Analysis of an Automatic Early Warning System Based on Fog Architecture

Gang Shi¹ and Yuechen Yang²*
¹,²College of Computer and Software, Chengdu Neusoft University

*Corresponding author email: YangYuechen@nsu.edu.cn

Abstract. In the process of analysing and processing terminal sensor information, a large number of terminal sensors are needed to collect front-end information. These front-end data collection, analysis and processing require high real-time, and need the support of location aware mobile computing services. Traditional cloud computing architecture is not the best choice for service scenarios with high real-time requirements. The fog computing architecture is to extend cloud computing services to the edge of the sensor network, coupled with appropriate fitness algorithms, can effectively improve the information analysis and early warning response speed of the geological disaster information early warning system.

Keywords: Automatic early warning; Fog computing; Genetic algorithm; Cuckoo search algorithm.

1. Introduction

In the large scale automatic early warning system, there are a huge number of front-end sensor nodes, and various types of sensors constitute an early warning information collection network. The information collection efficiency of various sensors, the completion of information collection coverage, and the speed of information analysis and processing directly affect the accuracy and real-time performance of the automatic early warning monitoring system.

Fog computing is a distributed computing infrastructure oriented to the Internet of Things, which can extend computing power and data analysis applications to the “edge” of the network. It enables customers to analyse and manage data locally to gain instant insights through connections.

Cuckoo Search Algorithm is an emerging heuristic algorithm proposed in 2009 by Professor Yang Xinshe and S. Daib of Cambridge University. This algorithm solves the optimization problem by simulating the parasitic brooding behaviours of cuckoos. Since the calculation can consider the contradiction between algorithm complexity and algorithm solving speed, this paper combines the fog algorithm with the cuckoo algorithm. On the one hand, it takes advantage of the fast front-end data processing speed of fog computing, and on the other hand, it combines the advantages of the simple flow of the cuckoo algorithm to facilitate accurate selection of high-coverage front-end computing nodes. Thereby, the monitoring accuracy of the geological disaster early warning system can be further improved, and the real-time response time of early warning can be reduced.
2. Sensor Network Monitoring Coverage Model

2.1. Characteristics of the Node Layout of Harm Monitoring Sensor Networks
In the geological hazard early warning system, the number of front-end sensors is very large, and there are many types of sensors, which are used to collect different geological hazard information. Due to the different types and functions of sensors, coupled with the application of integrated sensor terminals, there is a problem of a large number of redundant nodes in the deployment of sensors. It will not only cause data transmission conflicts, affect the stable operation of the sensor network, but also cause a lot of waste of sensor resources. Therefore, in the early stage of sensor layout, it is necessary to deploy sensor nodes as scientifically as possible under the premise of ensuring the coverage of the sensor monitoring range. At the same time, by moving the sensor node monitoring information processing method forward, the cloud server or central server sends it to the fog computing processing unit near the monitoring terminal for front-end processing, which can increase the sensor information processing speed of each specific monitoring area and improve the accuracy of early warning rate.

2.2. Monitoring Coverage Model
First assume that the monitoring radius (sensing radius) \( r \) of the same type of monitoring sensor is the same. For a specific type of sensor, the set of sensors in a certain monitoring area is \( G = (g_1, g_2, g_3, ..., g_n) \). In this set, the coordinates of each sensor placement point in the monitoring area correspond to the two-dimensional position coordinates \((x_i, y_i)\). Let \( A \) be any point in the two-dimensional plane of the monitoring area, and its corresponding coordinates are \((x_a, y_a)\). According to the above setting, the linear distance between point \( A \) and any point \( g_i \) in the sensor set is

\[
d = \sqrt{(x_i - x_a)^2 + (y_i - y_a)^2}
\]

(1)

For point \( A \), the probability of being monitored by the sensor network is

\[
P = \begin{cases} 
1 & (d_{\text{min}} \leq r) \\
0 & (d_{\text{min}} > r) 
\end{cases}
\]

(2)

Considering that point \( A \) may be sensed by multiple sensor nodes, the common perception probability of all nodes to point \( A \) can be further calculated as

\[
P_{\text{all}}(G_{\text{all}}, p) = 1 - \prod_{i=n} (1 - P(p, g_i))
\]

