Intelligent Sensor-Cloud in Fog Computer: A Novel Hierarchical Data Job Scheduling Strategy

Zeyu Sun 1,2, Chuanfeng Li 1,*, Lili Wei 1, Zhixian Li 1, Zhiyu Min 1 and Guozeng Zhao 1

1 School of Computer Science and Information Engineering, Luoyang Institute of Science and Technology, Luoyang 471023, China; lylgszy@163.com (Z.S.); nywily@163.com (L.W.); zxlee0402@163.com (Z.L.); mindayu@163.com (Z.M.); ly_zgz@163.com (G.Z.)
2 Key Laboratory of Intelligent IoT, Luoyang Institute of Science and Technology, Luoyang 471023, China
* Correspondence: lcflit@163.com; Tel.: +86-379-6592-8117

Received: 6 November 2019; Accepted: 19 November 2019; Published: 21 November 2019

Abstract: In the Fog Computer (FC), the process of data is prone to problems such as low data similarity and poor data tolerance. This paper proposes a hierarchical data job scheduling strategy Based on Intelligent Sensor-Cloud in Fog Computer (HDJS). HDJS dynamically adjusts the priority of the job to avoid job starvation and maximize the use of resources, uses the key frame to the resource occupied information, distributes the frame sequence to the unit, and then combines the intra frame distribution strategy to balance the load between the nodes. The experimental results show our proposed strategy may be possible to avoid the operation of hunger and resource fragmentation problems, make full use of the advantages of multi-core and multi-thread, improve system resource utilization, and shorten the execution time and response time.

Keywords: wireless sensor networks; job scheduling; sensor cloud; scheduling strategy

1. Introduction

The job-scheduling of the traditional wireless sensor network usually emphasizes a certain aspect but balances the property of the system from the whole view [1–3]. In addition to this, the division strategy of average node job makes the jobs between the nodes imbalanced, which leads to the problem that the nodes which are complicated in the task are put off the whole time of the job. Facing the increase in wireless sensor network data combined with the features of the job, developing a high speed job whose scheduling cost is low and that can also satisfy the users’ QoS needs is the key to improve the concurrent rate and to reduce the waiting time of user jobs. The distribution strategy of the average task is meaningful for reducing the influence of the finishing time of the nodes, wooden pail effect, and for improving the data computing ability [4–6]. Because the job scheduling and division strategy are meaningful for the improvement of the system property, scholars from all over the world currently study this area widely in relation to Wireless Sensor Networks (WSNs).

In WSNs, the most usual scheduling strategy, First Come First Service (FCFS), is based on the order of the jobs to schedule or to divide at the source [7]. The serious FCFS scheduling mechanism guarantees the fairness of the users but sacrifices the throughput efficiency and the source using rate of the system, which easily presents the problem of node free and node congestion [8–10]. The strategy of FirstFit based on the order that the jobs get into the queue of the jobs and run the first job whose source requirement is low, which makes it not be solved for a long period [11]. The aspect of the fairness is limited, and it can also prolong the average waiting time of the system. Although the Job Scheduling Level Method (JSLM) has high throughput efficiency, it can prolong the time of calculating complicated jobs and also
cannot satisfy fairness. BestFit strategy runs the job whose source is the highest and which can satisfy its requirement. At the same time, it can also lead to the phenomenon of job starvation [12,13].

In the WSNs based on sensor-cloud, with the rapid development of the high-performance computing and digital multimedia, the data render job scheduling becomes an important means to solve the single computing node [14,15]. The application has both computational and data intensive hybrid features [16]. At the same time, the emergence of big data technology and cloud computing technology further enhances the WSNs ability to render the extension.

The extension of WSNs computing scale has a high processing capacity, but if there is no system optimization strategy, the system cannot give full play to the computing power [17–19]. How to optimize the task scheduling strategy, assign the task to the nodes according to the computation, and improve the resource utilization and system performance is an important challenge for performance optimization research [20–23]. In addition, the expansion of the scale of the WSNs also means the increase of the energy consumption of the system [24–26]. How to design a reasonable scheduling strategy to reduce the energy cost of the system is also a very challenging problem in the WSNs.

The rest of this paper is organized as follows. Section 2 discusses the scheduling level method of job. The hierarchical job scheduling strategy was proposed in Section 3. Section 4 shows the simulation's experimental results, and Section 5 summarizes the full paper.

