electronics

MDPI

Article

Inverse Coupled Simulated Annealing for Enhanced OSPF Convergence in IoT Networks

Chengsheng Pan [1,*], Huangjie Lu [1], Huaifeng Shi [1], Yingzhi Wang [1] and Lishang Qin [2]

[1] School of Electronics and Information, Nanjing University of Information Science and Technology, Nanjing 210044, China; 202212180027@nuist.edu.cn (H.L.); 003162@nuist.edu.cn (H.S.); 20201118009@nuist.edu.cn (Y.W.)
[2] School of Software, Nanjing University of Information Science and Technology, Nanjing 210044, China; 202212210039@nuist.edu.cn
* Correspondence: 003150@nuist.edu.cn; Tel.: +86-15940934666

Abstract: The current Internet of Things (IoT) network structure is evolving from small-scale distributed systems to a large-scale hierarchical collaboration between backbone and access networks. In this context, the dynamic changes in backbone node connections and the surge in service demands, coupled with sluggish fault detection speeds, significantly shorten effective service transmission time. To address this issue, this paper proposes an inverse coupled simulated annealing for enhanced OSPF route convergence in IoT networks (OSPF-ICSA). Initially, the link state is derived from the statistical characteristics of Hello packets, while the aggregated characteristics of the link state are employed to characterize the node state, providing data support for the reverse coupled simulated annealing algorithm. Subsequently, the Hello packet is refined, and a mechanism is designed to synchronize OSPF intervals and transmit node states. This ensures that nodes within the same subnet synchronize their sending intervals and fault detection times while sharing their node states. Finally, building upon this foundation, the reverse coupled simulated annealing algorithm is introduced to jointly optimize the Hello packet sending interval and fault detection time. Compared to the traditional AODV protocol, OSPF-ICSA reduces the average fault detection time by over 37.38%, improves the average fault detection accuracy by more than 3.1%, decreases the average routing overhead by over 20%, and increases the average packet delivery rate by over 5.1%.

Keywords: OSPF; routing convergence; Hello messages; fault detection

check for updates

Citation: Pan, C.; Lu, H.; Shi, H.; Wang, Y.; Qin, L. Inverse Coupled Simulated Annealing for Enhanced OSPF Convergence in IoT Networks. *Electronics* **2024**, *13*, 4332. https://doi.org/10.3390/electronics13224332

Academic Editor: Djuradj Budimir

Received: 9 October 2024
Revised: 30 October 2024
Accepted: 1 November 2024
Published: 5 November 2024

1. Introduction

With the rapid development of the Internet of Things (IoT), the interconnectivity of smart devices through the internet has experienced explosive growth, resulting in an unprecedented scale of networks [1–3]. Correspondingly, the IoT network structure has gradually evolved from small-scale distributed systems to a large-scale hierarchical layout characterized by tight collaboration between backbone and access networks, showcasing strong cross-domain communication potential [4–6]. In this context, the dynamic characteristic of backbone node connections and the surge in service demands pose a pressing challenge: how to achieve rapid fault detection of backbone node links and real-time awareness of topological changes, thereby increasing the effective transmission time of services.

In the process of service transmission, the role of routing protocols becomes increasingly critical, especially in the Internet of Things (IoT), where the on-demand routing protocol AODV is widely used [7,8]. However, the path selection of AODV is often relatively singular, which can lead to route contention and result in network congestion issues. Furthermore, the delays associated with AODV's path discovery process conflict with the real-time awareness requirements of backbone network nodes regarding topological changes, thereby limiting the application of AODV within backbone networks. To optimize the issue of routing contention in the Internet of Things, Elappila et al. proposed a congestion- and interference-aware energy efficient routing technique for the IoT. This approach is based on the signal-to-noise ratio and noise ratio of links. By evaluating the survivability factor of the path from the next-hop node to the destination and its current congestion level, it effectively meets the data routing demands in high-load network environments [9]. Guo et al. introduced a routing algorithm based on reinforcement learning, which constructs a reward function using remaining energy and hop count, optimizing routing in wireless sensor networks (WSNs), extending network lifespan, and ensuring a good data flow transmission rate [10]. Younus et al. utilized reinforcement learning to optimize routing in software-defined networking (SDN) wireless sensor networks, proposing a reward function based on energy utilization and QoS requirements, significantly improving data flow transmission efficiency [11]. Although these algorithms have reduced routing contention and improved data transmission efficiency to some extent, they primarily avoid routing contention by optimizing paths. When link failures occur between nodes and fault detection is not timely, even the best routing algorithms struggle to mitigate the impact of faulty links, leading to packet loss and reduced transmission efficiency.

Open Shortest Path First (OSPF) is a proactive routing protocol that possesses a comprehensive network topology view of all nodes within the area [12–14]. In the event of a link failure, it can swiftly disseminate information to all nodes and update routing, ensuring rapid route convergence across all nodes simultaneously. This mechanism is particularly suitable for backbone nodes in the Internet of Things, as it avoids the delays associated with path discovery in AODV. However, its fault detection process is mechanical and slow, making it difficult to quickly identify failures, which severely impacts service transmission in the IoT and hinders its deployment. To accelerate fault detection speed, the literature [15–17] has introduced the BFD protocol. BFD establishes independent sessions between OSPF neighbors to send probe messages at millisecond intervals, allowing for real-time monitoring of link status and rapid detection of link failures. However, this high-frequency probing also results in increased routing overhead. Particularly in cases of link congestion or interference, BFD is prone to false alarms, leading to unnecessary route convergence and further exacerbating routing overhead in the network [18]. To reduce routing overhead, Manousakis et al. developed a tool based on an enhanced simulated annealing algorithm, aimed at automatically optimizing OSPF area partitioning to balance multi-objective performance requirements. This tool effectively reduces routing overhead, convergence time, and latency while improving bandwidth utilization [19]. However, its applicability is limited, primarily targeting network environments with minimal topological changes. Considering the dynamic characteristics of network topology in the IoT, this limitation affects the tool's feasibility in practical deployment. In response to time-varying network topologies, the IETF has introduced the OSPF-MPR [20,21], OSPF-MDR [22], and OSPF-OR [23] protocols to address topological changes in MANETs. These protocols introduce specific mechanisms to meet the demand for routing convergence in MANET topologies; however, their effectiveness in improving convergence speed is not significant. To address this gap, researchers have introduced centrality into the routing convergence process to enhance convergence speed while optimizing routing overhead to some extent. References [24,25] discuss the placement problem of Designated Routers (DRs) and propose optimizing DR layouts using betweenness centrality, closeness centrality, and degree centrality to shorten routing convergence time. References [26,27] adjust the sending frequency of Hello packets based on betweenness centrality, increasing the frequency for nodes with higher intermediary centrality to expedite fault detection and reduce routing overhead. References [28–30] optimize the sending of Hello packets based on load centrality, running the load centrality algorithm directly in distributed routers to reduce computational complexity, significantly improving routing convergence efficiency. Although centrality-based algorithms have achieved success in accelerating fault detection and controlling routing overhead, nodes with lower centrality can still become critical routing nodes. Their failures can delay link fault detection and convergence, leading to data loss. Furthermore, nodes with higher centrality may waste network resources due to the frequent sending of Hello messages.

