Chapter 4

Matching and Exchange-based Utility Maximization for Fog Computing-enabled Smart Healthcare

4.1 Introduction

With an increasing demand for real-time healthcare applications, smart healthcare system has gained significant attention and made substantial progress in recent years [107]. As part of the smart healthcare system, WBANs connect various on-body physiological sensors, enabling the collection of health data and delivery of healthcare services [5]. Specifically, on-body sensors transmit health data to an aggregator, such as an LD, which then collects and processes this data for healthcare services. However, these applications are time-sensitive (e.g., post-surgery monitoring) and involve processing critical disease data, which is often beyond the capabilities of resource-constrained LDs with limited computation and energy resources [4, 5]. A promising solution is the use of fog computing, which brings computing services and storage closer to patients, sup-

porting computation-intensive healthcare applications and improving QoS in terms of latency and energy.

In healthcare applications, patients require continuous connectivity to FSs for real-time, reliable services, which increases the energy consumption of LDs [4]. Moreover, the diverse functions available on LDs further contribute to higher energy usage. Consequently, the energy consumption of LDs in WBANs becomes a critical factor that restricts the lifespan of the health monitoring system [8, 9]. Offloading health data processing tasks to FSs can significantly reduce the energy consumption of LDs. However, as the number of WBAN users seeking healthcare services grows, the computational load on FSs increases, raising concerns about timely service delivery [5]. Furthermore, health data has strict delay constraints, particularly in time-sensitive healthcare applications where delays can have fatal consequences.

As cloud and fog computing platforms operate on a pay-as-you-go pricing model, patients only pay *HSPs* for the resources they use [108]. To meet diverse healthcare applications and QoS requirements, HSPs offer on-demand healthcare platforms, known as Healthcare-as-a-Service (He-aaS) [108]. *Moreover, patients access healthcare resources based on their specific service demands and pay accordingly.* However, existing works [3, 97] typically rely on flat pricing schemes with fixed payment model designed for specific services. *Thus, there is a need for energy- and latency-aware WBAN-based smart healthcare system that integrate a dynamic pricing model tailored to computational requirements.* Recent efforts have been made to address this issue either by minimizing energy consumption [4, 42] or latency [3], yet they do not consider HSPs' profit or latency and energy costs of patients in a multi-FS healthcare system.

Motivated by the aforementioned scenarios, we propose an efficient fog computing-enabled WBAN-based smart healthcare system. Moreover, we formulate an optimization problem aimed at maximizing system utility, defined as a linear combination of HSP's profit and patients' latency and energy costs, with a focus on prioritizing critical

patients' health data. To address the different entities with contrasting objectives, we employ a *matching* and *exchange*-based solution that leverages the concept of preferences [109]. Additionally, matching-based approaches are computationally inexpensive, scalable, and provide near-optimal solution, in contrast to non-cooperative or Stackelberg game approaches [15]. The main contributions of this chapter are summarized as follows:

- Propose a fog computing-enabled WBAN-based system for real-time remote health monitoring, focusing on enhancing latency and energy efficiency.
- Formulate an optimization problem that maximizes system utility by considering the HSP's profit, as well as the latency and energy costs of patients, while prioritizing critical patients' health data as an NP-hard problem. Moreover, introduce a dynamic pricing scheme for delivering health monitoring services based on the computational requirements of patients' health data.
- Propose a matching and exchange-based sub-optimal algorithm to solve the formulated problem within polynomial time complexity while adhering to several constraints. Additionally, analyze stability, convergence, and computational complexity of the proposed algorithm.
- Experimental and simulation results using real-world data show the efficacy of the proposed algorithm, achieving an average utility of 99.01% of the optimal value.

4.2 System Model and Problem Formulation

We consider a fog computing-enabled remote health monitoring system where an HSP deploys F FSs, denoted as $\mathbb{F} = \{1, \ldots, f, \ldots, F\}$, to provide health monitoring services to P patients, represented by $\mathbb{P} = \{1, \ldots, p, \ldots, P\}$, as illustrated in Fig. 4.1. Patients are wirelessly connected to FSs via a cellular 5G base station installed by the HSP. The health monitoring data collected by sensor s from patient p is characterized as

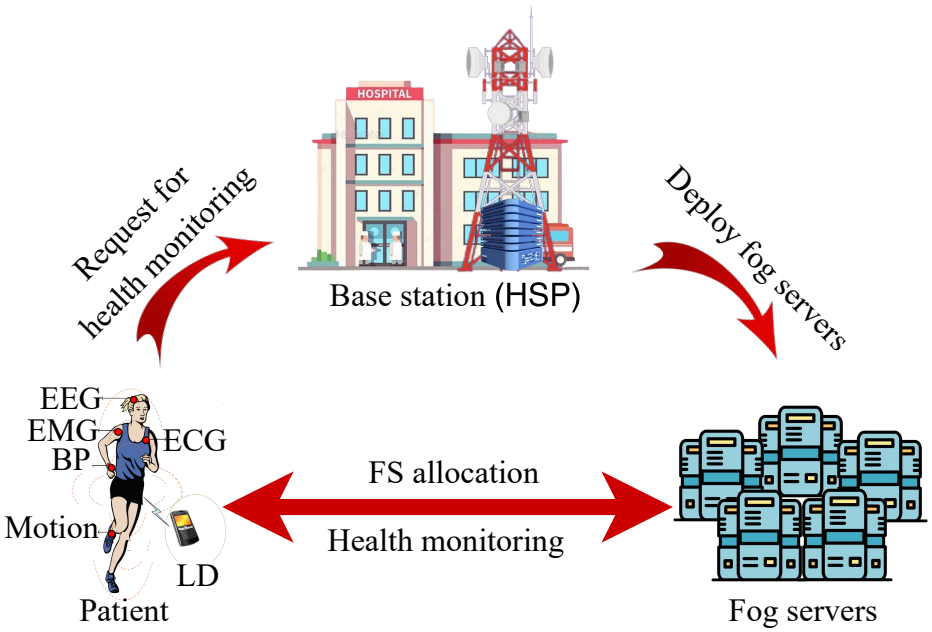


Fig. 4.1. Fog computing-enabled health monitoring system.

$\langle \theta_p^s, \mathrm{b}_p^s, \mathrm{y}_p^s \rangle$, where θ_p^s represents health data[1], b_p^s denotes data size in bytes, and y_p^s indicates CPU cycles required for processing. Subsequently, this data is transmitted to an LD for processing using either Bluetooth or ZigBee [40]. Upon receiving health data from the sensors, LDs request the base station to allocate FS for health monitoring services [3]. Subsequently, the base station assigns FSs to patients for remote health monitoring, considering various factors as discussed in the following subsections.

4.2.1 Health Data Relevance

Multiple heterogeneous physiological sensors collect diverse health data for monitoring purposes. For example, an ECG sensor captures heart rate and blood pressure, while a gyroscope insulin actuator tracks blood glucose levels [8, 40]. However, the relevance of health data collected by different sensors varies depending on the patient's condition [5]. For instance, to promptly detect recrudescence in a patient with a history of heart

[1]Health data θ_p^s is implicitly used in the calculation of the health severity index, $\mathfrak{s}_p^s$ (using the relation defined in Eq. (3.1) from Section 3.2.1 of Chapter 3), which is then used to determine the criticality index, c_p^s, as defined in Eq. (4.2).

attack, data such as heart rate and ECG should be prioritized and assigned higher importance. In contrast, for a diabetes patient, blood glucose data should be prioritized and given higher importance. Intuitively, the medical criticality of health data collected from various sensors differs; hence, health monitoring data from different sensors can be assigned priorities or ranks, as provided in the IEEE 802.15.6 standard [110]. Therefore, in the proposed health monitoring system, health data from various sensors are assigned priorities or ranks, $\mathcal{R}_s$, based on their relevance to the patient's disease, as done in [5]. Mathematically, the normalized priority or rank is expressed as follows:

$$\mathcal{H}_s = \frac{\mathcal{M} - \mathcal{R}_s}{\mathcal{M} - 1}, \tag{4.1}$$

where $\mathcal{M}$ denotes the highest priority. If the health data collected by sensor s is more relevant to the patient's disease compared to that collected by sensor s', then $\mathcal{H}_s > \mathcal{H}_{s'}$. Furthermore, the priority or rank serves as a fundamental scale, indicating the degree of medical professionals' preference for health data relevant to the disease [5].

