DRL-Enabled RSMA-Assisted Task Offloading in Multi-Server Edge Computing

Tri-Hai Nguyen*, Heejae Park*, Mucheol Kim†, and Laihyuk Park*
*Department of Computer Science and Engineering, Seoul National University of Science and Technology, Seoul, Korea
†School of Computer Science and Engineering, Chung-Ang University, Seoul, Korea
Email: haint93@seoultech.ac.kr, prkhj98@seoultech.ac.kr, kimm@sw.cau.ac.kr, lhpark@seoultech.ac.kr

***Abstract*—The growing demand for efficient and reliable wireless communication has fueled interest in Rate-Splitting Multiple Access (RSMA) as an advanced multiple access technique for future networks. Simultaneously, Multi-Access Edge Computing (MEC) has become a transformative solution for addressing emerging applications' latency and computing challenges. This study explores the integration of RSMA and MEC to enable simultaneous offloading of users' tasks to multiple MEC servers. We formulate a computation offloading problem to minimize the delay experienced by all users within the RSMA-aided multi-MEC server environment. To tackle this problem, we employ Deep Deterministic Policy Gradient (DDPG), a deep reinforcement learning technique known for its effectiveness in dynamic environments. Simulation results validate the superior performance of the DDPG-based approach compared to conventional methods.**

***Index Terms*—Deep reinforcement learning, multi-server edge computing, rate-splitting multiple access, task offloading.**

I. Introduction

Emerging as a frontrunner for multiple access schemes in the forthcoming sixth-generation (6G) networks is Rate-Splitting Multiple Access (RSMA) [1], [2]. Downlink RSMA entails splitting each user's data into common and private components. While the common message is sent out to all users simultaneously, the private message is exclusively transmitted to the designed recipient. Before decoding their private messages, users decode the common message and partially mitigate the interference caused by other users' transmissions. It enables RSMA to improve spectral efficiency over existing multiple access methods.

Furthermore, Multi-Access Edge Computing (MEC) is a distributed computing model that strategically positions storage and computational services closer to end-users [3], [4]. In MEC networks, resource-intensive and delay-sensitive tasks are offloaded from user devices (UDs) to edge servers for execution, reducing latency and enhancing the overall quality of service (QoS). The advent of 6G networks envisions deploying ultra-dense small-cell networks. This ensures that each UD remains within the wireless communication range of multi-MEC servers. However, the availability of multi-MEC servers and UDs introduces the challenge of making informed computation offloading decisions. While multi-MEC server scenarios have been explored in [3], [4], advanced multiple access schemes remain underexplored. Emerging research has delved into the application of RSMA for resource allocation and task offloading in wireless communication systems [5]–[10]. The studies in [5], [6] examine single-MEC server aerial networks, where users upload tasks over RSMA, and investigate the optimization of task offloading and RSMA parameters. Leveraging RSMA principles, an offloading scheme enables paired two users to unload their computations to a single MEC server simultaneously [7]. RSMA communications are seamlessly integrated into aerial and terrestrial base stations to serve multiple users [8], [9]. Focusing on an RSMA-enabled multi-server edge network, the work in [10] tackled a computation offloading and resource allocation problem by decomposing it into two sub-problems. Due to the dynamic nature of MEC systems, conventional optimization approaches are challenging to use for real-time decision-making and network scaling.

In time-varying environments, DRL has gained prominence as a robust method for addressing intricate decision-making issues [11]–[15]. In DRL, an agent learns to maximize long-term rewards through interactions with the environment. Deep Q-network (DQN) [11] is a well-known DRL technique; however, it is only suitable for tasks with discrete action spaces. Deep deterministic policy gradient (DDPG) [12] utilizes an actor-critic approach to optimize continuous action spaces, making it well-suited for tackling optimization problems in wireless networks [5], [6], [8], [9], [16].

