Chapter

Role of an Optimal Multiagent Scheduling in Different Applications Using ML

Fahmina Taranum, Sridevi K, Maniza Hijab, Syeda Fouzia Sayeedunissa, Afshan Kaleem and Niraja K.S

Abstract

Scheduling is regarded as one of the vital decision-making processes used frequently in many real-time cases. It manages everything from resource allocation to the task completion, with the goal to optimize the desired objectives. Subject to the problem, the resources, tasks, and goals can differ. The aim is to design a corporative multiagent system for optimal scheduling. Many of the scheduling available algorithms calculate optimality based on different perspectives. The proposal is to create the dataset using multiple algorithms with different performance metrics to find an optimal one. This data can be imported into machine learning tools for training and predicting, based on the selected performance metrics. The algorithm considered in the empirical analysis includes first come first serve, Round robin, and Ant colony approach. The major finding shows that scheduling using Ant colony is an optimal algorithm, which is based on speed and velocity. The future extension would be to check the correctness of optimality using machine learning tools.

Keywords: optimality, scheduling, machine learning

1. Introduction

1.1 Scheduling

The scheduling task can be comprehended as a distributed consecutive decision-making process, which is designed using multiagent reinforcement learning algorithms. These algorithms provide the agents with effective learning as they involve frequent interactions with the environment, thereby enhancing their relevance to numerous real-time cases. The scheduling entities used are resources and tasks, which are interchangeably assigned to each other. The resources can be classified as heterogeneous and homogeneous. Based on the generation of throughput, resources are classified as homogenous resources with similar or uniform throughput and heterogeneous refers to distinct type. Dependent tasks are interconnected job endeavors, in which the task cannot initiate until the accomplishment of a separate task. The characteristics based on defining the task include priority and QoS. Another widely used scheduling mechanism is based on the notion of priority rules. In this method, the resource allocation and task execution are scheduled by designating some priorities to them as per the situation, demand, requirement, or need.

The priority can be based on the deadline and load. The magnitude of work carried out to processes for manufacturing components is termed as load, which must be balanced to execute fair scheduling. Deadlines are the limits or constraints to complete the job, which can be classified as firm, soft, and hard. Based on the applicability the classification is named as soft if is not rigid, or if some limited consideration is given then it is firm, or if it is not accepted after the time limit then hard The components used for the entities are depicted in **Figure 1**.

1.2 Optimal schedule

Optimality can be measured with respect to task or resource using multiple metrics, as shown in **Figure 2**. Most of the optimization practices are extremely nonlinear and cross-media under different contexts and constraints with high convergence performance and low computational cost. It can be classified as gradient or derivative, stochastic or deterministic, or population-based or trajectory-based.

The word optimal means the best and most desirable solution and scheduling means arranging, controlling, and optimizing work and workloads. From that sequence of job, there should be maximum utilization of machines and less waiting time for the job and maximum work should be done by satisfying various constraints. It is an optimization problem in scheduling where the input to the machine is the list of jobs and the output is the schedule of all the jobs on the particular machine. The schedule should optimize the throughput and resource utilization. This is also known as machine scheduling, processor scheduling, multiprocessor scheduling, or just scheduling. The jobs can be scheduled on single machine or multiple machines based on the requirements. Either a set of jobs given to only one machine or parallel to more than one machine.

