Equity distribution of quality evaluation reports to doctors in health care organizations

Mahdi Jemmali[1,2,3], Loai Kayed B. Melhim[4], Abdullah Alourani[1] and Md. Moddassir Alam[4]

[1] Computer Science, Majmaah University, Zulfi, Riyadh, Saudi Arabia
[2] Computer Science, Mars Laboratory, Sousse, Tunisia
[3] Computer Science, Higher Institute of Computer Science and Mathematics, University of Monastir, Monastir, Monastir, Tunisia
[4] Computer Science, Hafr Batin University, Hafr Batin, Saudi Arabia

ABSTRACT

There are volumes of patient reports generated in any healthcare organization daily. The reports can be very lengthy or of few pages. Maintaining records of patients is essential for ensuring quality medical care. Doctors, apart from their routine activities, are also responsible to sort, examine and archive the generated reports. However, this process consumes doctors' time, who are already hard-pressed for time. The objective of this study is to search for a method that can assign reports to doctors to ensure equitable and fair distribution of the overall workload. As a part of the solution, a mathematical model will be proposed to perform different developed heuristics. An experimental evaluation using different classes with a total of 2,450 different instances will be tested to measure the performance of the developed heuristics in terms of, elapsed time and gap value calculations. The clustering heuristics which is based on two groups is the best heuristic with 96.1% for the small instances and 98% for the big scale instances. The contribution of this work is based on employing dispatching rules with several variants; randomization approach, clustering methods; probabilistic method, and iterative methods approach to assign all given reports to doctors while ensuring the equitable distribution of the paper workload.

Corresponding author
Loai Kayed B. Melhim,
l.banimelhim@uhb.edu.sa

INTRODUCTION

Doctors time in a healthcare organization is very precious. They spend time in face-to-face interaction with patients' gathering information, doing administrative work such as paper tasks related to visits, quality, *etc*. Furthermore, a doctor spending more time with a patient is a requirement for value and efficiency in health care delivery. Assigning excess paperwork tasks has to be compensated with the allotted time for patient–doctor interaction. Further, it may affect their performance, elevate stress, lead to job dissatisfaction, may cause disruptive behavior and may also affect how doctors provide health care. This can also lead to medical errors, deficiencies in safety and low-quality health care services.

Thus managing the workload of doctors is very imperative. As doctors differ in the way they work based on their specializations, experience, job position and other functional or administrative considerations. Keeping this into consideration, fair distribution of doctor workload is very essential. Although studies have been conducted on optimization of doctor's workload, to the best of the authors' knowledge, none of the available studies has stressed the equity distribution of paper workload to doctors. Based on the above arguments, the present research aims at searching the problem of distributing paperwork load between doctors by developing a suitable algorithm to equitably distribute these paper works. The proposed study will have several positive implications such as suitable load distribution will ensure adequate time for patient–doctor interaction, helping the doctor to understand the patient disease state and prescribe better treatment and care. It will also allow doctors to spend their time updating their medical knowledge and devoting more time to clinical research.

Load balancing or equity distribution is defined as: the effective distribution of task loads among a group of workers, whether they are people, computers or applications (*Jemmali & Alquhayz, 2020*). Load balancing helps ensuring scalability and availability of services, which enable organization to provide more services with less efforts (*Alharbi & Jemmali, 2020*; *Jemmali, 2019b*; *Alquhayz, Jemmali & Otoom, 2020*). It is important to note that the term equitable distribution here does not mean equal distribution, but rather committing to a set of preferences during the distribution of tasks in a way that ensures fairness among the group of workers according to given preferences. The problem of maximization of the minimum aim for the load balancing on machines is treated in several research works (*Haouari & Jemmali, 2008*; *Jemmali, Otoom & al Fayez, 2020*).

The problem of load balancing was discussed by many researchers who aimed to present a suitable solution for this problem. For example, *Jemmali (2019a)* presented several approximate solutions for the problem of load balancing. While mathematical modeling of load balancing with a new proposed objective function that determines the indicator to select among the proposed heuristics was presented by *Jemmali (2019b, 2021c)*. In the same context, the exact solution of the load balancing problem applied on the distribution of projects was presented in *Alharbi & Jemmali (2020)* and *Jemmali (2021b)*. The machine-learning technique (decision tree) and the application of particle swarm optimization (PSO) presented by *Bany Taha et al. (2020)* can be utilized to extend some meta-heuristics for the load balancing problem. Another application of the load balancing approach is proposed in *Jemmali (2021a)* to develop new solutions for the smart parking problem.

The importance of load balancing emerged through the different areas that utilized this concept. Load balancing algorithms were presented for the aviation sector to be applied for gas turbines engine in *Jemmali et al. (2019)* and *Jemmali, Melhim & Alharbi (2019)*. Recently, *Alquhayz, Jemmali & Otoom (2020)* utilized load balancing to present solutions for the dispatching files to different disc spaces. Moreover, equity distribution was implemented by *Jemmali & Alquhayz (2020)* where the authors proposed new network architecture by adding a new component called "scheduler". This component is

responsible to find the better schedule that ensures the best load balancing of packets to be forwarded to routers.

Also, *Taha & Chowdhury (2020)* proposed a new load balancing algorithm for E-Health system called (GALB) based on Genetic Algorithm (GA). To reduce latency and overhead while sending E-health encrypted data to the cloud, the tasks received by GALB algorithm will be distributed between main gateway and E-health nearby devices based on distance, task complexity and encryption policy length. In the area of health care, load balancing was employed by *Friesen et al. (2011)* to control the number of patients flow in six emergency departments within a mid-sized Canadian city. While *Kolomvatsos, Panagidi & Hadjiefthymiades (2015)* presented load balancing model that was oriented to distribute health care workers into a finite number of resources in a post-emergency scenario for both indoor or outdoor applications. Similarly, *Stock et al. (2021)* used load balancing to assist patients' load across 12 hospitals.

Utilizing load balancing in healthcare organizations to derive equitable or fair distribution, was adopted by many researchers. For example, *Schaus, Van Hentenryck & Régin (2009)* presented constraint programming models to balance the workload of assigning daily tasks of newborn infant patients to nurses, while satisfying many other constraints. The presented approach produces an approximate solution with an accepted performance and it did not consider the different nurse qualifications that would affect the ability to perform the assigned tasks, which in turn can affect the performance of the whole system. The concept of load balancing in health organizations to balance the loads of health care assigned tasks and paperwork loads, was presented by *Zhong et al. (2018)* where the authors investigated the staffing proper ratio of doctors and support staff to provide the required quality of the health care services.