To address this, this paper proposes an inverse coupled simulated annealing for enhanced OSPF route convergence (OSPF-ICSA), aimed at resolving the issues of time-varying topology in IoT backbone nodes, slow fault detection, and inadequate topological awareness, which result in shorter effective transmission times for services. This method integrates the OSPF protocol into the Internet of Things and improves upon the traditional OSPF protocol. This method first utilizes the statistical characteristics of Hello packets to assess the link state and characterizes the node state through aggregated link state features, dynamically reflecting the topological changes of nodes in the IoT. This process effectively captures the dynamic characteristics of connection relationships, providing real-time network state data support for the inverse coupled annealing algorithm. Secondly, based on the improvement of Hello packets, a mechanism for OSPF interval synchronization and node state transmission is designed to ensure that the sending intervals and fault detection times of all nodes within the same subnet are synchronized, facilitating the effective transmission of node status messages and laying the foundation for the efficient application of the inverse coupled annealing algorithm. Finally, based on this foundation, the inverse coupled annealing algorithm is introduced to collaboratively optimize the sending intervals of Hello packets and fault detection times through two coupled annealing algorithms, dynamically adjusting the sending frequency of Hello packets and fault detection times. In poor link states, this accelerates fault detection so that nodes can more swiftly and flexibly identify and respond to link failures, thus shortening overall routing convergence time. In good link states, it reduces the sending frequency of Hello packets to lower routing overhead.

2. Main Contribution

2.1. Design Overview

OSPF routing convergence refers to the process by which nodes in a network reach a consensus to update routing information following the detection of changes in links or nodes [12–14]. This process is illustrated in Figure 1.

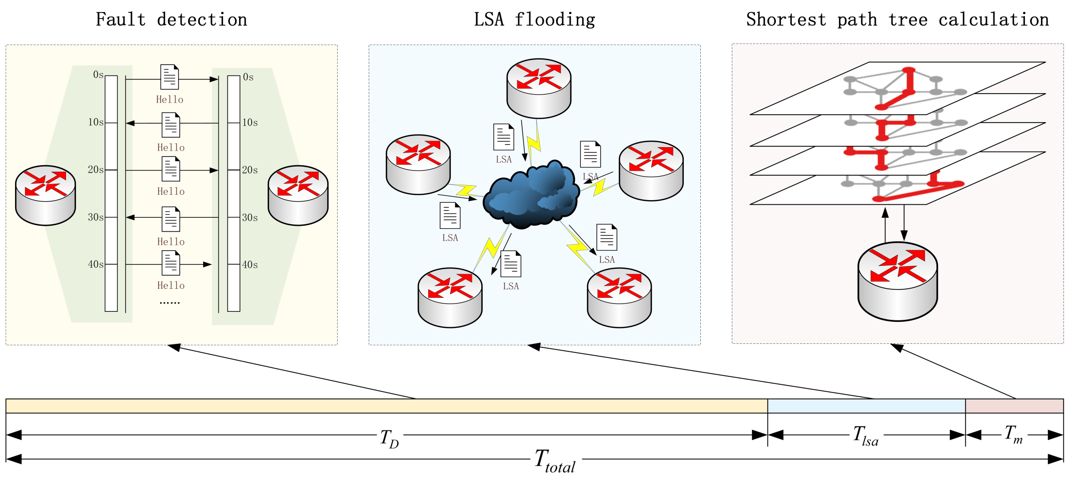


Figure 1. OSPF routing convergence time distribution diagram.

The process begins with fault detection. Node R periodically sends Hello packets via the Hello protocol to monitor the status of neighboring nodes. The sending interval for Hello packets is denoted as T_h, which is controlled by the Hello Interval parameter, typically set to a default value of 10 s. Additionally, the link fault detection time T_d is usually set to four times T_h. If no response to the Hello packets from the neighbors is received within T_d (governed by the Dead Interval parameter), the neighbor is considered unreachable, triggering link fault handling. Assuming that the occurrence of a fault is a random event, the time difference X from the occurrence of the link fault to when R detects the fault is uniformly distributed within the interval (30, 40). The probability density function for this distribution is given by:

$$f(x) = \begin{cases} \frac{1}{40-30} = 0.1 & \text{if } 30 \leq x \leq 40 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

At this point, the expected value $E(X)$ is 35. Therefore, the average fault detection time T_D is 35 s.

When a fault occurs, R generates a new Link State Advertisement (LSA) to reflect the topological change. Upon receiving the new LSA, neighboring nodes store it in their Link State Databases (LSDBs) and subsequently flood the LSA to all neighbors, achieving complete LSA flooding. To optimize LSA flooding, OSPF employs the mechanisms of Designated node (DR) and Backup Designated node (BDR). The DR is responsible for receiving and broadcasting LSAs within the network, which reduces redundant transmissions and minimizes bandwidth consumption. The BDR acts as a backup for the DR, ensuring continuity in LSA flooding. The cooperative operation of the DR and BDR enables OSPF to efficiently flood LSAs while alleviating routing overhead and network load. Consequently, the flooding time of an LSA within a broadcast segment can be expressed as:

$$T_l = \begin{cases} T_c^{DR} + \frac{s}{\beta} & \text{if } R = DR \\ T_c^{BDR/Others} + T_{\mathrm{m}}^{DR} + T_c^{DR} + 2 \cdot \frac{s}{\beta} & \text{if } R = BDR/Others \end{cases} \quad (2)$$

where T_c^{DR} and $T_c^{BDR/Others}$ represent the time taken by the DR and BDR/Others to generate the LSA, respectively, s is the size of the LSA packet in bytes, β is the interface transmission rate, and T_{m}^{DR} is the time taken by the DR to process the LSA. Therefore, the total time for LSA dissemination within the entire segment can be expressed as:

