MODEL FOR ACCEPTANCE OF PATIENTS DURING A PANDEMIC USING SPACE SEARCH OPTIMIZATION

Mandvi Fuloria [1] [ORCID 0000-0002-6574-3804], *Dušan Šormaz* [1] [ORCID 0000-0003-3726-3288]
[1]*Ohio University, Russ College of Engineering and Technology,*
Department of Industrial and Systems Engineering, Athens, OH, USA

Abstract: Resource allocation is one of the major problems in a healthcare system. The problem gets worse during a disaster situation like a pandemic. The demand is high, and the resources are limited, this puts a question on which patient should be admitted. This research focuses on increasing the number of patient admissions by deciding who should be admitted based on the patient's severity and length of stay. In the research, the patient has a fixed length of stay and if a patient is rejected, he/she is rejected forever. The research focuses on one critical resource, and it needs to be available to admit the patient. The daily capacity of resources is fixed. The demand during the pandemic is dynamic, so this research solves the problem using space search optimization which is one of the techniques of dynamic programming. The research uses depth-first and breadth-first space search algorithms. The experiments with randomized data were executed to see the effect of available capacity on the number of admitted patients. This research shows that space search algorithms are useful techniques for the optimization of resources and patients during a pandemic.

Keywords: pandemic, optimization, healthcare resources, space search

1. INTRODUCTION

A pandemic is a situation when a disease outbreak spreads to large areas, such as several countries or continents or the whole world (“WHO | What Is a Pandemic?” n.d.). There are approximately 1.7 million viruses that can infect humans from animals and birds, and, according to some scientists, the next pandemic can be around the corner (“Why Scientists Are Rushing to Hunt Down 1.7 Million Unknown Viruses | Live Science” n.d.). Situations like pandemics can disrupt almost every part of human life, but the most affected is the healthcare system.

The healthcare systems get overwhelmed with the patients’ load due to limited resources (Emanuel et al. 2020), (Baheti and Nayak 2022). Resources being limited means that not everyone can get treatment. This raises a lot of ethical questions as to who should be treated and how fair allocation can be made (Emanuel et al. 2020). During Covid-19 doctors' moral ethics were tested as they had to make decisions on who needed to be taken out of the ventilator so that it could be provided to other patients. Various researchers have done studies on what should be criteria for admissions and some have suggested age should be a priority. But there is not much done in creating a model for patient admission (Emanuel et al. 2020)

This research focuses on patient admission during situations like a pandemic. In the research, a patient's length of stay and day of arrival are known and the decision is to be made on who should be admitted to increase the overall number of patients taken in a given period. This research uses space search optimization to solve the patient admission problem.

This research is important in several ways: a) it provides hospital managers with a model to decide who to admit and who not to admit, and b) The research uses space search algorithms to solve the presented problem, space search has not been used in literature to solve patient admission problem. Section 2 provides a brief literature review of relevant research. Section 3 presents the problem statement. The solution methodology is given in Section 4, and its implementation in Section 5. The paper ends with conclusions in Section 6 and used references.

2. LITERATURE REVIEW

The literature on the pandemic generally focuses on the resource allocation problem, meaning how resources can be distributed to increase the number of patients taken. The research generally uses operation research techniques such as mathematical modeling or simulation to solve these problems. Prieto and Gomez solved the resource allocation using a risk-based algorithm. The authors’ objective was to distribute patient and resources among hospitals to minimize the spread. The authors model detects which locations needs to be prioritized to minimize the spread of pandemic (Prieto and Gomez 2021). Chen et. al. used the prediction to find the resource flow during a pandemic and then based on the predictions optimally allocated the resources in different regions (Chen, Sun, and Han 2022). Rico et.al did research on allocation of nurses in a hospital during influenza outbreak. The authors' objective was to find the optimal number of nurses to increase the number of patients taken. The authors used simulation to create various scenarios to find the optimal number of nurses in a hospital (Rico, Salari, and Centeno 2007).