The impact of load balancing on doctors' performance and on the quality of the delivered health care, was presented by many authors. For example, *Christino et al. (2013)* presented a discussion of an online survey, where they reported that paper workloads were recognized as potential barriers to doctors performance, provided services, health care quality, resident education and on the medical profession in general *i.e.* more workloads mean more interruptions and less productivity in health services. To reduce doctor's unnecessary interruptions, *Weigl et al. (2012)* investigated the relation between workflow interruptions and doctors' ability to manage their workload in a guarded and productive style. The objective of their study was to help doctors focus more on providing proper health care services by suggesting the smooth distribution of workloads within healthcare institutions. The smooth distribution may reduce unnecessary interruptions but it will not indicate clearly the amount of the workload assigned to each doctor. The answer to this was by attempted by *Woolhandler & Himmelstein (2014)*, who proposed a method to quantify the number of administrative tasks and paper workload assigned to each doctor in the healthcare organizations. The researchers utilized an online-survey that was distributed to more than 4,720 doctors in the United States. The authors stated that doctors assigned administrative tasks are likely to have more paper workloads, which may decrease their career satisfaction and productivity. The reviewed researches showed that the most common complaints among doctors working on health organizations was

the amount of paper workloads assigned to them. There are many proposed solutions for such problems, such as the equitable distribution of tasks especially the paper workloads or the load-balancing. Equitable distribution means the fair distribution or the load balancing of the assigned tasks among doctors in health organizations.

But no previous studies, till the writing of these lines utilize the load balancing techniques to equitably distribute paper workloads to achieve fair distribution of paper workloads among doctors in health organizations.

For the performance and safety of healthcare delivery, an equitable distribution of paper workload should be applied in health care organizations to give the doctors more space and more time, so that they have the ability to focus more on their clinical duties. The fair distribution of paperwork loads may be achieved by carefully distributing managerial workloads. In the proposed work, a set of main ideas will be presented, which are summarized as follows: The first idea, is to divide the doctors into groups based on the nature of the tasks assigned to them. The second idea, is the distribution of paperwork to the doctors in these groups, based on criteria that will be determined later. Finally, distribute the paperwork within the same group fairly among the group members so that the members of one group are equal in terms of the papers count for each of them. The nature of the paperwork assigned to each doctor must be taken into consideration in terms of specialization and in terms of the type of the required paperwork, as there is some paperwork that cannot be divided and requires completion by the same doctor who start processing it.

The rest of this paper will be organized as follows, "Methods" presents methods and the problem description, while the proposed heuristics will be presented in "Heuristics". Discussion of the experimental results will be presented in "Experimental Results". Finally, the conclusion will be presented in "Conclusion".

METHODS

This section presents the main stages that are used to solve the problem of entity paper workload distribution, which will be discussed in details in the coming sections.

Solution road map

A set of main ideas will be presented In this work, which can be summarized as follows: The first stage is to divide the doctors into groups based on the nature of the tasks assigned to them. The second stage is the distribution of paperwork to the doctors in these groups, based on criteria that will be determined later. Finally, distribute the paperwork within the same group fairly among the group members so that the members of one group are equal in terms of the number of papers required to be executed by each of them. The nature of the paperwork assigned to each doctor must be taken into consideration in terms of specialization and in terms of the type of the required report, as there is some paperwork that cannot be divided and requires completion by the same doctor who start processing it.

Problem description

The problem of paper workload distribution is described as follows. The paper work load is in the form of reports, let R be the set of reports that will be assigned to different doctors. These reports will be processed by the assigned doctors then will be sent to the concerned committees for final confirmation. While, the number of independent reports is denoted by n_r and the number of doctors in the concerned health organization is denoted by n_d. The set of doctors D is defined as follows $\{D_1, \cdots, D_{n_d}\}$. The index of each doctor is specified by i. Each report is characterized by the estimated processing time. This time depends on the number of pages of the assigned report. In this research the report pages count will be used as an indicator for the time estimation calculations. Therefore, each report j is defined by its pages count and will be denoted by np_j. All reports have pages counts that are defined in the set denoted by $\{np_1, \cdots, np_{n_r}\}$. The total number of pages assigned to each doctor i is denoted by N_{D_i}, while the number of pages assigned to doctor i when report j is assigned is defined as Cn_j. The objective of this research is to ensure the equity distribution of reports pages between doctors. To achieve this goal, this work should present an indicator that can calculate the efficiency of the distribution process. This indicator will be known as T and will be expressed in Eq. (1). T is the gap value between each total number of pages assigned to every doctor i and the minimum number of the assigned pages. This gap is the measure of the equity distribution process. A large gap value indicates that the equity distribution is not reached, while a small gap values indicates that a fair distribution is reached. Thus, the objective of this work, is to find a schedule that minimizes the gap values *i.e.* The goal is to minimize T.

$$T = \sum_{i=1}^{n_d}(N_{D_i} - N_{min}). \tag{1}$$

where N_{min} is the minimum number of assigned pages. $N = min_{i=\{1,\ldots,n_d\}}N_{D_i}$. Table 1 represents all notations used in this paper.

Example 1 *Let $n_r = 7$ and $n_d = 2$. The number of pages for each report is displayed in Table 2.*

Figure 1 presents a schedule to show the reports assignment to each doctor.

Figure 1 shows that the minimum total number of pages ($N_{D_2} = 52$) is assigned to Doctor 2. However, the maximum total number of pages ($N_{D_1} = 74$) is assigned to Doctor 1. The gap of total number of pages T between doctors, is calculated by applying Eq. (1) as $T = \sum_{i=1}^{n_d}(N_{D_i} - N_{min}) = (74 - 52) + (52 - 52) = 22$. The main objective is to develop another schedules that should ameliorate the gap between the doctors paperwork. For the same example if another schedule is applied then a different scenario will be obtained as shown in Fig. 2.

Figure 2 shows that the minimum total pages ($N_{D_1} = 59$) is assigned to Doctor 1, while the maximum total number of pages ($N_{D_2} = 67$) is assigned to Doctor 2. Thus, the new calculated gap between doctors after applying Eq. (1) is $(67 - 59) + (59 - 59) = 8$. Comparing with scheduling result shown in Fig. 1, it is clear that the schedule in Fig. 2 is

Table 1 Notations used in this paper with their definitions.