2. Related Works

In the WSNs, when optimizing energy consumption, the property also plays an important role as a target. Aiming at the question which is about the balance of energy consumption and the property, there are many researchers currently working on the topic. He et al. [27] reduced the consumption of energy through minimizing the number of activated physical devices. Promoting a resource management framework, which is oriented to the property and the energy, Liu et al. [28] achieved the balance of property and energy by improving the knowledge of the theory. Aiming at the issue about the decreasing of the user service quality, which was caused by excessive integration of the resource, Wang et al. [29] solve the average optimization between the energy and the property by using the dynamic migration of the virtual machine. According to the platform of the SaaS, Zhao et al. [30] proposed a sort of resource scheduling scheme to balance the energy consumption and the property. The technology of Job Scheduling Optimization Strategy (JSOS) appears as a new technology in the current low energy consumption design of the computer system, which leads to many new challenges and chances for the energy-saving scheduling of the progress. Using the feature of JSOS technology to dispatch the task can make the high-performance cluster decrease the cost of energy the most in the whole system, considering the task performance. Sun et al. [31] proposed Heterogeneity-aware Loads Balancing (HLB) technology to balance distribution of the different tasks, which decreases the energy consumption overall, and to set the constraint of the temperature. Based on the Task Distribution Layer Method (TDLM) technology, Wang et al. [32] promoted a sort of heuristic scheduling algorithm to reduce the executive energy consumption, which was produced by the parallel task in the cluster environment. On the condition that this kind of algorithm does not influence the time of the whole tasks, the algorithm reduces the consumption of the energy by reducing the node voltage of the task. Liu [33] also did the same research to put forward a kind of task scheduling algorithm based on the energy-aware in the integrated environment, which reduces the power consumption through controlling the proper level of the voltage. Aiming to the JSLM proposed a sort of task scheduling strategy of the energy-aware, which adjusts the needed voltage dynamically, according to dividing the length of the subtask and minimizing the energy consumption under the condition of satisfying the user’s deadline. Wu et al. [34] promoted a sort of the energy saving system of the energy-aware, which judges whether the users are in the situation of activation or not by setting the dynamic reconfiguration models. Through turning on or turning off the application selectively to achieve the purpose of energy saving, this kind of system can acquire a good energy use rate, and at the same time, it can also provide a different energy scheduling strategy. Aiming to solve the task scheduling problem in the
heterogeneous computing environment, Liu et al. [35] put forward an Energy-Conscious Scheduling (ECS) algorithm, which balances the task property and energy consumption by minimizing the value of Relative Superiority (RS) by making the RS of the task property and the cost of the energy consumption the optimization goal. On the basis of the optimization goal of the RS value, Sun et al. [36] combined the algorithm of the genes and proposed a meta-heuristic scheduling algorithm to optimize the property and the energy consumption of the tasks. Wang et al. [37] makes the energy consumption the constraint condition and makes the properties of the concurrent tasks the optimizing goals. Sun et al. [38] put forward a scheduling framework of the heterogeneous energy awareness resource, which makes the heat balance get rid of the ‘heat point’ by adjusting the voltage of the services dynamically. Li et al. [39] put forward energy awareness task fusion technology to optimize the energy consumption of the cluster, which reduces the energy consumption of the system through maximizing the method of using resources. Liu et al. [40] achieves the goal of optimizing the energy consumption by optimizing the free energy consumption and the running energy consumption effectively in the process of running the task. According to the parallel task scheduling, Shao et al. [41] put forward an optimizing algorithm, which concerns the running property, the cost, the reliability, even the energy consumption, and so on. However, this algorithm needs much exploration of the features of the task and even the features of the RS, and in the aspect of optimizing the energy, it also has much knowledge to explore.

In order to guarantee the fairness of the users’ work, the promoted work scheduling level method sets the dynamic priority for the work to guarantee the work priority scheduling method whose priority is high. When the source is limited, in the situation of not delaying the running of the original high priority work, the work priority of adjusting high priority dynamically will dispatch the work whose priority is low in advance, and at the same time, it will save resources for the work whose original priority is high [42]. Once the work requirement whose source satisfies the high level priority stopped the work whose priority is low and restarted to patch the work whose priority is high, the mission will send the frame sequence to the unit by way of making the unit the source particle size basic unit based on scene geometry, to choose the key frame. This uses the feedback of the source occupying information of the key frame and the related feature between the frames, combining the distributing strategy among the frames. The innovative points are as follows:

1. Setting the dynamic priority to guarantee the scheduling fairness.
2. Allowing the work whose priority is low and whose source requirement is small to run in advance to guarantee the adequate usage of the source, which also preserves the source for the work whose priority is high or whose requirement of source is big, and which also prevents situations of work starvation and reduction of source fragments.
3. The mission will send the frame sequence to the unit by way of making the unit the source particle size basic unit, based on scene geometry to choose the key frame, using the feedback of the source occupying information of the key frame and the related feature of between the frames. It combines the distributing strategy among the frames to render the work load between the nodes in balance.

3. Job Scheduling Level Method

3.1. Basic Concept

In the WSNs, the job scheduling layer mainly deals with the task submitted by the user. According to the needs of the users, the job is scheduled to the corresponding resource [43–45]. In order to better describe the method of job scheduling layer, the following definitions are introduced.

**Definition 1.** In the queue waiting for the scheduling job, the fill-in job is the job that, at the back of the queue, is scheduled first.

**Definition 2.** The unit is a collection of CPU cores in the node.
Definition 3. The use rate shows the extent to which the sources in the system are busy. In general, we use the percentage of using source. However, it uses the percentage of the CPU usage to show the source's use rate in the paper.

The use rate of the system includes the real-time source use rate and the average source use rate. We assume that the real-time source use rate is \( P_r \), and the average source use rate is \( P_{r,T} \) in the time period \( T \). So that the real-time source use rate is the following:

\[
P_r = \frac{N_b}{N} \tag{1}
\]

In the Formula, \( N_b \) indicates the amount of the CPU that is working; \( N \) shows the whole number of the CPU.

From the time period of \( T_a \) to \( T_b \), the average source use rate is the following:

\[
P_{r,T} = \frac{1}{T_a - T_b} \int_{T_a}^{T_b} P_r(t) dt \tag{2}
\]

Dividing this period of time into \( m \) areas, the time in every area is \( \Delta t \), and the result is as follows:

\[
\Delta t = \frac{T_a - T_b}{m} \tag{3}
\]

By using the conception of infinite segmentation, when \( n \to \infty, \Delta t \to 0 \), the average source rate can be shown as:

\[
P_{r,T} = \frac{1}{m \Delta t \to 0} \lim_{m} \sum_{i=1}^{m} P_r(i \cdot \Delta t) \tag{4}
\]

So that the average real-time source use rate in this paper states the source use rate in a certain time approximately.

Assuming a certain work \( J_i \) whose original time of moment is \( TJI(i) \) is the moment that the work \( J_i \) was submitted to the system by the users. Assuming that the start moment of the work is \( TJS(i, k) \), deadline moment is \( TJF(i, k) \), \( k \) is the number of the CPU that is needed. Therefore, the time of the work \( J_i \) which needs \( k \) CPU is:

\[
TJE(i, k) = TJF(i, k) - TJS(i, k) \tag{5}
\]

Definition 4. The whole time of the work indicates the whole finishing time of all the work in the work queue. It can also see it as the gap between the latest moment and the earliest moment.