(3)

Subsequently, the monitoring area is gridded and simplified into \( M \times N \) pixels, and the entire coverage of the monitoring sensor in the monitoring area is the ratio of all pixels that are sensed to the total pixels.

\[
P_{\text{cover}} = \sum_{x=1}^{M} \sum_{y=1}^{N} \frac{P_{\text{all}}(G_{\text{all}}, p)}{M \times N}
\]

(4)

3. Sensor Network Node Monitoring Program Optimization

3.1. Fog Computing
Fog computing was proposed in 2011. Compared with cloud computing, fog computing does not have strong computing power, but it can still be regarded as a branch and extension of cloud computing. Fog computer is the main application. It mainly uses the equipment in the edge network, and the data transmission has very low delay. Fog computing is a large-scale sensor network with a large number of network nodes. Fog computing has good mobility. Mobile phones and other mobile devices can communicate directly with each other. The signal does not have to go around the cloud or even the base station to support high mobility.
Fog computing is not a powerful server, but is composed of computers with weaker and more dispersed functions. Fog computing is between cloud computing and personal computing, and is a para virtualized service computing architecture model. Emphasizing quantity, no matter how weak a single computing node is, it must play a role. Compared with cloud computing, the architecture adopted by fog computing is more distributed and closer to the edge of the network. Fog computing concentrates data, data processing, and applications in devices at the edge of the network, instead of storing almost all of them in the cloud like cloud computing, data storage and processing rely more on local devices rather than servers. Fog computing is a new generation of distributed computing, in line with the "decentralized" characteristics of the Internet. Schematic diagram of the architecture of fog computing as Fig.1.

Figure 1. Schematic diagram of distributed computing process.

3.2. Cuckoo Search Algorithm
According to the long-term observation and research of entomologists, some cuckoos raise their young by parasitism. Instead of building nests, they lay their own eggs in other bird's nests, and other birds will hatch and brood. However, if these foreign bird eggs are found by the host, the host will abandon the eggs or build a new nest.

Popular understanding is that the process by which cuckoo eggs can successfully hatch in other nests is the process of finding the best. The Cuckoo Search Algorithm originated from the simulation of cuckoo breeding behaviours. In order to simplify the breeding habits of cuckoos in nature, Prof. Yang et al. assumed the cuckoo's spawning behaviours as three ideal states.

The cuckoo lays only one egg at a time and randomly chooses the nest position to hatch it. In a randomly selected set of bird nests, the best bird nest will be retained for the next generation. The number of selectable parasitic nests is fixed, and the probability that the owner of the parasitic nest finds foreign bird eggs is $p_a$, where $0 \leq p_a \leq 1$.

Based on these three ideal states, Yang et al. used equation (5) to update the position of the next-generation bird’s nest:

$$X_i^{t+1} = X_i^t + \alpha \otimes Levy(\lambda)$$

In the formula: $X_i^{t+1}$ represents the position of the first bird's nest in the t generation; $\otimes$ represents the point-to-point multiplication; $\alpha$ represents the step-length control amount, used to control the step-length size, usually set $\alpha = 1$. $Levy(\lambda)$ is Levy's random search path, which belongs to random walking. Using the Levy flight mechanism, the walking step meets a heavy-tailed stable distribution, and the random step length is the levy distribution:

$$Levy : \mu = i^{-\lambda}, 1 \leq \lambda \leq 3$$

The basic cuckoo search algorithm first updates the next-generation bird’s nest position according to equation (6), and calculates the fitness value of the objective function. If the value is better than the previous generation’s objective function value, then update the bird’s nest position, otherwise keep the
original position constant. After the location update, the randomly generated value $R$ that is uniformly distributed from 0 to 1 is compared with the probability $P_a$ that the bird’s nest owner finds the foreign bird eggs. If $R > P_a$, the will be changed randomly, and vice versa. Finally, keep a group of bird nest positions with good test values and record it as $X^{t+1}$. Judge whether the algorithm meets the set maximum number of iterations: if it is satisfied, the iterative optimization is ended, and output the global optimal value $f_{\text{min}}$. Otherwise, the iteration optimization is continued. The algorithm has the balance of global exploration and local development performance and the diversity of the population.