Symbols	Explanation
R	set of reports that will 104 assigned to different doctors
n_r	number of independent reports
n_d	number of doctors in the concerned health organization
$D = \{D1, \cdots, D_{n_d}\}$	set of doctors
i	index of each doctor
j	index of each report
np_j	number of pages
N_{D_i}	total number of pages assigned to each doctor i
Cn_j	number of pages assigned to doctor i when report j is assigned
N_{min}	minimum number of assigned pages
$limit$	number of iteration
H^*	the minimum T values returned after the execution of all heuristics H_1 to H_8.
H	represents the T values returned by the heuristics H_1 to H_8.
$Perc$	is the percentage for each heuristic to reach H^*.
$G = \dfrac{H - H^*}{H}$	if $H = 0$ then $G = 0$. gap value between H^* and H
Ag	average of G for a fixed number of instances.
$Time$	running time in seconds, or the result of "-" if the time is less than 0.001 s.

Table 2 7-2 instance of pages' number reports.

j	1	2	3	4	5	6	7
np_j	20	25	14	12	30	15	10
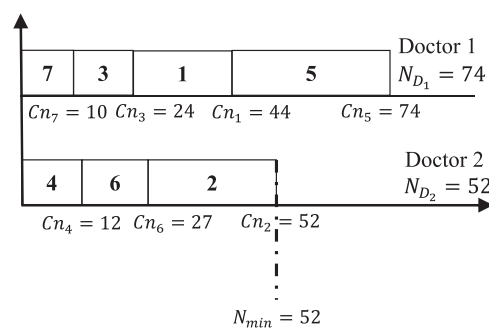
Figure 1 7-2 reports-doctors finishing time distribution.

better that schedule presented in Fig. 1. The gained gap between the two schedules is $22 - 8 = 14$.

Therefore, improving the calculated gap will enhance the fair distribution of reports to different doctors, which will gain more free time to doctors, more resting time and more
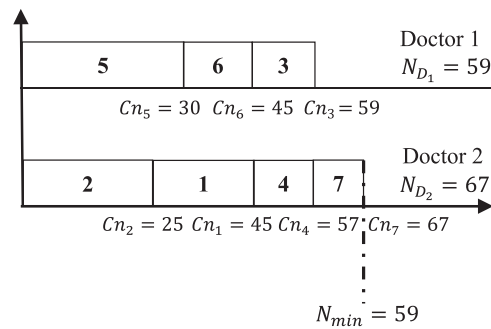
Figure 2 Max–min ameliorated schedule.

time to deliver improved health care services to patients. Considering the number of processed reports throughout the doctor's service period and the number of doctors involved in this process, will demonstrate the amount of time that can be saved and the benefits that can be achieved for doctors, patients and for the health care organizations, if proper scheduling is been adopted. This goal motivates us to present this research and to develop the required heuristics to achieve this goal.

HEURISTICS

In order to solve the proposed problem several heuristics is developed and will be presented in this section. The first two heuristics H_1 and H_2 are based on the dispatching rules. The heuristic H_3 is based on the randomization of the doctor choice, where the choice of the doctor will be performed randomly. Heuristic H_4 will apply the same steps of H_3, except that it will exclude the doctor whose report $j - 1$ has been assigned. While H_5 is obtained after excluding the doctor who has the most workload of reports. For heuristic H_6, it will be responsible for calling H_3, H_4 and H_5 heuristics with iteration and then selecting the best of the returned results. The iteration process will be performed separately for the invoked heuristics H_3, H_4 and H_5. Finally, heuristic H_8 is constructed based on the clustering method. Which means that the classification of reports into groups give us the possibility to find a good solution.

For iteration based heuristics, the choice of 2,000 times was selected based on an experimental test. Indeed, we tested the heuristics for several iterations number. When the iteration number is larger than 2,000, it consumes more time without any significant effect on the results. However, if the iteration number is less than 2,000, for example (50–1,000), this will consume less time, but with a remarkable difference in the results when compared with the 2,000 iterations.

Non-increasing order heuristic (H_1)

This heuristic is based on the arranging of all given reports in a non-increasing order, according to their number of pages. After that, assign the report that has the biggest number of pages count to the doctor who has the minimum number of total pages' count. The complexity of H_1 heuristic is $O(nlogn)$.

Non-decreasing order heuristic (H_2)

This heuristic is based on arranging all the given reports in a non-decreasing order according to their number of pages. After that, assign the report that has the lowest number of pages count to the doctor who has the minimum number of total pages' count. The complexity of H_2 heuristic is $O(nlogn)$.

Iterative random doctor choice without excluding heuristic (H_3)

This heuristic adopts the randomization method. In this heuristic, when there is a need to assign a report, the required doctor to process that report will be chosen randomly. After finishing the schedule of all reports, an iteration loop will be executed to repeat the choice of doctors, this will generate a different schedule for each iteration. For this type of heuristics, three variants are considered. The first variant is the choice of the report to be scheduled according to report index. The second variant is the choice of the report to be scheduled according to the increasing order of the report's pages count. The third variant is the choice of the report to be scheduled according to the decreasing order of the report's pages count. After the execution of these variants the best solution is selected. In practice the iteration loop is executed 2,000 times. The complexity of H_3 heuristic is $O(n^2)$.

The function random(x,y) returns a random integer between x and y. Schedule(r,j) is the function that assigns the report j to the doctor r. While, Incs() is the function that sorts the given reports in an increasing order based on their number of pages. Moreover, Decs() is the function that sorts the given reports in a decreasing order based on their number of pages. In practice the number of iterations is fixed with a value of *limit* = 2,000.

This heuristic is denoted by H_3 and the related heuristic is described in Algorithm 1.

Iterative random doctor choice excluding the last doctor heuristic (H_4)

This heuristic applies the same ideas described in H_3. However, there is a difference in the choice of the assigned doctor. At a given time t when there is a need to assign the report j, select randomly any doctor from the list of the available doctors excluding the doctor whose report $j - 1$ has been assigned. The three variants described in H_3 are also applied in this heuristic. After execution of these variants the best solution is selected. In practice the iteration loop is done 2,000 times. The complexity of H_4 heuristic is $O(n^2)$.

Iterative random doctor choice excluding the most loaded doctor heuristic (H_5)

This heuristic applies the same ideas described in H_3 with a difference the choice of the assigned doctor. At a given time t when there is a need to assign the report j, select randomly any doctor from the list of available doctors excluding the doctor who has the highest load. The three variants described in H_3 are also applied in this heuristic. After execution of these variants the best solution is selected. In practice the iteration loop is done 2,000 times. The complexity of H_5 heuristic is $O(n^2)$.