Harnessing the power of downlink RSMA principles, this work enables concurrent computational task offloading from multiple UDs to multiple MEC servers, minimizing overall user-perceived latency. To achieve this, we introduce a DDPG-based approach to the task offloading problem, jointly optimizing task assignment ratios, common rates, and transmit powers. Numerical simulations validate the effectiveness of our proposed DDPG-based solution.

II. Network Model and Problem Definition

A. Network Model

As illustrated in Fig. 1, a multi-server RSMA edge computing network is investigated. It includes a group of MEC servers $\mathcal{M} = \{1, \ldots, m, \ldots, M\}$ and a group of UDs $\mathcal{N} = \{1, \ldots, n, \ldots, N\}$. We use a discrete-time model with a time slot set $\mathcal{T} = \{1, \ldots, t, \ldots, T\}$. Each UD n generates a computational task as $C_n = \{I_n, R_n\}$, where I_n (bits) is input data and R_n (CPU cycles/bit) is required computational resources to process a data bit. Due to their limited energy, all UDs opt to offload their tasks fully. A set of accessible MEC

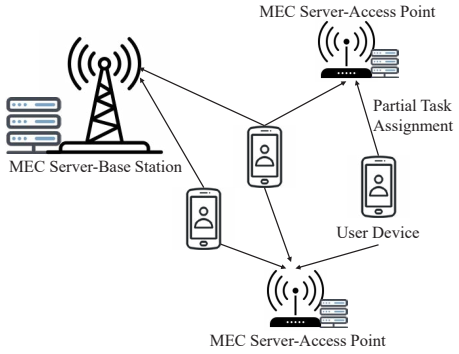

Fig. 1. Multi-MEC server network.

servers is utilized to execute different computation segments. A task assignment ratio of UD n to MEC server m is defined by $\alpha_{n,m} \in [0,1]$ such that $\sum_m^M \alpha_{n,m} = 1, \forall n$.

1) Communication Model: We utilize the one-layer RSMA principles [1] to allow each UD to offload each task to multiple MEC servers concurrently. A message $W_{n,m}$ containing the corresponding offloaded task $\alpha_{n,m} I_n R_n$ (CPU cycles) from UD n planned for MEC server m is decomposed into a common portion $W_{n,m}^c$ and a private portion $W_{n,m}^p$. All $W_{n,m}^c, \forall m$ for scheduled MEC servers from UD n are consolidated and encoded into a common signal $s_{n,0}$ sent to selected MEC servers with power $p_{n,0}$. Each $W_{n,m}^p$ is encoded into a private signal $s_{n,m}$ sent to the corresponding MEC server m with power $p_{n,m}$. Thus, the emitted signal of UD n is

$$x_n = \sqrt{p_{n,0}} s_{n,0} + \sum_m^M \sqrt{p_{n,m}} s_{n,m}. \tag{1}$$

The MEC server m receives the signal from UD n as

$$y_{n,m} = \sqrt{g_{n,m} p_{n,0}} s_{n,0} + \sum_{j=1}^M \sqrt{g_{n,m} p_{n,j}} s_{n,j} + z_{n,m}, \tag{2}$$

where $z_{n,m}$ represents the zero-mean additive white Gaussian noise with variance σ^2 and $g_{n,m}$ denotes the channel gain from UD n to MEC server m determined by a distance-dependent path loss formulation [10], [17]. The achievable rate (bps) for MEC server m decoding $s_{n,0}$ is

$$c_{n,m} = B_n \log_2 \left(1 + \frac{g_{n,m} p_{n,0}}{g_{n,m} \sum_{j=1}^M p_{n,j} + \sigma^2} \right), \tag{3}$$

where B_n (Hz) is the equally allocated bandwidth for the UD n. The channel gains from UD n to offloading MEC servers are assumed to be arranged in an increasing order, such that $g_{n,1} \leq \cdots \leq g_{n,m} \leq \cdots \leq g_{n,M}$ [10], [17]. To guarantee that each MEC server m is able to successfully decode $s_{n,0}$ of UD n, the allocated rates $\beta_{n,m}, \forall m$ (bps) must adhere to the condition as