3.3. Design of Cuckoo Algorithm Based on Fog Computing

In the meta-heuristic algorithm, the design of the Bird’s Nest fitness function will directly affect the quality of the results obtained. When applied to the monitoring and early warning system, it will directly affect the speed and accuracy of monitoring and early warning. The time complexity of the fitness function determines the final computer speed of the function. Without changing the fitness calculation method (algorithm), if you want to increase the signal collection and analysis speed of each abnormal monitoring point in the geological disaster monitoring area, it is a feasible way to adopt a fog computing architecture based on the distributed principle.

The placement of sensor nodes corresponds to the spawning position in the cuckoo algorithm (appropriate bird's nest). With the simple process of the cuckoo algorithm, few parameters and easy implementation, it can be easily transplanted to the fog computing platform of the distributed architecture. In order to realize the cuckoo algorithm based on fog computing. The cuckoo algorithm based on the fog computing platform designed in this paper does not change the basic steps of the original cuckoo algorithm. The main design is to put the fitness algorithm before running on the fog computing platform. For the entire sensor early warning information processing and other parts of the program for issuing early warning signals, it is not necessary to place them on the distributed fog platform to run. This design can minimize the algorithm changes involved in using the fog computing platform, and minimize the impact of algorithm changes on the final result.

Combining the above analysis and discussion based on the distributed fog computing platform and cuckoo algorithm, based on the basic flow of the cuckoo algorithm, combined with the distributed fog computing platform, the design of the flow of the cuckoo algorithm based on the distributed fog computing platform is given as follows:

a) Initialize the population and randomly generate $n$ bird nest positions (There are unusual monitored points);
b) Place the $n$ bird's nest positions generated in step a) on the distributed platform, and calculate the fitness of these bird's nests in parallel;
c) Find the best bird's nest location and keep it for the next generation;
d) Use formulas on the distributed platform to update the position of the bird’s nest, the fitness of the computer after the update, and compare it with the fitness of the original bird's nest. If the fitness is better than the original bird’s nest position, replace the original bird’s nest, otherwise directly discard it;
e) generates a random number greater than 0 and less than 1 on the distributed platform, which is used as the probability that the original owner of the bird's nest finds a foreign bird egg. It is compared with $P_a$. If it is less than $P_a$, the original bird's nest is discarded and a new one is randomly generated the position of the bird’s nest is used to recalculate the position adaptability;
f) If the algorithm termination condition is reached, find the optimal position of the bird's nest output (Further warning processing). If the algorithm termination condition is not reached, jump to step c) to continue to execute the algorithm again.

The algorithm flow design is shown in the Figure 2 below:
Calculate the fitness of the bird’s nest

Start

Initial population

Calculate the fitness of the bird’s nest

Distributed Fog Computing

Calculate the fitness of the bird’s nest

Find the best individual

Update The location

Abandon the bird’s nest with a certain probability

Distributed Fog Computing

Abandon the bird’s nest with a certain probability

End condition reached

Yes

Stop

Output the best nest position

No

Update The location

Abandon the bird’s nest with a certain probability

Figure 2. Schematic diagram of the CSA flow based on distributed fog computing.

4. Algorithm Test

Finally, by comparing the solution of the coverage optimization problem of monitoring sensor nodes in different two-dimensional areas, compare the optimization effect and calculation speed of the Cuckoo Algorithm (PFCSA) based on the fog computing platform and the traditional Cuckoo Algorithm (CSA) under different conditions. Different conditions mainly refer to different fitness functions (different time complexity).

Basic conditions of the experiment: Assuming that all monitoring sensors are of the same type and their sensing radius is 10m, they are tested in 4 different monitoring areas of 50m × 50m, 80m × 80m, 120m × 120m, and 160m × 160m. 30, 80 and 120 monitoring sensors.