Algorithm 1 Iterative random doctor choice without excluding heuristic (H_3).
1: Initialize $k = 1$ and $it = 1$
2: **for** ($k = 1$ to 3) **do**
3:       **if** ($k = 2$) **then**
4:           Incs(R)
5:       **else if** ($k = 3$) **then**
6:           Decs(R)
7:       **end if**
8:       **for** ($it = 1$ to $limit$) **do**
9:           **for** ($j = 1$ to n_r) **do**
10:              Set $r = $ random($1, n_d$)
11:              Call schedule(r, j)
12:           **end for**
13:           Calculate T_k^{it}
14:       **end for**
15:       Calculate $T_k = \min\limits_{1 \le it \le limit} T_k^{it}$
16: **end for**
17: Calculate $T = \min\limits_{1 \le k \le 3} T_k$
18: Return T

Iterative random doctor choice heuristic (H_6)

This heuristic is based on the iterative method. For each iteration and for each type of report pages sorting, apply the heuristic H_3, H_4, and H_5 with some modifications. These modifications are based on the functions of these heuristics as described below. The best obtained value is returned. The complexity of H_6 heuristic is $O(n^2)$.

$H_3()$ is the function of the heuristic H_3 without the 2,000 iterations loop. $H_4()$ is the function of the heuristic H_4 without the 2,000 iterations loop. $H_5()$ is the function of the heuristic H_5 without the 2,000 iterations loop. $V()$ is the value returned by the applied heuristic. Indeed, V_k^{it} represents the value returned by $H_k()$ for the iteration it. In practice the number of iterations is fixed to $limit = 2,000$. The instructions of the heuristic H_6 are described in Algorithm 2.

Randomly repeating iteratively doctor choice heuristic (H_7)

This heuristic is based on the iterative method. For each iteration and for each type of report pages sorting, apply the heuristics H_3, H_4, and H_5 with the modifications that are based on the functions of these heuristics as described below. Compared to H_6 the difference will be in the nature of performing the loop iterations. Indeed, for H_6, a loop of $limit$ iteration is performed on all heuristics $H_3()$, $H_4()$ and $H_5()$ for 1 time. However, for H_7, the iteration will be performed $limit$ iterations for each heuristic separately. The best value is returned. The complexity of H_7 heuristic is $O(n^2)$.

Algorithm 2 Iterative random doctor choice algorithm (H_6).

1: **for** (k = 1 to 3) **do**
2:      **if** (k = 2) **then**
3:          Incs(R)
4:      **else if** (k = 3) **then**
5:          Decs(R)
6:      **end if**
7:      **for** (it = 1 to $limit$) **do**
8:          Call H_3() and calculate V_3^{it}
9:          Call H_4() and calculate V_4^{it}
10:          Call H_5() and calculate V_5^{it}
11:      **end for**
12:      Calculate $T_k = \min\limits_{3 \leq i \leq 5} \left(\min\limits_{1 \leq it \leq limit} V_i^{it} \right)$
13: **end for**
14: Calculate $T = \min\limits_{1 \leq k \leq 3} T_k$
15: Return T

$H_3^*()$ is the function of the heuristic H_3 with only one sorting choice. $H_4^*()$ is the function of the heuristic H_4 with only one sorting choice. $H_5^*()$ is the function of the heuristic H_5 with only one sorting choice. $W()$ is the value returned by the applied heuristic. Indeed, W_k^{it} represents the value returned by $H_k()$ for the iteration it. In practice the number of iterations is fixed by $limit$ = 2,000. The instructions of heuristic H_7 are described in Algorithm 3.

Clustering-based heuristic (H_8)

This heuristic is based essentially on the classification method. This research identifies two groups of reports. These groups are denoted by G_1 and G_2. The first phase of this heuristic is to choose the reports sorting method. The sorting methods will be in three variants. The first variant will consider to assign a report directly after it is initiated. The second variant will consider to sort the reports ascending based on their number of pages. The third variant will consider to sort the reports descending based on their number of pages. The second phase of this heuristic is the construction of G_1 and G_2 groups. Initially G_1 and G_2 are empty.

The third phase is to initiate G_1 and G_2 groups by distributing the set of reports R. The first step is to choose the first report and to distribute it to either G_1 or G_2, then the second report will be sent to the most available group between G_1 and G_2. The most available group will be the group that has the minimum pages count. After that assign the next report to the most available group and so on until all reports in R are distributed.

The fourth phase is the scheduling of the reports to the available doctors. This phase is based on the randomly selection between G_1 and G_2, by generating a probability α to select a report from the two groups. The selected report will be assigned to the doctor who

Algorithm 3 Randomly repeating iteratively doctor choice heuristic (H_7).
1: **for** (k to 3) **do**
2:      **if** ($k = 2$) **then**
3:           Incs(R)
4:      **else if** ($k = 3$) **then**
5:           Decs(R)
6:      **end if**
7:      **for** ($it = 1$ to $limit$) **do**
8:           Call $H_3^*()$ and calculate W_3^{it}
9:      **end for**
10:      **for** ($it = 1$ to $limit$) **do**
11:           Call $H_4^*()$ and calculate W_4^{it}
12:      **end for**
13:      **for** ($it = 1$ to $limit$) **do**
14:           Call $H_5^*()$ and calculate W_5^{it}
15:      **end for**
16:      Calculate $T_k = \min\limits_{3 \leq i \leq 5} \left(\min\limits_{1 \leq it \leq limit} W_i^{it} \right)$
17: **end for**
18: Calculate $T = \min\limits_{1 \leq k \leq 3} T_k$
19: Return T

has the minimum total number of pages. This procedure will be repeated many times. In practice the number of iterations is fixed to $limit = 2{,}000$.

Finally, the heuristic is repeated again with another sorting method variant and iterated for $limit = 2{,}000$ times, after all sorting variants are chosen the best final results are considered. The complexity of H_8 heuristic is $O(n^2)$.

EXPERIMENTAL RESULTS

This section describes the performance of the developed heuristics. All heuristics proposed in this paper were implemented in Microsoft Visual C++ and executed on an Intel(R) Core (TM) i5-3337U CPU @ 1.8 GHz and 8 GB RAM. Several classes of instances were being tested in this paper to measure the performance of the developed heuristics. These classes were generated by methods derived from *Jemmali (2019b)*. The number of pages np_j for each report are generated by using two different distributions. The first distribution is the uniform distribution which is denoted by $U()$ and the normal distribution that is denoted by $N()$.