Therefore, the whole finishing time \( TAJF(n) \) of the \( n \) works in the work queue is:

\[
TAJF(n) = \text{MAX}(TJF(i, k)) - \text{MIN}(TJI(i)) \tag{6}
\]

Definition 5. The response time of the work is the gap of the time between the original and the first time it starts, and it is also the waiting time of the work.

Definition 6. The turnover time of the work is the time gap between the start moment and the over moment. It not only includes the running time but the response time as well.
Assuming that $T_{Jres}(i,k)$ shows the response time of $k$ CPU source work $J_i$, and $T_{ave\_res(n)}$ shows the average response time of $n$ works, we can conclude that:

$$T_{ave}(i,k) = T_{JS}(i,k) - T_{JI}(i,k)$$  \hspace{1cm} (7)

$$T_{ave\_res(n)} = \frac{1}{n} \sum_{i=1}^{n} T_{ave\_res(i)}$$  \hspace{1cm} (8)

Assuming that $T_{Jturn}(i,k)$ is the finishing time of $k$ CPU source work $J_i$, and $T_{ave\_turn(n)}$ is the average turnover time of $n$ works, we can conclude that:

$$T_{Jturn}(i,k) = T_{JF}(i,k) - T_{JI}(i,k)$$  \hspace{1cm} (9)

$$T_{ave\_turn(n)} = \frac{1}{n} \sum_{i=1}^{n} T_{ave\_turn(i)}$$  \hspace{1cm} (10)

The analysis of the characteristics of operation is a sequence of frames, a group of mutual dependence and parallel split, in the WSNs [46]. Therefore, this method can be adjusted by dynamic priority and preemption mechanism to ensure fairness. The idea is to ensure maximizing the utilization of resources so that the free resources can be low priority and resources need little homework ahead of time, at the same time that resources are set aside for high priority and resource demanding work, in the WSNs.

3.2. Constraint Conditions for Data Jobs

Due to it being a data intensive application in terms of the memory requirements of the distribution of a wide range, there are dozens of MB of memory occupied by a simple scene, and it also has more than a dozen GB of memory of complex scenes [47,48]. Therefore, it is necessary to take account of the memory footprint in relation to the unit’s size. If you run a unit through all nodes of the process of the physical memory, the operating system will use the swap partition to ensure the normal operation of the process; so frequently, the disk data exchange will bring an extra-heavy overhead, and it will greatly extend the time. Therefore, the unit partition strategy needs to satisfy the following constraints:

1. The memory of all the processes running on the unit does not exceed the physical memory capacity of the node,

$$\sum_{i=1}^{p} m_i \leq M - \sigma$$  \hspace{1cm} (11)

in which $m_i$ is the memory for each process, $M$ is the machine memory capacity, $p$ is the number of processes running simultaneously, and $\sigma$ is the memory of the system.

In addition, it is also a computationally intensive application processor; task switching overhead will usually offset the boost processor sharing performance and even lead to performance decline. Therefore, the unit partition strategy should satisfy the following constraints:

2. The sum of the number of threads that run on the unit cannot exceed the number of processors in the node,

$$\sum_{i=1}^{p} t_i \leq C$$  \hspace{1cm} (12)

in which $t_i$ is the number of threads in the $i$ process, $C$ is the number of processor cores for the node, and $p$ is the number of simultaneous processes.

According to the above two constraints, the switching cost and the lack of memory can be eliminated. Therefore, to maximize the use of resources, the goal of the division of the unit needs to
meet the above two constraints to start the concurrent process to reduce the time as much as possible. Assume that the optimal number of units in each node is $C_o$, the number of units corresponding to $N_u$, then the optimal process number $P_o$ and the number of threads allocated by each process can be given as

$$\begin{align*}
N_u &= P_o \\
P_o &= \max\{p | \text{pm}_1 \leq M - \sigma, C\%p = 0\} \\
T_o &= C_o \\
C_o &= \frac{T_o}{P_o}
\end{align*}$$

(13)

According to the above analysis, it can be seen that the unit partition strategy first selects the corresponding resource for the user according to the parameters of the task submitted by the user. Here the use of CPU as a resource for the number of units and then the method of scene geometry selection based on key frames and key frames from the pre-rendered are used to estimate resource usage according to the resource of that unit optimal kernel number, and each CPU is assigned a number of processes, and finally, assigned tasks according to a certain strategy in the WSNs.

After submitting the task, the schedule that every node finishes all tasks is:

$$T_1 : \{t_1, t_2, \ldots, t_{r-1}\},$$
$$T_2 : \{t_r, t_{r+1}, \ldots, t_{2r-1}\},$$
$$\vdots$$
$$T_n : \{t_{nr}, t_{nr+1}, \ldots, t_{m}\}$$

(14)

Giving the example of the node $i$, the paper describes the energy consumption of the single node. Assuming that $T_j$ is the whole time of the node $i$, it is as follows:

$$T_i = \sum_{j=ir}^{(i+1)r-1} t_j$$

(15)

Therefore, the finishing time $T_{\text{max}}$ of the whole nodes are the latest nodes. The accurate calculating process is as follows:

$$T_{\text{max}} = \max(T_1, T_2, \ldots, T_n)$$

(16)

In order to illustrate the energy consumption $E_i$ of the node, which is number $i$, Equation (17) illustrates the process of the calculation.

$$E_i = W_i + I_i$$

(17)

In the Equation (17), $W_i$ is the whole energy consumption of the node $i$ which finishes the task list $L: \{l_{ir}, l_{ir+1}, \ldots, l_{(i+1)r-1}\}$. This part of energy consumption is decided by the power $P_i$ of the node $i$ and the running time of the task. $I_i$ shows the energy consumption of the node $i$ in this scheduling, which is in the free situation; this part of the energy consumption is decided by the free power $p_i$ of the node $i$ and the free waiting time. This paper assumes that if and only if the nodes finish the whole task in the queue, they can be in the situation which is free. The Equations (18) and (19) describe two kinds of counting process of the energy consumption which are above.

