotemporal-Oriented Deep Ensemble Learning Model to Defend Link Flooding Attacks in IoT Network,” *Sensors*, vol. 21, no. 4: 1027, Feb. 2021.

[9] J. Li, X. Shen, L. Chen, and J. Chen, “Bandwidth Slicing to Boost Federated Learning Over Passive Optical Networks,” *IEEE Commun. Lett.*, vol. 24, no. 7, pp. 1492–1495, Jul. 2020.

[10] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[11] L.-K. Chen, P.-T. Jan, Y.-H. Chen, R.-Z. Hung, and Y.-J. Lee, “A MAP Overhead Aware Two-Dimensional OFDMA Burst Construction Algorithm,” *Applied Sciences-Base* (ISSN 2076-3417), vol. 9, no. 2, 2019.

[12] Y.-H. Chen, Y.-C. Lai, C.-N. Lai, and P.-T. Jan, “Modeling Machine to Machine Vehicular Safety Communication,” Proceeding of 19th International Conference on Circuits, Systems, Communications and Computers, (CSCC 2015), Zakynthos Island, Greece, July 16-20, 2015.

[13] Y.-C. Lai and Y.-H. Chen, “Designing and Implementing an IEEE 802.16 Network Simulator for Performance Evaluation of Bandwidth Allocation Algorithms,” *High Performance Computing and Communications, South Korea*, pp.432-437, 2009.

Yen-Hung Chen
Associate Professor,
Department of Information Management, National Taipei University of Nursing and Health Sciences, Taiwan. Research focus: National policy making, Business strategies planning and evaluation, Networking protocol design and efficiency estimation.