The generated classes are as follows:

- Class A: $np_j \in U(10, 20)$.
- Class B: $np_j \in U(20, 30)$.
- Class C: $np_j \in U(10, 30)$.

- Class D: $np_j \in N(20, 5)$.
- Class E: $np_j \in N(20, 10)$.

Two types of instances are adopted for the experimental part. The small instances type and the big scale type. Both types were used to generate the np_j for all the classes above. For the small type instances, several values for the pair (n_r, n_d) were selected as Table 3 illustrates.

For each tuple $(n_r, n_d, class)$, this work will generate 10 different instances of the number of pages. Therefore, the total number of the generated instances will be $(3 + 2 \times 5 + 5 \times 4) \times 10 \times 5 = 1,650$. For the big scale instances, generate 800 instances as detailed in the end of this section. So, in total 2,450 instances were tested. Now, we start the analyses of results related to the 1,650 small instances.

To assess the performance of the proposed heuristics, the metrics *Perc*, *Ag* and *Time* are defined in Table 1. Hereafter, the analyses of the experimental results are concerning the small instances type.

The overall performance of the developed heuristics is shown in Table 4. As it can be noticed from the table, the best performance was obtained by the heuristic H_8. The obtained percentage for this heuristic was 96.1% which is the highest among all heuristics. The gap calculations for H_8 gained zero results for most of the instances except for 64 instances out of 1,650 leading to an average gap of 0.02, while the elapsed time was 0.006 s. For heuristic H_7 the percentage was 59.1% with an average gap of 0.29 and an average time of 0.021s, which makes it the second best heuristic. However, the heuristic that has the worst performance measurements was H_2, with the highest average gap of 0.74 and the lowest percentage of 0.8%.

Table 5 represents the variation of *Ag* and *Time* according to n_r for all developed heuristics. This table shows that the highest gap was obtained for heuristic H_2 when $n_r = 35$, while the lowest gab of 0 was obtained by H_8, H_7, H_6 and H_3 for $n_r = 5$ and $n_r = 7$. As it can be noticed from Table 5, most of the heuristics perform well when $n_r = 5$, $n_r = 7$ and $n_r = 10$ and the performance values was close to each other, but for $n_r = 35$ the developed heuristics showed a different performance. So, based on gap calculations heuristic H_2 showed the worst performance while heuristic H_8 showed the best performance even for high n_r values such as 30 and 35. Based on time calculations heuristic H_6 showed the worst performance at $n_r = 35$ then heuristic H_7 also at $n_r = 35$. For heuristics H_3 to H_7 time increases as n_r is increases. The remarkable performance was shown by heuristic H_8 as it can be noticed by the results shown in Table 5 the gap values = 0 and the running time was 0.001 for all n_r values.

Table 6 represents the variation of *Ag* and *Time* according to n_d for all developed heuristics. The 0 gap values were obtained by many heuristics at different n_d values, even heuristic H_1 had a zero gap value at $n_d = 6$. But in general as the value of n_d increases, the gap value and the elapsed time increases for all heuristics except for heuristic H_8, where it again shows a better performance values than the rest of the heuristics even with large n_d values. So, based on gap values heuristic H_2 shows the worst performance and

Algorithm 4 Randomly repeating iteratively doctor choice heuristic (H_7).

1: **for** (k = 1 to 3) **do**
2:      **if** (k = 2) **then**
3:          Incs(R)
4:      **else if** (k = 3) **then**
5:          Decs(R)
6:      **end if**
7:      Determine G_1 and G_2
8:      **for** (it = 1 to $limit$) **do**
9:          **for** (j = 1 to n_r) **do**
10:             r =random(1,2)
11:             **if** (r = 1) **then**
12:                 schedule the first report in G_1
13:             **else**
14:                 schedule the first report in G_2
15:             **end if**
16:         **end for**
17:         Calculate T_k^{it}
18:     **end for**
19:     Calculate $T_k = \min\limits_{1 \leq it \leq limit} T_k^{it}$
20: **end for**
21: Calculate $T = \min\limits_{1 \leq k \leq 3} T_k$
22: Return T

Table 3 Generation of (n_r, n_d).

n_r	n_d
5	2, 3, 4
7, 10	2, 3, 4, 5, 6
15, 20, 25, 30, 35	3, 5, 7, 10

Table 4 Overall performance of all heuristics.

	H_1	H_2	H_3	H_4	H_5	H_6	H_7	H_8
Perc	45.2%	0.8%	48.1%	37.5%	50.7%	57.6%	59.1%	96.1%
Ag	0.39	0.74	0.40	0.45	0.34	0.30	0.29	0.02
Time	–	–	0.003	0.005	0.014	0.022	0.021	0.006

Table 5 *Time* and *Ag* variations according based on n_r for all heuristics.

		n_r							
		5	7	10	15	20	25	30	35
H_1	Ag	0.14	0.21	0.34	0.27	0.49	0.64	0.42	0.57
	Time	–	–	–	–	–	–	–	–
H_2	Ag	0.46	0.51	0.73	0.73	0.84	0.86	0.89	0.90
	Time	–	–	–	–	–	–	–	–
H_3	Ag	0.00	0.00	0.08	0.42	0.61	0.67	0.73	0.74
	Time	0.001	0.002	0.002	0.003	0.004	0.004	0.005	0.006
H_4	Ag	0.24	0.21	0.29	0.37	0.57	0.62	0.67	0.69
	Time	0.002	0.003	0.003	0.004	0.005	0.006	0.007	0.009
H_5	Ag	0.10	0.13	0.10	0.31	0.46	0.53	0.58	0.59
	Time	0.003	0.005	0.006	0.012	0.015	0.019	0.023	0.027
H_6	Ag	0.00	0.00	0.01	0.29	0.46	0.49	0.60	0.60
	Time	0.006	0.008	0.011	0.019	0.025	0.030	0.036	0.041
H_7	Ag	0.00	0.00	0.00	0.28	0.43	0.49	0.59	0.60
	Time	0.007	0.008	0.010	0.018	0.024	0.029	0.035	0.040
H_8	Ag	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Time	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001

Table 6 *Time* and *Ag* variations according based on n_d for all heuristics.