4.2.2 Health Data Criticality

Estimating the severity level of patients offers valuable insights into their health condition, which reflects the severity level of the collected health data [5]. The health severity index, denoted as $\mathfrak{s}_p^s$, for the health data of patient p collected by sensor s, θ_p^s, is calculated using Eq. (3.1) in Chapter 3. This index measures how much a patient's health data deviates from its normal reading, with a higher value indicating more severe health data. Additionally, the criticality index, denoted as c_p^s, is defined for the health data collected by sensor s on patient p as the product of the priority or rank and the health severity index of patient p's health data, expressed as follows [3, 8]:

$$c_p^s = \mathcal{H}_s \mathfrak{s}_p^s. \tag{4.2}$$

Then, the criticality level of patient p is determined as the average of the criticality index for all of patient p's collected health data, expressed as follows:

$$\rho_p = \frac{1}{S} \sum_{s \in \mathbb{S}} c_p^s. \tag{4.3}$$

4.2.3 Computation at LD

LD aggregates health data received from body sensors, with the total size (in bytes) of the aggregated health data from patient p is given by: $\eta_p = \sum_{s \in \mathbb{S}} \mathrm{b}_p^s$. Similarly, the number of CPU cycles needed to compute patient p's health data is expressed as: $\beta_p = \sum_{s \in \mathbb{S}} \mathrm{y}_p^s$. Let Υ be the computation capacity of LD for patient p, which is assumed to be uniform across all LDs [3]. Furthermore, we introduce a binary variable, u_p, to indicate whether the computation of patient p's health data occurs on the LD. If the computation takes place on the LD, then u_p is assigned the value 1; otherwise, it is set to 0. Thus, the local computation latency for patient p is defined as follows:

$$T_p^{c,l} = u_p \frac{\beta_p}{\Upsilon}. \tag{4.4}$$

Moreover, the energy consumption per cycle at LD, while operating at Υ cycles per second, is given by $\psi \Upsilon^2$. Therefore, the energy consumption of patient p to locally compute its health data is calculated as follows [111]:

$$e_p^c = u_p \psi \beta_p \Upsilon^2, \tag{4.5}$$

where ψ denotes the effective capacitance, which depends on the chip architecture of the LD [112]. Then, the energy cost of the patient due to local computation is defined as follows:

$$\mathcal{E}_p^l = \alpha_p e_p^c, \tag{4.6}$$

where α_p is the cost per unit energy consumption.

Due to the critical nature of health data, prompt monitoring without delays is essential. Therefore, inspired by [4] and [108], we define the latency cost of patient p due to local computation as a function of both the local computation latency and the criticality level of the patient, which is expressed as follows:

$$\mathcal{C}_p^l = \rho_p T_p^{c,l}. \tag{4.7}$$

4.2.4 Computation at FS

Many LDs lack the computation capacity to process health data related to critically sensitive diseases within the desired latency constraint, primarily due to unstable power supply and excessive energy consumption [3,8]. As a result, transmitting health data to FSs with high computational power to facilitate faster computation becomes essential. Therefore, patients transmit their health data to an FS for health monitoring via the cellular 5G network. The health data transmission latency depends on the transmission rate between the patient and the FS, which is calculated using Eq. (3.9).

We further define a binary variable to indicate the choice of FS for computing patient p's health data, as follows:

$$h_p^f = \begin{cases} 1, & \text{if patient } p\text{'s health data is computed by FS } f; \\ 0, & \text{otherwise.} \end{cases} \tag{4.8}$$

The transmission latency between patient p and FS f is defined as follows [113]:

$$T_p^{tr,f} = \frac{h_p^f \eta_p}{r_p^f}. \tag{4.9}$$

The energy consumption of patient p to transmit health data to FS f depends on data size (η_p), transmission rate (r_p^f), and transmitting power ($\mathfrak{k}$) [4]. Thus, it is expressed

as follows:

$$\mathrm{e}_p^{tr,f} = T_p^{tr,f} \mathfrak{k}. \tag{4.10}$$

Then, the energy cost of the patient for transmitting health data to the FS is defined as follows:

$$\mathcal{E}_p^{tr,f} = \alpha_p \mathrm{e}_p^{tr,f}. \tag{4.11}$$

Thus, the total energy cost for all patients is given as follows:

$$\Omega = \sum_{p \in \mathbb{P}} \left(u_p \mathcal{E}_p^l + \sum_{f \in \mathbb{F}} (1 - u_p) \mathcal{E}_p^{tr,f} \right). \tag{4.12}$$

Let $\mathbb{L} = \{\Gamma_1, \ldots, \Gamma_f, \ldots, \Gamma_F\}$ be the set of computation capacities of the F FSs. Additionally, we assume that each patient equally utilizes FS resources, following the approach in [3, 8]. Thus, the computation latency of FS f for computing patient p's health data is determined as follows:

$$T_p^{c,f} = \frac{h_p^f \beta_p n^f}{\Gamma_f}, \tag{4.13}$$

where n^f represents the number of patients, including patient p, using FS f without violating the latency constraint. However, allocating patients' health data to an FS must satisfy the following constraint: $n^f \leq F_f^{max}$, where F_f^{max} denotes the maximum capacity of FS f, indicating the number of patients it can handle concurrently.

Similar, the latency cost for patient p to process its health data at FS f is defined as a function of transmission latency, computation latency at the FS, and the patient's criticality. Mathematically, it is expressed as follows:

$$\mathcal{C}_p^f = \rho_p (T_p^{c,f} + T_p^{tr,f}). \tag{4.14}$$

Thus, the total latency cost for all the patients in the system is expressed as follows:

$$\Pi = \sum_{p \in \mathbb{P}} \left(\rho_p T_p^{c,l} + \sum_{f \in \mathbb{F}} \rho_p (T_p^{c,f} + T_p^{tr,f}) \right). \tag{4.15}$$

From Eq. (4.15), it is evident that the incurred cost is higher for critical patients compared to less critical ones; thus, minimizing transmission and computation latency for critical patients is essential to reduce the cost.

4.2.5 Profit of HSP

The primary goal of the HSP is to maximize the profit generated from providing health monitoring services to patients. Unlike our previous work [3], the profit from a patient p depends on the number of CPU cycles required to compute their health data. Let o and a be the prices per mega CPU cycle for computing a patient's health data at LD and FS, respectively, with $a > o$ [3]. Then, the total price charged by the HSP from all patients is defined as follows:

$$\nu = \sum_{p \in \mathbb{P}} (u_p \beta_p o + (1 - u_p) \beta_p a). \tag{4.16}$$

Let υ be the cost to HSP per mega byte of health data computed at FS. Then, the total expenses of the HSP can be calculated as follows:

$$\partial = \upsilon \sum_{p \in \mathbb{P}} \sum_{f \in \mathbb{F}} h_p^f \eta_p, \tag{4.17}$$

Now, the HSP's profit is determined as follows:

$$\Lambda = \nu - \partial = \sum_{p \in \mathbb{P}} (u_p \beta_p o + (1 - u_p) \beta_p a) - \upsilon \sum_{p \in \mathbb{P}} \sum_{f \in \mathbb{F}} h_p^f \eta_p. \tag{4.18}$$

To ensure an increase in the HSP's profit when a patient's health data is processed

by FS, the values of a and o must satisfy the following constraint:

$$a - o \geq \upsilon \frac{\eta_{max}}{\beta_{min}}, \tag{4.19}$$

where $\eta_{\max}$ represents the maximum size of the patients' health data, and $\beta_{\min}$ denotes the minimum number of CPU cycles required for computing the health data.