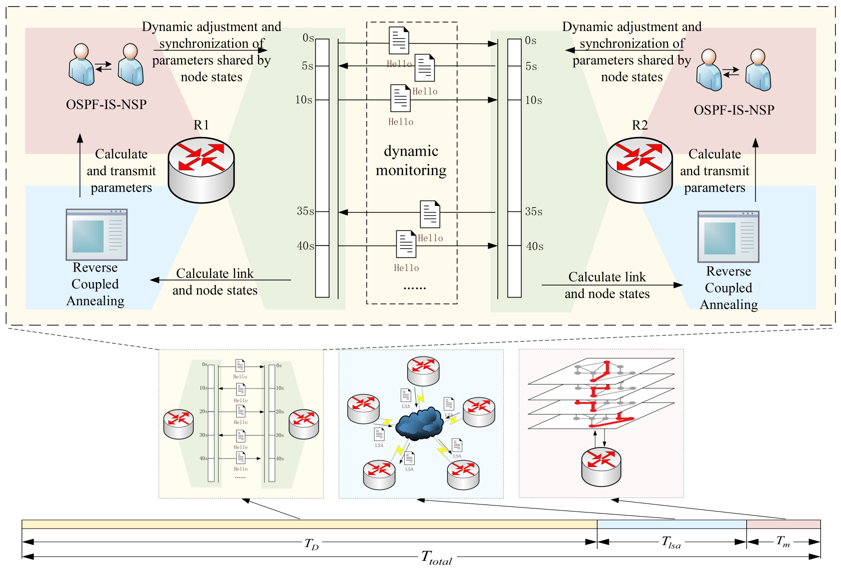
$$\mathrm{T}_{lsa} = \sum_{i=1}^{n} T_l^i \quad (3)$$

where n represents the maximum number of broadcast segments for flooding the LSA to all nodes, and T_l^i is the time required for the i-th broadcast segment that is traversed.

Once the LSA flooding is complete, the node recalculates the Shortest Path Tree (SPT) using Dijkstra's algorithm based on the updated LSDB and updates the routing table. Let the time taken for this process be denoted as T_m. Therefore, the total routing convergence time can be expressed as:

$$\mathrm{T}_{total} = T_{\mathrm{D}} + T_{lsa} + T_m \quad (4)$$

In T_{total}, the fault detection phase is typically the most time-consuming part. This is primarily because confirming a link failure depends on not receiving a response to the Hello packets within the time T_d, which is set to 40 s by default. During this period, the average T_D requires 35 s. In contrast, the LSA flooding process, which involves the generation and flooding of LSAs, generally consumes less time than the fault detection phase, although the time required may increase with network size. Additionally, the routing table calculation phase, based on Dijkstra's algorithm for shortest path computation, also operates within a limited time frame. Therefore, this paper restructures the protocol based on the traditional OSPF protocol, dynamically optimizing the Hello packet transmission frequency and fault detection time, aiming to accelerate fault detection speed, thereby enhancing routing convergence speed and, to some extent, reducing routing overhead, as illustrated in Figure 2.



Dynamic adjustment and synchronization of parameters shared by node states
OSPF-IS-NSP
R1
Calculate and transmit parameters
Reverse Coupled Annealing
Calculate link and node states
0s
5s
10s
35s
40s
Hello
dynamic monitoring
......
R2
T_D
T_{lsa}
T_m
T_{total}

Figure 2. OSPF-ICSA architecture diagram.

The specific improvements are as follows:

Utilize the statistical characteristics of Hello packets to obtain the link status, and characterize the node state based on the aggregated features of the link state, thereby assessing the state of links and nodes in the network and providing data support for the reverse coupled annealing algorithm.

Design an OSPF interval synchronization and node state propagation mechanism (OSPF-IS-NSP) to synchronize the sending intervals of Hello packets and fault detection times of nodes within the same subnet and propagate the node state of the local node.

Propose a reverse coupled annealing algorithm, which consists of two coupled annealing algorithms: the upward optimization annealing algorithm and the downward optimization annealing algorithm. When one annealing algorithm is executed, the temperature of the other gradually rises, and vice versa. Through this mechanism of alternating heating and cooling, and based on the link and node status, the two algorithms collaboratively optimize the Hello packet transmission interval and fault detection time.

2.2. Link and Node State Monitoring

In the OSPF protocol, the instability of links or nodes can trigger excessive LSA flooding and frequent routing updates, leading to routing oscillations. This situation not only results in a significant increase in routing overhead but also places a greater burden on the node's CPU. In severe cases, it may even lead to network storms, resulting in network failures. To address this issue, this section introduces link and node states to monitor the statuses of links and nodes in real time, thereby providing data support for subsequent algorithms.

A network topology can be represented as $G(N, E)$, where N is the set of network node nodes and E is the set of communication links. Assume there exists a link $l(l \in E)$ between nodes $u \in N$ and $v \in N$, which can be denoted as: $l = uv$. In this case, the link state is evaluated based on the absence of Hello packets in link l during the assessment period, as expressed in the following formula:

$$\eta(uv) = \frac{T - t_d}{T} \tag{5}$$

where $\eta(uv)$ is the link state evaluation function, T is the assessment period, and t_d represents the time segment formed by the absence of Hello packets during the assessment period. This is defined as the difference between the expected time of receiving the next Hello packet and the actual last received time, as illustrated in Figure 3.

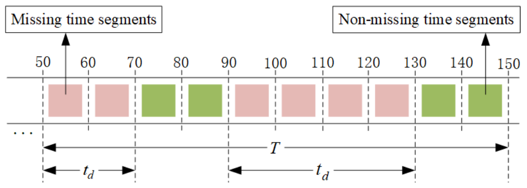


Figure 3. Link state diagram.

Additionally, based on the aggregated average link state of all interfaces on node v, the node state is assessed using the following formula:

$$\gamma(v) = \frac{\sum_{uv \in E} \eta(uv)}{|E|} \tag{6}$$

where $\gamma(v)$ is the node state assessment function, and $|E|$ represents the cardinality of E, which is the number of links associated with node v.

Building on this, this paper introduces weight parameters α and β for a comprehensive assessment of link and node states. The specific evaluation formula is:

$$\varphi(u) = \alpha * \eta(vu) + \beta * \gamma(u) \tag{7}$$

where $\varphi(u)$ is the comprehensive assessment function used to evaluate the overall state of the link between node v and its neighboring node u. The link state is denoted by $\eta(vu)$, indicating the condition of the link between nodes v and u, while $\gamma(u)$ represents the state of node u. The parameters α and β are weight coefficients for link and node states, respectively, which adjust their influence on $\varphi(u)$.

2.3. OSPF Interval Synchronization and Node State Propagation Mechanism

In the OSPF protocol, unifying the configuration of the Hello Interval (T_h) and Dead Interval (T_d) parameters for all nodes within the same segment is essential for ensuring the stable establishment and continuous maintenance of neighbor relationships. Hello packets primarily facilitate neighbor discovery and fault monitoring. Inconsistent parameter configurations can lead to asymmetries in neighbor state perception, adversely affecting the correct establishment of adjacency relations and timely fault detection. Additionally, parameter consistency synchronizes the time window for fault detection, preventing delays or misjudgments due to variations in T_d. This unification also enables the synchronous propagation of topological state information within the segment, minimizing delays in topology updates. Consequently, this section introduces the OSPF interval synchronization mechanism to ensure parameter consistency among all nodes, providing foundational support for the deployment of the reverse coupled annealing algorithm, as illustrated in Figure 4.