		n_d						
		2	3	4	5	6	7	10
H_1	Ag	0.52	0.54	0.18	0.33	0.00	0.51	0.28
	Time	–	–	–	–	–	–	–
H_2	Ag	0.80	0.83	0.51	0.80	0.36	0.81	0.70
	Time	–	–	–	–	–	–	–
H_3	Ag	0.00	0.19	0.05	0.62	0.01	0.72	0.69
	Time	0.002	0.003	0.002	0.004	0.002	0.005	0.005
H_4	Ag	0.73	0.18	0.11	0.58	0.04	0.68	0.68
	Time	0.003	0.006	0.002	0.005	0.002	0.006	0.006
H_5	Ag	0.44	0.03	0.00	0.49	0.01	0.62	0.65
	Time	0.004	0.008	0.005	0.012	0.006	0.020	0.031
H_6	Ag	0.00	0.02	0.00	0.48	0.00	0.61	0.63
	Time	0.009	0.017	0.008	0.021	0.011	0.031	0.043
H_7	Ag	0.00	0.01	0.00	0.46	0.00	0.61	0.63
	Time	0.008	0.017	0.007	0.020	0.010	0.029	0.041
H_8	Ag	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Time	0.001	0.001	0.001	0.001	0.001	0.001	0.001

Table 7 Time and Ag variations according based on Class for all heuristics.

		Class				
		1	2	3	4	5
H_1	Ag	0.34	0.24	0.43	0.47	0.45
	Time	0.000	0.000	0.000	0.000	0.000
H_2	Ag	0.72	0.61	0.79	0.76	0.83
	Time	0.000	0.000	0.000	0.000	0.000
H_3	Ag	0.39	0.35	0.40	0.40	0.44
	Time	0.003	0.003	0.003	0.003	0.003
H_4	Ag	0.43	0.39	0.48	0.45	0.52
	Time	0.005	0.005	0.005	0.005	0.005
H_5	Ag	0.32	0.26	0.38	0.36	0.40
	Time	0.014	0.013	0.014	0.014	0.013
H_6	Ag	0.27	0.25	0.32	0.30	0.35
	Time	0.022	0.022	0.022	0.022	0.022
H_7	Ag	0.26	0.23	0.31	0.30	0.34
	Time	0.021	0.021	0.021	0.021	0.021
H_8	Ag	0.01	0.01	0.01	0.01	0.01
	Time	0.001	0.001	0.001	0.001	0.001

heuristic H_8 showed the best performance. Heuristics H_1 and H_2 have the minimum average time, while heuristic H_8 showed an average time of 0.001 for all n_d values.

Table 7 illustrates the variation of Ag and Time according to Class for all developed heuristics. As was noticed in the results shown in Tables 6 and 5, the performance measurement of heuristic H_2 showed the worst performance compared to the rest of the heuristics, heuristic H_2 returned the highest average gap when class = 5 with a value equal to 0.83. The rest of the heuristics showed comparable performance for all used classes except for heuristic H_8. The average gap values ranged from 0.23 to 0.52, the highest gap values were obtained for classes 3 and 5. But for heuristic H_8, again it shows the best performance measures with average gap value = 0.01 with an elapsed time of 0.001 s.

The experimental results show that the maximum average gap reaches 0.97 for H_2 when ($n_r = 20$, $n_d = 3$), ($n_r = 25$, $n_d = 3$), and ($n_r = 35$, $n_d = 3$). The 0.00 average gap was obtained by H_3, H_6, H_7 and H_8 for $n_r = 5$ and $n_r = 7$ for all n_d values, but when n_r values begin to be larger than 10 the average gap results, start to increase for most of the heuristics except for H_8 where the heuristic shows a 0.00 average gap results for all n_r and n_d values.

Hereafter, Ind is the index of each pair (n_r, n_d). In total we have 33 different pairs.

Figure 3 illustrates the variation of the average gap for heuristics H_1 and H_2 according to Ind. The behavior of the gap variations for both heuristics shows a similar behavior for most of Ind. This behavior is expected, because of the similarity in the way the two heuristics work. Despite the similar behavior, H_1 performs better than the H_2, there are
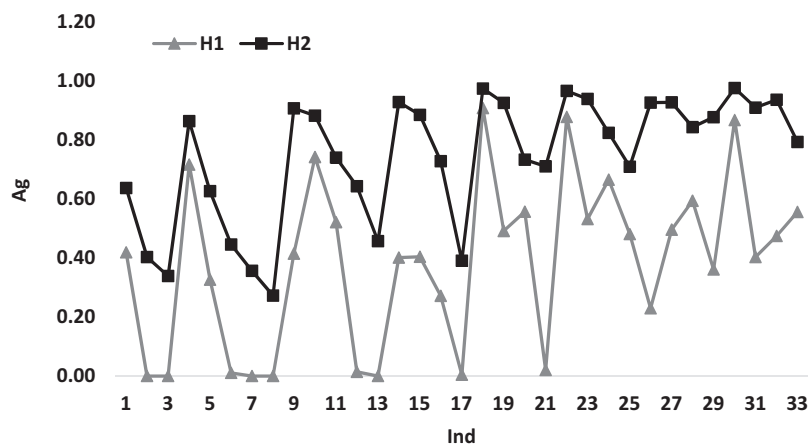
Figure 3 The average gap for H_1 and H_2 according to *Ind*.
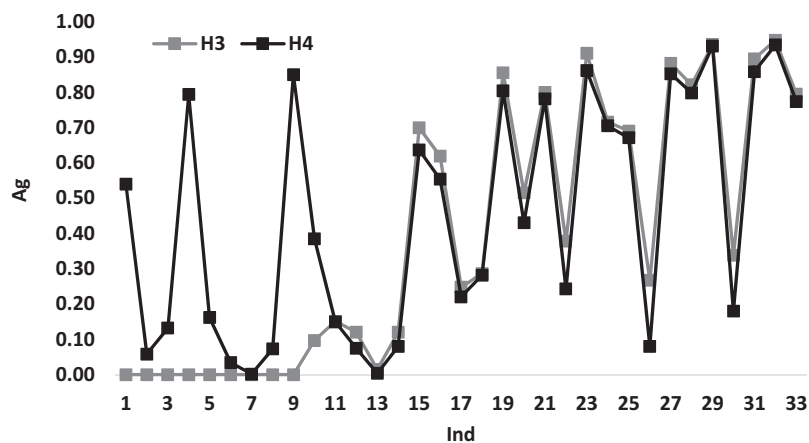
Figure 4 The average gap for H_3 and H_4 according to *Ind*.

some points where H_1 gap values reaches 0.00 while H_2 gap values never reaches 0.00 for all *Ind*.

