Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 4, October 2023 e-ISSN : 2541-2019
DOI : https://doi.org/10.33395/sinkron.v8i4.12887 p-ISSN : 2541-044X

Multidimensional Knapsack 0-1 Solution With Algorithm Evolution Pso-Ga

Yudistira Arya Sapoetra[1)*], Azwar Riza Habibi[2)]
[1,2)]Institut Teknologi dan Bisnis Asia Malang, Indonesia
[1)]yudistira@asia.ac.id, [2)]riza.bj@gmail.com

Submitted : Aug 15, 2023 | **Accepted** : Aug 22, 2023 | **Published** : Oct 1, 2023

Abstract: This paper develops the particle swarm optimization (PSO) method and uses a genetic algorithm (GA) by changing the distribution of articles in the initialization of the initial position. PSO at this time the search and speed of particles will always go to the best solution so that by narrowing the search area will be faster by updating the best position of PSO. While the Genetic algorithm plays a role to get an expanded search area for PSO solutions by utilizing crossover and mutation in GA. So that GA will expand the range of candidates for the best solution in PSO. From each of the advantages of PSO Update and GA will be combined to get Evolutionary PSO-GA (EVPGA) that can minimize error and speed up computation (itation) in finding the best solution. By using the Multidimensional Knapsack data set, the results of EVPGA get an average speed of 24.9s with an error of 1.49%.

Keywords:
Particle swarm optimization, Genetic Algorithm, Evolusi

INTRODUCTION

Evolutionary computing is a computational technique that uses a process of iteration. In each iteration there is a population. From the population, one best individual will be selected as a local solution. Local solutions from each iteration will become a global solution if there is at least one local solution that is best compared to other local solutions and this solution will become the solution to the problem being solved. The process of obtaining a solution is generally inspired by biological evolutionary mechanisms or by swarm intelligent mechanisms. Some examples of evolutionary computing models that are often used and can still be developed include Genetic algorithm, particle swarm optimization, ant colony optimization, and bee algorithm(Abidin, 2018).

(Hayashida et al., 2019) The 0-1 multidimensional Knapsack problem is a combinatorial problem of selecting items to be stored in a place with capacity constraints. The selection of a small number of items is very easy to do manually, but if the number of items is large and must meet several capacity constraints, it will make a lot of combinatorial choices and difficult to do manually.

in their paper try to analyze some algorithms used to solve the 0-1 Knapsack problem. The 0-1 Knapsack problem is a simplified form of the 0-1 Knapsack multidimensional problem. (Z. Liu, 2020) used Brute Force approach, Dynamic Programming, Greedy algorithm. As a result, all algorithms get the same solution as the exact solution except Greedy algorithm. However, the Greedy algorithm is far superior in terms of operating speed and the amount of memory used to perform its operations while the Genetic algorithm has the longest operating time and the largest amount of memory(K. Wang et al., 2018).

(Gupta, 2018) to use hyper parameters to use Particle Swarm Optimization (PSO) and Bayesian optimization (BO) to get effective initial parameters in running the particle area in the best position. These hyper parameters are used to solve the 0-1 multidimensional Knapsack problem. However, the results obtained are only close to the exact solution for some 0-1 multidimensional Knapsack problems

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 4, October 2023 e-ISSN : 2541-2019
DOI : https://doi.org/10.33395/sinkron.v8i4.12887 p-ISSN : 2541-044X

with small dimensions and get the exact solution for one problem with large dimensions (Jindal & Bansal, 2019).

Knowing the fact that with one algorithm alone the solution obtained is not satisfactory, (Luo et al., 2023) tried to develop the Genetic algorithm (GA) by combining the Genetic algorithm with the Dantzig algorithm to solve the multidimensional Knapsack 0-1 problem (Gu, 2020).

The results obtained show that the combined algorithm has better capabilities than the GA algorithm variant. (X. Wang & Li, 2018) also developed the PSO algorithm by combining the PSO algorithm with a mimetic strategy of profit per weight (p/w) comparison. The results obtained by the new combined algorithm created were able to obtain better results than the basic algorithm (PSO). There are many more combined algorithms that have been made.