$$\begin{aligned} \sum_m^M \beta_{n,m} &\leq \min_{m \in \mathcal{M}} c_{n,m} = c_{n,1} \\ &= B_n \log_2 \left(1 + \frac{g_{n,1} p_{n,0}}{g_{n,1} \sum_m^M p_{n,m} + \sigma^2} \right), \end{aligned} \tag{4}$$

where the equality is a result of the ordered channel gains. In addition, given the ordered channel gains, the condition must hold for UD n to guarantee the successful application of the Successive Interference Cancellation (SIC) technique at MEC server m as [10], [17]

$$g_{n,1} p_{n,0} - g_{n,1} \sum_m^M p_{n,m} \geq p_n^{thr}, \tag{5}$$

where p_n^{thr} is the SIC detection threshold. Each MEC server m decodes $s_{n,m}$ with an achievable rate of

$$r_{n,m} = B_n \log_2 \left(1 + \frac{g_{n,m} p_{n,m}}{g_{n,m} \sum_{j=1, j \neq m}^M p_{n,j} + \sigma^2} \right). \tag{6}$$

The total achievable rate from UD n to MEC server m is $r_{n,m}^{tot} = \beta_{n,m} + r_{n,m}$. As a result, the uploading delay (s) of $\alpha_{n,m} I_n$ (bits) transmitted from UD n to MEC server m is

$$T_{n,m}^{up} = \frac{\alpha_{n,m} I_n}{r_{n,m}^{tot}}. \tag{7}$$

2) Computing Model: To simplify the computational resource allocation, we denote $f_{n,m}$ (CPU cycles/s) as the MEC m's computing resources F_m (CPU cycles/s) equally allocated to each UD n [5], [6]. In addition, the downlink transmission delay of the computed output is disregarded due to its insignificant impact on the overall delay [5], [6], [10]. The processing delay (s) incurred by MEC server m processing $\alpha_{n,m} I_n R_n$ (CPU cycles) for UD n is

$$T_{n,m}^{exec} = \frac{\alpha_{n,m} I_n R_n}{f_{n,m}}. \tag{8}$$

Then, the delay endured by UD n due to task uploading and execution at MEC server m is $T_{n,m} = T_{n,m}^{up} + T_{n,m}^{exec}$. Consequently, the delay experienced by UD n for dealing with the task C_n is $T_n = \max\{T_{n,m}\}, \forall m$.

B. Problem Formulation

We jointly optimize task assignment ratios $\boldsymbol{\alpha}_n = \{\alpha_{n,m}, \forall m\}$, allocated common rates $\boldsymbol{\beta}_n = \{\beta_{n,m}, \forall m\}$, and transmit powers $\boldsymbol{p}_n = \{p_{n,0}, p_{n,m}, \forall m\}$ with the goal of minimizing experienced delay as

$$\min_{\boldsymbol{\alpha}_n, \boldsymbol{\beta}_n, \boldsymbol{p}_n, \forall n} \sum_n^N T_n \tag{9a}$$

$$\text{s.t. } \alpha_{n,m} \in [0,1], p_{n,0}, p_{n,m}, \beta_{n,m} \geq 0, \forall n, m, \tag{9b}$$

$$\sum_m^M \alpha_{n,m} = 1, \forall n, \tag{9c}$$

$$\sum_m^M \beta_{n,m} \leq c_{n,1}, \forall n, \tag{9d}$$

$$g_{n,1} p_{n,0} - g_{n,1} \sum_m^M p_{n,m} \geq p_n^{thr}, \forall n, \tag{9e}$$

$$p_{n,0} + \sum_m^M p_{n,m} \leq p_n^{max}, \forall n \tag{9f}$$

$$r_{n,m}^{tot} \geq r_n^{min}, \forall n, m, \tag{9g}$$

where (9b) specifies the permissible domain for the optimization variables, (9c) ensures that each task of UD n is entirely offloaded to the accessible MEC servers, (9d) and (9e) impose the requirements for the SIC technique, (9f) constitutes the maximum transmit power constraint with p_n^{max} being UD n's maximum power budget, and (9g) ensures the QoS with r_n^{min} (bps) being the minimum uplink transmission requirement.