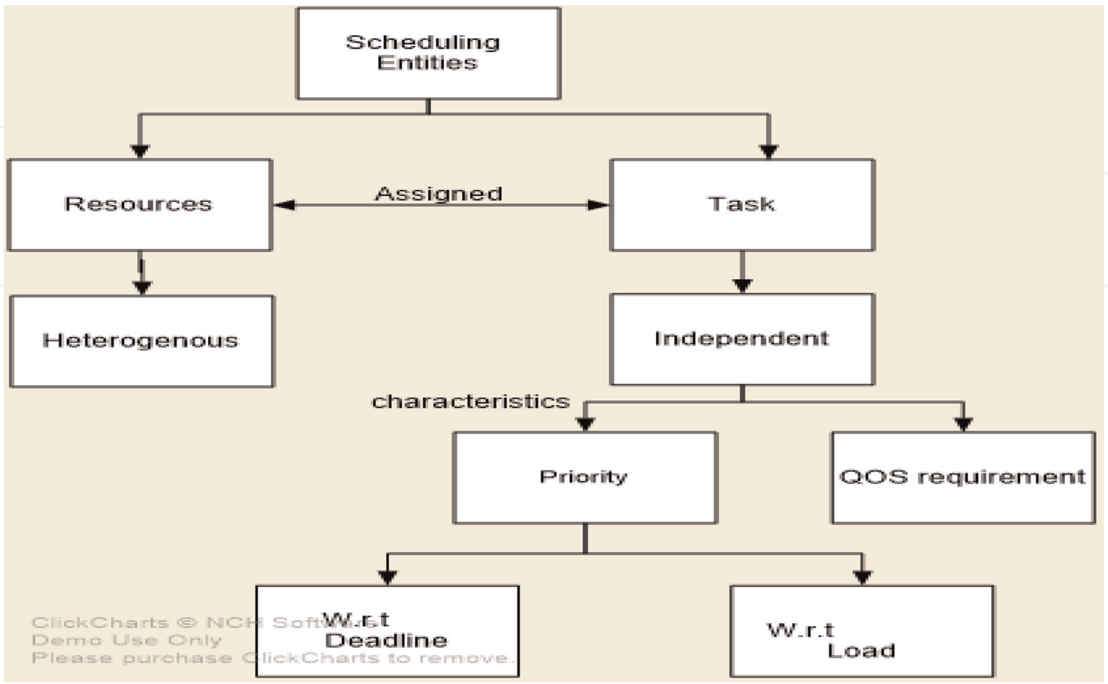

Figure 1.
Scheduling entities.

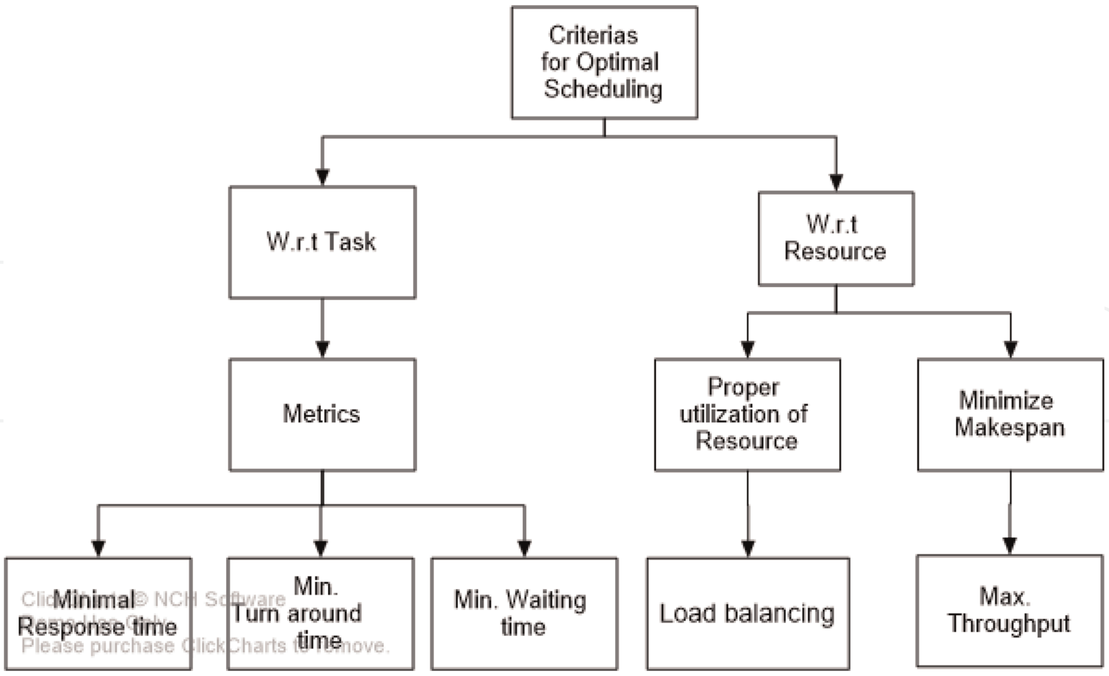

Figure 2.
Criteria for optimal scheduling.

The criteria for scheduling a resource are its proper utilization, generating minimum makespan, and maximum throughput. Utilization refers to the instance of manufacturing a component based on its real-world or commercial usage. Makespan refers to the time taken by the units of resources to properly complete executing all the jobs. Throughput refers to the process of executing the job in a unit amount of time.