The specific steps of the OSPF interval synchronization mechanism are as follows:

- When the Designated Router (DR) calculates the new parameters for T_h and T_d (assumed to be 10 s and 40 s, respectively), the DR appends its T_h and T_d parameters to the Hello message and broadcasts it to all neighbors in the subnet.
- The Backup Designated Router (BDR) and other non-DR routers have their previous T_h and T_d values assumed to be 7 s and 28 s, respectively.
- Once the BDR or other routers receive the Hello message from the DR containing T_h and T_d, they will immediately update their T_h and T_d to 10 s and 40 s, respectively. At the same time, the nodes will adjust their Hello message sending timers and failure detection timers accordingly. As shown in Figure 4, due to the timer adjustments, the sending of Hello messages is delayed by 3 s compared to before. When the next Hello message sending cycle begins, the node will send a Hello message containing the new parameters.

- When the DR receives the Hello messages sent by the BDR or other routers, it can detect that these nodes' T_h and T_d have been updated to 10 s and 40 s, thereby achieving global synchronization of T_h and T_d across the entire subnet.

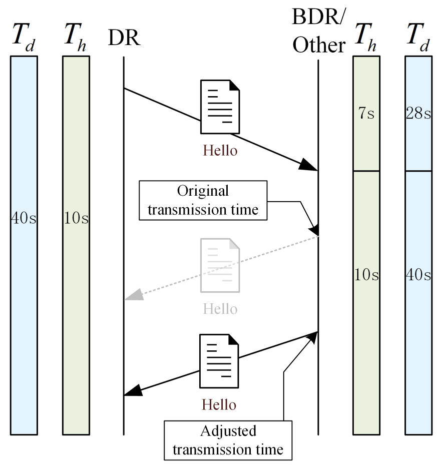


Figure 4. OSPF interval synchronization mechanism operating process diagram.

Additionally, this section designs a node state propagation mechanism, which transmits the node state by mapping it to the Priority field in the Hello message. This approach not only optimizes the DR election process based on the Priority field—allowing nodes with higher node state to be prioritized as DR or BDR—but also ensures that nodes with superior states take on critical roles within the network, enhancing stability and routing efficiency. Furthermore, since the OSPF protocol lacks a native mechanism for propagating node state, this mechanism enables nodes to share node state information in real-time among neighbors.

2.4. Reverse Coupled Annealing Algorithm Design

To address the dynamic characteristic of link relationships and the frequent link failures in IoT networks, it is essential to dynamically adjust the sending frequency of Hello messages and the fault detection time based on the characteristics of link and node states in real time. The annealing algorithm can explore and approximate the optimal Hello interval and fault detection time through multiple iterations based on the network environment, allowing for a broader search space in the initial stages to prevent the algorithm from getting trapped in local optima [31–34]. As the algorithm's temperature gradually decreases, the search range converges, accelerating convergence while maintaining solution diversity. The algorithm begins by randomly generating an initial solution x_0 and searches for solutions within its neighborhood. In each iteration, based on the Metropolis criterion, the algorithm decides whether to accept a worse solution according to the acceptance probability formula:

$$P(\Delta E) = \begin{cases} 1 & \text{if } \Delta E < 0 \\ e^{-\frac{\Delta E}{T_k}} & \text{if } \Delta E \geq 0 \end{cases} \tag{8}$$

where $\Delta E = f(x_{\text{new}}) - f(x_{\text{current}})$ is the difference in objective function values between the current and new solutions.

As the internal "temperature" of the algorithm gradually decreases, the temperature change can be expressed by the formula:

$$T_k = T_0 \cdot \alpha^k \tag{9}$$

where T_0 is the initial temperature and α is a decay factor less than 1. As the temperature decreases, the probability of accepting inferior solutions also diminishes. After each iteration, the algorithm updates the current solution:

$$x_{k+1} = \begin{cases} x_{\text{new}} & \text{if new solution accepted} \\ x_k & \text{if new solution rejected} \end{cases} \tag{10}$$

The optimization process gradually stabilizes, ultimately converging to the global optimum x^*, thereby minimizing $f(x^*)$.

However, traditional annealing algorithms primarily focus on unidirectional optimization, gradually reducing the temperature to minimize perturbation and slowly approaching the optimal solution. Once the temperature reaches its lowest point, the algorithm loses its effectiveness. This unidirectional cooling strategy gradually diminishes the algorithm's perception of the network environment during execution, leading to a decrease in adaptability to dynamic changes. Particularly when faced with failures, the algorithm may not respond promptly, thereby affecting the overall fault detection and recovery speed. Therefore, this paper proposes the reverse coupled annealing algorithm, which consists of two coupled annealing algorithms: Algorithms 1 and 2. These algorithms collaborate to optimize link and node states based on $\varphi(u)$ and implement dynamic fault detection through the OSPF interval synchronization mechanism. When a node assesses the overall state of node v, neighboring node u, and the link between them through the comprehensive evaluation function $\varphi(u)$, it triggers DOSA if $\varphi(u)$ falls below the preset threshold ε, executing downward optimization to reduce the values of T_h and T_d; otherwise, it triggers UOSA. Through this alternating operation of heating and cooling, the two annealing algorithms dynamically couple and interact, thereby adjusting the parameters T_h and T_d based on the network environment. The details are as follows.

Algorithm 1: Upward optimization simulated annealing algorithm (UOSA)

Input: UOSA current temperature: T_0^u; UOSA termination temperature: T_k^u; UOSA annealing weight: δ^u; DOSA current temperature: T_0^d; UOSA heating weight: β^u; DOSA maximum temperature: T_m^d; UOSA inner loop iterations: M_1^u; UOSA current optimal sending interval: s_{cb}^u Current Hello packet sending interval: T_h ; Current fault detection time: T_d;
Output: next Hello packet sending interval: s_{nh}; next fault detection time: s_{nd}; UOSA optimal sending interval: s_b^u; DOSA next temperature: T_n^d

```
1:  Initialize: $s_{nh} = T_0^u$, $s_b^u = s_{cb}^u$, $s_{nd} = T_d$
2:  if $T_0^u > T_k^u$ then
3:      Initialize: $k = 0$
4:      while $k < M_1^u$ do
5:          $s_{nh} = f_u(s_{nh})$
6:          difference value: $\Delta s = s_{nh} - T_h$
7:          if $\Delta s \geq 0$ then
8:              $s_{nh} = \min\{30, s_{nh}\}$
9:          else if $e^{(\Delta s / T_0^u)} > random(0,1)$ then
10:             $s_{nh} = \min\{30, s_{nh}\}$
11:         end if
12:         if $s_{nh} > s_b^u$ then
13:             $s_b^u = s_{nh}$
14:         end if
15:         $k = k + 1$
16:     end while
17:     $T_n^u = \delta^u * T_0^u$
18:     $s_{nd} = \min\{40, s_{nh} * 40\}$
19: end if
20: $T_0^d = \min(\beta^u * T_0^d, T_m^d)$
```