Figure 4 illustrates the variations of the average gap for heuristics H_3 and H_4 according to *Ind*. From the figure it can be noticed that H_3 and H_4 shows similar behavior variations starting from *Ind* = 9, while before that point H_4 has 0.00 gap values. The similar behavior is expected as both heuristics, mostly use the same technique most of the time with a difference in the choice of the assigned doctor.

Figure 5 illustrates the variations of the average gap for heuristics H_5 and H_6 according to *Ind*. It can be noticed that H_6 has a better performance measures with a gap value of 0.00 till *Ind* = 14 and beyond this point, both heuristics have similar behavior variations.

Figure 6 illustrates the variations of the average gap for heuristics H_7 and H_8 according to *Ind*. Both heuristics show a similar variation behavior with a gap value of 0.00 till *Ind* = 14m after this point the variations changed for H_7 and has different gap values
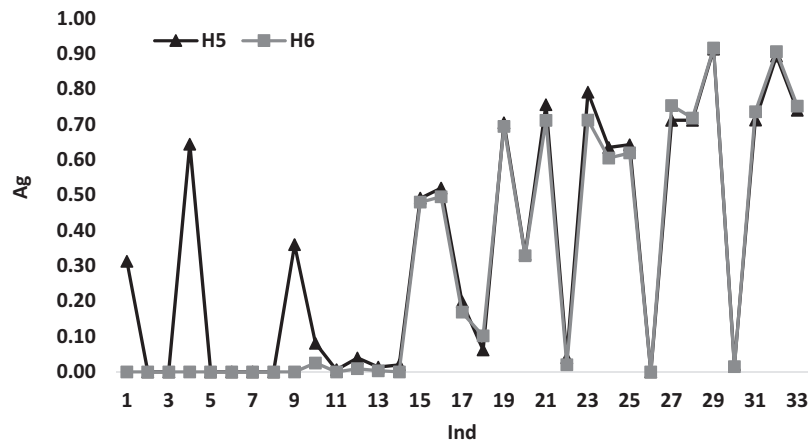
Figure 5 The average gap for H_5 and H_6 according to *Ind*.
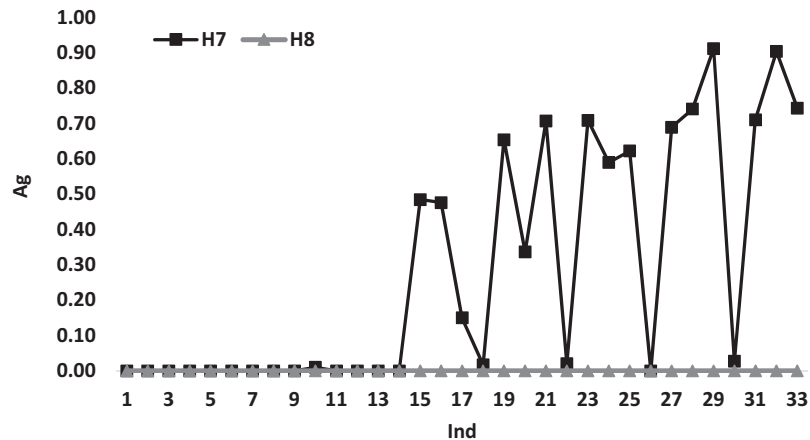
Figure 6 The average gap for H_7 and H_8 according to *Ind*.

reaches to 0.90 at some points, while the gap variations of H_8 continues to have a 0.00 gap values for all *Ind*, which shows the dominance performance of H_8 over all the used heuristics.

The results presented above, indicate that the developed heuristics were capable to derive an accepted approximate solution with a good execution time, but none of the given heuristics were capable to produce exact or optimal solution, which requires more work and research to enable these heuristics to produce optimal solutions for the given problem. Besides, the algorithms presented by *Taha, Ould-Slimane & Talhi (2020)* can inspire these heuristics to develop new algorithms for the studied problem.

Moreover, the developed heuristics can be used as an initial solution for other meta heuristics, such as, genetic algorithm, particle swarm optimization algorithms to enhance the given solution.

Table 8 Overview of heuristics according to *Perc*, *Ag* and *Time* for the big scale instances.

	H_1	H_2	H_3	H_4	H_5	H_6	H_7	H_8
Perc	44.4%	0.0%	6.0%	10.9%	28.6%	28.1%	28.3%	98.0%
Ag	0.41	0.90	0.88	0.82	0.63	0.64	0.63	0.02
Time	0.000	0.000	0.036	0.049	0.209	0.302	0.291	0.068

The performance of the developed heuristics was compared across each other, since to the best of our knowledge this problem was not studied previously in any of the literature that was within our reach.

Other type of instances regarding the big scale of number of reports. Indeed, for this type of instances the number of reports n_r will be in {50, 100, 200, 500} and the number of doctors n_d will be in {3, 5, 10, 15}. Therefore, we generate 800 instances of this type. The overview of the proposed heuristics according to *Perc*, *Ag* and *Time*, is shown in Table 8.

Table 8 shows that the heuristics H_1, H_2 and H_8 keep the same range of percentages comparing with the small instances illustrated in Table 4. However, for the other heuristics the percentages range are remarkably increased. Indeed, for H_3 the percentage for the small instances is 48.1% (see Table 4), while for the big scale *Perc* = 6%. The H_8 heuristic is always the best, with 98%.

CONCLUSION

This work presented the solution to the problem of paper workload distribution between doctors in healthcare organizations to ensure fair distribution of the overall workload based on the total number of pages of patients' reports assigned to each doctor. This problem was solved by developing eight heuristics based on dispatching rules with several variants, randomization approach, clustering methods, probabilistic method and iterative methods approach. The performance of the developed heuristics was measured experimentally based on different classes with 2,450 instances. The obtained results showed a clear variation in the performance measurements of the developed heuristics based on the elapsed time and the gap value calculations. The given results showed that heuristic H_8 was dominant among all the developed heuristics with remarkable performance and 0.00 gap values for all the used instances. For future work, the developed heuristics can be adopted in other domains like education, finance, industry, aviation, and many others. Also, it can be used in branch and bound for exact methods.