There are various parameters considered to get the optimal job or task for setup.

- Start dates and deadlines of the job
- Costs depend on the completion times of activities
- Possibilities of leaving some jobs “unperformed.” due to some interrupt.
- Initial Setup times and costs.

If there is a machine M with a set of jobs J1, J2, and J3 that needs to be executed by M, with maximum work in minimum time to complete all jobs with the best sequence is called optimal job scheduling.

1.3 Types of scheduling

There are many algorithms for scheduling of dependent or independent tasks in a single processor or multiprocessor environment with different speeds. And using dynamic programming algorithms, the best optimal scheduling can be obtained by considering the time taken to complete the job or priority of the job and any other constraints need to satisfy to finish the particular tasks.

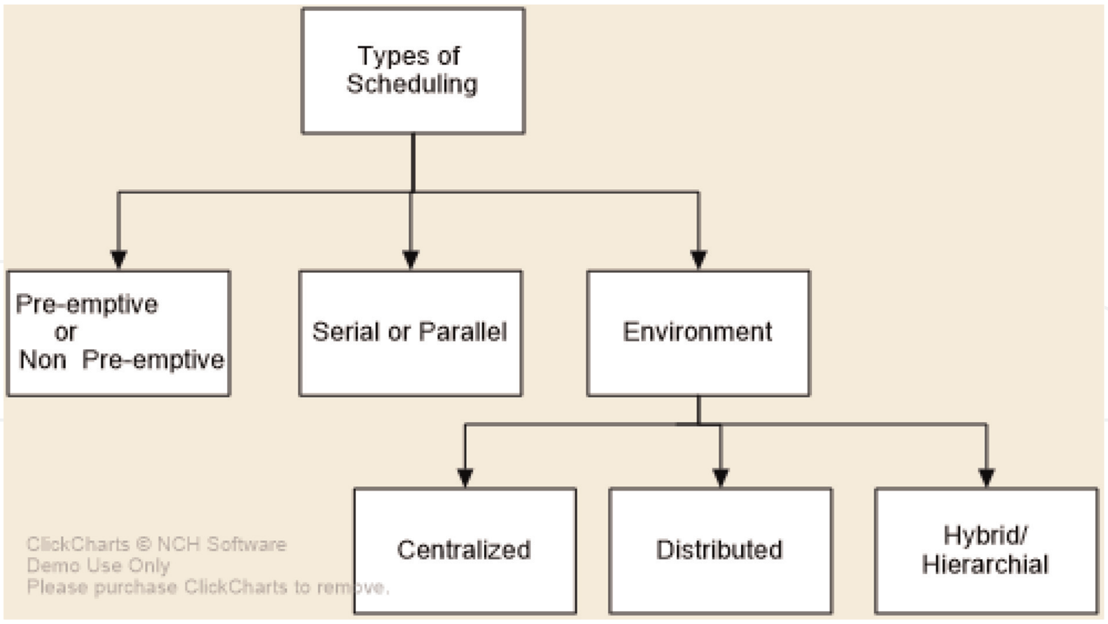

Figure 3.
Types of scheduling.

The effective allocation of the order of the jobs to a machine to get the maximum profit and minimum cost from the process leads to optimal scheduling is depicted in **Figure 3**.

Classification of the scheduling can be preemptive or nonprimitive, in which the decision to take the resource from the low-priority process and allocate it to a higher priory one is taken. For the simultaneous, linear or nonlinear class of execution, the second level of classification is used. Finally, the decision to distribute the resource can be done at central, distributed, or combined by using different environments based on the demand and architecture used. The most widely used scheduling algorithms are Round-robin, First come first serve, Shortest job first, Earliest Deadline First, Priority Scheduling, Multi-Level Queue Algorithm, Multi-level Feedback Queue Scheduling Algorithm, and shortest Remaining Time.

1.4 Approaches used in machine learning

Reinforcement learning, which is usually preferred is an area of machine learning which emphasizes how intelligently an agent decides an environment to achieve the optimum reward. This mechanism allows learning using trial-and-error communication with the environment. Scheduling helps to interpret the data accurately, if collaborated with machine learning as it generates the optimal results based on predictions as depicted in **Figure 4**. Through frequent trials, the agents come upon different possible outcomes for an action and thereby find the most appropriate action to be done in any given situation.