Algorithm 2: Downward optimization simulated annealing algorithm (DOSA)

```
Input: DOSA current temperature: $T_0^d$; DOSA termination temperature: $T_k^d$; DOSA annealing weight: $\delta^d$; UOSA current temperature: $T_0^u$; DOSA heating weight: $\beta^d$; UOSA maximum temperature: $T_m^u$; DOSA inner loop count: $M_1^d$; DOSA current optimal sending interval: $s_{cb}^d$; Current hello Packet transmission interval: $T_h$; Current fault detection time: $T_d$
Output: next Hello packet transmission interval: $s_{nh}$; next fault detection time: $s_{nd}$; DOSA optimal sending interval: $s_b^d$; UOSA next temperature: $T_n^u$
1:  Initialize: $s_{nh} = T_0^{\mathrm{d}}, s_b^d = s_{cb}^d$
2:  if $T_0^d > T_k^d$ then
3:      Initialize: $k = 0$
4:      while $k < M_1^d$ do
5:          $s_{nh} = f_d(s_{nh})$
6:          difference value: $\Delta s = s_{nh} - T_h$
7:          if $\Delta s \leq 0$ then
8:              $s_{nh} = \max\{1, s_{nh}\}$
9:          else if $e^{(-\Delta s / T_0^u)} > random(0, 1)$ then
10:             $s_{nh} = \max\{1, s_{nh}\}$
11:         end if
12:         if $s_{nh} < s_b^d$ then
13:             $s_b^u = s_{nh}$
14:         end if
15:         $k = k + 1$
16:     end while
17:     $T_n^d = \delta^d * T_0^d$
18:     $s_{nd} = \min\{40, s_{nh} * 40\}$
19: end if
20: $T_0^u = \min(\beta^d * T_0^u, T_m^u)$
```

The objective of the upward optimization simulated annealing (UOSA) algorithm is to optimize the Hello packet transmission interval T_h and the fault detection time T_d, gradually increasing both parameters to their maximum values. In line 2 of the pseudocode, it first checks whether the initial temperature T_0^u has reached the termination temperature T_k^u; if not, it enters the inner loop. Lines 4 to 16 describe the inner loop process of the simulated annealing algorithm. The algorithm generates a random number in the range of -2 to 4 using the random seed generation function $f_u(s_{nh})$ and adds it to s_{nh} to obtain a new s_{nh}. Subsequently, it assesses whether the generated difference Δs meets the Metropolis criterion, ultimately updating s_{nh} and the current optimal state s_b^u. Line 17 implements the cooling operation of the annealing process, and line 18 calculates the new fault detection time s_{nd} based on the updated s_{nh}. Line 20 executes the heating operation for the downward optimization simulated annealing (DOSA); each time UOSA is completed, the initial temperature T_0^d for DOSA is increased. In summary, the code structure of this algorithm consists of only one outer loop, with the number of iterations being M_1^u. The algorithm's complexity is $O(M_1^u)$.

The downward optimization simulated annealing algorithm (DOSA) also targets the Hello message transmission interval T_h and the fault detection time T_d, but its optimization direction is opposite to that of UOSA, aiming to gradually reduce these two parameters to their minimum values. The process is similar to UOSA, so it will not be elaborated here.

3. Simulation and Evaluation

3.1. Evaluation Metrics

The experiment primarily evaluates routing convergence performance through four metrics: fault detection time (FDT), fault detection accuracy (FDA), routing overhead

(RO), and packet delivery rate (PDR). The definitions and formulas for these metrics are as follows:

(1) Fault detection rime (FDT): The fault detection time is defined as the average difference between the time τ when all OSPF nodes detect a fault during network simulation and the actual time t when the link fails [35,36]. A shorter time indicates faster routing convergence under the same conditions, reflecting the efficiency of the fault detection mechanism's impact on overall routing convergence speed.

$$FDT = \frac{1}{n}\sum_{i=1}^{n}(\tau_i - t_i) \tag{11}$$

(2) Fault detection accuracy (FDA): The fault detection accuracy is defined as the ratio of the number of successful link fault detections F_n by all OSPF nodes during network simulation to the total number of actual link failures F_l. A higher accuracy indicates better detection precision in the face of link failures, reflecting the responsiveness of the detection mechanism to actual fault events.

$$FDA = \frac{F_n}{F_l} \times 100\% \tag{12}$$

(3) Routing overhead (RO): Routing overhead is expressed as the ratio of the total number of bytes in control packets B_r to the total number of bytes in successfully transmitted packets B_s [37,38]. A smaller routing overhead indicates lower costs associated with protocol routing convergence.

$$RO = \frac{B_r}{B_s} \times 100\% \tag{13}$$

(4) Packet delivery rate (PDR): The packet delivery rate is defined as the proportion of successfully transmitted packets P_r to the total packets sent P_s by a node [39,40]. A higher packet delivery rate indicates greater reliability of the routing convergence method.

$$PDR = \frac{P_r}{P_s} \times 100\% \tag{14}$$

3.2. Results, Analysis, and Discussion

The simulation platform for this experiment is based on the Ubuntu 16.04 operating system, running on hardware equipped with a 12th Gen Intel(R) Core(TM) i7-12700H processor and 16 GB of RAM. The experiment utilizes the NS3.33 network simulation software to conduct the simulations. Through these simulations, performance evaluations were carried out on four key metrics: fault detection time (FDT), fault detection accuracy (FDA), routing overhead (RO), and packet delivery rate (PDR). The experiment compared the performance of the OSPF-ICSA algorithm, the AODV protocol [7,8], the OSPF protocol, the BFD protocol [15,16], and the routing convergence algorithm based on load centrality (LC) [28,30]. The experiments set up two network topologies based on the literature [1,2,4,5,12]. The algorithm and topology parameters are shown in Table 1.

Testing was conducted under two fault scenarios, randomly introducing between 1 and 20 link failures, and 1 and 5 node failures. The experimental results are as follows.

