Multi-Agent Based Coordinated Scheduling in Hospitals

J. Dhiviya Rose\textsuperscript{1} and S. Christalin Nelson\textsuperscript{2}

\textsuperscript{1} School of Computer Science, University of Petroleum and Energy Studies (UPES) 
Bidholi, Dehradun. INDIA 248007
\textsuperscript{2} School of Computer Science, University of Petroleum and Energy Studies (UPES) 
Bidholi, Dehradun. INDIA 248007

Email: dhiviyanelson@gmail.com

Abstract. — Patient Scheduling is the process of scheduling the patients by creating an optimal schedule for the patients in using the resources in hospitals. The real-time scheduling with various constraints like emergency situations like accident case makes this problem complex. The manually scheduling algorithms and the traditional approaches needs more coordination, time and manpower to achieve its efficiency. Agents are a natural choice because they are known to represent a distributed environment well and they are capable of coordinating with each other in dynamic environment. The architectural components like Belief that maintains the internal state of the system, Desires that represents the motivational state of the agent and Intentions that helps to deliberate the state of the goal make this autonomous system adaptable and fault tolerant. In the proposed system each resources and patient is constructed to be software agents, which has the ability to coordinate among them. The optimal schedule is calculated in minimal time using the Genetic Algorithm (GA), an efficient search-based optimization technique. When a patient arrives, he/she is served by a common agent where the priority, resource requirement, processing time, and sequence are assigned to each patient. Hence in this highly dynamic domain optimal schedule generation with minimal time and hardware/software requirements will provide the customers satisfaction with their visit to the hospitals.

Keywords— Multi-agent Systems, Scheduling, Optimization

1. Introduction

In hospitals the use of the various services in the hospitals like blood testing, scans, X-Rays etc. can be optimized if scheduled properly. Patient Scheduling is the process of scheduling the patients by creating an optimal schedule for the patients in using the resources in hospitals. The real-time scheduling with various constraints like emergency situations like accident case makes this problem complex. More generally, scheduling problems are subjected to certain constraints to optimize some objective function [1]. This is usually done manually in most systems to load balance a system effectively. Effective Patient Scheduling can improve hospital service quality and operation efficiency. The manually managed and traditional scheduling algorithms needs high coordination, time and manpower to achieve its efficiency. In the
autonomous system designed by [6] it’s proved that it have improved the service delivery by generating and maintaining proper schedule in spite various scheduling problem differ in its characteristics. The finding out of MAS has derived the way of deploying software agent to solve the scheduling problem [2]. Agents are a natural choice because they can be used to represent a distributed environment well and they have the ability in coordinating among the other agents in dynamic environment. Agents based model follows Belief-Desires-Intentions (BDI) architecture that supports in developing intelligence to the autonomous systems. This helps in selecting and achieving plans without disturbing the real time systems execution [3]. The adapting and dynamic property of the autonomous systems made these agents meet out the real time needs [4].

The architectural components like Belief that maintains the internal state of the system, Desires that represents motivational of the desires state of the agent and Intentions that helps to deliberate the state of the goal make this autonomous system adaptable and fault tolerant [5]. Group of heterogeneous agents form an Multi Agent Systems(MAS). MAS allows the subproblems of a constraint satisfaction problem to be subcontracted to different problem-solving agents with their own interests and goals. The most important reason to use MAS when designing a system is that some domains require it people who have various BDI but intent towards achieving a common goal without conflict. These systems can be well developed using the BDI architecture of MAS. The software agents can be programmed to act with their own capabilities and abilities. Multi-agent systems also have more scalability when agents have to be added. Systems whose capabilities and parameters are likely to need to change over time or across agents can also benefit from this advantage of MAC [6]. This paper describes about the basic of scheduling in chapter I follow with the survey of scheduling in healthcare domain and system design for optimization for patient scheduling using multi agents ended with its implementation.