For instance, upon encountering an unexpected situation, reinforcement learning can be found helpful, as it enables learning from prior results and altering the parameters as per the need before passing for the subsequent iterations thereby ensuring that the solutions are as desired and robust.

The learning algorithms could benefit from this idea, by associating the priorities with the feedback signal the agents receive when executing the actions. Later by incorporating the priority rules with the feedback received for each action, the learning algorithm can be improvised. The aim is to generate multiple cases from different

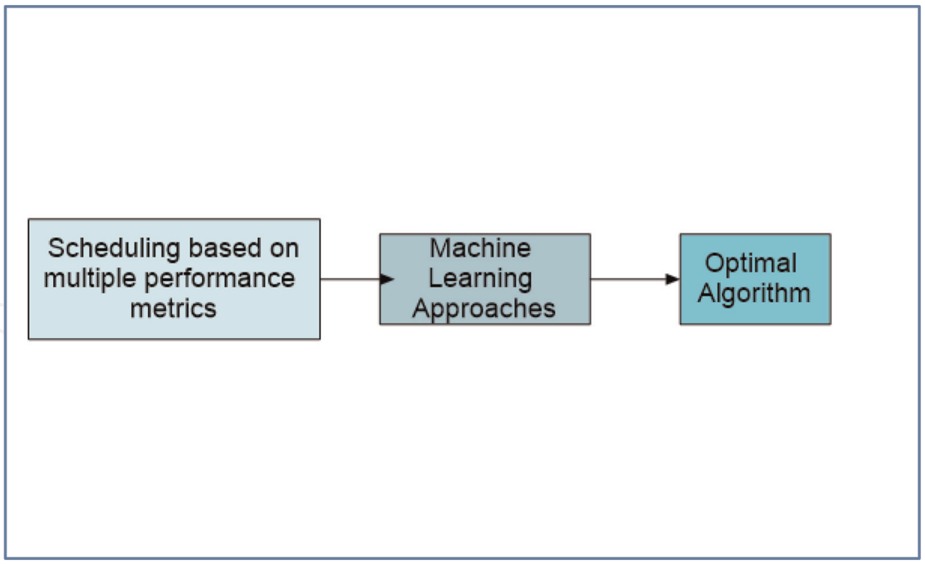
Scheduling based on multiple performance metrics
Machine Learning Approaches
Optimal Algorithm

Figure 4.
Machine learning and optimality.

algorithms and train a machine with the data generated from execution, to validate the optimality of scheduling using machine-generated test results.

1.5 Applications

Based on the usage of multitask applications in every field, a need for efficient utilization of optimal multiagent scheduling algorithms is highlighted. Machine learning algorithms are widely used to improve the optimality of multiagent scheduling in the field of production and transportation. The major domains where some of these challenges can be improved have been discussed below.

1.5.1 Transportation domain

Every individual had to rely on some mode of transportation which can be through land, air, or water. With the advancement of technology, the usage of air and land transportation has drastically improved due to efficient time management and comfort. The development of many multiagent system software has been of the major reasons for this. Though the multiagent scheduling softwares still exist, there are some challenges that need to be improved.

1.5.2 Railway domain

Transportation is the major source of people especially above 75% of people relies on railway networks throughout the globe. When it comes to railway networks, most railways run on a single line where more limitations exist for decades. Although multiagent systems have applied to this, many of the existing challenges are unavoidable. The train delay rate is a challenging issue to date, even with the advances in technology, it cannot be handled perfectly. With the advancement of machine learning algorithms, we can find a path with the optimal multiagent scheduling algorithm. Electricity transport management plays a vital role, especially in railway networks with the utilization of machine learning algorithms. We can find an optimal solution for scheduling the multiagent system to choose the proper routing to overcome the major challenges faced by the transportation domain.

1.5.3 Airline reservation domain

Airline flight bidding software mostly uses multiagent system prototype software, where the buyers buy the e-ticket. The optimal multiagent scheduling algorithm with machine learning can direct the buyer to a better price line system through which buyers take preferences and correlate the parameters with the available flights.