Figures 5 and 6 present a comparative experimental analysis of fault detection times under link failure and node failure scenarios for two different scales of topologies. As shown in the figures, the BFD algorithm demonstrates the best performance in terms of fault detection time, with the shortest detection time. However, the detection speed of OSPF-ICSA is also impressive, with an average fault detection time of approximately 4 s. Compared to the AODV protocol, OSPF-ICSA reduces fault detection time by an average of 74% and 37.38% in the two scenarios, respectively. In comparison to the traditional OSPF protocol, OSPF-ICSA achieves reductions of 86.24% and 87.37%, respectively. Additionally, compared to the load centrality (LC) algorithm, fault detection time is reduced by an average of 34.73% and 40.14%, respectively. These results indicate that OSFP-ICSA significantly enhances fault detection efficiency, second only to the BFD algorithm. Its lower fault detection time is attributed to the OSFP-ICSA algorithm's ability to respond flexibly and rapidly in the network environment. When network conditions deteriorate, OSFP-ICSA minimizes fault detection time through the reverse coupled annealing algorithm, significantly reducing the detection duration. Topology 1 in Figure 5 shows that, with one link failure, the detection time is 12 s. This occurs because both annealing algorithms reach a dynamic equilibrium at the 12 s mark. Subsequently, as the number of link failures increases, the fault detection time continues to decrease until it reaches its minimum value.

Table 1. Algorithm and topology parameter configuration table.

Parameter Name	Topology 1	Topology 2
UOSA Maximum Temperature T_m^u	100	100
UOSA Termination Temperature T_k^u	0	0
UOSA Heating Weight β^u	1.2	1.2
UOSA Annealing Weight δ^u	0.8	0.8
UOSA Inner Loop Iterations M_1^u	1	1
DOSA Maximum Temperature T_m^d	100	100
DOSA Termination Temperature T_k^d	0	0
DOSA Heating Weight β^d	1.2	1.2
DOSA Annealing Weight δ^d	0.8	0.8
DOSA Inner Loop Iterations M_1^d	1	1
Threshold ε	0.8	0.8
Number of Backbone Nodes	40	80
Number of Access Networks	13	19
Number of Flows	16	26

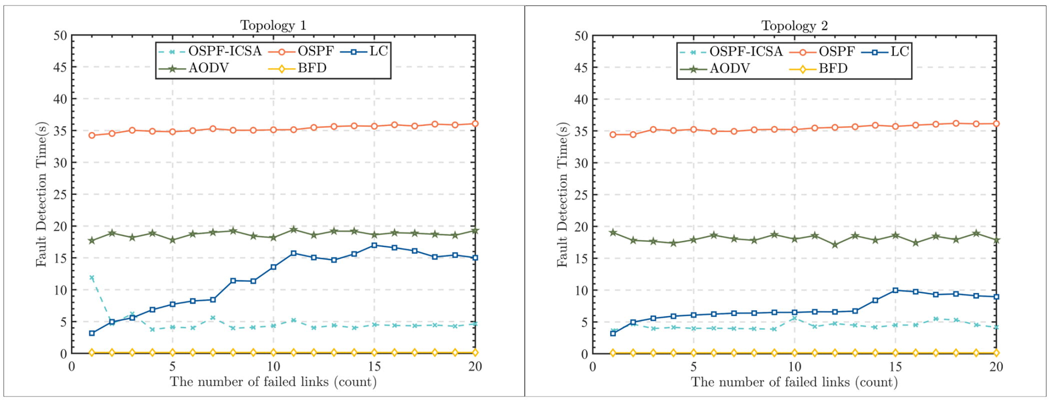


Figure 5. Comparison of link failure count vs. fault detection time.

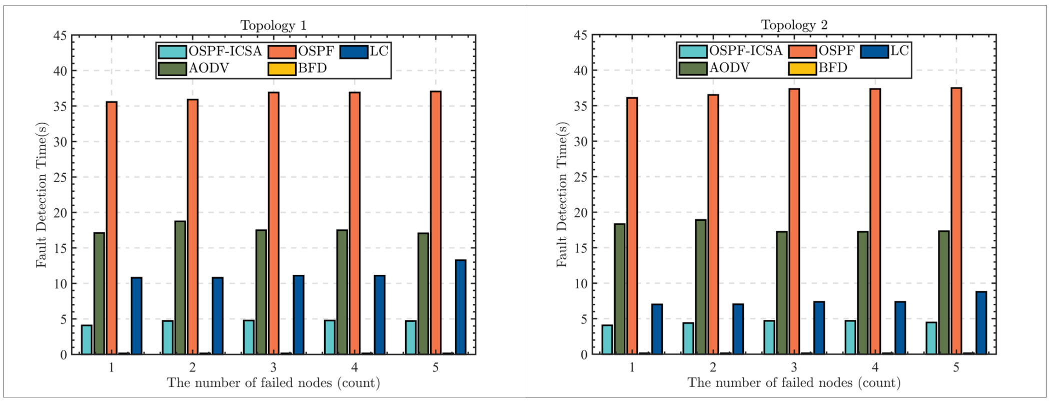


Figure 6. Comparison of node failure count vs. fault detection time.

Figures 7 and 8 display the fault detection accuracy of five algorithms under link failure and node failure scenarios for two different scales of topologies. It can be observed that the accuracy of the OSPF-ICSA algorithm is nearly identical to that of the BFD algorithm in both scenarios, with both approaching 100%. Compared to the AODV protocol, OSPF-ICSA improves the fault detection accuracy by an average of over 20.19% and 3.1% in the link failure and node failure scenarios, respectively. When compared to the traditional OSPF protocol, the accuracy is increased by an average of over 10.54% and 27.68% in the two scenarios. Additionally, compared to the load centrality (LC) algorithm, OSPF-ICSA's detection accuracy is improved by an average of over 3.56% and 1.14% in the respective scenarios. These results indicate that OSPF-ICSA significantly enhances fault detection accuracy, performing comparably to the BFD algorithm, which achieves faster detection by increasing routing overhead. The advantage of OSPF-ICSA lies in its ability to dynamically adjust the frequency of Hello message transmissions and fault detection times based on real-time feedback from link and node states, thus enabling efficient and flexible fault detection that achieves the same level of precision as the BFD algorithm.

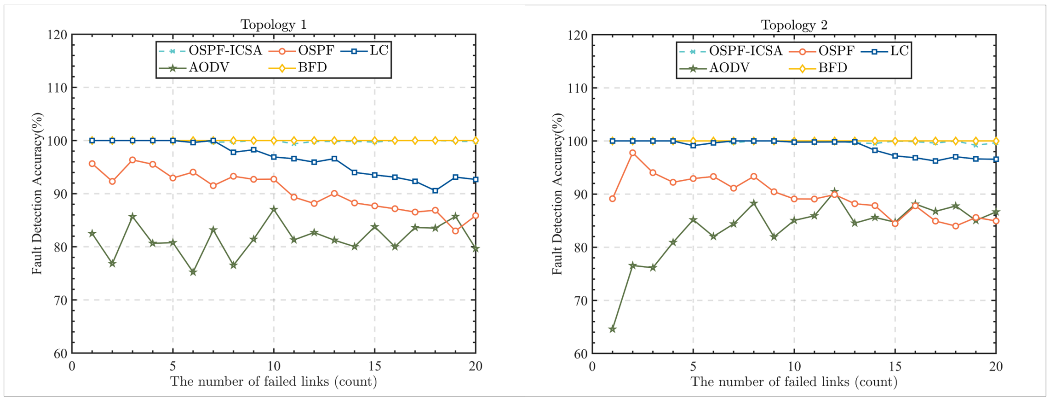


Figure 7. Comparison of link failure count vs. fault detection accuracy.