Based on previous research on multidimensional Knapsack 0-1 which is used to find the minimum and maximum value of a function, this research will try to be developed again with the evolution of the PSO and GA algorithms to solve the multidimensional Knapsack 0-1 problem as a test problem to find out the optimal value and computation time.

LITERATURE REVIEW

Particle swarm optimization

The PSO algorithm is an optimization algorithm based on swarms. This means that given a herd of animals that are then spread over an area. This area is called the search space (Abdollahi et al., 2021). Therefore, it can be said that the speed of the PSO algorithm to converge to an exact solution is influenced by the initialization of the initial population and the speed of one individual in finding an exact solution so that other individuals in the population can follow in its footsteps (Cui et al., 2020).

$$v_{ik}^{t+1} = w.v_{ik}^{t} + c_1.r_1^{t}.(pb_{ik}^{t} - x_{ik}^{t}) + c_2.r_2^{t}.(gb_{g1}^{t} - x_{ik}^{t})$$

Each iteration will be weighted with random terms, stochastically adjusted cognitive and social acceleration with weights r_1 and r_2 (Shahbandegan & Naderi, 2020). Weights used by utilizing inertial weights $w \in \Re^+$, $r_1 \in [0,2]$, $c_1 \in \Re^+$, $r_2 \in [0,2]$, $c_2 \in \Re^+$. This weight w is called the inertia weight with the lower the coefficient w, the stronger the convergence. While w> 1 then causes particle divergence, this inertia weight causes a balance between exploration and exploration of the best solution. This is directly proportional to c_1 and c_2 the greater the solution value of c_1 and c_2 there is no convergence in PSO (Saire & Singh, 2019). The PSO optimization process in addition to determining the value of *w*, c_1 and c_2 in order to converge quickly, it is necessary to increase the exploration of the search radius for global minimum convergence by updating *w*, c_1 and c_2 in the following equation.

$$w^t = 0.4\frac{(t-N)}{N^2} + 0.4$$

$$c_1^t = -3\frac{t}{N} + 3.5$$

$$c_2^t = 3\frac{t}{N} + 0.5$$

According to the paper by (S. Liu et al., 2019) to determine the standard Particle Swarm Optimization, the best static parameters are $w = 0.72984$ and $c_1 + c_2 > 4$ More precisely $c_1 = c_2 = 2.05$.

Genetic Algorithm

Genetic Algorithm is an algorithm that is still being developed. Genetic Algorithms have a wide variety of possible candidate solutions(Rikatsih & Mahmudy, 2018). So it can also be said that with Genetic algorithms all candidate solutions can be found (search in a wider search space) so that the possibility of getting an exact solution becomes greater. The crossover and mutation operators cause this to happen(Nand & Sharma, 2019).

Research variables and parameters can be divided into four parts because this research involves three algorithms that have variables and parameters, namely the PSO algorithm, Genetic algorithm, EVPGA algorithm (Evolutionary Particle Swarm Optimization Genetic algorithm).

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 4, October 2023 e-ISSN : 2541-2019
DOI : https://doi.org/10.33395/sinkron.v8i4.12887 p-ISSN : 2541-044X

The first step in this research is to analyze the needs according to the needs. The data used in the experiment is taken from the OR-library test data set which can be accessed from the website: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/. The data used are mknap1.text, mknap2.text, mknap3.text, and mknap4.text.

METHOD

At this stage, system design and implementation planning will be carried out to solve the multidimensional Knapsack problem that is currently running. In terms of system and flow, the model can be seen as below:

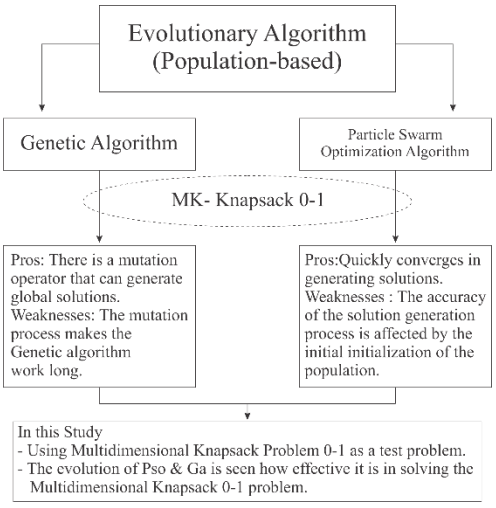

Fig. 1 Research Framework

The first stage will be the evolution of particle swam optimization with genetic algorithm using 0-1 Multidimensional Knapsack operation research data.