Singhrova created an intelligent healthcare system to allocate the movable resources such as ventilators, ambulances among hospitals to increase the number of patients getting treatment during Covid-19 (Anu and Singhrova 2022). Mbah and Gillian showed that applying quantifiable strategies such as using treatment measure and reproductive number may provide fair treatment to everyone. They said that fair treatment to everyone can minimize the spread (Mbah and Gilligan 2011). Sun demonstrated through mathematical model that in pandemic resource sharing and patients among hospitals can reduce the patient load and overall burden on hospitals. The author said that it is important to share resources and patients during such overwhelming situations. The author's model has two objectives: first is to minimize total travels distance by patient to hospital and second is to minimize the maximum distance patient travel to a hospital. The author's model also predicts the resource shortage and optimizes the resource allocation when they become available, the allocation is done based on patients' cost to access care, minimizing travel distance and burden. The authors showed through their model that patient and resource allocation can reduce the overall burden during such crisis (Sun 2011).

Vaccine distribution is equally important to control the spread of the virus. Venkatramanan et al. showed that optimal vaccine allocation reduces the spread of influenza outbreak (Venkatramanan et al. 2019). Tuite et.al studied the effects of vaccines to control the spread of H1N1 virus. The authors used simulation to create various strategies to optimally allocate vaccines and found that regardless of the age if the vaccines are allocated to high severe patients it can decrease the morbidity and spread of the virus (Tuite et al. 2010).

Sormaz and Malik proposed a mathematical model for the patient admission to find out who should be admitted to increasing the number of patients taken. The patients in their model had priority based on the severity of the patient. The author's model has capacity as a constraint and studied the number of resources required to increase the number of patients (Sormaz and Malik 2023). Fattahi et. al. created a multi-stage stochastic program model for the distribution of resources and patients in a pandemic. The authors' model is multi-stage model and uses stochasticity to capture the uncertainty in demand and resource requirement during pandemic. Since, the situation in pandemic is dynamic, meaning the demand and resource requirement continuously change over time and is not linear. To address this issue, they used the approach of rolling horizon, to make their model closer to the real-world problem The authors' tested their model on two real world case studies, in one case study they focused on distributing resources and the other they focused in both resources and patients distribution. The authors concluded that resource distribution and patient sharing can increase the number of patients getting treatment (Fattahi et al. 2023). Liu et. al. used integer programming to solve the resource allocation problem during pandemic dynamically. The authors' objective was to minimize the total cost of logistics of medical resources. Their problem concentrated on transshipment of resources from supplier to distribution centers and from distribution centers to hospitals and found that dynamic models are more effective for resource distribution during pandemic (Liu and Li 2022). Space search optimization has not been explored much for optimization of resources and patient distribution during emergency like situations such as pandemic. But the technique has been proven to be useful in allocating resources. Costa Filho et al. used the depth search technique with heuristics to allocate the physician in a healthcare setting. The problem they solved has constraints such available hours of physicians, number of days a physician works. The authors' objective was to efficiently allocate human resources to increase the overall number of patient treatments. The authors used combination of heuristics such as minimum remaining values, grade heuristic to solve their problems and found that their method is effective in distributing the resources and minimizes the cost at the same time increases the patients getting treatment (Costa Filho et al. 2012).

Space search is a problem-solving technique where a search space is being explored to find the solution (Luger 2005). The space search uses blind or informed search algorithms to make decisions on patients in a period when the capacity is a constraint. Space search is an artificial intelligence technique that uses search algorithms to find the optimal solution (Cortes et al. 2017). To our knowledge this is the first research that explores the methodology of space search on patient admission during methodology. The research will help in understanding the use of space search algorithms in solving problems of improvement and optimization of health systems operations.

3. PROBLEM STATEMENT

This research proposes a solution to the problem of patient admission in situations where resources are limited. The research uses random data of patients, where patient length of stay and day of arrival is known to the hospital managers in advance. If a patient is rejected, the patient is rejected forever in current research. To illustrate the problem, let us consider ad sample data shown in table 1 there are four patients and five days. The arrival of patients P1 and patient P2 is day 1 and they are going to stay for four and three days respectively. The arrival of patient P3 is on day 2 and he/she stays for 3 days, while patient P4 arrives on day 5 and stays only one day. Since the capacity is two and if we take patient P1 and P2 on day one, we cannot take patient P3 on day 2, or we can take one of the two patients on day 1 and take patient P3 on day 2. All four patients cannot be taken, since in days 2 and 3 we have demand for 3 units of the resource, but we have capacity of only 2. In day 5 we cannot utilize all capacity.