4.2.6 Utility Maximization Problem

In the health monitoring system, patients aim to minimize both latency and energy consumption for computing their health data. In other words, minimizing latency and energy consumption for critical patients is essential. Additionally, maximizing the HSP's profit from providing health monitoring services is crucial. Therefore, we define a utility function that considers both the HSP's profit and the latency and energy costs of patients as follows:

$$\mathcal{W} = \lambda_1 \Lambda - \lambda_2 \Pi - \lambda_3 \Omega, \tag{4.20}$$

where λ_1, λ_2, and λ_3 are positive weights, with $\lambda_1 + \lambda_2 + \lambda_3 = 1$. They are assigned units inversely proportional to profit, latency cost, and energy cost, respectively, ensuring the utility is unitless. These weights vary based on the system requirements and should be considered accordingly [3, 8, 114]. When the system prioritizes energy efficiency alongside HSP's profit, it assigns higher weight to profit and energy factors ($\lambda_1, \lambda_3 > \lambda_2$), making it an energy-sensitive system. Conversely, if the system prioritizes low latency while considering HSP's profit, it assigns higher weight to profit and latency factors ($\lambda_1, \lambda_2 > \lambda_3$), defining it a latency-sensitive system. A system that equally balances all three factors ($\lambda_1 = \lambda_2 = \lambda_3$) is considered normal system. Additionally, we introduce a latency constraint, denoted as δ, to ensure that no patient experiences a delay exceeding δ/ρ_p. Therefore, our main objective is to maximize the HSP's profit while minimizing latency and energy costs for patients. Thus, we formulate the optimization problem for

the system as follows:

$$\textbf{P2:} \arg\max_{u_p, h_p^f} \ \mathcal{W} \tag{4.21}$$

Subject to the constraints:

$$u_p \frac{\beta_p}{\Upsilon} + (1 - u_p)\left(\frac{\beta_p n^f}{\Gamma_f} + \frac{\eta_p}{r_p^f}\right) \leq \frac{\delta}{\rho_p}, \tag{4.21a}$$

$$a - o \geq \upsilon \frac{\eta_{max}}{\beta_{min}}, \tag{4.21b}$$

$$\sum_{f \in \mathbb{F}} h_p^f + u_p = 1, \tag{4.21c}$$

$$\sum_{p \in \mathbb{P}} h_p^f \leq F_f^{max}, \tag{4.21d}$$

$$u_p, h_p^f \in \{0, 1\}, \tag{4.21e}$$

$\forall p \in \mathbb{P}, \forall f \in \mathbb{F}$. Eq. (4.21a) represents the latency constraint, while Eq. (4.21b) refers to the constraint on Eq. (4.19). Eq. (4.21c) ensures that each patient's health data is allocated to at most one FS, and Eq. (4.21d) sets the maximum number of patients whose health data can be allocated to an FS. Eq. (4.21e) denotes the binary variables.

In the formulated problem, each patient can either be allocated to one FS from multiple FSs or not allocated to any FS. To describe this allocation, the decision variable h_p^f can only take the values 0 or 1, as defined in Eq. (4.8), indicating whether patient p's health data is allocated to FS f or not. Additionally, the decision variable u_p denotes whether the patient's data is computed at the LD or not, and thus can only take the values 0 or 1. Therefore, the formulated problem in Eq. (4.21) is a Binary Integer Programming problem with decision variables u_p and h_p^f. Such problems are typically NP-hard, as its feasibility problem is strongly NP-complete [98]. Given the complexity of the formulated problem, this chapter proposes a sub-optimal solution for maximizing utility using matching and exchange-based approach, as discussed in the following.

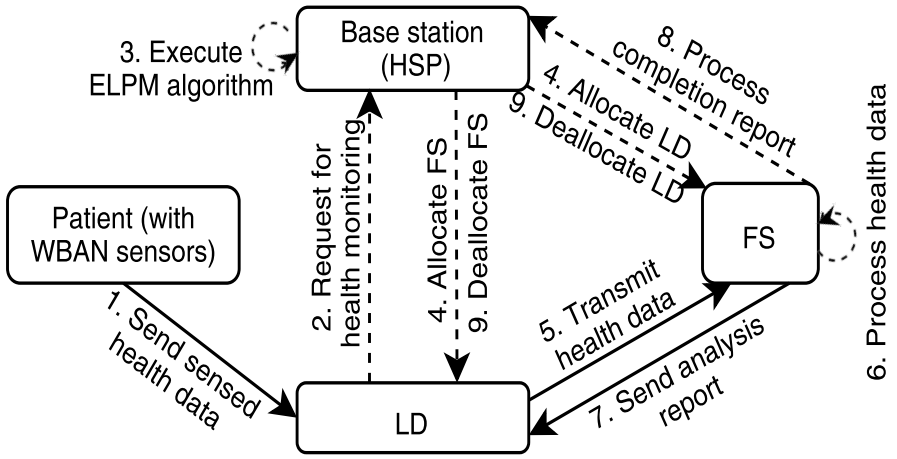


Fig. 4.2. Data flow diagram of the FC-enabled healthcare system.

4.3 Proposed Solution

In this section, we introduce matching and exchange-based sub-optimal algorithm to efficiently solve the formulated problem, namely Energy and Latency-aware Patient Monitoring (ELPM) algorithm. We assume that the base station possesses sufficient computational power to execute the ELPM algorithm for allocating patients' health data to FSs. Moreover, Fig. 4.2 illustrates the data flow among patients, FSs, and the base station. Labels 2, 3, 4, 6, 8, and 9 in Fig. 4.2 are managed by control signals, such as beacons [15], while labels 1, 5, and 7 are managed by data signals. ELPM algorithm utilizes matching and iterative repositioning of patients' health data to FSs via unidirectional and bidirectional exchange algorithms, as described in the following.

4.3.1 ELPM Algorithm

Patients prefer to choose local computation to reduce the high charges at the FS for computing their health data. However, ensuring the desired latency may not be feasible for all patients who opt for local computation. Therefore, the ELPM algorithm identifies patients who do not satisfy the desired latency constraint when their data is computed locally at LDs (lines 2-6) based on the following condition: $\rho_p T_p^{c,l} > \delta$. Then,

preference lists for patients and FSs are generated (line 7). A patient prefers an FS with the highest data transmission rate between them and a higher share of computation capacity. Therefore, the preference list of patient p over FS f is created based on the data transmission rate and the portion of computation resources allocated to the patient. However, the allocation of computation resources to the patient relies on the total number of patients assigned to an FS, which cannot be determined initially. As a result, we assume a worst-case scenario in which the number of assigned patients equals the FS's full capacity (F_f^{max}). Then, the preference list of patient p is defined as follows:

$$f \succ_p f' \Leftrightarrow W_p^f < W_p^{f'}, \tag{4.22}$$

where W_p^f and $W_p^{f'}$ represent the costs of patient p if its health data is processed by FSs f and f', respectively. Specifically, patient p prefers f over f' because f can compute its health data more efficiently in terms of latency and energy consumption. Mathematically, the costs of patient p when its health data is processed by FS f is defined as follows:

$$W_p^f = h_p^f \left(\frac{\rho_p \beta_p F_f^{max}}{\Gamma_f} + (\rho_p + \alpha_p \mathfrak{k}) \frac{\eta_p}{r_p^f} \right). \tag{4.23}$$

On the other hand, FS prefers patients from whom it can maximize its profit, considering the required CPU cycles for computation and the total size of the patient's health data. Then, the preference list for FS f is defined as follows:

$$p \succ_f p' \Leftrightarrow \mathfrak{W}_p^f > \mathfrak{W}_{p'}^f, \tag{4.24}$$

where $\mathfrak{W}_p^f$ and $\mathfrak{W}_{p'}^f$ represent the profit of FS f if it accepts patients p's and p''s health data for computation, respectively. In particular, FS f favors p if the profit gained from processing their health data is greater than that of p'. Mathematically, FS f's profit

Algorithm 4.1: ELPM Algorithm

Input: $a, o, \upsilon, \delta, \Upsilon, \omega, k, \alpha_p, \mathfrak{k}, \rho_p, \beta_p, \eta_p, F_f^{max}, \Gamma_f, V_p^f, SINR_p^f, \forall f \in \mathbb{F}, \forall p \in \mathbb{P}$.
Output: $h_p^f, \forall p \in \mathbb{P}, \forall f \in \mathbb{F}$.