Step:

1. Initial initialization of the PSO algorithm, and input of PSO algorithm all parameters. $x_{ik}, v_{ik}, w, c_1, c_2, r_1, r_2, \max iter$, where x_{ik} is the kth candidate solution. The values are randomly generated with the rule.

   $$x_{i(k-1)} = \begin{cases} 1, rand(0,1) \geq 0.5 \\ 0, \quad other \end{cases}$$

   Where $k=1,\ldots$, total population, And Genetic algorithm parameters : α, β.

   $$w^t = 0.4\frac{(t-N)}{N^2} + 0.4$$

   $$c_1^t = -3\frac{t}{N} + 3.5$$

   $$c_2^t = 3\frac{t}{N} + 0.5$$

2. Calculate the fitness value of each individual.

   $$F_k(p,x) = \begin{cases} F_k(p,x), H_k(w_j,x) \leq W_j \\ 0, \quad others \end{cases}$$

   Where $F_k(p,x) = \sum_{i=1}^{n} p_i x_i$ and $H_k(w_j,x) = \sum_{i=1}^{n} w_{ij} x_i$

3. Determine pbest and gbest.

   $$pb_{ik}^{t+1} = \begin{cases} x_{ik}^{t+1}, F_k(p,x_{ik}^{t+1}) \geq F_k(p,pb_{ik}^t) \\ pb_{ik}^t, \quad others \end{cases}$$

   As for gbest, the initial value is *gbest=zeros*(*size*(x_{ik})).

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 4, October 2023 e-ISSN : 2541-2019
DOI : https://doi.org/10.33395/sinkron.v8i4.12887 p-ISSN : 2541-044X

$$gb_{i1}^{t+1} = \begin{cases} pb_{ik}^{t+1}, F_k(p, pb_{i1}^{t+1}) \geq F_k(p, gb_{i1}^{t}) \\ pb_{i1}^{t}, \quad others \end{cases}$$

4. Update the *x* and *v* values of each individual.
$$v_{ik}^{t+1} = w.v_{ik}^{t} + c_1.r_1^{t}.(pb_{ik}^{t} - x_{ik}^{t}) + c_2.r_2^{t}.(gb_{g1}^{t} - x_{ik}^{t})$$
To update the value of *x* sigmoid limiting transformation is used $S(v_{ik}^{t+1})$.
$$S(v_{ik}^{t+1}) = \frac{1}{1+e^{-v_{ik}^{t+1}}} \text{ and } x_{ik}^{t+1} = \begin{cases} 1, rand(0,1) \geq S(v_{ik}^{t+1}) \\ 0, \quad others \end{cases}$$
5. Perform the crossover process on the new individual formed.
$$x_{ik}^{t+1} = \begin{cases} x_{ik}^{t+1}, 0 < rand(0,1) \leq \alpha \\ pb_{ik}^{t}, \alpha < rand(0,1) \leq 2\alpha \\ gb_{i1}^{t}, 2\alpha < rand(0,1) \leq 1 \end{cases}$$
With α is the ratio of the probability of crossover occurring. The value ranges from $0-50\%$ Followed by the mutation process which is done by :
$$x_{ik}^{t+1} = \begin{cases} inv(x_{ik}^{t+1}), rand(0,1) \leq \beta \\ x_{ik}^{t+1}, \quad others \end{cases}$$
Which $inv(x_{ik}^{t+1})$ denotes the binary inverse of x_{ik}^{t+1} and β is the probability of the mutation occurring.
6. Perform the mutation process.
7. Repeating steps 2-6 until the stop condition is met, that is, until the loop value is equal to the maximum.
8. Output: gbest solution.

RESULT

For initial initialization in this study using the weights specified in the following table used for running the results of evolutionary Particle swarm optimization and Genetic Algorithm (EVPGA)(Li & Zhang, 2020).