$$E_i = W_i + I_i$$

(18)

$$I_i = p_i(T_{\text{max}} - T_i)$$

(19)

Then, if we put the Equations (8) and (9) into (10), then we can conclude the energy consumption of the number $i$ node:
According to the energy consumption model of the node \(i\), in the nodes list \((R_1, R_2, \ldots, R_n)\), the Equation (21) describes the counting process of the whole nodes’ energy consumption.

\[
E = \sum_{j=1}^{m} P_it_j + \sum_{i=1}^{n} p_i(T_{\text{max}} - T_i)
\]  

(21)

In the WSNs, the whole energy consumption includes the energy consumption of all tasks and time of node and even the free waiting energy consumption. If Equation (15) is put into Equation (20), eventually we can get the energy consumption computing model.

The accurate estimation of resource occupancy is the premise of the efficient implementation of the multi-process mode, and how to divide the granularity of computing resources based on the feedback resource information is the key to improve the performance of multi-process [49,50]. The traditional way of dividing the resources of the unit is to improve the performance of the node; dividing it into coarse granularity and multi-thread is not obvious, therefore, this paper puts forward the concept of “unit” as the resource partition unit.

Multi-frame sequences are usually assigned to this set for multi thread, and the unit contains the number of cores that show how many threads are rendered. In this paper, we assume that a kernel corresponds to a thread, and each sub job (4 frames) is rendered in the order of the number of frames. Therefore, for nodes with multi-core processors, according to the resource information feedback, the number of nuclear units is the optimal unit allocation by starting a process to render sequence frame multi-thread allocation and resource utilization and by maximizing the increased efficiency.

In order to eliminate the influence of the accumulation of inter frame differences on the load balance among the nodes, the allocation strategy uses the correlation between the frame and the frame to allocate the frame to the unit. In the traditional way, the continuous frames are assigned to the nodes sequentially, and the correlation between frames can only guarantee the difference between adjacent frames. However, if both ends of the sequence, frame node 1 and node \(R\), are assigned to the total operation in the sequence frame, this will bring a larger load imbalance; especially in the abrupt shot explosion, some background conversion, the unbalanced phenomenon is more serious. Therefore, in the unit distribution frame, according to the frame number, staggered modes are allocated for each node frame number of successive frames will be assigned to the unit adjacent to the purpose is to use inter frame correlation to minimize the load of each unit. Therefore, the unit \(r\) is assigned to the frame sequence set \(S_r\) can be given as

\[
S_r = \{f | f = Rz + r, z \in N, 1 \leq f \leq F\}
\]  

(22)

The frame allocation strategy based on unit belongs to a static strategy, while dynamic strategy needs to start the new process and reload the additional cost of the scene file. The implementation of this strategy is relatively simple; there is no preprocessing or task migration overhead.

### 3.3. Data Job Scheduling

The user submits 5 jobs to the system. According to the job scheduling strategy proposed in this paper, the job scheduling process is shown in Figure 1.

Figure 1a shows the initial state of the scheduling, the current system has idle resources, the user submitted to the 5 jobs, \(J_1-J_5\), waiting for job management system scheduling. \(J_1, J_3,\) and \(J_5\) have the highest priority, priority scheduling; the current system has 15 idle states of CPU, to meet the needs of the resources of these three operations. As a result, \(T_0, J_1, J_3,\) and \(J_5\) are assigned resources to perform. Since the current system does not have an idle resource, it cannot meet the resource requirements of job \(J_2\) and \(J_4\), so job \(J_2\) and \(J_4\) remain in the waiting queue.
First of all, it is based on the key frame before rendering the granularity of the resources.

Secondly, the correlation based on frame will render the frame sequence assigned to the unit.

Finally, combined with intra partition strategy, it will not be divisible by the number of the remaining frames unit assigned to the unit.
Sensors 2019, 19, 5083

Figure 2. The process of FCFS and FirstFit job scheduling (a) $t = 100\text{s}$; (b) $t = 150\text{s}$.

Aiming at the problems such as low data similarity and poor data tolerance, this paper proposes a hierarchical job scheduling strategy (HDJS). The implementation of the proposed HDJS is as follows (Algorithm 1):

**Algorithm 1** Hierarchical Job Scheduling Strategy

**Input:** Wait for scheduling job queue  
**Output:** Job scheduling sequence

1. Set flag $= 0$;
2. Sort jobs in $S_{jobs}$ by priority from high to low;
3. for each $J_i$ in $S_{jobs}$ do
4. if $\text{Num}_{J_i} <= \text{freeNum}$ then
5. Allocate the free CPUs to $J_i$;
6. Update the state of CPUs and $R_{jobs}$;
7. if flag $== 1$ then
8. Set $J_i$ as fill-in job and corresponding CPUs as reservation CPUs;
9. Update order $N$;
10. endif
11. break
12. else
13. if $\text{Num}_{J_i} > \text{freeN} \&\& \text{Num}_{J_i} <= (\text{freeN} + \text{orderN})$ then
14. Select jobs paused in $I_{jobs}$ and put them to $b_{jobs}$;
15. Select CPUs corresponding to jobs in $b_{jobs}$ and put them to $b_{Cs}$;
16. Update $b_{CsN}$;
17. if $b_{Cs} ! = \text{NULL}$ then
18. Change the state of CPUs to free and CPUs as no reservation CPUs;
19. Update free$N$, order$N$;
20. endif
21. endif
22. endif
23. if $\text{Num}_{J_i} <= \text{freeN}$ then
24. Allocate the free CPUs to $J_i$ and update the state of CPUs;
25. Update busy$N$;
26. if flag $== 1$ then
27. Set $J_i$ as fill-in job and corresponding CPUs as reservation CPUs;
28. Update order$N$;
29. endif
30. endif
31. flag $= 1$;
32. endfor
The Algorithm 1 is used to judge whether the current job initialization is a sign of queue operations (flag) algorithm simulation having completed the process of scheduling a period in the WSNs. If the current idle state of the CPU resource is able to meet the resource requirements of the job $J_i$, the idle state CPU is assigned to the job $J_i$, and then the resource state and the busy state CPU number are updated. If the next one and the reservation status of homework then set the $J_i$ queue operations, and the corresponding CPU is set to CPU resources reservation number at the same time, update the appointment booking status CPU. The time complexity of the algorithm is $O(n \log n)$, and the time is mainly consumed in the job priority ranking of the waiting list. In this paper, we use a stable merge sort, and the space complexity is $O(n)$.

