Appendix

Peng Chen[1] and Jixian Zhang[1,2](✉)

[1] School of Information Science and Engineering, Yunnan University
[2] Yunnan Key Laboratory of Intelligent Systems and Computing, Yunnan University
pengchen@stu.ynu.edu.cn, zhangjixian@ynu.edu.cn

1 Network Architecture

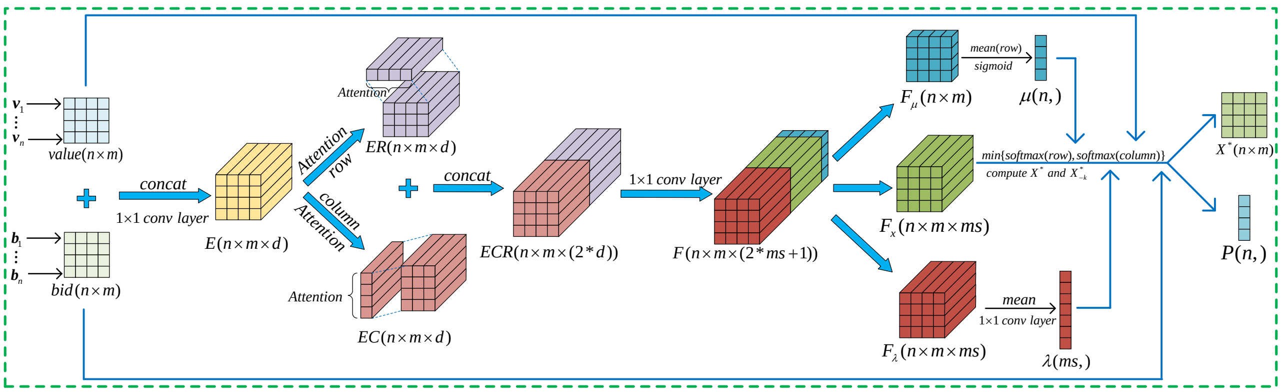

Fig. 1. The network architecture diagram.

In this section, we describe the reverse affine auction network architecture. As shown in Fig.1, the neural network is mainly built based on attention mechanism [2]. The attention mechanism effectively captures high-order feature interactions, making it suitable for obtaining interaction information between users and ROI. Additionally, due to the size of the allocation result set is $|\mathcal{X}| = \sum_{i=0}^{min(n,m)} C_n^i C_m^i A_i^i$, when n and m increase, the number of parameters is too large to be calculated. Therefore, we adopt the approach from [1], using the neural network to generate candidate allocation set instead of $\mathcal{X}$, and set the candidate allocation set size to $menu\ size(ms)$. This network can compute allocation and payment schemes based on the user's value and bidding information.

First, the users' $bid \in \mathcal{R}_{\geq 0}^{n\times m}$ and $value \in \mathcal{R}_{\geq 0}^{n\times m}$ matrix are concatenated, and then a 1×1 convolution operation is then performed to map the UAV value and bid information to d dimensions without changing the input size:

$$E = conv(concat(bid, value)) \in \mathcal{R}^{n\times m\times d}$$

Next, to capture the interaction information between UAVs and ROI, interactions are allowed in both row and column directions using attention mechanisms, and the interaction results are concatenated:

$$ER = Attrntion - row(E) \in \mathcal{R}^{n\times m\times d}$$
$$EC = Attrntion - column(E) \in \mathcal{R}^{n\times m\times d}$$
$$ERC = concat(ER, EC) \in \mathcal{R}^{n\times m\times (2*d)}$$

To facilitate the computation of μ, λ, and the candidate allocation set, a 1×1 convolution operation is used to adjust the third dimension of ERC to $2 * ms + 1$:

$$F = conv(ECR) \in \mathcal{R}^{n\times m\times (2*ms+1)}$$

Finally, F is used to compute the task allocation scheme $X^* \in [0,1]^{n\times m}$ and the payment scheme $P = (p_1, p_2, ..., p_n) \in \mathcal{R}^n_{\geq 0}$. F is divided into three parts: $F_\mu \in \mathcal{R}^{n\times m}$, $F_x \in \mathcal{R}^{n\times m\times ms}$ and $F_\lambda \in \mathcal{R}^{n\times m\times ms}$. These components are used to compute μ, λ, and the candidate allocation set respectively:

$$\mu = Sigmoid(mean(F_\mu, row)) \in (0,1)^n, \quad Sigmoid(x) := \frac{1}{1+e^{-x}} \in (0,1)$$

$$\mathcal{X} = min\{Softmax(F_x, row), Softmax(F_x, column)\} \in [0,1]^{n\times m\times ms}, \quad Softmax(x_j) := \frac{e^{x_j}}{\sum_{j=1}^{m} x_j} \in (0,1)$$

$$\lambda = conv(mean(mean(F_\lambda, column), row)) \in \mathcal{R}^{ms}$$

Based on μ, λ, and the candidate allocation set, the allocation scheme $X^* \in [0,1]^{n\times m}$ and payment scheme $P = (p_1, p_2, ..., p_n) \in \mathcal{R}^n_{\geq 0}$ can be calculated according to the definition of reverse affine auction(this process is non-differentiable, and during training, the $Softmax$ temperature parameter is used for approximation). The hyperparameters of the neural network are provided at https://github.com/ynu-chp/RAA.

2 Experimental Data

We conducted experiments with users $n = 5$ and ROIs $m \in \{2, 4, 6, 8, 10, 12\}$, the specific experimental data is as follows:

Table 1. social welfare($n = 5$)

setting	$m = 2$	$m = 4$	$m = 6$	$m = 8$	$m = 10$	$m = 12$
OPT	1.673	3.228	4.159	4.446	4.613	4.73
RVCG	1.673	3.228	4.159	4.446	4.613	4.73
$\gamma = 1$	1.672	3.202	4.131	4.433	4.604	4.73
$\gamma = 0.5$	1.672	3.214	4.13	4.428	4.607	4.723
$\gamma = 0$	1.671	3.205	4.132	4.431	4.607	4.723

Table 2. VSP utility($n = 5$)

setting	$m = 2$	$m = 4$	$m = 6$	$m = 8$	$m = 10$	$m = 12$
OPT	1.673	3.228	4.159	4.446	4.613	4.73
RVCG	1.258	1.947	0.2909	0.141	0.0875	0.0637
$\gamma = 1$	1.652	3.135	4.04	4.345	4.516	4.641
$\gamma = 0.5$	0.836	1.606	2.067	2.217	2.304	2.362
$\gamma = 0$	0	0	0	0	0	0

Table 3. users utility($n = 5$)

setting	$m = 2$	$m = 4$	$m = 6$	$m = 8$	$m = 10$	$m = 12$
OPT	1.673	3.228	4.159	4.446	4.613	4.73
RVCG	0.415	1.281	3.869	4.305	4.526	4.667
$\gamma = 1$	0.0203	0.0671	0.0913	0.0881	0.0886	0.0897
$\gamma = 0.5$	0.836	1.608	2.063	2.211	2.303	2.361
$\gamma = 0$	1.671	3.205	4.132	4.431	4.607	4.723

References

1. Duan, Z., Sun, H., Chen, Y., Deng, X.: A scalable neural network for dsic affine maximizer auction design. In: Adv. neural inf. proces. syst. vol. 36, pp. 56169–56185. Curran Associates, Inc. (2023)
2. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: Adv. neural inf. proces. syst. vol. 30. Curran Associates, Inc. (2017)