Experimental setup: Set the number of bird’s nests to 25, Pa to 0.3, and the number of iterations to 1000. Computer equipment used in the experiment: 1 main server node, 4 distributed computing nodes. The hardware configuration of each node is CPU: Intel Core i5 4590 LG A 1150, 3.3GHz; memory: DDR4 8G × 2; 1G adaptive network card. Through the experimental conditions based on the above-mentioned software and hardware environment, the test results of the PFCSA algorithm and the CSA algorithm are shown in Table 1 as follows.

| Monitoring range | CSA | PCSA |
|------------------|-----|------|
|                  | Accuracy % | Response time(s) | Accuracy % | Response time(s) |
| 50m × 50m        | 82.9      | 9.4      | 84.3      | 7.2       |
| 80m × 80m        | 81.7      | 14.3     | 83.1      | 11.6      |
| 120m × 120m      | 79.5      | 44.1     | 80.3      | 26.5      |
| 160m × 160m      | 77.3      | 89.5     | 78.5      | 35.3      |

It can be seen from the experimental results that the CSA and PFCSA algorithms have little difference in the effect of monitoring sensor network coverage optimization, because the core basic processes of
the CSA and PFCSA algorithms are completely the same. In the comparison of the time complexity of the algorithm, the advantage of PFCSA is very obvious. PFCSA is shorter than the traditional CSA algorithm in terms of optimization problems of different scales. And with the expansion of the problem scale, the time-consuming growth rate of the CSA algorithm is much higher than that of the PFCSA algorithm. As can be seen from Tab 1, the algorithmic advantage of PFCSA is not obvious when the scale is 80m\times80m and below. However, after the test area is larger than 100m\times100m, the time used for the CSA algorithm increases several times, while the PFCSA still increases linearly.

Under the current test conditions, when the test area is larger than 100m\times100m, the host load of a single node is obviously saturated, and the load of the distributed host nodes of PFCSA has a lot of room for improvement. And with the increase of computational complexity, the optimization effect of PFCSA is relatively stable, and in the course of four tests, it is better than CSA algorithm. Therefore, it can be concluded that by combining the PFCSA algorithm based on the distributed fog level calculation and the cuckoo algorithm, not only the execution efficiency of the algorithm can be greatly improved, but the calculation results are also stable and optimized.

5. Conclusion
This paper combines the distributed fog computing platform with the traditional cuckoo algorithm, and proposes a PFCSA-based automatic early warning monitoring sensor monitoring and early warning optimization algorithm. Under complex conditions, this algorithm can increase the speed of the fitness algorithm. The test results show that compared with the traditional CSA algorithm, the optimization effect of this algorithm is basically equivalent or slightly better, and its computing speed (real-time warning response) has been greatly improved when the problem complexity reaches a certain scale. At the same time, it should be noted that this algorithm is based on the same type of monitoring sensors and the same monitoring radius. In large-scale engineering automatic monitoring and early warning practice, there are different types of monitoring sensors with different monitoring radius, physical response speed and other parameters. How to optimize sensor monitoring accuracy and real-time response speed in combination with the fog computing platform cuckoo algorithm in a mixed deployment environment of multiple types of sensors requires further in-depth analysis and research.

Acknowledgments
This paper was supported by Project: RFID-based smart storage cabinet, No.: 2018GY03. Dujiangyan Science and Technology Bureau.

References
[1] Pinar Civicioglu, Erkan Besdok. A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms[J]. Artificial Intelligence Review. 2013 (4)
[2] Xin-She Yang, Suash Deb. Engineering optimisation by cuckoo search[J]. Int. J. of Mathematical Modelling and Numerical O. 2010 (4)
[3] Xin-She Yang, Amir Hossein Alavi. C-Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems[J]. Amir Hossein Gandomi, Engineering with Computers. 2013 (1)
[4] Srikanth R., Bikshalu K. Multilevel thresholding image segmentation based on energy curve with harmony Search Algorithm[J]. Ain Shams Engineering Journal, 2021, 12(1)
[5] Mateev and I. Marinova, Distributed Internet of Things System for Wireless Monitoring of Electrical Grids[D]. 20th International Symposium on Electrical Apparatus and Technologies (SIELA), 2018