If frames cannot be divisible by the number of nodes, there are some nodes in idle state when there is a residual frame; resources cannot be fully utilized, and finishing time will be a prolonged operation. Fine grained intra frame distribution is helpful to balance the load of each node, but the efficiency is not high. As a result, an intra-frame distribution strategy is used to render a smaller residual frame, to avoid the idle resources, to further balance the load between nodes, and to reduce the completion time of the job.

By assigning the fine-grained task to the unit, the number of blocks per frame $N_{tile}$ can be given as:

$$N_{tile} = \left\lfloor \frac{R}{((F - R) \mod R)} \right\rfloor$$

(23)

4. Experimental Results and Analysis

In order to evaluate the proposed scheduling strategy performance in the WSNs, this paper builds WSNs based on B/S architecture using local server. In the simulation of the WSNs, the management node undertakes the management job and resource management functions. The system deployed on the management node is the Tomcat version 6.02; the version of the database is mysql-5.32. In addition, the 200 nodes responsible for the hardware configuration of each node are the same, belonging to the isomorphic WSNs.

First, Systems test the resource utilization, and the priority of each task, and the proportion of resource requirements is different. Figure 3 shows the comparison of resource utilization ratio for different strategy. It is clearly evident from Figure 3 that our proposed HDJS has better performance and higher resource utilization. When the number of jobs is 100, the resource utilization ratio of HDJS is increased by an average of 8.8% compared with FCFS, and the best case is increased by 10.28%. When the number of jobs is 200, the resource utilization ratio of HDJS is increased by an average of 8.51% compared with FCFS, and the best case is increased by 10.02%. With the increase of job resources and high priority, the HDJS can obtain higher resource utilization ratio in the WSNs.
while the low priority and high degree of idle resources work to fill in this part for resources, not only to improve their utilization rate but also to make sure the operations can complete part of the schedule. In this way, each job has the opportunity to run ahead of time not because of low priority and long wait for the scheduling state. Of course, this part of the low priority and early scheduling of the work in the system resources to meet the needs of high priority resources will automatically release the resources, also to meet the principle of fairness. This kind of advance plug in execution can reduce the average turnaround time in the WSNs.

Figure 4 shows the comparison of average completion times for different strategies. As shown in Figure 4, when the number of jobs is 100, the average execution time of HDJS is reduced compared with FCFS average execution time. When the amount of jobs is 200, the average execution time of HDJS is reduced compared with FCFS average execution time. Although FirstFit is based on the principle of matching the system resources, the task can be scheduled as soon as possible so as to avoid the resource congestion caused by the lack of system resources. However, in the process of matching resources, FirstFit will produce the phenomenon of idle resources, which cannot make full use of the resources. HDJS can achieve the operations and does not make the resources idle in idle waiting state, while the low priority and high degree of idle resources work to fill in this part for resources, not only to improve their utilization rate but also to make sure the operations can complete part of the schedule. In this way, each job has the opportunity to run ahead of time not because of low priority and long wait for the scheduling state. Of course, this part of the low priority and early scheduling of the work in the system resources to meet the needs of high priority resources will automatically release the resources, also to meet the principle of fairness. This kind of advance plug in execution can reduce the average turnaround time in the WSNs.
Figure 4 shows the comparison of average completion time for different strategies; (a) \( N = 100 \); (b) \( N = 200 \).

Figure 5 shows the comparison of data delay ration for a different strategy. It is clearly evident from Figure 5 that our proposed HDJS is superior to the other two algorithms in data delay ration. The user experience is not only the number of hours of work but also includes the fairness factor. Usually, the user’s needs are not only related to the resources but also to the fairness of their request. For users, fairness is a very important factor. Users usually, from their own point of view, do not want to submit their own work whose execution has been delayed, which is the key point to render the user needs to deal with WSNs. The total completion time is one of the key indicators to evaluate the performance of the system. Compared with the traditional job scheduling strategies FCFS and FirstFit, HDJS can achieve higher performance and better user experience in WSNs management in the WSNs. The traditional FCFS job scheduling strategy can guarantee absolute fairness, which is strictly based on the order of the user submitted jobs. However, the algorithm does not consider the improvement of system performance. FirstFit scheduling strategy with respect to the FCFS strategy readjusts the scheduling queue in order to achieve the purpose of increasing the throughput of the system, but its consideration in terms of fairness is poor; this is often not acceptable for users. In this paper, job scheduling strategy, according to the job priority scheduling, will break absolute fairness.
only when the job is blocked. With high priority, resource needs work temporarily to wait; when the priority is low, resource needs little homework ahead of time scheduling, thus speeding up the process of scheduling, reduce the resource idle time, and make full use of the current free system resources. For the scheduling jobs in the current system, the dynamic priority policy can improve the throughput of the system, and the reservation mechanism ensures the fairness of job scheduling with high priority and large resource requirements. Therefore, the method of this paper can shorten the total completion time of the work and can provide the overall performance of the system.