So, in the current problem the focus is given in periods when the capacity is a constraint for who should be admitted, if we take a patient today then this decision might decrease the overall number of patients in a whole period. Thus, the objective is to find the best combination of patients based on the capacity to increase patients taken. We will consider this problem as a sequence of one day problems, where for every day we need to make a decision on which patients to admit, which is suitable for the space search algorithms. The objective will be to maximize the number of patients for given capacity constraints.

Table 1: Description of sample data

Patients	Day 1	Day 2	Day 3	Day 4	Day 5
P1	1	1	1	1	
P2	1	1	1		
P3		1	1	1	
P4					1
Total Patients	2	3	3	2	1
Capacity	2	2	2	2	2

The search algorithm creates combinations of patients based on the number of available capacity and makes decisions based on it.

4. METHODOLOGY

This research uses the space search technique to solve the patient admission problem. The search algorithms uses the graph theory and graph representation of the problem space and traverse through nodes of the graph to solve the problem (Luger 2005). The following subsections will describe state representation and transition, breadth-first algorithm and depth-first algorithm as applied for the presented problem.

4.1 State Representation and Transition

State representation reflects the decisions made by the hospital for any given day in operation. It carries the following details: current day, current patients and provides method for state transition and evaluation. The state space search transition for the example in Table 1 is shown in Figure 1. In Figure1 the arrows going out of a node (represented as a circle) are the resulting states for decisions from a current state or node. The root of the state space search graph is start and its resulting children according to capacity two on day 1 are (none), P1, P2, or (P1,P2) where P1 and P2 for various choices at day one. Based on choice the appropriate capacity will be reserved for patients for day 1, but also for future days. That means if we choose the state (P1,P2), then in day 2 we cannot choose P3, since there is no available capacity. After day one we expand each state based on the day 1 decisions. The children of none nodes are P3 and (none) – patients P1 and P2 are lost, as we can still have none patients on day 2. Similarly children of node P1 are

none, P1 and (P1,P3). Children of P2 are (none and P2), (P2 and P3). The children of the (P1,P2) state are (none) as there is not enough capacity. This expansion then continues until the end of the whole period of operation. In the end, the solution in the path from start to goal which admits the most patients. In our example it would be with admitting three patients, three alternative solutions, S1(P1,P2,P4), S2(P1,P3,P4), S3(P2,P3,P4). For larger problems it is not trivial to come with those solutions.
The search algorithms are applied on the state space graph to find the optimal or a good solution. The research uses two types of space search algorithms to solve this problem: breadth first algorithm and depth first algorithm.

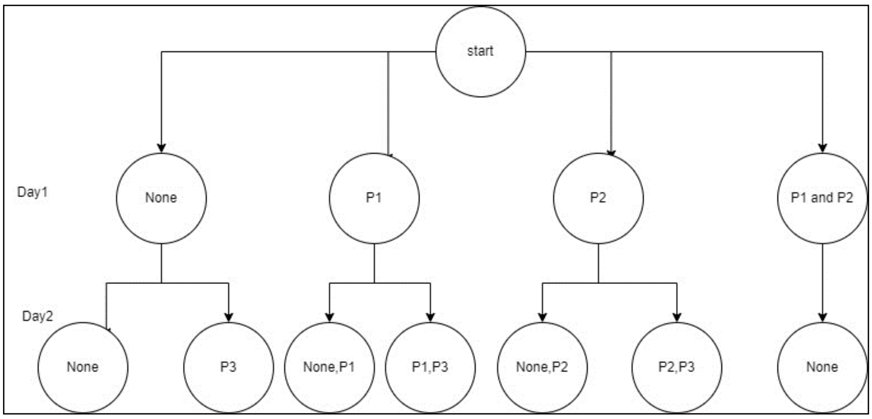

Figure 1: State transition diagram

4.2 Breadth First Algorithm

The breadth first algorithm traverses the graph nodes level by level. If there are no more states to investigate in a level, it advances to the next level (Luger 2005).The breadth first algorithm for Table 1 example would start by examining the start state, which is start, according to the breadth first approach. After visiting this stage, it will be closed, which means it will not be examined or explored further. The open states are the offspring of the start state, which are none, P1, P2, and P1,P2, where P1, P2 represent the patients in this problem. It would next visit state none and add its children in the queue of open states. The algorithm explores states in a systematic way, moving through them level by level. It doesn't stop until it has examined all possible states.