Funding

This work was supported by the Deanship of Scientific Research, University of Hafr Al-Batin under Project Number No. G-114-2020. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Grant Disclosures

The following grant information was disclosed by the authors:

Deanship of Scientific Research, University of Hafr Al-Batin: G-114-2020.

Competing Interests

The authors declare that they have no competing interests.

Author Contributions

- Mahdi Jemmali conceived and designed the experiments, performed the experiments, analyzed the data, performed the computation work, prepared figures and/or tables, authored or reviewed drafts of the paper, and approved the final draft.
- Loai Kayed B. Melhim analyzed the data, prepared figures and/or tables, authored or reviewed drafts of the paper, and approved the final draft.
- Abdullah Alourani performed the experiments, performed the computation work, authored or reviewed drafts of the paper, and approved the final draft.
- Md. Moddassir Alam performed the computation work, authored or reviewed drafts of the paper, and approved the final draft.

Data Availability

Raw data and code are available in the Supplemental Files.

Supplemental Information

Supplemental information for this article can be found online at http://dx.doi.org/10.7717/peerj-cs.819#supplemental-information.

REFERENCES

Alharbi M, Jemmali M. 2020. Algorithms for investment project distribution on regions. *Computational Intelligence and Neuroscience* **2020(4)**:1–13 DOI 10.1155/2020/3607547.

Alquhayz H, Jemmali M, Otoom MM. 2020. Dispatching-rule variants algorithms for used spaces of storage supports. *Discrete Dynamics in Nature and Society* **2020**:1–9 DOI 10.1155/2020/1072485.

Bany Taha M, Talhi C, Ould-Slimane H, Alrabaee S. 2020. Td-pso: task distribution approach based on particle swarm optimization for vehicular ad hoc network. *Transactions on Emerging Telecommunications Technologies* **37(10)**:e3860 DOI 10.1002/ett.3860.

Christino MA, Matson AP, Fischer SA, Reinert SE, DiGiovanni CW, Fadale PD. 2013. Paperwork versus patient care: a nationwide survey of residents' perceptions of clinical documentation requirements and patient care. *Journal of Graduate Medical Education* **5(4)**:600–604 DOI 10.4300/JGME-D-12-00377.1.

Friesen MR, McLeod R, Strome T, Mukhi S. 2011. Load balancing at emergency departments using 'crowdinforming'. In: *2011 IEEE 13th International Conference on e-Health Networking, Applications and Services*. Piscataway: IEEE, 364–370.

Haouari M, Jemmali M. 2008. Maximizing the minimum completion time on parallel machines. *4OR* **6(4)**:375–392 DOI 10.1007/s10288-007-0053-5.

Jemmali M. 2019a. Approximate solutions for the projects revenues assignment problem. *Communications in Mathematics and Applications* **10(3)**:653–658 DOI 10.26713/cma.v10i3.1238.

Jemmali M. 2019b. Budgets balancing algorithms for the projects assignment. *International Journal of Advanced Computer Science and Applications* **10(11)**:574–578 DOI 10.14569/issn.2156-5570.

Jemmali M. 2021a. Intelligent algorithms and complex system for a smart parking for vaccine delivery center of covid-19. *Complex & Intelligent Systems* **368(6494)**:1–13 DOI 10.1007/s40747-021-00524-5.

Jemmali M. 2021b. An optimal solution for the budgets assignment problem. *RAIRO: Recherche Opérationnelle* **55(2)**:873–897 DOI 10.1051/ro/2021043.

Jemmali M. 2021c. Projects distribution algorithms for regional development. *Advances in Distributed Computing and Artificial Intelligence Journal* **10(3)**:293–305 DOI 10.14201/ADCAIJ2021103293305.

Jemmali M, Alquhayz H. 2020. Equity data distribution algorithms on identical routers. In: *International Conference on Innovative Computing and Communications*. Berlin: Springer, 297–305.

Jemmali M, Melhim LKB, Alharbi M. 2019. Randomized-variants lower bounds for gas turbines aircraft engines. In: *World Congress on Global Optimization*. Berlin: Springer, 949–956.

Jemmali M, Melhim LKB, Alharbi SOB, Bajahzar AS. 2019. Lower bounds for gas turbines aircraft engines. *Communications in Mathematics and Applications* **10(3)**:637–642 DOI 10.26713/cma.v10i3.1218.

Jemmali M, Otoom MM, al Fayez F. 2020. Max-min probabilistic algorithms for parallel machines. In: *Proceedings of the 2020 International Conference on Industrial Engineering and Industrial Management*. 19–24.

Kolomvatsos K, Panagidi K, Hadjiefthymiades S. 2015. A load balancing module for post-emergency management. *Expert Systems with Applications* **42(1)**:657–667 DOI 10.1016/j.eswa.2014.07.055.

Schaus P, Van Hentenryck P, Régin J-C. 2009. Scalable load balancing in nurse to patient assignment problems. In: *International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems*. Berlin: Springer, 248–262.

Stock C, Cocchi M, Clardy P, Liesching T, Nesto R, Talmor D. 2021. 103: How a healthcare system leveraged ICU patient load balancing to manage the covid-19 pandemic surge. *Critical Care Medicine* **49(1)**:35 DOI 10.1097/01.ccm.0000726300.38763.1c.

Taha MB, Chowdhury R. 2020. Galb: Load balancing algorithm for CP-ABE encryption tasks in e-health environment. In: *2020 Fifth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*. Piscataway: IEEE, 165–170.

Taha MB, Ould-Slimane H, Talhi C. 2020. Smart offloading technique for CP-ABE encryption schemes in constrained devices. *SN Applied Sciences* **2**:274 DOI 10.1007/s42452-020-2074-z.

Weigl M, Müller A, Vincent C, Angerer P, Sevdalis N. 2012. The association of workflow interruptions and hospital doctors' workload: a prospective observational study. *BMJ Quality & Safety* **21(5)**:399–407 DOI 10.1136/bmjqs-2011-000188.

Woolhandler S, Himmelstein DU. 2014. Administrative work consumes one-sixth of us physicians' working hours and lowers their career satisfaction. *International Journal of Health Services* **44(4)**:635–642 DOI 10.2190/HS.44.4.a.

Zhong X, Lee HK, Williams M, Kraft S, Sleeth J, Welnick R, Hauschild L, Li J. 2018. Workload balancing: staffing ratio analysis for primary care redesign. *Flexible Services and Manufacturing Journal* **30(1)**:6–29 DOI 10.1007/s10696-016-9258-2.