Figure 5. Comparison of data delay ration for different strategies; (a) \(N = 100\); (b) \(N = 200\).

Figure 6 shows the sensor-cloud processing time of the different testing cases in the low cyber conditions. From Figure 6a,b, we can conclude that HDJS has the superior property, compared with other two optimizing approaches. In the best situation, the property of the HDJS increased by 13%, compared with HLB, and increased by 7%, compared with JSOS. Figure 6c,d shows the localization execution rate of the different testing cases in WSNs period. From Figure 6, it can be concluded that the HDJS has the highest localization execution rate, compared with other two kinds of approaches. For example, when the data block is 64 KB, the localization execution rate of the HDJS can increase
to 23.7%; however, the localization execution rates of the HLB and JSOS are only 12.28% and 15.51%, respectively. Compared with the HLB, the property of the HDJS and localization execution rate of it increased dramatically. This is because, in the approach of the HLB, the data blocks were put on the working nodes randomly according to the hard disk of the nodes using rate. This strategy does not concern the differences of the working nodes calculating in WSNs, which caused that, after the nodes whose calculating abilities finish the local mission stole the local sensor-cloud tasks of the working nodes which have low computing abilities, which leads to abundant non-localization execution of the task sensor-cloud. The non-localization execution of the sensor-cloud task caused the distant communication of the nodes in WSNs, so that it can prolong the time that the jobs stay in WSNs. The property of the HDJS and the localization execution rate of the sensor-cloud have improved to some extent, compared with JSOS. This is because when the JSOS starts to select the input data nodes, according to the calculating rate of different working nodes, it preserves the proper amount of the data block, and the nodes whose calculating ability is high preserve a bigger number of the sensor-cloud task inputting data blocks, which reduced the number of the data blocks movement. However, in the approach of JSOS, the execution time of the task has a linear relation with the amount of the data that is inputted.

Figure 6. Cont.
According to the analysis in the third chapter, the execution time of the task has no simple linear relationship with the amount of data that is imputed. It also has some relationship with the features of the loads and the features of the working nodes. Therefore, the JSOS approach can also lead to that part of the working nodes to execute the non-localization sensor-cloud task. The HDJS proposed in the paper uses the property forecasting model to test the process abilities of the working nodes, whose forecast accuracy is higher than JSOS. In addition to this, in the period of sensor-cloud, the HDJS can dynamically monitor different working nodes in the local sensor-cloud task list execution process. Because of this, it reduces the waiting cost of the inputting data node-hop cyber communication that was caused by the non-localization sensor-cloud mission in the WSNs.

5. Conclusions

In big data environments and WSNs, data plays an important role in improving system performance and job scheduling. With the strategy proposed in this paper, it may be possible to avoid the operation of hunger and resource fragmentation problems, make full use of the advantages of multi-core and
multi-thread, and improve system resource utilization, considering the idle energy consumption of the system and the energy consumption of the task runtime; according to the natural parallelism of the rendering application frame and the frame, the rendering task energy consumption model is established. According to this model, the task scheduling queue is split into sub-queues, and the sub-sequence is optimized. Task scheduling improves node utilization; avoids waste of idle node energy consumption, thus completing the optimization of energy consumption of the rendering system’s global task; and shortens the execution time and response time. In future work, we will combine simulated annealing algorithm and GPGPU platform with the proposed strategy so as to reduce the system execution time more effectively.

Author Contributions: Conceptualization and methodology: Z.S., C.L., and G.Z.; data curation: Z.S., L.W., Z.L., and Z.M.; writing—original draft preparation: Z.S., C.L., and Z.M.; writing—review and editing: Z.S., Z.L., Z.M., and G.Z.; experiment analyses: Z.S., C.L., L.W., and G.Z.; funding acquisition: Z.S.

Funding: This work was supported in part by the Henan Province Education Department Cultivation Young Key Teachers in University under Grant 2016GGJS-158, in part by the Luoyang Institute of Science and Technology High-Level Research Start Foundation under Grant 2017BZ07, in part by the Henan Province Education Department Natural Science Foundation under Grant 19A520006 and 20A520027, and in part by the Key Science and Technology Program of Henan Province under Grant 192102210116 and 182102210428.