4.3 Depth First Algorithm

The depth first method adheres to the principles of last in, first out, and the search space becomes deeper as it explores in length. The backtrack algorithm is then used to discover the solution(Luger 2005). The depth first algorithm for the Table 1 example would start with the start node and then investigate it. The open states would be the start node's children (none, P1,P2, and P1,P2). The program would then investigate (P1,P2) and place it in a end states. (none,P1,P2, none) is the open state. The solution will then investigate the none state and place it in the close state ((P1,P2), none) because it has no children; the open state will remain unchanged.
The following state would be P2, and the open state would add its children (none, P1,(none,P2), and (P2,P3). (P2, P3) is the next state to be visited. The solution would continue to go deeper and would come to an end after visiting all the states or after finding the goal state.

4.4 State Evaluation

As it was mentioned, the objective in our probelm i sto maximize number of admitted p[atiens, which is equivalent to minimize teh number of rejected patients. When loking teh the state transition explaind in section 4.1, four possible choices on day 1, namely (Npone), P1, P2, adn (P1,P2) have 2, 1, 1, and 0 rejected patients respectively. Therefore, it would be beneficial to introduce informed search and expend states with minimal number of rejected patietns first.

The search termination happens when we reach the last day of the whole period (in our example day 5) or when there is no chance to further improve. In this paper we just report teh blind search, so termination happens at teh end of the whole period.

5. IMPLEMENTATION

In this section we present the current implementation of the space search algorithm within SpaceSearcher tool developed in previous work. We include a brief introduction to SpaceSearcher, and then details of implementation for this research.

5.1 SpaceSearcher for Patient Admission

The proposed space search procedure has been implemeted within the framework of *SpaceSearcher*, java based tool for exectuion and visualization of problem-solving algorithms (Sormaz, Dusan and Rajaraman, Srinivas 2003). The tool has been applied on variety of problems in manufacturing (Šormaz and Rajaraman 2008). The tool implements generic search algorithms while problem specific state transition and evaluation has to be done for each problem model for patient This research imports the SpaceSearcher library into its worksapce to use the algorithms, but it provides state representation, state trnasition, and state evaluation metohds.

5.2 PeriodicProblem Implemetation

Implementation inlcude two classes: *Patient*, to represent the data about each patient who comes to the hospital and *PeriodicProblemDay*, which extends *ComparableSpaceState* from *SpaceSearcher*, to implement state represetation, state transition, and stateevaluation methods needed by *SpaceSearcher*. The following methods are created in the model to solve this problem:

- main *Method:* This method creates the current state and goal state of the problem. The method calls the SpaceSearcher and graphically displays the results of the algorithms being used.
- createPatients *Method:* This method uses the *Patient* class. The Patient class provides information about the patients, the class has constructor that creates objects of patient with length of stay , name of the patient and their arrival day. The *createPatients* method uses this class and stores the information of the patients based on the data provided in a map or from a file. The method sorts patients on a daily basis and groups patients who come on the same day into one list in the map.
- makeNewStates *Method:* This method creates new states based on the current state of the problem. The code for *makeNewStates* method is shown in Figure 2. The method runs as follows, in the beginning of the method a set of searchable states, list of patients "newStPat" and nextDyPat are being created.

The nextDyPat stores the information of the next day patients. The code begins with the for loop on the decisions that resulted in the current state of the search, if the patient in the current state is staying over nextday , the code then add this patient in the newStPat and in the states which stores the information of the resulting states from the current state. This ensures that the patients that are staying over next day are added in the resulting state from the current state decisions.
Once the patients from previous states are added in the current state, the method then check there are patients on next day of the problem and is there is enough capacity.
If both are true , it then runs the creates the combinations on the number of next day patients based on the group of remaining capacity. The combinations are created by calling createCombinations method. The combinations are then added in the nextDayComb list. A new list of patient is created that carries the patients from the previous decisions, the elements from the combination list are then added in the new patient list created and finally added in the states. The states are then returned as the result of the method.

```
public Set<Searchable> makeNewStates()
{

    Set<Searchable>  states=  new HashSet<Searchable>();
    ArrayList <Patients> newStPat= new ArrayList <Patients> ();
    ArrayList <Patients> nextDyPat= map.get(currentDay+1);
    for (Patients p2 : statePat)
    {
        if(p2.isStayingDay(currentDay+1))
        {
            newStPat.add(p2);
        }
    }
    states.add(new PeriodicProblemDay(this,newStPat,currentDay+1));
    if (nextDyPat !=null)
    {
        int nextDayCap= nextDayCap();
        if (nextDayCap>0)
        {
        List<Patients> nxtDyPatIter = nextDyPat;
        List<List<Patients>> nextDayComb=new ArrayList <>();
        try {
            for (int i=1;i<=nextDayCap;i++)
            {
            List<List<Patients>> combinations = Comb.createCombinations(nxtDyPatIter, i);
            nextDayComb.addAll(combinations);

            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        for (List<Patients> ndi :nextDayComb)
        {
            ArrayList <Patients> allNewPat= new ArrayList <Patients> (newStPat);
            for (Patients i : ndi )
        {
            allNewPat.add(i);
        }
            states.add( new PeriodicProblemDay(this,allNewPat,currentDay+1));
        }
    }
    }

    return states;
```