III. DDPG-Based Approach

As the non-convex nature of (9), it is converted into a Markov decision process (MDP), a well-established framework for modeling and solving sequential decision-making challenges. The objective is finding a policy mapping states to actions that maximize the expected long-term reward. The MDP framework consists of the elements as follows.

- State space: The state $s(t)$ encompasses the task information, channel gains, and MEC servers' computational resources allocated to UDs, as represented by $s(t) = \{C_n, f_{n,m}, g_{n,m}, \forall n, m\}$.
- Action space: The action $a(t)$ is defined by the optimization variables as $a(t) = \{\boldsymbol{\alpha}_n, \boldsymbol{\beta}_n, \boldsymbol{p}_n, \forall n\}$. To ensure that the actions adhere to the system's constraints, action normalization techniques can be employed [5], [6], [8].
- Reward function: The reward $r(t)$ is defined by negative system delay as $r(t) = -\sum_n^N T_n(t)$. The agent endeavors to maximize the expected cumulative reward as $R = \max_{a(t)} \mathbb{E}\left[\sum_{t=1}^T \gamma^{t-1} r(t)\right]$ with $\gamma \in [0, 1)$ being a discount factor.

To effectively tackle the MDP model, we employ the DDPG algorithm [12]. This model-free actor-critic DRL technique includes two deep neural networks: an actor network and a critic network. The former generates a deterministic policy that maps states to actions. The latter produces a value function that approximates the expected long-term reward for a given state. To ensure the stability of the training process, DDPG employs experience replay, which involves learning from past experiences rather than just the most recent ones. Additionally, it utilizes two target networks to prevent the actor network from overfitting to the training data. The training procedure is detailed in Algorithm 1. This results in a well-trained actor network that can be effectively employed for online execution.

Algorithm 1 Proposed DDPG-based training framework.

Set up the environment with its specified parameters
Create an actor network $\mu(s|\theta^\mu)$ and a critic network $Q(s, a|\theta^Q)$ with initial parameters θ^μ and θ^Q
Create target networks μ' and Q' with $\theta^{\mu'} \leftarrow \theta^\mu$, $\theta^{Q'} \leftarrow \theta^Q$
Create a replay buffer $\mathcal{R}$
for each episode **do**
  for $t = 1, 2, \ldots, T$ **do**
    Generate action with noise $a(t) = \mu(s(t)|\theta^\mu) + \mathcal{Z}(t)$
    Execute $a(t)$, obtain $r(t)$ and $s(t+1)$
    Archive $\{s(t), a(t), r(t), s(t+1)\}$ in $\mathcal{R}$
    Randomly sample a batch S from $\mathcal{R}$
    Adjust actor network parameter θ^μ according to

$$\nabla_{\theta^\mu} J = \frac{1}{S} \sum_i^S \left(\nabla_a Q(s_i, a_i|\theta^Q)|_{a_i=\mu(s_i)} \nabla_{\theta^\mu} \mu(s_i|\theta^\mu) \right)$$

    Adjust critic network parameter θ^Q according to

$$L = \frac{1}{S} \sum_i^S \left(y_i - Q(s_i, a_i|\theta^Q) \right)^2$$

$$\text{with } y_i = r_i + \gamma Q'\left(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'}\right)$$

    Adjust target networks with soft update coefficient τ by

$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}, \quad \theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$

  end for
end for

IV. Simulation Results

A. Simulation Settings

In the simulations, $N = 10$ UDs are randomly distributed and moving within a 200 m$\times$200 m area. $M = 3$ MEC servers with $F_m = 10$ GHz are located in the area. The channel gains are determined using the distance-based path loss model $128.1 + 37.6 \log_{10}(d_{n,m})$, where $d_{n,m}$ (km) represents the distance between UD n and MEC server m [10]. The shadow fading standard deviation is 4 dB, $\sigma^2 = -104$ dB, $p_n^{max} = 20$ dBm, $r_n^{min} = 1$ Mbps, $p_n^{thr} = -94$ dBm, and $B_n = 1$ MHz. The tasks are randomly generated with $I_n \in [0.8, 1.2]$ Mbits and $R_n \in [800, 1000]$ CPU cycles/bit.