1.5.4 Manufacturing domain

Every business domain has its own production department where different jobs have to be scheduled with multiagent approaches. Machine Learning usage has become unimaginable heights in the last few years where the manufacturing industry's economic growth rate can be improved with the utilization of optimal multiagent scheduling algorithms.

1.5.5 Electronic commerce domain

This domain will show its increased growth rate in the near future as most of the day-to-day transactions are carried out these days through e-commerce combined with the multiagent scheduler and machine learning algorithms.

2. Literature review

The proposal by Kumar et al. [1] is to enterprise a scheme for processing tasks to allocate resources at runtime. The experimentation was performed on Cloudsim to validate the effective technique for optimum solutions. The proposal aims at tracing results with initial high velocity, which is gradually decreased improved exploitation.

The proposal of reinforcement learning is used by author Chi Zhang et al. [2], to achieve corporation in the scheduling of multiagent working in a team, which is an agent feedback-based learning used to learn from the results of the environment and perform action. The concept of Markov chain and Proximal Policy Optimization is used to check for optimal scheduling. The performance metrics used are load consumption and price of electricity to check for optimal results. The optimality of the results is based on the trained dataset or historical data used. Multiagent reinforcement learning is used when manifold inhabitants with multiple global constraints interact with different scheduling errands. This approach is used in a realistic environment to obtain optical results using predictions.

The population diversity control problem is selected in the proposal by author Zha oyun Song, Bo Liu et al. [3] to improve optimization. The method is based on the runtime selection for adaptive and diversified controlling parameters. The accuracy is calculated for the cluster of movable particles.

The concept of using low voltage over harmonic distortion of modulated index is used to set the image threshold for increasing the image accuracy by the author [4].

The scheduling of multiagent in a cooperative, spatial and temporal constraints is used by the author Julie Shah [5] to obtain an optimal task assignment. The experimental analysis is done on the hill climbing algorithm using intricate computation and the accuracy versus optimality is compared with the conventional algorithm. An empirical result proves that the optimality of hill climbing is better using the mixed integer collaboration approach of linear programming.

The author Khaled M. Khalil [6] has used the Netlogo simulator for experimentation. The aim is to maximize the agent group by maximizing the reward to the agent working in the group using Q-learning algorithm (action value function) along with working in an interactive, autonomous, and seamless environment. The approaches of high relevance to realistic solutions grounded on AI was proposed by authors Martin Riedmiller et al. [7] to optimize, manufacture and control the scheduling based on distributed sequential approaches for employment related decision making with multi agent reinforcement learning.

The Authors E. Grace Mary Kanaga [8] have used an approach of ANT colony for an optimal scheduling algorithm. The continuous optimization problems are solved using velocity, inertia weight location of the particle, and global and local best positions. Scheduling patients for an optimal accurate solution is performed.

Zhi hui et al. [9] have aimed to implement adaptive behavior for population distribution along with fitness.

The target is to improve and validate the global convergent ability by authors Jianchao Zeng, Jing Jie et al. [10]. Kennedy et al. [11] have invented the perception using non-linear functions of Ant colony using particle swarm methodologies for optimization. Kennedy et al. [11] is the inventor behind the use of non-linear functions to design methodologies based on cluster optimization. The relationships amongst PSO and its integration with artificial life using genetic algorithms are proposed and discussed.

The author Wilfried Brauer et al. [12], aims at scheduling multimachine, and assigning jobs based on demand like cost, the effectiveness of results, and time. The approach of artificial intelligence is used to collect and train the data in distributed, parallel, asynchronous and corporative multi-agent environments. To generate the results two learning steps known as successor selection and estimate adjustment are repeatedly applied and experimented on individual machines.

3. Proposal

The idea is to generate optimal scheduling, which is tested for set of scheduling algorithms and the major finding is ANT colony scheduling gave the best results. The algorithm aims to predict and generate the best position using parameters like inertia, weight, speed, velocity, distance, acceleration constants, direction, and local and global position. The local and global positions are the best positions for each and the group of jobs, respectively.