Acknowledgments: We are grateful to the experimental foundation provided by the Key Laboratory of Intelligent IoT.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sun, Z.Y.; Wang, H.H.; Liu, B.L.; Li, C.F.; Pan, X.Y.; Nie, Y.L. CS-FCDA: A compressed sensing-based on fault-tolerant data aggregation in sensor networks. *Sensors* 2018, 18, 3749. [CrossRef] [PubMed]
2. Bittencourt, L.F.; Diaz-Montes, J.; Buyya, R.; Rana, O.F.; Rashar, M. Mobility-aware application scheduling in fog computing. *IEEE Cloud Comput.* 2017, 4, 26–35. [CrossRef]
3. Bitam, S.; Zeadally, S.; Mellouk, A. Fog computing job scheduling optimization based on bees swarm. *Enterp. Inf. Syst.* 2018, 12, 373–397. [CrossRef]
4. Wang, T.; Zhang, G.X.; Liu, A.F.; Alam Bhuiyan, M.Z.; Jin, Q. A Secure IoT Service architecture with an efficient balance dynamics based on cloud and edge computing. *IEEE Internet Things J.* 2019, 6, 4831–4843. [CrossRef]
5. Zhang, G.X.; Wang, T.; Wang, G.J.; Liu, A.F.; Jia, W.J. Detection of hidden data attacks combined fog computing and trust evaluation method in sensor-cloud systems. *Concurrency Comp. Pract. E.* 2018. [CrossRef]
6. Sun, Z.Y.; Liu, J.; Xing, X.F.; Li, C.F.; Pan, X.Y. A dynamic cluster job scheduling optimization algorithm based on data irreversibility in sensor networks. *Int. J. Embedded Syst.* 2019, 11, 551–561. [CrossRef]
7. Li, Q.; Wu, W.G.; Sun, Z.Y.; Wang, L.; Huang, J.H.; Zhou, X.X. A novel hierarchal scheduling strategy for rendering system. In Proceedings of the International Conference on Identification, Information, and Knowledge in the Internet of Things (IIKI 2015), Beijing, China, 22–23 October 2015; pp. 206–209.
8. Liu, X.X.; Qiu, T.; Zhou, X.B.; Wang, T.; Yang, L.; Chang, V. Latency-aware anchor-point deployment for disconnected sensor networks with mobile sinks. *IEEE Trans. Ind. Inf.* 2019. [CrossRef]
9. Liu, X.M.; Guo, Y.; Li, W.; Hua, M.; Ding, E.J. A complete feasible and nodes-grouped scheduling algorithm for wireless rechargeable sensor networks in tunnels. *Sensors* 2018, 18, 3410. [CrossRef]
10. Sun, Z.Y.; Lv, Z.G.; Hou, Y.; Xu, C.; Yan, B. MR-DFM: A multi-path routing algorithm based on data fusion mechanism in sensor networks. *Comput. Sci. Inf. Syst.* 2019, 16, 867–890. [CrossRef]
11. Li, Q.; Wu, W.G.; Zhou, X.X.; Sun, Z.Y.; Huang, J.H. R-FirstFit: A reservation based firstfit priority job scheduling strategy and its application for rendering. In Proceedings of the 17th IEEE International Conference on Computation Science and Engineering (CSE 2014), Chengdu, China, 19–21 December 2014; pp. 1078–1085.
12. Stavrinides, G.L.; Karatzas, H.D. A hybrid approach to scheduling real-time IoT workflows in fog and cloud environments. *Multimed. Tools. Appl.* 2019, 78, 24639–24655. [CrossRef]
13. Biason, A.; Pelli, C.; Zanella, A.; Zorzi, M. Access control for IoT nodes with energy and fidelity constraints. *IEEE Trans. Wirel. Commun.* 2018, 17, 3242–3257. [CrossRef]
14. Wang, T.; Luo, H.; Zheng, X.; Xie, M.D. Crowdsourcing mechanism for trust evaluation in CPCS based on intelligent mobile edge computing. ACM Trans. Intel. Syst. Tec. 2019, 10. [CrossRef]

15. Sun, Z.Y.; Wei, L.L.; Song, B.; Nie, Y.L.; Shao, H.X. Mobile intelligent computing in internet of things: An optimized data gathering method based on compressive sensing. IEEE Access 2019, 7, 66110–66122. [CrossRef]

16. Lyu, X.C.; Ni, W.; Tian, H.; Liu, R.P.; Wang, X.; Giannakis, G.B.; Paulraj, A. Distributed online optimization of fog computing for selfish devices with out-of-data information. IEEE Trans. Wirel. Commun. 2018, 17, 7704–7717. [CrossRef]

17. Wang, T.; Ke, H.X.; Zheng, X.; Wang, K.; Sangaiah, A.K.; Liu, A.F. Big data cleaning based on mobile edge computing in industrial sensor-cloud. IEEE Trans. Ind. Inf. 2019. [CrossRef]

18. Sthapit, S.; Thompson, J.; Robertson, N.M.; Hopgood, J.R. Computational load balancing on the edge in absence of cloud and fog. IEEE Trans. Mob. Comput. 2019, 18, 1499–1512. [CrossRef]

19. Wang, T.; Zhang, G.X.; Alam Bhuiyan, M.Z.; Liu, A.F.; Jia, W.J.; Xie, M.D. A novel trust mechanism based on fog computing in sensor-cloud. Future Gener. Comput. Syst. 2018. [CrossRef]

20. Zeng, D.Z.; Gu, L.; Guo, S.; Cheng, Z.X.; Yu, S. Joint optimization of task scheduling and image placement in fog computing supported software-defined embedded systems. IEEE Trans. Comput. 2016, 65, 3702–3712. [CrossRef]

21. Wang, T.; Zhou, J.Y.; Liu, A.F.; Alam Bhuiyan, M.Z.; Wang, G.J.; Jia, W.J. Fog-based computing and storage offloading for data synchronization in IoT. IEEE Internet Things J. 2019, 6, 4272–4282. [CrossRef]

22. Wang, X.L.; Veeravalli, B.; Rana, O.F. An optimal task-scheduling strategy for large-scale astronomical workloads using in-transit computation model. Int. J. Comput. Int. Sys. 2018, 11, 600–607. [CrossRef]

23. Lyu, X.C.; Ni, W.; Tian, H.; Liu, R.P.; Wang, X.; Giannakis, G.B.; Paulraj, A. Distributed online optimization of fog computing for selfish devices with out-of-data information. IEEE Trans. Wirel. Commun. 2018, 17, 7704–7717. [CrossRef]

24. Wang, T.; Zeng, J.D.; Lai, Y.X.; Cai, Y.Q.; Tian, H.; Chen, Y.H.; Wang, B.W. Data collection from WSNs to the cloud based on mobile fog elements. Future Gener. Comput. Syst. 2017. [CrossRef]

25. Liu, X.X.; Zhang, P.Y. Data drainage: A novel load balancing strategy for wireless sensor networks. IEEE Commun. Lett. 2018, 22, 125–128. [CrossRef]

26. He, S.; Dong, M.X.; Ota, K.; Wu, J.; Li, J.H.; Li, G.L. Software-defined efficient service reconstruction in fog using content awareness and weighted graph. In Proceedings of the 2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017; pp. 1–6.