2. Background Studies

Effective resource utilization can improve hospital service quality and operation efficiency. Traditional algorithms like FIFO, linear programming are been used manually and computerized used to solve this in many hospitals. The existing studies try to not only improve the service quality but also to help the allocators do their works efficiently. However, each problem has its own characteristics [7]. Beside those different application’s sectors, these approaches are linked by a common objective, which is also ours: create cooperation between agents in systems where decisions and information are highly distributed. The auction based system can help in deriving the best optimized solution for any real time scenario [8]. Using a specific agent as coordinator could be considered for preparing the plan and different care providers’ supply numerous services [9]. The negotiation is highlighted by the agents to compromise the policies between the doctors and the various agents which is made for administrators [10]. Objectives used for the proposed systems to reduce the patient waiting time while using the labs is mentioned in [11]. However, the specific problem characteristics including time and hardware/software requirements for generating an effective and efficient schedule that are seldom addressed in the existing studies. Recent research has taken a new
dimension on the possibility of deploying software agents to solve the allocation problem [12] in the various domain. Agents are a natural choice because they are known to represent a distributed environment well and they are also capable of coordinating with each other which works well in dynamic environment[13][14]. Agents also has the capability to interact with each and coordinate to achieve a common goal with intelligence adds its social behaviour and dynamic nature[15].

Predictive scheduling techniques are used for static environments, but rescheduling is often done to aid the recovery of system from disruptions. Reactive scheduling in dynamic environments often uses a variety of priority rules. R. Haupt [16] has given a list of 26 different scheduling rules. Some of the popular dispatching rules are FCFS (First Come First Serve), MS (Minimum Slack), SPT (shortest processing time), LPT (longest processing time), WPT (Weighted Processing Time), EDD (earliest due date), SL (smallest slack) and CR (critical ratio). The advantage of traditional scheduling algorithm is that these algorithms can be guaranteed to produce a schedule which is robust and reliable. However, many of these algorithms require a central computational unit and do not represent the distributed nature of the patient scheduling problem. Also they are inflexible, expensive and they do not handle disruptions very well.

3. System Description

From the earlier discussion we can conclude that although the hospitals are trying to provide better services to the patients, but they are still not able to provide the required services in optimized manner. Motivated by these factors, my research aim is use multiagent coordination and optimization in healthcare domain. The main objective of the proposed research is to develop a computerized system to achieve the following:

- Patient satisfaction by providing the required resources at the right time.
- Reducing the waiting time of the patient
- Automatic negotiation between the agents avoid human conflicts during the criticality of each and every patient
- Increase the usage of resources by providing all time availability.
- Prioritizing the usage pattern

In the proposed system each resource and patient is represented as agents, which has the ability to coordinate among them. The optimal schedule is calculated in minimal time using the genetic algorithm (GA), an efficient search-based optimization technique. When a patient arrives, he/she is served by a common agent where the priority, resource requirement, processing time, and sequence are assigned to each patient. An optimal schedule is generated using genetic algorithm. The generated schedule is compared with the exact methods like dynamic integer programming for comparative study of time complexity. This helps to prove the effectiveness of the algorithm. The framework in which the whole system works is given in the fig. 3.1. When a patient arrives, he/she is served by a common agent (CA), a general physician who asks the patient about their overall health condition. After the consultation with the general
physician, the CA makes the entry about the various operations they have to undergo for a specific time in different resources like scan and blood test.

Thus, the overall plan of patient’s treatment is generated by the CA. The CA also creates individual agents for each patient called patient agent (PA) and resource agent (RA) for all available resources in the hospital domain. PA interacts with the scheduling agent (SA) supporting the distributed nature of the problem. The treatment plan generated by CA is communicated to SA, where genetic algorithm is used to find the optimal schedule at that given time. The generated schedule is then informed to the PA and RA for their usage. The priority of the patients and resources are also allocated by the CA.

Genetic Algorithm (GA) is one of the methods that demonstrated significant improvement in various related domains like job shop scheduling. The initial path was reported by Davis who proposed a new encoding scheme for scheduling problem into the GA solution space and achieved an effective time based schedule without violating the constraint[17]. On the research towards identifying an optimal schedule in dynamic scheduling problems using GA various contributions were made in the areas like chromosome representation, cross over operators, mutation operators and hybrid GA operators. Powered with a minimal time and hardware/software requirement, GA can find its place for generating optimal schedules in health care domains. Making the schedule distributed through the use of multi-agents i.e. accessible will reduce the manual works like telephone operator, crowding at a place and minimizes the patient waiting time and increase the staff productivity in the hospitals [4].