The movement of the job is in the direction of the best location value. The velocity is calculated using Eq. (1).

$$\mathrm{Ve}_i^{z+1} = w\mathrm{Ve}_i^{z} + ac_1 p_1\left(\mathrm{Lbst}_i^{z} - x_i^{z}\right) + ac_2 p_2\left(\mathrm{Gbst}^{k} - x_i^{z}\right) \tag{1}$$

Lbst is the minimum value of the experience position of each job and the group's minimum value of experience position is Gbst. The movement is monitored using the kth step of ith job to get the next new place. The first term of Eq. (1) gives the inertia with respect to previous velocity; second and third terms are used to fetch the direction of each and group of job respectively. An accelerating object (ac) is used to persuade a uniform alteration in its velocity at each instant. Acceleration constants used in Eq. (1) are ac1 and ac2. The random numbers are p1 and p2 [0, 1] of range are chosen. Velocity and Position update is derived using Eq. (1) as given by Kennedy and Eberhart [11].

$$\underset{j}{x^{z+1}} = \underset{j}{x^{z}} + \underset{j}{Ve^{z+1}} \tag{2}$$

Eq. (2) is used to get the new position from the previous position and the velocity of its movement. After getting the best and optimal position, all the particles synchronize with each other. According to Kennedy et al. [11], the search space is epitomized in a D-dimensional vector and the velocity and position update are calculated using Eqs. (1) and (2). The jth particle's position in D-dimensional vector is calculated using Eq. (3). The local best (Lbst) and the global best (Gbst) of the jth particle are denoted in Eqs. (3) and (4) respectively.

$$x_j = \left(x_{j1}, x_{j2}, x_{j3}, x_{jk}, \dots, x_{jd}\right) \tag{3}$$

$$vex_j = \left(x_{j1}, x_{j2}, x_{j3}, x_{jk}, \dots, x_{jd}\right) \tag{4}$$

$$l_j = \left(x_{j1}, x_{j2}, x_{j3}, x_{jk}, \dots, x_{jd}\right) \tag{5}$$

$$g_j = \left(x_{1j}, x_{2j}, x_{3j}, x_{ij}, \dots, x_{jd}\right) \tag{6}$$

For scheduling a process with recourse, the problem is to schedule 'n' jobs with appropriate resources; each process has a set of sequential tasks (operations) and an index of the job. The count of operations used in scheduling is denoted by D in (7).

$$D = \sum n_i \tag{7}$$

$$Y = \sum_{i=1}^{i-1} n_i + 1 \tag{8}$$

3.1 Optimization for job scheduling

To generate an ideal solution for Job-scheduling multiple scheduling algorithms are available, which works on different parameters. To generate an optimal one, the need to select the parameter for performance metric becomes mandatory, based on which the complete experimentation can be carried out. Let the proposal be defined on the continuous optimization parameters like particle position xj, Velocity vej, acceleration coefficients ac1 and ac2, and inertia weight ω. Scheduling a task is a conjunctional and feasible optimization with sequence of selected resource along with its operation. The aim is to find an optimal schedule without busy waiting and with fair scheduling.

3.2 Task scheduling with optimization

The scenario consisting of n task (or jobs) denoted as J = {J1, J2, … , Jn}, sequential task T={1 … n}, Resource R = {R_1, R_2, … , R_m}, and O_{ij} are the operations with i, j are the indices of the jobs and task respectively. Every task of the v^{th} job is numbered as v and for same number of v tasks in **Table 1**, and y is defined in Eq. (8). One sample representation of the job for optimization of scheduling is depicted in **Table 1**.

a. Phase I: Scheduling for three jobs with three resources is depicted in **Table 2**. The experimentation is done in multiple permutations (3p3 ways for 3,3 job, resource allocation) giving in total 9 such task possibility. **Table 2** uses a few of the nine possible permutations, where R1, R2, and R3 are the distinct resources. The example for job representation along with position and velocity vector initialization of scheduling is given in **Table 3**.

   Initialization of the particle position is done with random numbers from x_{min} to x_{max}, and x_{min} is set to 0 and x_{max} is set to 2. The velocity vector is initialized with the random numbers limitation of v_{min} as −4 and v_{max} as 4 as shown in **Table 3**.

b. Phase II: Decoding Particles with job solutions: An integration of optimizing for job scheduling cannot be deployed directly as a solution to particle position. Hence, indirect ways are adopted to decode particle representation as a solution to schedule job problem. For decoding jobs into a schedule, the algorithmic steps are listed below:

Task	**1**	**2**	**…**	**V**	**…**	**n**
Position X_{ij}	Xi1	Xi2	…	Xiy	…	Xid

Table 1.
Job's representation for resource scheduling.