27. Liu, Z.; Zhang, J.W.; Li, Y.N.; Bai, L.; Ji, Y.F. Joint jobs scheduling and lightpath provisioning in fog computing micro datacenter networks. J. Opt. Commun. Netw. 2018, 10, B152–B163. [CrossRef]

28. Wang, T.; Qu, L.; Sangaiah, A.K.; Xu, G.Q.; Liu, A.F. Energy-efficient and Trustworthy Data Collection Protocol Based on Mobile Fog Computing in Internet of Things. IEEE Trans. Ind. Informat. 2019. [CrossRef]

29. Zhao, G.Z.; Gao, X.; Zheng, W.D.; Lv, Z.G. A novel optimization strategy for job scheduling based on double hierarchy. J. Eng. Sci. Technol. Rev. 2017, 10, 61–67. [CrossRef]

30. Liu, X.X.; Zhang, P.Y. Data drainage: A novel load balancing strategy for wireless sensor networks. IEEE Trans. Ind. Inf. 2019. [CrossRef]

31. Liu, X.X. Node deployment based on extra path creation for wireless sensor networks on mountain roads. IEEE Commun. Lett. 2017, 21, 2376–2379. [CrossRef]

32. Wu, Y.K.; Huang, N.Y.; Wang, Y.; Alam Bhuiyan, M.Z.; Wang, T. An incentive-based protection and recovery strategy for secure big data in social networks. Inf. Sci. 2020, 508, 79–91.

33. Liu, X.X.; Wang, T.; Jia, W.J.; Liu, A.F.; Chi, K.K. Quick convex hull-based rendezvous planning for delay-harsh mobile data gathering in disjoint sensor networks. IEEE Trans. Syst. Man Cybern. B Cybern. 2019. [CrossRef]

34. Wu, Y.K.; Huang, N.Y.; Wang, Y.; Alam Bhuiyan, M.Z.; Wang, T. An incentive-based protection and recovery strategy for secure big data in social networks. Inf. Sci. 2020, 508, 79–91.

35. Liu, X.X. Node deployment based on extra path creation for wireless sensor networks on mountain roads. IEEE Commun. Lett. 2017, 21, 2376–2379. [CrossRef]

36. Sun, Z.Y.; Ji, X.H. HDAC: High-dimensional data aggregation control algorithm for big data in wireless sensor networks. Int. J. Inf. Technol. Web. Eng. 2017, 12, 72–86. [CrossRef]

37. Wan, J.F.; Chen, B.T.; Wang, S.Y.; Xia, M.; Li, D.; Liu, C.L. Fog computing for energy aware load balancing and scheduling in smart factory. IEEE Trans. Ind. Inf. 2018, 14, 4548–4556. [CrossRef]
38. Sun, Z.Y.; Li, L.X.; Xing, X.F.; Lv, Z.G.; Xiong, N.N. A novel nodes deployment assignment scheme with data association attributed in wireless sensor networks. J. Internet Technol. 2019, 20, 509–520.
39. Li, Z.X.; Su, D.D.; Zhu, H.J.; Li, W.; Zhang, F.; Li, R.R. A fast synthetic aperture radar raw data simulation using cloud computing. Sensors 2017, 17, 113. [CrossRef]
40. Liu, X.X.; Qiu, T.; Wang, T. Load-balanced data dissemination for wireless sensor networks: A nature-Inspired approach. IEEE Internet Things J. 2019. [CrossRef]
41. Shao, Y.L.; Li, C.L.; Fu, Z.; Jia, L.Y.; Luo, Y.L. Cost-effective replication management and scheduling in edge computing. J. Netw. Comput. Appl. 2019, 129, 46–61. [CrossRef]
42. Sun, Z.Y.; Tao, R.; Xiong, N.X.; Pan, X.Y. CS-PLM: Compressive sensing data gathering algorithm based on packet loss matching in sensor networks. Wirel. Commun. Mob. Comput. 2018. [CrossRef]
43. Wang, T.; Wang, X.; Zhao, Z.M.; He, Z.X.; Xia, T.S. Measurement data classification optimization based on a novel evolutionary kernel clustering algorithm for multi-target tracking. IEEE Sens. J. 2018, 18, 3722–3733. [CrossRef]
44. Hou, X.J.; Zhao, G.Z. Resource Scheduling and load balancing fusion algorithm with deep learning based on cloud computing. Int. J. Inf. Technol. Web. Eng. 2018, 13, 54–72. [CrossRef]
45. Wang, T.; Alam Bhuiyan, M.D.Z.; Wang, G.J.; Rahman, M.A.; Wu, J.; Cao, J.N. Big data reduction for smart city’s critical infrastructural health monitoring. IEEE Commun. Mag. 2018, 56, 128–133. [CrossRef]
46. Sun, Z.Y.; Zhang, Y.S.; Nie, Y.L.; Wei, W.; Lloret, J.; Song, H.B. CASMOC: A novel complex alliance strategy with multi-objective optimization of coverage in wireless sensor networks. Wirel. Netw. 2017, 23, 1201–1222. [CrossRef]
47. Wang, T.; Zhao, D.; Cai, S.B.; Jia, W.J.; Liu, A.F. Bidirectional prediction based underwater data collection protocol for end-edge-cloud orchestrated system. IEEE Tran. Ind. Inform. 2019. [CrossRef]
48. Sun, Z.Y.; Zhao, G.Z.; Xing, X.F. ENCP: A new energy-efficient nonlinear coverage control protocol in mobile sensor networks. EURASIP J. Wirel. Commun. Netw. 2018, 1. [CrossRef]
49. Wang, Q.Y.; Guo, S.T.; Liu, J.D.; Yang, Y.Y. Energy-efficient computation offloading and resource allocation for delay-sensitive mobile edge computing. Sustainable Comput. Inf. Sys. 2019, 21, 154–164. [CrossRef]
50. Wu, Y.K.; Huang, H.Y.; Wu, Q.; Liu, A.F.; Wang, T. A risk defense method based on microscopic state prediction with partial information observations in social networks. J. Parallel Distrib. Comput. 2019, 131, 189–199. [CrossRef]

© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).