Figure 2: The code for makeNewStates *method*

5. RESULTS

The model was run on a randomly generated data. The input data is shown in *SpaceSearcher* GUI in the Figure 3 in the left side of right panel. According to the table patient one comes on day 1 and stay for three days, patient two come on day 1 and stay for two days and so on. The capacity is two. This is the shortened versionj of the problem in Section 3 inorder to show completion of algorithms runs. The model has also been run on models with 20 patients and 10 days. The results of the algorithms are described below:

- Breath First Alorithm: Based on the capacity and patient arrivals, the states are created and each state creates its children states. The algorithm was run and the goal state in this case was the number of days defined in the data set. Figure 3 shows the breadth first search, the left side of the figure shows the expansion of the state space after every satet is explored. Integre numbers in brackets ([1], [3], [4], [5], etc) show the order of state expansions. State transitions are shown as children nodes for a given node, with color codes as: red are the states that have been already explored, magenta is the current state, blues are the children or decisions for each state that have not been explored yet. . Accoding to breadth first, it first investigated state number [3]. Decision number three takes no patients on day one, thus if it takes zero patients on day one, it can accept zero or patient number three who arrives on day two.

The search then moves on to the state [4], where patient one is taken, therefore children from this state would be patient one on the second day as it stays for another two days and does not take anyone else, or it can admit patient three. The algorithm currently runs until the number of days indicated in the objective state is completed. The state tree shown visually how breadth first expands level by level (in our problem day by day) before moving to the next day. The highlighted state [24] shows the one of best states which will be explored in the next round of the algorithm. The panel on the far right shows the current patients in this state, P3 and P4 (It does not show P2, since that patient has already left).