Job	**Arrival time**	**Task Sequence**	**Processing time**			**Age**
1	**1**	**R1 R2 R3**	2	**1**	3	**38**
2	3	R2 R3 R1	1	3	2	40
3	5	R3 R2 R1	3	1	2	44

Table 2.
Job scheduling problem for 3:3 jobs and resource allocation.

Task	**1**	**1**	**1**	**2**	**2**	**2**	**3**	**3**	**3**
Position xij (Values)	xi1 (.81)	xi2 (1.12)	xi3 (1.86)	xi4 (1.09)	xi5 (.56)	xi6 (0.93)	xi7 (.68)	xi8 (1.95)	xi9 (1.76)
Velocity vij (Values)	vi1 (−2.96)	vi2 (1.67)	vi3 (−2.93)	vi4 (.99)	vi5–3.63	vi6 (.86)	vi7 (−3.10)	vi8 (3.26)	vi9 (.09)

Table 3.
Job representation along with initialization of position and velocity vector for i^{th} job.

Task unit	**2.1**	**3.1**	**1.1**	**2.2**	**2.3**	**1.2**	**3.2**	**1.3**	**3.3**
Position Pij	Xi5	Xi7	Xi1	xi6	Xi4	Xi2	Xi9	xi3	Xi8
(Values)	0.56	0.68	0.81	0.93	1.09	1.12	1.76	1.86	1.95
Order of execution	1	2	3	4	5	6	7	8	9

Table 4.
Result obtained for sequenced particles after phase II.

R3		**J 3.1**			**J 2.2**			**J 1.3**							
R2	**J 2.1**		**J 1.2**	**J 3.2**											
R1	J1.1				J 3.3		J2.3								
Time slot	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Table 5.
Decoded schedule with optimization.

Step I: sort in ascending order the values of the position vector.
Step II: arrange the tasks in the corresponding order of the values of the position vector obtained in step I.
Step III: The resultant is with sequential order of task along with the corresponding positions as shown in **Table 4**.

Using the sequence obtained with operation-based permutation is $\pi = (2,3,1,2,2,1,3,1,3)$. An element of π with a value i for Job Ji. The jth occurrence of i in π refers to operation Oij for the jth task (operation) of Job ith. The precedence of the task is determined simply by the order of the elements of π and all the task are ready for scheduling as per the first row of **Table 4**. Based on the permutation, the first element selected for scheduling according to the permutation is 2, therefore, first unit of the third Job is processed on resource R3.

Followed with the first task of the second Job is processed on R2, and then the first unit of the task1 is scheduled on R1. The obtained decode schedule is shown in **Table 5**.

Jobs 1,2, and 3 complete the execution at 9, 8, and 6 time respectively as concluded from **Table 5** after applying optimization algorithm based on the order of tasks, as depicted below:

$$O = \{O31, O21, O11, O12, O22, O32, O33, O13, O23\}$$

It is optimal scheduling as all the processes have been completed within time limit of 9, with the time of completion for jobs as {J1, J2, J3 as 9,8,6}.

3.3 First come first server algorithm

The working concept of FCFS is to serve the first in jobs on high priority, and results obtained after applying the scheduling are shown in **Table 6**.

R3						J 1.3		J 2.2			J 3.1									
R2			J 1.2	J 2.1										J3.2						
R1	**J 1.1**										**J 2.3**					**J3.3**				
Time slot	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

Table 6.
Decoded schedule with FCFS.

R3					J2,2		J3.1			J2.3			J1.3							
R2				J2.1	J1.2					J3.2										
R1		**J1,1**									**J3.3**									
Time slot	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

Table 7.
Job decoded schedule using priority.

Jobs 1,2, and 3 complete the execution at 6, 11, and 16 times respectively after applying the FCFS scheduling algorithm as concluded from **Table 6**. The order is based on first-entered jobs and would be the first to be catered.

The selected operation of tasks is as depicted below:

$$O = \{O11, O12, O21, O13, O22, O31, O23, O32, O33\}$$

Hence, the conclusion is drawn as FCFS is not optimal when compared to optimization algorithm. Job 1 has arrived first and he/she is processed for task 1 on resource R1 which is available. Next Job 1 task 2 has to be processed on resource R2 and after completion of the task; Job 1 has to get the resource R3 for task 3. Thus, based on the FCFS description **Table 6** decoded schedules are obtained. Based on FCFS the calculated total job completion time is:

$$\{J1 = 6, J2 = 11 \text{ and } J3 = 16\}$$

3.3.1 Based on age priority (using preemptive manner)

The working concept of priority is based on age priority, and results obtained after applying the scheduling are depicted in **Table 7**.