DDPG uses an actor network with 512 and 128 neurons of two hidden layers and a critic network featuring 1024 and 256 neurons of two hidden layers. In the actor network, only the output layer uses the sigmoid function; all other networks' layers use the rectified linear unit function. There are 300 time steps. The training process involves 2000 episodes, while 10 episodes are allocated for evaluation. Regarding hyperparameters, we used a batch size of 32, a replay buffer capacity of 1×10^5, a discount factor of 0.9, and a soft update coefficient of 0.01. The two networks' learning rate is the same and is determined through experimentation.

B. Simulation Results

First, the DDPG convergence performance is analyzed, focusing on the impact of the learning rate. As shown in Fig. 2, a learning rate of 1×10^{-4} achieves the highest reward compared to other values. While a higher learning rate of 1×10^{-3} leads to faster coverage, it compromises the reward at the end of the training process. Conversely, a low learning rate of 1×10^{-5} results in slower coverage.

Second, since RSMA has demonstrated superior performance compared to other multiple access schemes in various settings [5], [7], [8], [10], we focus on evaluating the effectiveness based on the average task delay metric of different solution methods: DDPG (proposed method), DQN, and Local Search (random action generation with local search improvement). For DQN and Local Search, the action variables are discretized within the allowable ranges. As shown in Fig. 3, DDPG outperforms DQN and Local Search. This superior

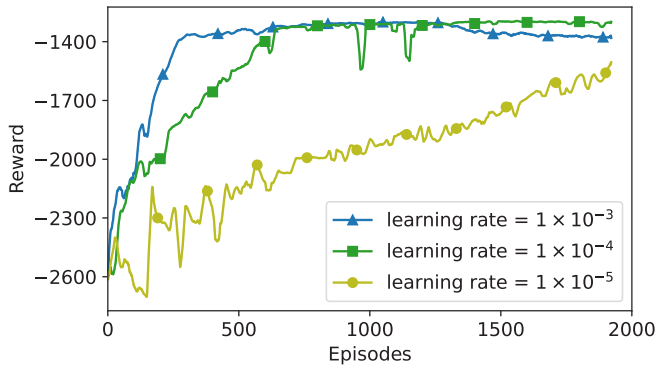

Fig. 2. Influence of learning rate on the training stability of DDPG.

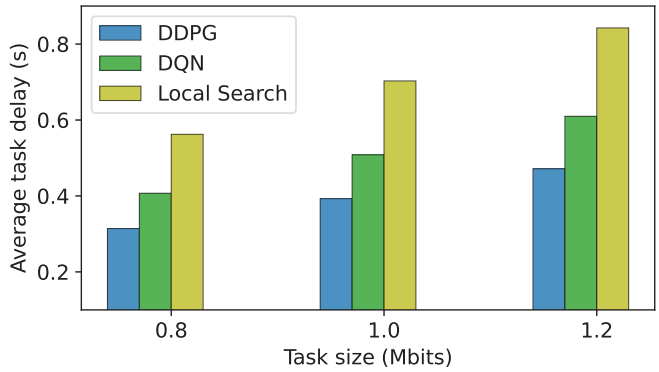

Fig. 3. Performance of three methods under different task size parameters.

performance can be attributed to DDPG's continuous action space optimization, while DQN and Local Search are constrained by their discretized action spaces. Additionally, Local Search easily gets stuck in local optima, contributing to its lower performance.

V. Conclusion

This study demonstrated that integrating RSMA and MEC presents a promising solution for addressing emerging applications' latency and computing demands. By employing RSMA to enable concurrent offloading of users' computational tasks to multiple MEC servers, the proposed network architecture minimizes the overall delay experienced by all users. The DDPG-based DRL technique effectively optimized computation offloading in this dynamic environment, outperforming baseline solutions in delay minimization.