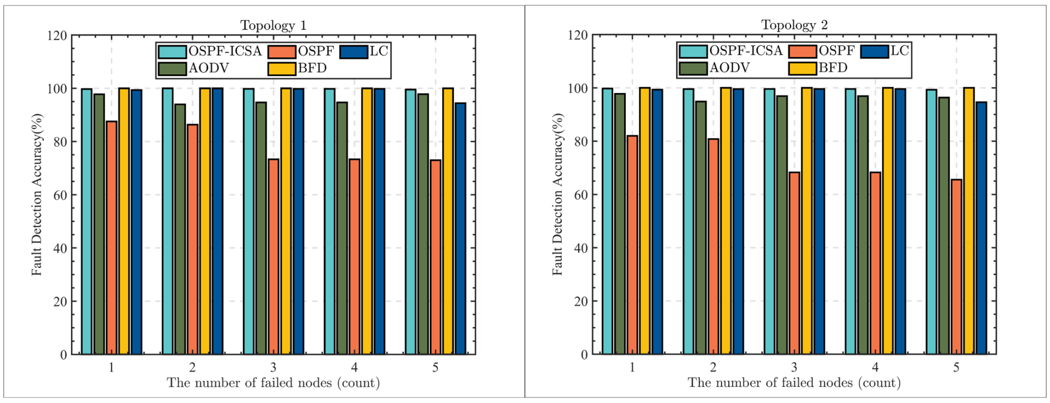


Figure 8. Comparison of node failure count vs. fault detection accuracy.

Figures 9 and 10 present a comparative analysis of routing overhead for five algorithms under link failure and node failure scenarios across two different network scales. It is evident that the routing overhead of OSPF-ICSA is generally lower than that of other algorithms. In both link failure and node failure scenarios, OSPF-ICSA achieves lower overhead than the AODV protocol under most conditions, with the exception of the larger topology in the node failure scenario, where it is slightly higher than AODV. Compared to the traditional OSPF protocol, OSPF-ICSA reduces overhead by an average of 21.66% and 44.44%, respectively. When compared to the BFD algorithm, OSPF-ICSA decreases overhead by an average of 63.77%. Additionally, it reduces overhead compared to the load centrality (LC) algorithm by an average of 28.14% and 20.47% in the respective scenarios. This can be attributed to two main factors: first, OSPF-ICSA significantly shortens fault detection time, enabling nodes in the network topology to quickly sense changes, thus increasing the total amount of successfully transmitted packets compared to traditional OSPF and LC algorithms. Second, OSPF-ICSA dynamically adjusts the frequency of Hello message transmissions, reducing the frequency of Hello messages on normal links and, consequently, lowering the overhead of control messages. In contrast, the BFD algorithm incurs higher routing overhead, partly due to its high frequency of probe message transmissions and partly due to false alarms caused by link jitter and congestion, which further increases routing overhead.

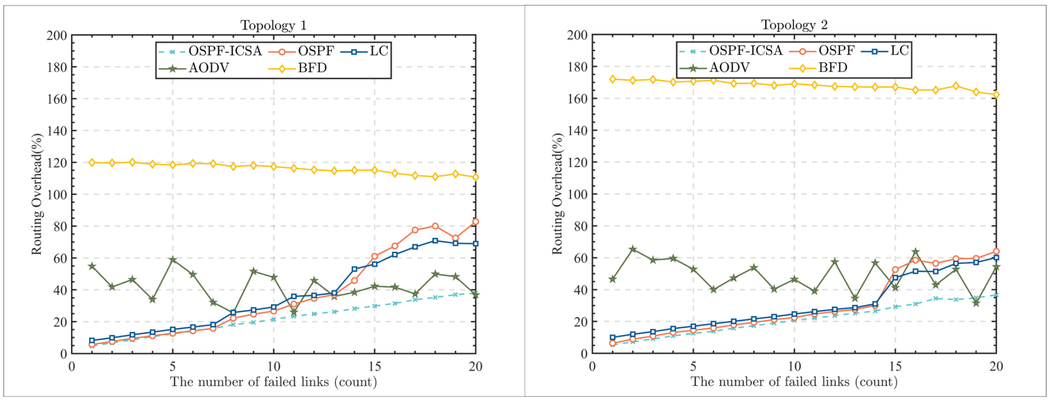


Figure 9. Comparison of link failure count vs. routing overhead.

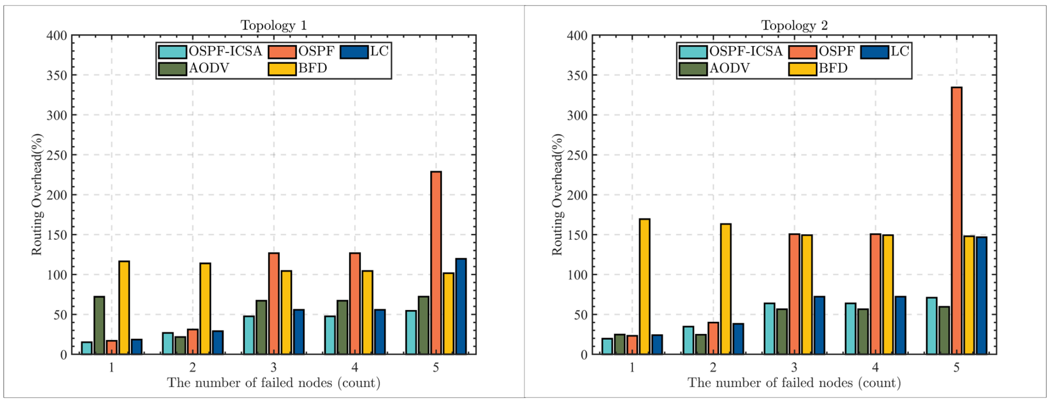


Figure 10. Comparison of node failure count vs. routing overhead.