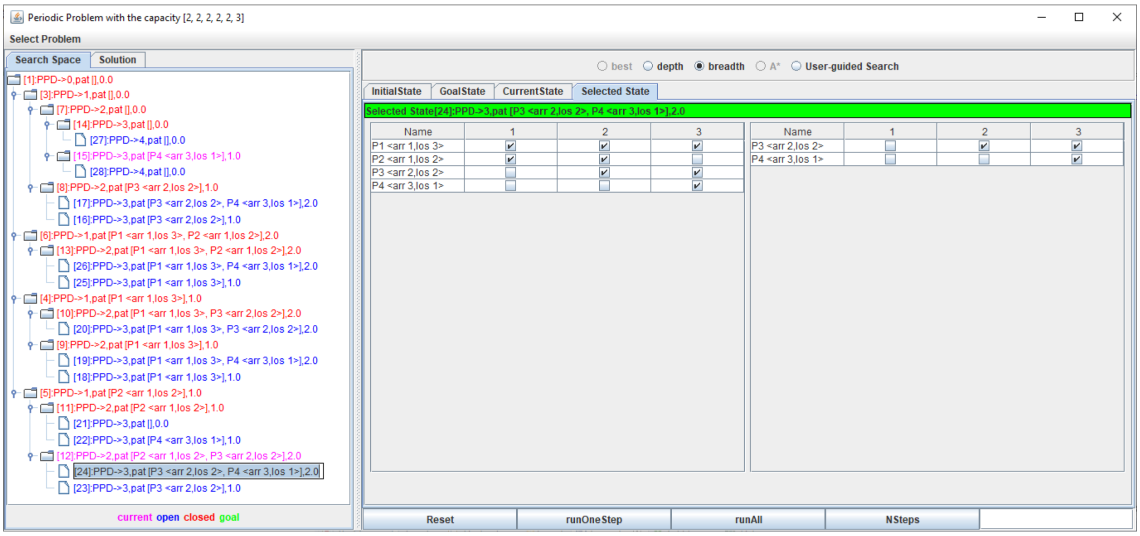

Figure 3: Breadth First Algorithm Result

Depth First Algorithm: According to the depth first algorithm it first explores state [6], which means taking patients one and two. The total patients count is 2 as shown in the Figure 4. After this, it explores state [7], where it can not take new patient P3, then state [9] where Patient P2 is gone, and in the next step I twill explore state [10] and take patient P4. The total patient count is 3. Then it explores state 10 . In state 10 the day is three and patient 4 comes the total patients in the system currently is 2 (patient 1 and patient 4). The states continue to explore until day 3. The state tree on the left shows the depth first behavior in which only one state in each day has been explored so far. The panel on the far right shows the current patients in this state, P3 and P4 (It does not show P2, since that patient has already left), the same as in breadth-first algorithm.

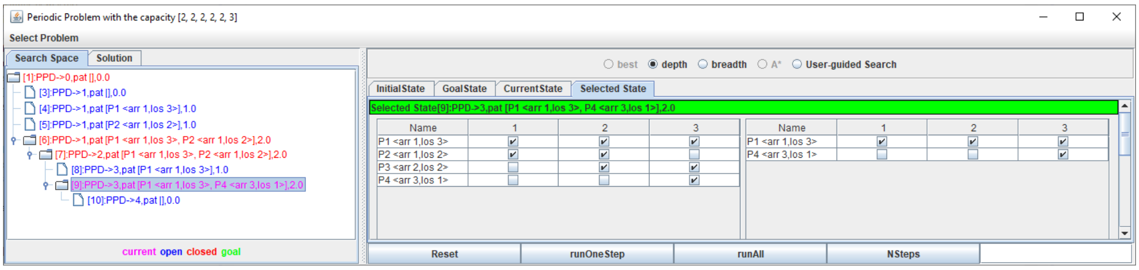

Figure 4: Results of depth first algorithm

According to the current scenario the maximum number of patients taken would be 3, as shown in the both algorithms. The algorithm shows combinations of patients based on the capacity and shows successfully the patient that needs to be taken to maximize the result. Running algorithms further wwoudl also detect other optimal states as mentiond in Section 3.

6. CONCLUSION

This paper presents the initial results of an ongoing research. The research explores the applications of space algorithms in creating models for patient admissions. The research shows that space search algorithms can be used in making decision on patient admission when capacity is the constrained. The future work in this research includes the implementation of state evaluation method and search termination criteria that would avoid the exhaustive search, The future work also includes running the data sets with larger number of patients and days and experimentation and optimization on real data.

7. REFERENCES

Anu, and Anita Singhrova. 2022. "Optimal Healthcare Resource Allocation in Covid Scenario Using Firefly Algorithm." *International Journal of Engineering Trends and Technology* 70 (5): 240–50. https://doi.org/10.14445/22315381/IJETT-V70I5P226.

Baheti, Akshay D., and Prakash Nayak. 2022. "Covid-19 in India: Oxygen Shortages and a Real World Trolley Problem." *BMJ* 376 (February): o369. https://doi.org/10.1136/bmj.o369.

Chen, Zhen-Yu, Minghe Sun, and Xi-Xi Han. 2022. "Prediction-Driven Collaborative Emergency Medical Resource Allocation with Deep Learning and Optimization." *Journal of the Operational Research Society* 0 (0): 1–14. https://doi.org/10.1080/01605682.2022.2101953.

Cortes, Adrian, Calliope A Dendrou, Allan Motyer, Luke Jostins, Damjan Vukcevic, Alexander Dilthey, Peter Donnelly, Stephen Leslie, Lars Fugger, and Gil McVean. 2017. "Bayesian Analysis of Genetic Association across Tree-Structured Routine Healthcare Data in the UK Biobank." *Nature Genetics* 49 (9): 1311–18. https://doi.org/10.1038/ng.3926.