Acknowledgment

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2023-RS-2022-00156353) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation). It was also supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1F1A1062881).

References

[1] Y. Mao, B. Clerckx, and V. O. Li, "Rate-splitting multiple access for downlink communication systems: bridging, generalizing, and outperforming SDMA and NOMA," *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, no. 1, May 2018.

[2] B. Clerckx, Y. Mao, E. A. Jorswieck, J. Yuan, D. J. Love, E. Erkip, and D. Niyato, "A primer on rate-splitting multiple access: Tutorial, myths, and frequently asked questions," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 5, pp. 1265–1308, May 2023.

[3] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 856–868, Jan. 2019.

[4] P. A. Apostolopoulos, E. E. Tsiropoulou, and S. Papavassiliou, "Risk-aware data offloading in multi-server multi-access edge computing environment," *IEEE/ACM Transactions on Networking*, vol. 28, no. 3, pp. 1405–1418, Jun. 2020.

[5] T. P. Truong, N.-N. Dao, and S. Cho, "HAMEC-RSMA: Enhanced aerial computing systems with rate splitting multiple access," *IEEE Access*, vol. 10, pp. 52 398–52 409, 2022.

[6] T.-H. Nguyen and L. Park, "HAP-assisted RSMA-enabled vehicular edge computing: A DRL-based optimization framework," *Mathematics*, vol. 11, no. 10, p. 2376, May 2023.

[7] P. Chen, H. Liu, Y. Ye, L. Yang, K. J. Kim, and T. A. Tsiftsis, "Rate-splitting multiple access aided mobile edge computing with randomly deployed users," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 5, pp. 1549–1565, May 2023.

[8] T.-H. Nguyen, L. V. Nguyen, L. M. Dang, V. T. Hoang, and L. Park, "TD3-based optimization framework for RSMA-enhanced UAV-aided downlink communications in remote areas," *Remote Sensing*, vol. 15, no. 22, p. 5284, Nov. 2023.

[9] T. P. Truong, T.-H. Nguyen, A.-T. Tran, S. V.-T. Tran, V. D. Tuong, L. V. Nguyen, W. Na, L. Park, and S. Cho, "Deep reinforcement learning-based sum-rate maximization for uplink multi-user SIMO-RSMA systems," in *Intelligence of Things: Technologies and Applications*, N.-N. Dao, T. N. Thinh, and N. T. Nguyen, Eds. Cham: Springer Nature Switzerland, 2023, pp. 36–45.

[10] M. Diamanti, C. Pelekis, E. E. Tsiropoulou, and S. Papavassiliou, "Delay minimization for rate-splitting multiple access-based multi-server MEC offloading," *IEEE/ACM Transactions on Networking*, 2023.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[12] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *4th International Conference on Learning Representations (ICLR 2016)*, 2016.

[13] T.-H. Nguyen and L. Park, "A survey on deep reinforcement learning-driven task offloading in aerial access networks," in *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, Oct. 2022.

[14] T.-H. Nguyen, H. Park, and L. Park, "Recent studies on deep reinforcement learning in RIS-UAV communication networks," in *2023 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. IEEE, Feb. 2023.

[15] T.-H. Nguyen, H. Park, K. Seol, S. So, and L. Park, "Applications of deep learning and deep reinforcement learning in 6G networks," in *2023 Fourteenth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, Jul. 2023.

[16] T.-H. Nguyen, T. P. Truong, N.-N. Dao, W. Na, H. Park, and L. Park, "Deep reinforcement learning-based partial task offloading in high altitude platform-aided vehicular networks," in *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, Oct. 2022.

[17] Z. Yang, M. Chen, W. Saad, and M. Shikh-Bahaei, "Optimization of rate allocation and power control for rate splitting multiple access (RSMA)," *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 5988–6002, Sep. 2021.