The scheduling agent generates an optimal schedule by getting the treatment plan for each patient from the common agent in minimal time using GA. This works by defining the goal in the form of a quality criteria and move forward by a stepwise refinement to reach a better solution. The generalized algorithm for the GA based optimization proposed for patient scheduling is as follows shown in Figure 3.1.

**Figure 3.1:** System Architecture

![System Architecture Diagram](image-url)
Read Parameters
BEGIN
Generate initial population of size 50 by random from possible permutations
Compute the fitness of each schedule
WHILE NOT finished DO
BEGIN
FOR population_size/2 DO
BEGIN (if crossover rate is less than 0.6 to 1.0 )
Select 2 individuals for crossover
Combine the two individuals by order based crossover
Compute the fitness of offspring generated
Insert offspring in new generation
END
FOR each best individual apply Mutation
BEGIN (if mutation rate is less than 0.001 )
Select individual with high fitness value
Apply insertion mutation
Compute Fitness
Insert Offspring in new generation
END
IF population has converged THEN
finished := TRUE
END
END
Output the best schedule with high fitness

**Figure 3.2:** Algorithm for Schedule Generation

The generation of GA based schedules is robust and can be successful with a wide range of problem areas, including those which are difficult for other methods to solve. Initially whole new populations of possible schedules are produced. The best individuals with high fitness were selected and made to reproduce new generation of off springs. The improvement process is accomplished by GA operators such as Crossover and Mutation. The converged optimal schedule is compared for its efficiency with schedules generated by traditional scheduling algorithms like First-Come-First-Serve and Earliest-deadline-first for time complexity. The exploration and exploitation used by GA makes this algorithm a good solution to calculate an optimal schedule for patient scheduling.
4. Implementation

A GA based Optimization for patient scheduling using multi-agents is designed using Java Agent Development Environment (JADE). Each resource and patient is represented as agents, has the ability to coordinate among them. The optimal schedule is generated in minimal time using the genetic algorithm, an efficient search-based optimization technique. When a patient arrives, each person is served by a Common agent where the priority, resource requirement, processing time, and sequence are assigned to each patient. An optimal schedule is generated using GA by a Schedule agent. The schedule generated is compared with the exact methods like dynamic integer programming and traditional scheduling techniques such as First-come First-serve and EDF for comparative study of time complexity to prove the advantages of the proposed system. Different genetic encoding is used to generate an optimal solution. Choosing the representation remains critical in the solution. The genes of the chromosome represent the completion time of the operation in each patient.

The system is constructed using the agent framework JADE in windows operating system. An agent is created in JADE by defining a class that extends jade.core.Agent and implements the setup() method as shown below. A sample ACL.INFORM message is sent by a common agent to the scheduling agent is given below.

```java
public class MyAgent extends Agent
{
  protected void setup()
  {
    System.out.println("Adding waker behaviour");
    addBehaviour(new WakerBehaviour(this, 10000)
    |
    protected void handleElapsedTimeout()
    |
    perform operation X
    )
  }
}
ACLMessage msg = new ACLMessage(ACLMessage.INFORM);
msg.addReceiver(new AID("Peter", AID.ISLOCALNAME));
msg.setLanguage("English");
msg.setOntology("Weather-forecast-ontology");
msg.setContent("Today it’s raining");
send(msg);
```

**Figure 4.1**: Sample of Patient name and number of resource agent

The experimental design of patient scheduling system creates two agents named: Common Agent (CA) and Schedule Agent (SA) under the main container and number of patient agents with patient name and number of resource agent with resource name under two separate
containers as shown in figure 4.1. The CA starts the JADE runtime environment. It supports the patient and resource registration by entering the related data into the database. It also creates individual agents for each patient and resource entry. It informs the SA to create the optimal schedule using GA through ACL (Agent Communication Language) messages. The optimized schedule generated by the scheduling agent is communicated to all registered patients and resource agents through ACL.INFORM per formative message.