Job 1 is processed on task 1 with a request for resource R1, Job 2 needs resource R2 for its first task and based on its availability is allocated for 1 slot. Makespan time for jobs is {J1 = 14–1 = 13, J2 = 14–3 = 4, J3 = 12–5 = 7}, i.e{J1, J2, J3 is 13, 4 and 7, respectively}.

4. Conclusion

Multi-agent technologies and approaches can be applied to machine learning, based on their capabilities of flexibility, adaptability and self-sufficiency. The applications designed using these methodologies are realistic, dynamic and distributive in nature. The idea is to build a best decision making model using the latest approaches of machine learning. The experiments are tested for four types of Scheduling Round robin, FCFS, Priority-based and Ant colony or particle swarm optimization technique. The results validate the Ant colony approach with the optimal answer. The dataset generated with this experimentation for multi-agent scheduling is collected with different performance metrics. The future work of this proposal is to justify the validation of multiagent scheduling using a machine learning tool, by training the machine using the generated dataset.

Author details

Fahmina Taranum[1], Sridevi K[1], Maniza Hijab[1]*, Syeda Fouzia Sayeedunissa[1], Afshan Kaleem[1] and Niraja K.S[2]

1 Muffakham Jah College of Engineering and Technology affiliated to Osmania University, Hyderabad, Telangana, India

2 BVRIT Hyderabad College of Engineering for Women, Hyderabad, Telangana, India

*Address all correspondence to: hijabmaniza@gmail.com

References

[1] Kumar M, Sharma SC. PSO-based novel resource scheduling technique to improve QoS parameters in cloud computing. Real-world optimization problems and meta-heuristics. Neural Computing and Applications. 2019; **134**(16):1-24. DOI: 10.1145/3302505.3310069

[2] Zhang C, Kuppannagari SR, Kannan R, Xiong C, Prasanna VK. A cooperative multi-agent deep reinforcement learning framework for real-time residential load scheduling. Association for Computing Machinery. ACM. 2019. pp. 59-69. ISBN: 978-1-4503-6283-2/19/04 DOI: 10.1145/3302505.3310069

[3] Song Z, Liu B, Cheng H. Adaptive particle swarm optimization with population diversity control and its application in tandem blade optimization. Journals of Mechanical Engineering Science, Sage. 2018;**233**(6):1-17

[4] Dulhare U. Prediction system for heart disease using Naïve-Bayes and particle swarm optimization. Biomedical Research. 2018;**29**(12):2646-2649

[5] Shah J, Zhang C. Co-optimizating multi-agent placement with task assignment and scheduling. Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence. ACM. 2016. pp. 3308-3314. ISSN No: 978-1-57735-770-4

[6] Khalil KM, Abdel-Aziz M, Nazmy TT, Salem A-BM. MLIMAS: A framework for machine learning in interactive, multi-agent systems. Procedia Computer Science. 2015;**65**:827-835. DOI: 10.1016/j.procs.2015.09.035

[7] Riedmiller M, Sperschneider V, Brockmann W, Nuchter A. Multi-agent reinforcement learning approaches for distributed job-shop scheduling problems. 2009

[8] Kanaga EGM, Valarmathi ML. Multi-agent based patient scheduling using particle-swarm optimization. Procedia Engineering. 2012;**30**:386-393

[9] Zhan Z-H, Zhang J, Li Y, Chung HS-H. Adaptive particle swarm optimization. IEEE Transactions on Systems, Man, and Cybernetics. 2009;**39**(6):1362-1381

[10] Zeng J, Jie J, Hu J. Adaptive particle swarm optimization guided by acceleration information. International Conference on Computational Intelligence and Security. IEEE, 1-4244. 2006. pp. 351-355

[11] Kennedy J, Eberhart R. Particle swarm optimization. IEEE Xplore. 2002. pp. 1942-1948

[12] Brauer W, Weiß G. Multi-machine scheduling – a multi-agent learning approach