Table 1. initial initialization

PSO		GA	EVPGA
Population size=30 & maximum iterations =100			
PSO :	$c_1 = c_2 = 2.05$	GA:	Crossover probability = 0.33
	Inersia Weight $(w) = 0.72984$		Mutation probability= 0.05
PSOGA :	$c_1 = c_2 = 2$		Crossover probability = 0.33
	Inersia Weight $(w) = 0.72984$		Mutation probability= 0.05

Results on the first data set to compare GA, PSO & EVPGA (Evolutionary Particle Swarm Optimization Genetic algorithm) algorithms. Shows that EVPGA has a faster convergent value based on iterations and computation time:

Table 2. Comparison of convergent and iteration time

No	Algorithm	GA	PSO	PSOGA	EVPGA
1	Max Solution	8481.7	8706.1	8706.1	8706.1
2	Exact Convergent (iteration)	Not yet converged	46	30	20
3	Computation Time (s)	49.715	45.731	39.76	24.9760

The exact convergent comparison shows that EVPGA converges faster than GA which has not converged until the 100th iteration limit, while PSO with a shorter time than GA gets an exact solution at the 46th iteration. The combined PSOGA obtained an exact solution with the 30th convergent with a

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 4, October 2023
DOI : https://doi.org/10.33395/sinkron.v8i4.12887
e-ISSN : 2541-2019
p-ISSN : 2541-044X

faster time than PSO and GA, while EVPGA from the average comparison of convergence and computation time is far superior to GA, PSO, and combined PSOGA.

Table 3. Comparison of average exact values

	Running Algorithm	1	2	3	4	5	Rata – rata
DATA 1 Exact (8706.1)	GA	8481.7	8530.9	8552.7	8503.7	8603.4	8534.48
	PSO	8704.2	8705.1	8705.4	8706.1	8705.9	8705.34
	PSOGA	8706.1	8706.1	8706.1	8706.1	8706.1	8706.1
	EVPGA	8706.1	8706.1	8706.1	8706.1	8706.1	8706.1
DATA 2 Exact (4015)	GA	3816.7	3786.2	3806.5	3786.5	3812.5	3801.68
	PSO	4015.2	4013.3	4012.6	4014.6	4012.6	4013.6
	PSOGA	4015	4015	4015	4015	4015	4015
	EVPGA	4015	4015	4015	4015	4015	4015
DATA 3 Exact (6120)	GA	5687.5	5649.5	5604	5663	5904	5701.6
	PSO	6120	6120	6118.9	6120	6120	6119.78
	PSOGA	6120	6120.1	6120	6120	6120	6120.02
	EVPGA	6120	6120	6120	6120	6120	6120
DATA 4 Exact (16537)	GA	15877	15837	15397	15887	15891	15777.8
	PSO	15897	15628	15697	15834	15843	15779.8
	PSOGA	15834	15999	15789	15829	15514	15793
	EVPGA	16537	16537	16537	16537	16537	16537

Table 3 shows the overall average solution obtained using different data, the results show that EVPGA on data 1-4 is close to the exact solution. This is shown in Figure 2. Which shows that the comparison of the average solutions from data 1-4.

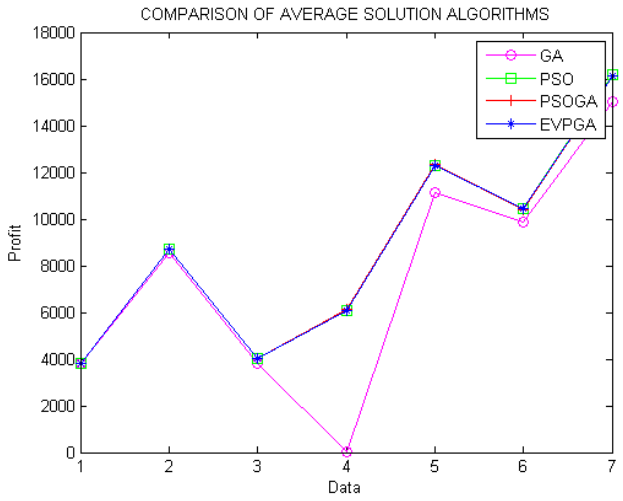

Fig. 2 Solution Comparison

The following graph compares the average computation time of all the data used, and shows that EVPGA is much faster than GA, PSO, and PSOGA.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 7, Number 4, October 2023
DOI : https://doi.org/10.33395/sinkron.v8i4.12887
e-ISSN : 2541-2019
p-ISSN : 2541-044X