Figure 4.2: Patient scheduling runtime environment in JADE

The Optimal schedule for patient scheduling problem is solved using GA in minimal time. The Scheduling Agent gathers the patient and resource data from database and generates different possible schedules randomly. The schedules are encoded and evaluated using the fitness function. The algorithm will take those schedules and allows them to reproduce by subjecting it to genetic operators like Crossover and Mutation. This goes on until certain criterion is met such as finding a best solution or reaching a fixed number of iterations. The simulation can be run with the experimental inputs. The results obtained should be compared against traditional scheduling algorithms like First-Come-First-Serve and Earliest-deadline-first. The overall completion time is used as a performance metric for the comparative study. LEKIN software can be used to obtain the result for traditional scheduling algorithms. The figure 4.3 shows the screenshot of the experimentation outputs.
5. Conclusion

The multi-agent based solution for patient scheduling has its advantage during the real time situation like scheduling in case of emergency situations. The social and dynamic nature of agents in a multi-agent system makes the task easier. The mobile agent helps in announcing the updated. The schedule to the patients in a timely approach. A proper selection of Genetic algorithm parameters for this application is still an open issue. The crossover and mutation operators presented in this paper should be further investigated in order to improve the performance of the proposed system. Heuristic optimization techniques will surely benefit the time management in constructing an efficient schedule in the very large combinatorial search problems like patient scheduling. Hence in this highly dynamic domain optimal schedule generation with minimal time and hardware/software requirements will improve the quality of livelihood.

References

[1] D. Karger, C. Stein, and J. Wein, “Scheduling Algorithms 1 Introduction,” Environment, pp. 1–52.
[2] I. Vermeulen, S. Bohte, K. Somefun, and H. La Poutré, “Multi-agent Pareto appointment exchanging in hospital patient scheduling,” Serv. Oriented Comput. Appl., vol. 1, no. 3, pp. 185–196, 2007.
[3] J. M. Corchado, D. I. Tapia, and J. Bajo, “A multi-agent architecture for distributed services and applications,” Int. J. Innov. Comput. Inf. Control, vol. 8, no. 4, pp. 2453–2476, 2012.
[4] M. Hannebauer, “From Formal Work ow Models to Intelligent Agents The Case Study Modeling Work ows Static Aspects,” 1998.
[5] W. Shen, S. Member, L. Wang, and Q. Hao, “International membranes directory,” Filtr. Sep., vol. 35, no. 1, pp. 45–58, 1998.
[6] J. Sauer, “Knowledge-Based Systems Techniques and Applications in Scheduling,” Knowledge-Based Syst., vol. 231, no. 4741, p. 1554, 2000.
[7] N. Benhajji, D. Roy, and D. Anciaux, “Patient-centered multi agent system for health care,” IFAC-PapersOnLine, vol. 28, no. 3, pp. 710–714, 2015.
[8] M. Reid, “Multi-Agent System Applications in Healthcare: Current Technology and Future Roadmap,” Procedia Comput. Sci., vol. 52, pp. 252–261, Jan. 2015.
[9] H. K. Lee, S. Member, J. Li, A. J. Musa, P. A. Bain, and K. Nelson, “Modeling and Analysis of Patient Transitions in Community Hospitals: A Systems Approach,” pp. 1–14, 2017.

[10] D. L. Hudson and M. E. Cohen, “Intelligent agents in home healthcare,” Ann. des Telecommun. Telecommun., vol. 65, no. 9–10, pp. 593–600, 2010.

[11] R. J. Vwhpv, C. G. Toader, and S. Busnati, “Patient Flow Control Using Multi-Agent Systems.”

[12] J. Xie and C.-C. Liu, “Multi-agent systems and their applications,” J. Int. Counc. Electr. Eng., vol. 7, no. 1, pp. 188–197, Jan. 2017.

[13] D. Borrajo and S. Fernández, “Efficient approaches for multi-agent planning,” Knowl. Inf. Syst., vol. 58, no. 2, pp. 425–479, 2019.

[14] J. Terán, J. Aguilar, and M. Cerrada, “Integration in industrial automation based on multi-agent systems using cultural algorithms for optimizing the coordination mechanisms,” Comput. Ind., vol. 91, pp. 11–23, 2017.

[15] Y. Lyu et al., “Designing and optimizing a healthcare kiosk for the community,” Appl. Ergon., vol. 47, no. 277, pp. 157–169, 2015.

[16] R. Haupt, “A survey of priority rule-based scheduling,” OR Spectrum, vol. 11, no. 1, Mar. 1989.

[17] P. E. Plsek and T. Greenhalgh, “Complexity science: The challenge of complexity in health care. [Article 1 in series of 4],” BMJ, vol. 323, no. 7313, pp. 625–8, 2001.