1 $\mathbb{P}^v \leftarrow \varnothing$; `// Set of patients who fail to meet the latency constraint`
2 **for** *every* p *in* $\mathbb{P}$ **do**
3 **if** $\rho_p T_p^{c,l} > \delta$ **then**
4 Include patient p in the set $\mathbb{P}^v$;
5 **else**
6 Update $u_p = 1$ and $h_p^f = 0, \forall f \in \mathbb{F}$;
7 Create preference lists for patients in $\mathbb{P}^v$, as $\mathcal{L}_p$, and for FSs, as $\mathcal{L}_f$, based on the relations in Eqs. (4.22) and (4.24);
8 **while** *set* $\mathbb{P}^v$ *is not empty* **do**
9 **for** *every* p *in* $\mathbb{P}^v$ *such that* $\mathcal{L}_p \neq \varnothing$ **do**
10 $f \leftarrow$ most preferred FS from $\mathcal{L}_p$;
11 **if** $n^f < F_f^{max}$ **then**
12 Allocate patient p's health data to FS f;
13 Remove p from the set $\mathbb{P}^v$;
14 Remove f from p's preference list $\mathcal{L}_p$;
15 Update corresponding decision variables;
16 **else**
17 $p' \leftarrow$ least preferred patient of f from $\mathcal{L}_f$;
18 **if** $p \succ_f p'$ **then**
19 Allocate p to f and add p' back to $\mathbb{P}^v$;
20 Update decision variables;
21 **else**
22 Remove f from p's preference list $\mathcal{L}_p$;
23 break;
24 **repeat**
25 Execute Algorithm 4.2;
26 Execute Algorithm 4.3;
27 **until** *No exchange increases utility*;

for processing patient p's data is defined as follows:

$$\mathfrak{W}_p^f = h_p^f(\beta_p a - \upsilon \eta_p). \tag{4.25}$$

After creating preference lists for both patients and FSs, patient p's health data is allocated to its most preferred FS f if the the capacity of FS f is not already full (lines

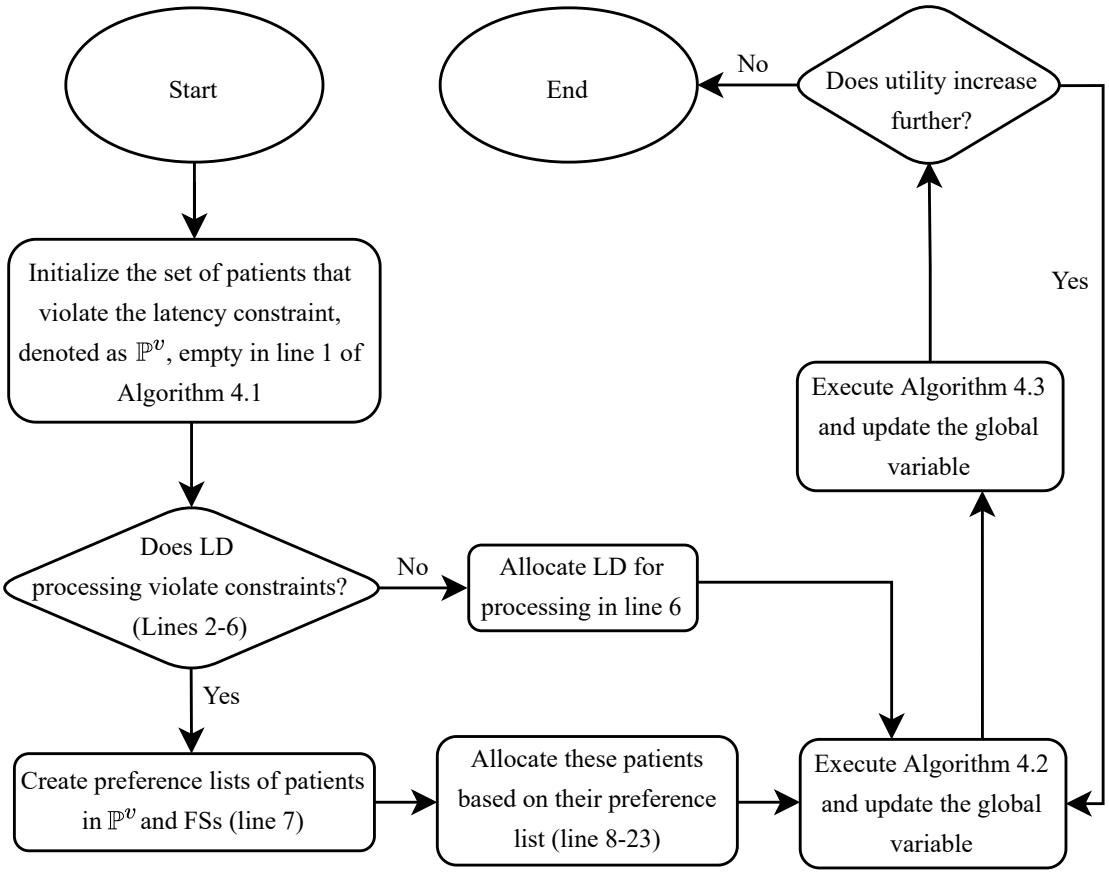


Fig. 4.3. Flow chart of ELPM algorithm.

10-12). Then, both patient p and FS f are removed from the set $\mathbb{P}^v$ and preference list $\mathcal{L}_p$ respectively (lines 13 and 14), and the corresponding decision variables are updated in line 15. If the capacity of f is full, the least preferred allocated patient p' of FS f is selected (line 17). If the preference of p is higher than that of p', then FS f is chosen for processing patient p's health data, and patient p' is added back to $\mathbb{P}^v$ (lines 18 and 19). Subsequently, the corresponding decision variables are updated in line 20. If p is not the more preferred one, FS f is removed from p's preference list (line 22). This process repeats until the set $\mathbb{P}^v$ is empty (lines 8-23). Then, ELPM repeatedly executes Algorithms 4.2 and 4.3 until there is no further increase in utility by reallocating patients' health data to different FSs in lines 24-27 (Fig. 4.3). Algorithms 4.2 and 4.3 improve the utility of the system by exchanging patients' health data among

FSs, as discussed in Subsections 4.3.1.1 and 4.3.1.2, respectively.

4.3.1.1 Unidirectional Exchange

In each iteration, Algorithm 4.2 selects a patient (p) whose health data is already assigned to an FS (f), i.e., select a patient p from the set $\mathbb{P}^f$ (lines 2-3), where $\mathbb{P}^f$ represents the set of patients allocated to FS f. It then chooses a different FS that has not yet reached its maximum capacity (lines 4-5). After selecting this FS, it computes the change in utility if the patient's health data is reassigned to the chosen FS (line 6), as follows:

$$\begin{aligned}\mathcal{W}^{uni} = (\alpha_p \mathfrak{k} + \rho_{p'}) \frac{\eta_{p'}}{r^{f}_{p'}} + \sum_{p \in \mathbb{P}^f} \frac{\rho_p \beta_p n^f}{\Gamma_f} + \sum_{p \in \mathbb{P}^{f'}} \frac{\rho_p \beta_p n^{f'}}{\Gamma_{f'}} \\ - (\alpha_p \mathfrak{k} + \rho_{p'}) \frac{\eta_{p'}}{r^{f'}_{p'}} - \sum_{p \in \mathbb{P}^f - p'} \frac{\rho_p \beta_p \left(n^f - 1\right)}{\Gamma_f} \\ - \sum_{p \in \mathbb{P}^{f'}} \frac{\rho_p \beta_p \left(n^{f'} + 1\right)}{\Gamma_{f'}} - \frac{\rho_{p'} \beta_{p'} \left(n^{f'} + 1\right)}{\Gamma_{f'}}. \quad (4.26)\end{aligned}$$

If reassigning the data to this FS increases utility, the patient's data is reassigned to that FS, and the corresponding values are updated accordingly (lines 7-9). This process continues until no further increase in utility is achievable (lines 1-10).