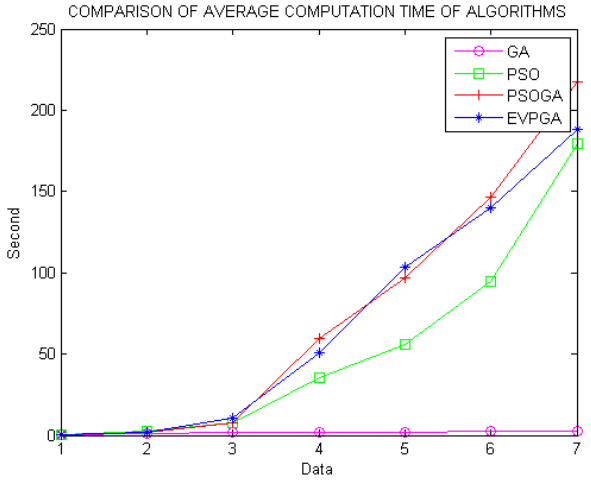

Fig. 3 Comparison of average computation time

The following fig. 4 compares the percentage error from the comparison of GA, PSO, PSO-GA, and EVPGA algorithms using 0-1 Multidimensional Knapsack data. The results show that GA gets 3.94%, PSO = 0.93%, HPSOGA = 1.511% and PSOGA Evolution gets an error of 1.49%.

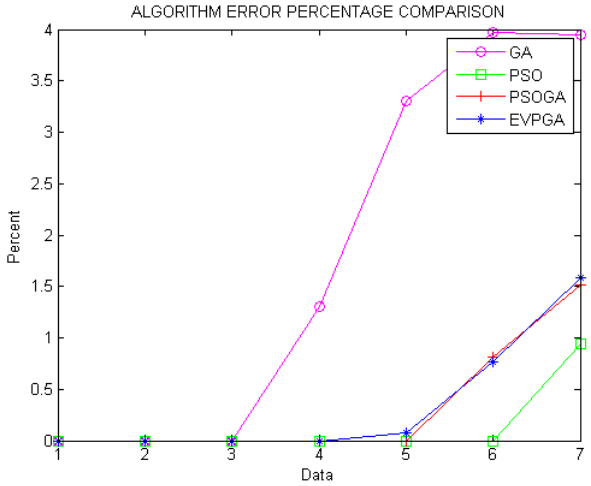

Fig. 4 Comparison of algorithm error percentage

DISCUSSIONS

In this paper, PSO-GA Evolutionary algorithm is used to solve multidimensional Knapsack problem. The main idea of PSO and GA Evolutionary Algorithm is to use GA for global search and PSO for local search. This new algorithm integrates the advantages of both algorithms. In GA algorithm the new population is selected as the surviving population for the new generation. Meanwhile, PSO has a role in the solution search process by modifying the inertia weights and search area to shorten the computation time. However, GA can affect the stability of the resulting solution, so it is necessary to further study the convergence of EVPGA, to prevent being trapped in local optimality that affects the computation time and error rate.

CONCLUSION

EVPGA method is able to get effective results in computation time and faster iterations compared to PSO, GA, PSO-GA methods. In the comparison of errors EVPGA is better than GA and HPSOGA by getting 1.49% but PSO without the combination of other algorithms gets a better error of 0.93%. however, PSO in computation time and iterations is longer than EVPGA. This is because the initial rate in EVPGA is determined with a minimum limit and the search area is minimized.

*name of corresponding author

REFERENCES

Abdollahi, S., Deldari, A., Asadi, H., Montazerolghaem, A., & Mazinani, S. M. (2021). Flow-Aware Forwarding in SDN Datacenters Using a Knapsack-PSO-Based Solution. *IEEE Transactions on Network and Service Management*, *18*(3), 2902–2914. https://doi.org/10.1109/TNSM.2021.3064974

Abidin, D. (2018). A Hybrid Genetic—Differential Evolution Algorithm (HybGADE) for a Constrained Sequencing Problem. *2018 International Conference on Artificial Intelligence and Data Processing (IDAP)*, 1–6. https://doi.org/10.1109/IDAP.2018.8620825