Costa Filho, Cicero Ferreira Fernandes, Dayse Aparecida Rivera Rocha, Marly Guimarães Fernandes Costa, and Wagner Coelho de Albuquerque Pereira. 2012. "Using Constraint Satisfaction Problem Approach to Solve Human Resource Allocation Problems in Cooperative Health Services." *Expert Systems with Applications* 39 (1): 385–94. https://doi.org/10.1016/j.eswa.2011.07.027.

Emanuel, Ezekiel J., Govind Persad, Ross Upshur, Beatriz Thome, Michael Parker, Aaron Glickman, Cathy Zhang, Connor Boyle, Maxwell Smith, and James P. Phillips. 2020. "Fair Allocation of Scarce Medical Resources in the Time of Covid-19." *New England Journal of Medicine* 382 (21): 2049–55. https://doi.org/10.1056/NEJMsb2005114.

Fattahi, Mohammad, Esmaeil Keyvanshokooh, Devika Kannan, and Kannan Govindan. 2023. "Resource Planning Strategies for Healthcare Systems during a Pandemic." *European Journal of Operational Research* 304 (1): 192–206. https://doi.org/10.1016/j.ejor.2022.01.023.

Liu, Ya, and Zhaojin Li. 2022. "Dynamic-Programming-Based Heuristic for Multi-Objective Operating Theater Planning." *Journal of Industrial and Management Optimization* 18 (1): 111. https://doi.org/10.3934/jimo.2020145.

Luger, George F. 2005. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Addison-Wesley.

Mbah, Martial L. Ndeffo, and Christopher A. Gilligan. 2011. "Resource Allocation for Epidemic Control in Metapopulations." *PLOS ONE* 6 (9): e24577. https://doi.org/10.1371/journal.pone.0024577.

Prieto, Jeisson, and Jonatan Gomez. 2021. "Multi-Objective Risk-Based Resource Allocation for Urban Pandemic Preparedness: The COVID-19 Case in Bogotá, Colombia." medRxiv. https://doi.org/10.1101/2021.02.24.21252407.

Rico, Florentino, Ehsan Salari, and Grisselle Centeno. 2007. "Emergency Departments Nurse Allocation to Face a Pandemic Influenza Outbreak." In *2007 Winter Simulation Conference*, 1292–98. Washington, DC, USA: IEEE. https://doi.org/10.1109/WSC.2007.4419734.

Šormaz, D. N., and S. N. Rajaraman. 2008. "Problem Space Search Algorithm for Manufacturing Cell Formation with Alternative Process Plans." *International Journal of Production Research* 46 (2): 345–69. https://doi.org/10.1080/00207540601138544.

Sormaz, Dusan, and Mandvi Malik. 2023. "Model for Acceptance and Rejection of Patients Using Critical Resources During Pandemic-Like Situation." In . New Orleans: Institute of Industrial and Systems Engineers.

Sormaz, Dusan, and Rajaraman, Srinivas. 2003. "Interactive Tool for Modeling and Visualisation of Cell Formation Algorithms." In *Proceedings of Group Technology/Cellular Manufacturing World Symposium*. Ohio University.

Sun, Li. 2011. "Optimization Models for Patient Allocation during a Pandemic Influenza Outbreak." University of Louisville. https://doi.org/10.18297/etd/1405.

Tuite, Ashleigh, David N. Fisman, Jeffrey C. Kwong, and Amy Greer. 2010. "Optimal Pandemic Influenza Vaccine Allocation Strategies for the Canadian Population." *PLoS Currents* 2 (January): RRN1144. https://doi.org/10.1371/currents.RRN1144.

Venkatramanan, Srinivasan, Jiangzhuo Chen, Arindam Fadikar, Sandeep Gupta, Dave Higdon, Bryan Lewis, Madhav Marathe, Henning Mortveit, and Anil Vullikanti. 2019. "Optimizing Spatial Allocation of Seasonal Influenza Vaccine under Temporal Constraints." Edited by Cecile Viboud. *PLOS Computational Biology* 15 (9): e1007111. https://doi.org/10.1371/journal.pcbi.1007111.

"WHO | What Is a Pandemic?" n.d. WHO. World Health Organization. Accessed June 1, 2021. http://www.who.int/csr/disease/swineflu/frequently_asked_questions/pandemic/en/.

"Why Scientists Are Rushing to Hunt Down 1.7 Million Unknown Viruses | Live Science." n.d. Accessed September 7, 2023. https://www.livescience.com/61848-scientists-hunt-unknown-viruses.html.