4.3.1.2 Bidirectional Exchange

Algorithm 4.3 selects a pair of patients whose data is currently allocated to different FSs (lines 3-6). It then calculates the change in utility resulting from exchanging the

Algorithm 4.2: Unidirectional Exchange

Input: $h_p^f, \forall p \in \mathbb{P}, \forall f \in \mathbb{F}$, information of all patients (as in Algorithm 4.1) and FSs.

Output: $h_p^f, \forall p \in \mathbb{P}, \forall f \in \mathbb{F}$.

```
1  repeat
2      for every FS f in 𝔽 do
3          for every p in ℙ^f do
4              for every f′ in 𝔽 do
5                  if f′ != f and n^{f′} ≤ F_{f′}^{max} then
6                      Compute 𝒲^{uni} using Eq. (4.26);
7                      if 𝒲^{uni} ≥ 0 then
8                          Assign p's data to f′ and remove p's data from f;
9                          Update values correspondingly;
10 until No exchange increases utility;
```

allocations of these selected patients' health data (line 7), as follows:

$$\mathcal{W}^{bi} = \rho_p \left(\beta_p \left(\frac{n^f}{\Gamma_f} - \frac{n^{f'}}{\Gamma_{f'}} \right) + \left(\frac{\eta_{p'}}{r_{p'}^f} - \frac{\eta_p}{r_p^{f'}} \right) \right) + \rho_{p'} \left(\beta_{p'} \left(\frac{n^{f'}}{\Gamma_{f'}} - \frac{n^f}{\Gamma_f} \right) + \left(\frac{\eta_p}{r_p^{f'}} - \frac{\eta_{p'}}{r_{p'}^f} \right) \right) + \alpha_p \mathfrak{k} \left(\eta_p \left(\frac{1}{r_p^f} - \frac{1}{r_p^{f'}} \right) + \eta_{p'} \left(\frac{1}{r_{p'}^{f'}} - \frac{1}{r_{p'}^f} \right) \right). \quad (4.27)$$

Then, it identifies the pair of patients for which the utility change is maximized (lines 8-10). If the utility increases, the patients are exchanged, and the corresponding decision variables are updated (lines 11-13). This process continues until no further increase in utility is achievable (lines 1-14).

4.3.2 Analysis of ELPM Algorithm

This section provides an analysis of the HSP's profit, stability, convergence, and computational complexity with respect to the proposed ELPM algorithm.

Theorem 4.1 *HSP's profit either increases or remains constant when a patient's health*

Algorithm 4.3: Bidirectional Exchange

Input: $h_p^f, \forall p \in \mathbb{P}, \forall f \in \mathbb{F}$, information of all patients (as in Algorithm 4.1) and FSs.

Output: $h_p^f, \forall p \in \mathbb{P}, \forall f \in \mathbb{F}$.

```
repeat
    W^bi_{t_max} = 0;
    for every FS f in F do
        for every p in P^f do
            for every f' in F do
                for every p' in P^{f'} if f'! = f do
                    Calculate W^bi using Eq. (4.27);
                    if W^bi ≥ W^bi_{t_max} then
                        W^bi_{t_max} = W^bi;
                        p_1 = p and p_2 = p';
    if W^bi_{t_max} > 0 then
        Exchange p_1 and p_2;
        Update corresponding decision variables;
until No exchange increases utility;
```

data is computed at FS.

Proof: Let's consider an allocation of patients' health data in the system. Then, the HSP's profit can be calculated as:

$$\Lambda = \sum_{p\in\mathbb{P}}\sum_{f\in\mathbb{F}} h_p^f \beta_p a + \sum_{p\in\mathbb{P}} u_p \beta_p o - \upsilon \sum_{p\in\mathbb{P}}\sum_{f\in\mathbb{F}} h_p^f \eta_p. \tag{4.28}$$

Take any patient p' who is utilizing its LD and allocate its health data to an FS without violating any constraints, while keeping all other patients' allocations unchanged. Then, the HSP's new profit can be calculated as follows:

$$\Lambda' = \sum_{p\in\mathbb{P}}\sum_{f\in\mathbb{F}} h_p^f \beta_p a + \sum_{p\in\mathbb{P}} u_p \beta_p o - \upsilon \sum_{p\in\mathbb{P}}\sum_{f\in\mathbb{F}} h_p^f \eta_p + \beta_{p'} a - \beta_{p'} o - \upsilon \eta_{p'}. \tag{4.29}$$

Now, the change in profit $\Lambda' - \Lambda$, is calculated as follows:

$$\Lambda' - \Lambda = \beta_{p'}(a - o) - \upsilon\eta_{p'}. \tag{4.30}$$

From Eqs. (4.19) and (4.30), it can be concluded that $\Lambda' - \Lambda \geq 0$, i.e., HSP's profit either increases or remains constant when a patient's health data is processed at FS. □

Theorem 4.1 demonstrates that the proposed model ensures that the HSP's profit either increases or remains the same when a patient's health data is processed at an FS. Furthermore, to establish the stability of the proposed ELPM algorithm, it is essential to establish the non-existence of Blocking Pairs (BPs). Therefore, we formally define a BP in Definition 4.1.

Definition 4.1 (BP) *A pair consisting of a patient and an FS, denoted as (p, f), is considered BP if both prefer to be matched with each other rather than with their currently allocated pairs, i.e., (p, f') for patient p and (p', f) for FS f.*

Definition 4.2 (Stability) *The proposed ELPM algorithm is stable if it does not contain any BPs.*

Theorem 4.2 *The proposed ELPM algorithm results in stable allocation of patients' health data to FSs.*

Proof: To establish the stability of the proposed ELPM algorithm, we need to show the absence of BPs. Let us assume that BP exists, denoted as (p, f) for patient p, after its allocation to FS f'. This implies that $W_p^f < W_p^{f'}$. However, for patient p not to be allocated to FS f, the costs of patient p must be lower when its health data is processed by FS f' than the costs derived from the BP (p, f). This contradicts the assumption that (p, f) is a BP. Therefore, after applying the ELPM algorithm, no BPs exist, thereby proving the theorem. □

Theorem 4.3 *The proposed ELPM algorithm converges.*

Proof: The ELPM algorithm terminates its execution when the loop in lines 8-24 of Algorithm 4.1, as well as in Algorithms 4.2 and 4.3, converge. Lines 8-24 repeat until no patient remains in the set $\mathbb{P}^v$, which is bounded by PF. Therefore, the loop in lines 8-24 terminates. Furthermore, Algorithms 4.2 and 4.3 only exchange patients when utility increases; otherwise, they terminate. Since the total possible exchanges are finite, exchange algorithms terminate in finite time. Additionally, Algorithms 4.2 and 4.3 iterate multiple times, converging in each iteration and proceeding to the next iteration only when utility increases. Hence, the proposed ELPM algorithm converges. □

Theorem 4.4 *Computational time complexity of the proposed ELPM algorithm is* $O(PF \log PF + P^2)$.