Cui, Y., Qiao, J., & Meng, X. (2020). Multi-stage multi-objective particle swarm optimization algorithm based on the evolutionary information of population. *2020 Chinese Automation Congress (CAC)*, 3412–3417. https://doi.org/10.1109/CAC51589.2020.9326666

Gu, H. (2020). Optimal Lagrangian Multipliers for the Multidimensional Knapsack Problem: A Bayesian Optimisation Approach. *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, 3149–3155. https://doi.org/10.1109/SSCI47803.2020.9308380

Gupta, I. K. (2018). A hybrid GA-GSA algorithm to solve multidimensional knapsack problem. *2018 4th International Conference on Recent Advances in Information Technology (RAIT)*, 1–6. https://doi.org/10.1109/RAIT.2018.8389069

Hayashida, T., Nishizaki, I., Sekizaki, S., & Takamori, Y. (2019). Improvement of Two-swarm Cooperative Particle Swarm Optimization Using Immune Algorithms and Swarm Clustering. *2019 IEEE 11th International Workshop on Computational Intelligence and Applications (IWCIA)*, 101–107. https://doi.org/10.1109/IWCIA47330.2019.8955042

Jindal, A., & Bansal, S. (2019). Effective Methods for Constraint Handling in Quantum Inspired Evolutionary Algorithm for Multi-Dimensional 0–1 Knapsack Problem. *2019 4th International Conference on Information Systems and Computer Networks (ISCON)*, 534–537. https://doi.org/10.1109/ISCON47742.2019.9036166

Li, Y., & Zhang, Y. (2020). *Hyper-parameter estimation method with particle swarm optimization* (arXiv:2011.11944). arXiv. http://arxiv.org/abs/2011.11944

Liu, S., Gao, X., He, H., & Zhang, S. (2019). Self-adaptive chaotic local search particle swarm optimization for propylene explosion region parameter identification. *2019 Chinese Control And Decision Conference (CCDC)*, 1702–1707. https://doi.org/10.1109/CCDC.2019.8833290

Liu, Z. (2020). An Analysis of Particle Swarm Optimization of Multi-objective Knapsack Problem. *2020 9th International Conference on Industrial Technology and Management (ICITM)*, 302–306. https://doi.org/10.1109/ICITM48982.2020.9080345

Luo, D., Ji, W., & Hu, X. (2023). Parameter Optimization and Control Strategy of Hybrid Electric Vehicle Transmission System based on Improved GA Algorithm. *Processes*, *11*(5), 1554. https://doi.org/10.3390/pr11051554

Nand, R., & Sharma, P. (2019). Iteration split with Firefly Algorithm and Genetic Algorithm to Solve Multidimensional Knapsack Problems. *2019 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, 1–7. https://doi.org/10.1109/CSDE48274.2019.9162422

Rikatsih, N., & Mahmudy, W. F. (2018). Adaptive Genetic Algorithm Based on Crossover and Mutation Method for Optimization of Poultry Feed Composition. *2018 International Conference on Sustainable Information Engineering and Technology (SIET)*, 110–114. https://doi.org/10.1109/SIET.2018.8693167

Saire, J. E. C., & Singh, A. (2019). A Comparative Analysis of Quantum Inspired Evolutionary Algorithm with Differential Evolution, Evolutionary Strategy and Particle Swarm Optimization. *2019 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, 1–6. https://doi.org/10.1109/LA-CCI47412.2019.9037039

Shahbandegan, A., & Naderi, M. (2020). A Binary Butterfly Optimization Algorithm for the Multidimensional Knapsack Problem. *2020 6th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS)*, 1–5. https://doi.org/10.1109/ICSPIS51611.2020.9349589

*name of corresponding author

Wang, K., Shang, W., Liu, M., Lin, W., & Fu, H. (2018). A Greedy and Genetic Fusion Algorithm for Solving Course Timetabling Problem. *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*, 344–349. https://doi.org/10.1109/ICIS.2018.8466405

Wang, X., & Li, C. (2018). Prediction Model of MBR Membrane Flux for Elman Neural Network Based on PSO-GA Hybrid Algorithm. *2018 Eighth International Conference on Instrumentation & Measurement, Computer, Communication and Control (IMCCC)*, 737–740. https://doi.org/10.1109/IMCCC.2018.00159

*name of corresponding author