Proof: In Algorithm 4.1, identifying patients who do not met latency constraint takes P iterations (lines 2-6) and creating preference lists for patients and FSs (line 7) takes $O(PF \log PF)$. Lines 9-10 select the most preferred FS for each patient with a non-empty preference list, and lines 11-15 update allocations and variables if the capacity constraint is met, each in constant time ($O(1)$). If capacity constraints aren't met, selecting and swapping the least preferred patient (lines 17-19), updating decision variables (line 20), and adjusting the patient's list (line 22) also takes $O(1)$. Therefore, time complexity of lines 9-23 is $O(P)$. This process continues until all patients in set $\mathbb{P}^v$ are allocated or dropped due to FS capacity limits (lines 8-23), resulting in $O(P^2)$ time complexity. Algorithm 4.2 considers P^2 pairs of patients and repeats until it converges. Since this number of iterations is bounded by a finite value (P^2), the time complexity of Algorithm 4.2 is $O(P^2)$. Similarly, the time complexity of Algorithm 4.3 is $O(PF)$ as the possible number of exchanges is PF. Hence, the time complexity of the ELPM algorithm is $O(PF \log PF + P + PF + 2P^2)$, i.e., $O(PF \log PF + P^2)$. □

Table 4.1: Parameters used in simulation

Parameters	Value
P, F, η_p [8]	[80-400], [20-80], [1, 3] MB
β_p [8]	[100, 1000] Megacycles
δ [3], ρ_p [3], Υ [3]	250 ms, [0, 1], 2.4 GHz
o, a, υ, α_p	0.7, 1.4, 23.33, 0.3
ω, Γ_f [3]	15 MHz, [18-22.4] GHz
$SINR_p^f$ [15], V_p^f [3]	[13-20] dB, [5-15]
ψ [111], F_f^{max}, $\mathfrak{k}$ [4]	10^{-24}, [1-4], 10^5 micro-Watt
$\lambda_1, \lambda_2, \lambda_3, \mathcal{M}$	0.33, 0.33, 0.33, 3
Blood pressure ($\theta_{low,s}$, $\theta_{up,s}$) [89]	91 mmHg, 169 mmHg
Cholesterol level ($\theta_{low,s}$, $\theta_{up,s}$)	200 mg/d, 239 mg/d
Heart rate ($\theta_{low,s}$, $\theta_{up,s}$) [89]	51 bpm, 139 bpm

4.4 Performance Study

In this section, we evaluate the performance of the proposed model through extensive simulations conducted on a Windows 10 Home PC equipped with an Intel® Core™ i7-10750H @ 2.60 GHz processor. We simulate a smart healthcare system where the HSP provides health monitoring services using FSs, considering scenarios with 80 to 400 patients requesting health monitoring services and 20 to 80 FSs. Each patient's health data size is randomly selected between 1 to 3 MB, and the required CPU cycles for computation are randomly chosen from 100 to 1000 Megacycles [8]. We set a constant value of δ to 250 ms [115]. The criticality of patients is chosen within the range of [0, 1] as per Eq. (3.3). Bandwidth, number of PRBs, and SINR are set to 15 MHz, [5-15] [3], and [13-20] dB [15], respectively. The transmission power of LDs is fixed at 10^5 micro-Watt [4]. The detailed simulation parameters are provided in Table 4.1.

To the best of our knowledge, most existing works do not simultaneously consider patient criticality, HSP's profit under a dynamic pricing scheme, latency, and energy costs of patients (refer Table 2.2). We compared our proposed ELPM algorithm with the UMPM scheme [3], which closely aligns with our work unlike other existing works. The UMPM algorithm strategically integrates criticality, HSP's profit, and latency costs

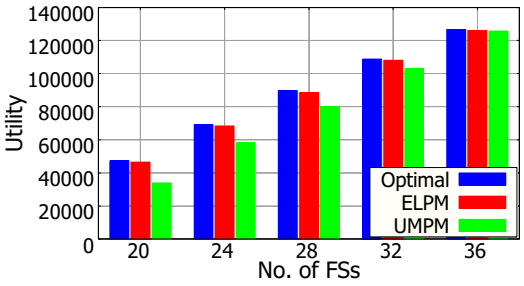
to achieve efficient sub-optimal utility using swapping-based heuristics for allocating and reallocating patients' health data. To ensure fairness, we excluded FS allocation where patients' latency constraints were not violated. Additionally, we used the Gurobi optimization tool [103] to obtain an optimal solution for further comparison with our proposed model.

4.4.1 Experimental Results

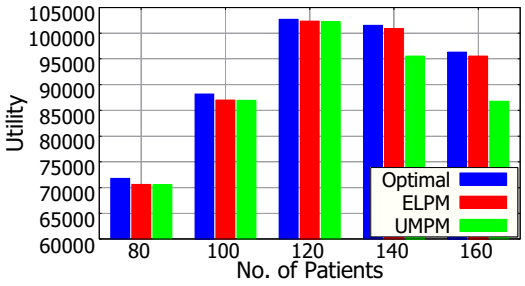
In this section, we evaluate the effectiveness of the proposed model on various aspects as discussed below.

Utility Analysis: Fig. 4.4 compares the utility of various schemes. In Fig. 4.4a, with the number of patients fixed at 150 and the number of FSs varying from 20 to 36, utility increases as the number of FSs rises. Fig. 4.4b shows utility with the number of FSs fixed at 30 and the number of patients varying from 80 to 160, where utility initially increases but then decreases due to the increased load on FSs with a higher number of patients. The ELPM algorithm achieves an average utility of 99.01%, whereas UMPM achieves 94% of the optimal value. The reason is that UMPM relies solely on latency and criticality for choosing the FS for computing patients' health data, without considering dynamic pricing and energy costs. In contrast, ELPM incorporates all these parameters using matching and exchange-based approach. However, the optimal solution outperforms ELPM as it evaluates all possible combinations of allocations and selects the best ones to maximize utility.

Latency Analysis: Fig. 4.5 presents a comparison of latency costs among different schemes. We observe from the results that the ELPM algorithm yields lower latency costs compared to UMPM. This is because the ELPM algorithm uses matching theory for allocating FSs to process patients' health data and adjusts the allocation to maximize utility by minimizing latency costs. We also observe that the ELPM algorithm incurs higher latency costs compared to the optimal solution, which explores all

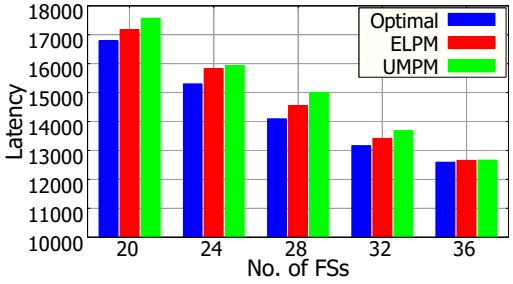


(a) Utility vs FSs at P=150

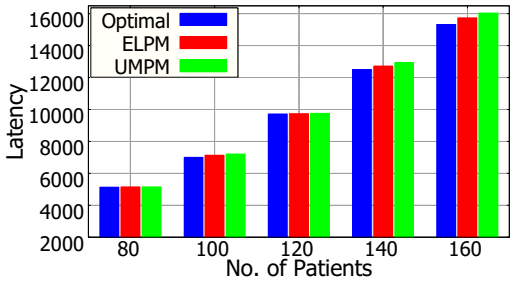


(b) Utility vs patients at F=30

Fig. 4.4. Comparison of utility among different schemes.



(a) Latency vs FSs at P=150

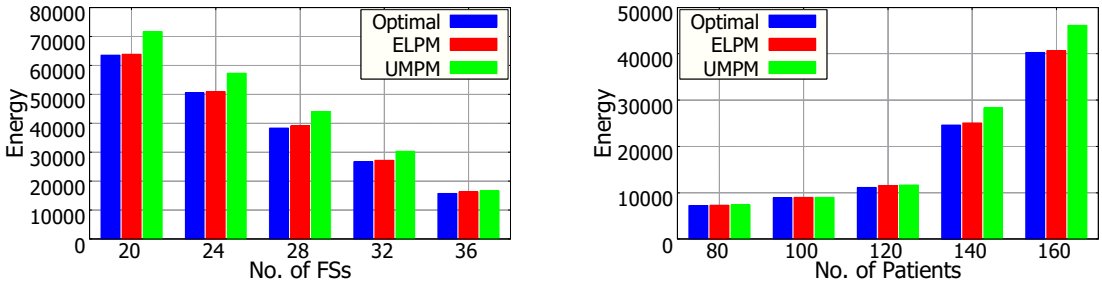


(b) Latency vs patient at F=30

Fig. 4.5. Comparison of latency among different schemes.

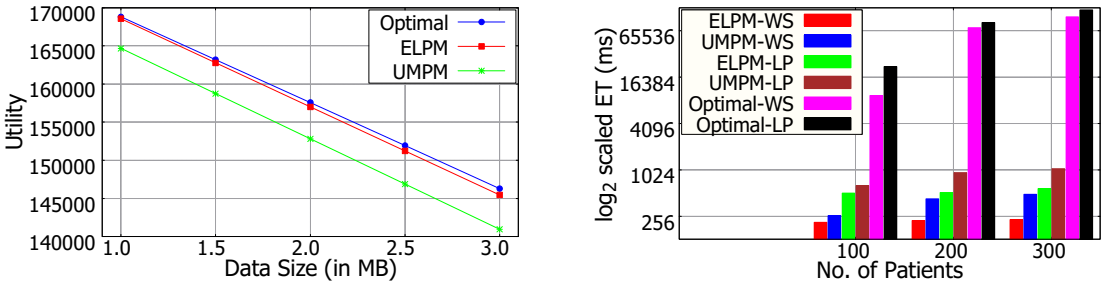
possible allocations to minimize latency costs. Moreover, latency costs decrease with an increasing number of FSs, as seen in Fig. 4.5a, due to the increase in total computation capacity of the FSs, resulting in decreased computation latency. Furthermore, latency costs increase with a higher number of patients, as shown in Fig. 4.5b, because processing a larger number of patients' health data increases the load on the FSs, leading to higher latency costs.

Energy Analysis: Fig. 4.6 compares the energy costs of patients across different schemes under various scenarios. We observe from the results that the ELPM algorithm yields lower energy costs compared to UMPM. This is because the ELPM algorithm considers energy consumption as a parameter when allocating FSs for computing patients' health data, whereas UMPM does not account for energy consumption in its allocation process. Additionally, Fig. 4.6a shows a decrease in energy costs as the number of FSs



(a) Energy vs FSs at P=150 **(b)** Energy vs patients at F=30

Fig. 4.6. Comparison of patients' energy consumption.



(a) Utility vs data size **(b)** Execution time

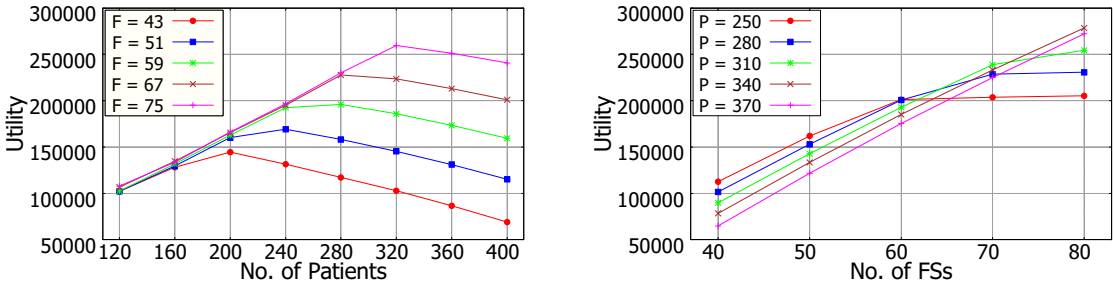
Fig. 4.7. Utility vs data size and execution time.

increases. This is because, as the number of FSs increases, more patients' health data can be computed at FSs, reducing the need for local computation and resulting in lower energy costs. Furthermore, we observe an increase in energy costs with the number of patients, as shown in Fig. 4.6b. This increase is due to the higher demand for local computation when the number of patients grows while the total capacity of the FSs remains constant, leading to a rise in energy costs.

Impact of Data Size on Utility: Fig. 4.7a illustrates the impact of data sizes on utility. We observe from the result that utility decreases as data size increases. The highest utility is observed when the patient's health data size is minimal (1.0 MB). In contrast, utility diminishes as the health data sizes of patients increase to 3.0 MB. This is due to the increased transmission latency and the higher expenses incurred by the HSP, which consequently reduce the system utility.

Table 4.2: Device specifications

Devices	Specification
Laptop	**Processor:** Intel(R) i5-9300H @2.40 GHz, **RAM:** 8 GB
Workstation	**Model:** Dell Precision 3640 Workstation, **Processor:** 11th Gen Core-i7 -10700 CPU (8 Core(s)) @ 4.10 GHz, **RAM:** 32 GB



(a) Utility vs patients **(b)** Utility vs FSs

Fig. 4.8. Utility of ELPM algorithm on various parameters.

Execution Time Analysis: Fig. 4.7b compares the Execution Time (ET) across various scenarios using a Laptop (LP) and a Workstation (WS)[2], with the number of FSs fixed at 50. For the optimal solution, we consider the ET required by the Gurobi optimization tool to solve the formulated optimization problem. Additionally, Fig. 4.7b is plotted using a $\log_2$ scale on the y-axis for better readability. We observe from the result that the ET increases as the number of patients increases. This is due to the growing number of possible allocation combinations, which leads to longer ET. Furthermore, we observe that the ET varies with different machine configurations, demonstrating that it depends not only on the number of patients requesting health monitoring services and the number of FSs but also on the machine's configuration, specifically the computational power at the base station. Moreover, the optimal solution requires significantly more time compared to the ELPM algorithm, as it evaluates all possible allocation combinations to maximize utility.

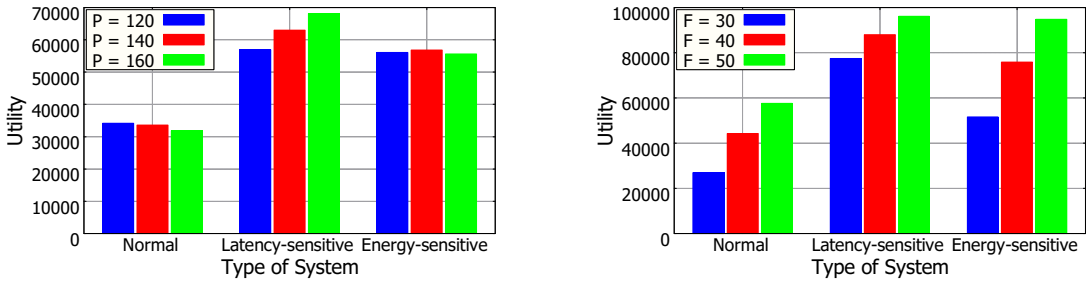
Utility of ELPM on Various Parameters: Fig. 4.8a illustrates different

[2]Specifications of LP and WS are provided in Table 4.2.

scenarios with varying numbers of patients and FSs. We observe from the result that for a given number of patients, utility increases with the number of FSs. This is due to the decrease in computation latency as the total computation capacity of the FSs increases with the growing number of FSs. Additionally, for a fixed number of FSs, utility initially rises with the number of patients but starts to decrease after reaching a certain number of patients. This trend occurs because, at first, the computation required to process patients' health data is less than the total computation capacity of the FSs. However, as the number of patients continues to grow, the total computation demand exceeds the total computation capacity of FSs, leading to higher latency and energy costs, thereby reducing utility.

Fig. 4.8b considers different scenarios with varying numbers of patients, and the number of FSs ranging from 35 to 80. The result shows that utility increases as the number of FSs grows. This is due to the increased in total computation capacity of FSs, while the total required CPU cycles for computing patients' health data remain constant, leading to lower latency and energy. Additionally, the result shows that for a given number of FS, utility decreases as the number of patients increases. This decrease is due to increased computation latency associated with a higher number of patients, coupled with the fixed total computation capacity of the FSs, resulting in higher latency and energy costs.

Utility on Different Types of Systems: Fig. 4.9 compares utility across three different types of system. In the normal system, profits, latency, and energy costs are considered equally, with $\lambda_1 = \lambda_2 = \lambda_3 = 0.33$. The latency-sensitive system prioritizes profit and latency, ignoring energy costs, i.e., the weights are set as $\lambda_1 = \lambda_2 = 0.5$ and $\lambda_3 = 0$. The energy-sensitive system emphasizes profit and energy cost, i.e., the weights are set as $\lambda_1 = \lambda_3 = 0.5$ and $\lambda_2 = 0$. In Fig. 4.9a, we observe that utility decreases as the number of patients increases due to higher load on FSs and increasing energy consumption of LDs. In latency-sensitive system, utility rises due to exclusion



(a) No. of FSs = 30 **(b)** No. of patients = 200

Fig. 4.9. Utility for different types of systems.

of patients' energy cost as $\lambda_3 = 0$. The increase in HSP's profit exceeds the rise in latency cost, thus increasing utility. In energy-sensitive system, the increase in HSP's profit is balanced by the increase in energy costs of patients; thereby, utility remains same. Furthermore, we observe from Fig. 4.9b that utility increases with the number of FSs as they can process more health data, reducing local computation needs and lowering latency and energy costs for patients.

Convergence Analysis: Fig. 4.10 shows the convergence of ELPM algorithm in three distinct scenarios. We observe that the utility increases with the number of patients and FSs, as does the number of iterations, due to the increasing allocation possibilities between patients and FSs. However, after a certain number of iterations, the increase in utility becomes negligible in all scenarios, and the process eventually stops within a finite number of iterations. Therefore, the proposed ELPM algorithm converges in a finite time.

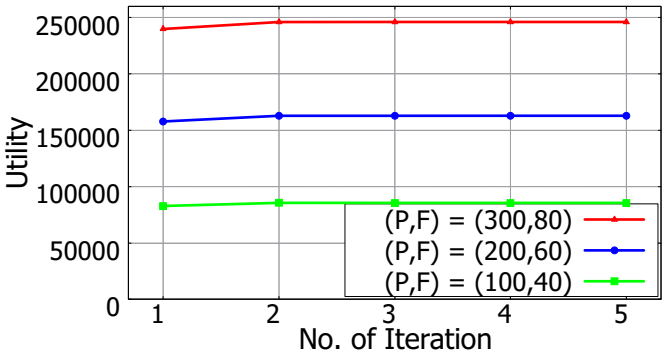
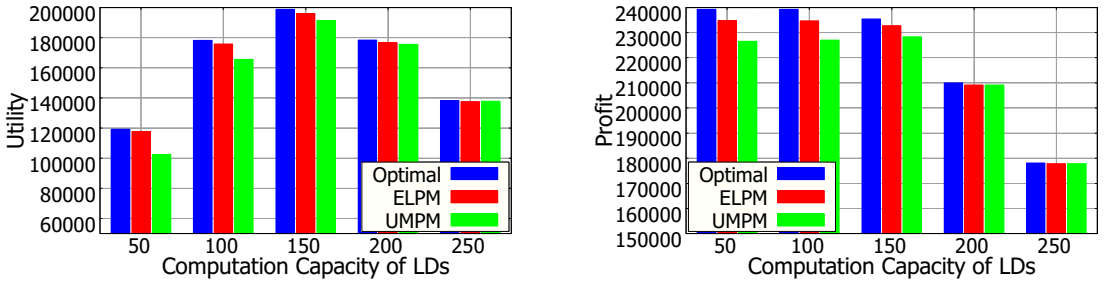


Fig. 4.10. Convergence analysis.



(a) Utility vs computation capacity **(b)** Profit vs computation capacity

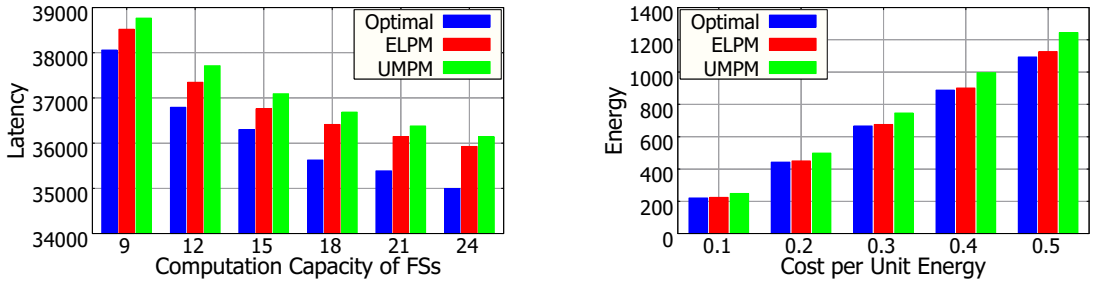
Fig. 4.11. Utility and profit vs computation capacity of LDs.

4.4.2 Experimental Results using Real-World Data

In this section, we evaluate the effectiveness of our model using SimPy simulation tool [116] under various settings. Our simulation involves 30 FSs and 270 patients requesting heart monitoring services from the HSP (base station). We simulate a scenario using the real-world Statlog (Heart) dataset [106], which contains 270 samples with 13 attributes. We focus on three key attributes: cholesterol level, blood pressure, and heart rate data, to represent the heart disease data collected by WBAN-enabled sensors attached to patients. Other simulation parameters are as provided in Table 4.1. To establish the priority or rank (refer Section 4.2.1), cholesterol level data is given the highest priority, followed by blood pressure and heart rate data. Accordingly, the priority ranks for cholesterol level, blood pressure, and heart rate data are assigned as 1, 2, and 3, respectively. Furthermore, we use a deep neural network model to monitor patients' heart disease condition[3].

Fig. 4.11a compares the utility of various schemes while varying the computation capacity of LDs from 50 MHz to 250 MHz. We observe from the result that utility increases as the computation capacity of LDs rises until a certain point, after which it gradually decreases with further increases in LDs' computation capacity. This is because increasing the computation capacity of LDs reduces computation time, resulting in

[3]Developing a heart disease monitoring model is beyond the scope of this chapter, but it can be achieved using methods given in work [112].



(a) Latency vs computation capacity **(b)** Energy vs cost

Fig. 4.12. Latency and energy vs computation capacity of FSs and cost.

higher utility. However, beyond a certain point, the increase in energy consumption due to local computation rises polynomially, leading to a decreasing trend in utility.

Fig. 4.11b shows the profit comparison of the proposed ELPM algorithm with the optimal solution and the UMPM, while varying the computation capacity of LDs. We observe from the result that initially, the profit remains constant when the computation capacity of LDs is low. This is because when LDs have low computation capacity, patients' health data cannot be processed within the desired latency, necessitating the allocation of FSs, thereby increasing profit due to higher fees charged for FS computations compared to LDs. However, as the computation capacity of LDs increases further, more patients' data can be processed within the desired latency at LDs, leading to fewer patients' health data being computed at FSs and consequently lower profit due to the lower fees charged for computations at LDs.

Fig. 4.12 compares how latency and energy costs vary across different schemes as the computation capacity of FSs and the cost per unit energy consumption change. Fig. 4.12a compares the latency costs of different schemes with a fixed computation capacity of LDs at 150 MHz, while varying the computation capacity of FSs from 9 to 24 GHz. We observe from the result that the latency costs decrease as the computation capacity of FSs increases. This is because the higher computation capacity of the FSs enables faster processing of patients' health data, thereby reducing computation latency.

Fig. 4.12b depicts how the cost per unit energy consumption affects energy when the computation capacity of FSs and LDs is fixed at 22 GHz and 150 MHz, respectively. We observe that the energy cost increases as the cost per unit energy consumption for LDs rises. This is because higher costs lead to increased energy expenses for patients during local computation or data transmission to FSs, thereby reducing overall energy efficiency.

4.5 Summary

This chapter proposed a fog computing-enabled WBAN-based system for smart healthcare applications. We formulated a utility maximization problem that considers HSP's profit, as well as latency and energy costs of patients, while prioritizing critical patients' health data. Furthermore, we applied matching and exchange-based sub-optimal algorithm to efficiently solve the formulated problem. We analyzed the impact of various factors on utility, profit, latency, and energy, including the number of patients, the number of FSs, and the health data size. Extensive experimental and simulation results using real-world data validated the effectiveness of the ELPM algorithm, achieving an average utility of 99.01% of the optimal solution within polynomial time complexity. This chapter primarily focused on optimizing latency, energy, and profit in a remote health monitoring system while prioritizing critical health data. However, to further improve healthcare quality, it is equally important to design a system that enables the development of effective ML models for detecting or predicting patients' health conditions. Therefore, the next chapter focuses on developing a system that enables the creation of effective ML models while also addressing key aspects such as privacy concerns and interference issues during data collection for building these models.