Figures 11 and 12 illustrate the comparison of packet delivery rates for five algorithms under link failure and node failure scenarios across two different network scales. It is evident that OSPF-ICSA improves packet delivery rates by an average of 14.04% and 5.1% compared to the AODV protocol in these two scenarios, respectively. When compared to the traditional OSPF protocol, packet delivery rates increase by an average of 34.03% and 153.93%. Furthermore, it is noticeable that the traditional OSPF protocol exhibits poor packet delivery rates during multiple node failures. Additionally, OSPF-ICSA outperforms the load centrality (LC) algorithm, with average increases in packet delivery rates of 15.78% and 28.09% under link failure and node failure scenarios, respectively. However, compared to the BFD algorithm, OSPF-ICSA shows a slight decrease, with reductions of 3.46% and 7.72%. The advantage of OSPF-ICSA in packet delivery rates primarily stems from its ability to perceive changes in network topology and dynamically adjust fault detection times, leading to faster fault diagnosis and LSA flooding. This enables the entire network to detect link failures more swiftly. Although the LC algorithm has optimized fault detection, its response speed remains inferior to that of OSPF-ICSA. The slight disadvantage of OSPF-ICSA compared to BFD is due to BFD's ability to adjust fault detection times to extremely low frequencies, significantly lower than the detection frequency of OSPF-ICSA, placing them in different magnitudes. However, under non-extreme conditions, these slightly reduced packet delivery rates remain acceptable.

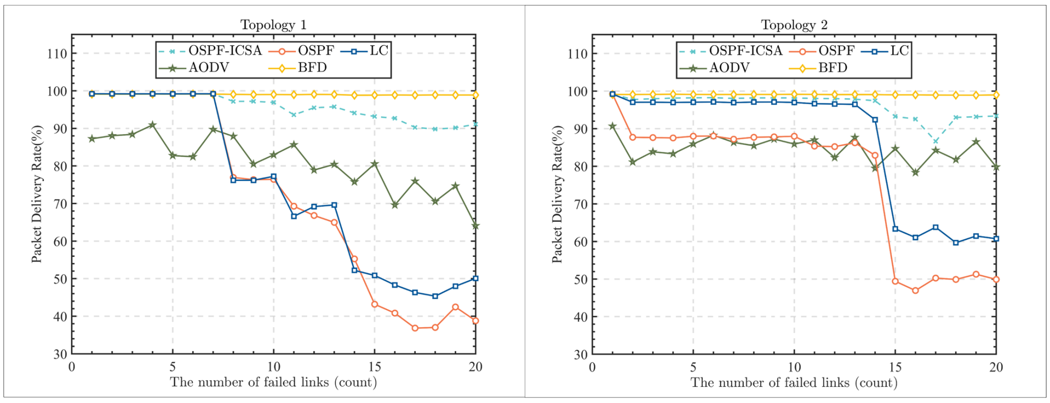


Figure 11. Comparison of link failure count vs. packet delivery rate.

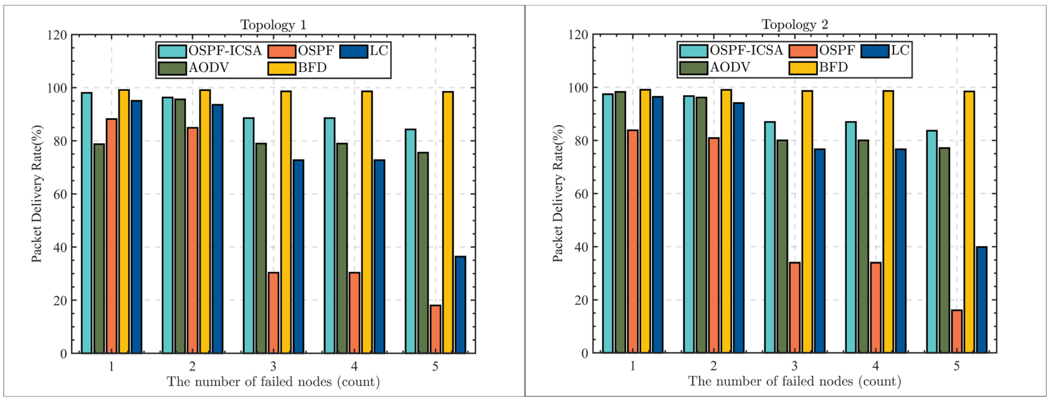


Figure 12. Comparison of node failure count vs. packet delivery rate.

4. Conclusions

In the context of the current evolution of IoT network structures from small-scale distributed systems to large-scale hierarchical collaboration between backbone and access networks, the dynamic changes in backbone node connections and the surge in service demands pose significant challenges. The sluggish fault detection speed leads to a substantial reduction in the effective service transmission time. This paper proposes an OSPF dynamic routing convergence method based on reverse coupled simulated annealing (OSPF-ICSA). This method utilizes the statistical characteristics of Hello packets to acquire the link status and characterizes the node state based on the aggregated features of the link status, providing data support for the reverse coupled simulated annealing algorithm regarding the network conditions. Furthermore, the Hello packet is improved and an OSPF interval synchronization and node state transmission mechanism is designed to synchronize the sending intervals and fault detection times of nodes within the same subnet, while sharing the node state of nodes. Building on this foundation, the reverse coupled simulated annealing algorithm is introduced to collaboratively optimize the Hello packet sending interval and fault detection time. Experimental results demonstrate that OSPF-ICSA exhibits outstanding performance across various fault scenarios, particularly in four key indicators: fault detection time, detection accuracy, routing overhead, and packet delivery rate. This algorithm effectively addresses the trade-off between routing convergence time and routing overhead, achieving an optimized convergence speed while significantly reducing resource consumption. The aim is for OSPF-ICSA to provide a new perspective on routing convergence in large-scale hierarchical IoT network topologies. In future research, we plan to integrate machine learning, large models, and other cuttingedge algorithms to achieve more efficient and rapid routing convergence, ushering in a new era of network optimization.

Author Contributions: Conceptualization, methodology, and software, C.P.; validation, formal analysis, and investigation, H.L.; resources, data curation, and writing—original draft preparation, H.S.; writing—review and editing, visualization, and supervision, Y.W.; project administration, and funding acquisition, L.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 61931004.

Data Availability Statement: Data are contained within the article.

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

References

1. Madakam, S.; Ramaswamy, R.; Tripathi, S. Internet of Things (IoT): A literature review. *J. Comput. Commun.* **2015**, *3*, 164–173. [CrossRef]
2. Mouha, R.A.R.A. Internet of things (IoT). *J. Data Anal. Inf. Process.* **2021**, *9*, 77–101.
3. Xia, F.; Yang, L.T.; Wang, L.; Vinel, A. Internet of things. *Int. J. Commun. Syst.* **2012**, *25*, 1101–1102. [CrossRef]
4. Ortiz, A.M.; Hussein, D.; Park, S.; Han, S.N.; Crespi, N. The cluster between internet of things and social networks: Review and research challenges. *IEEE Internet Things J.* **2014**, *1*, 206–215. [CrossRef]
5. Shahraki, A.; Taherkordi, A.; Haugen, Ø.; Eliassen, F. A survey and future directions on clustering: From WSNs to IoT and modern networking paradigms. *IEEE Trans. Netw. Serv. Manag.* **2020**, *18*, 2242–2274. [CrossRef]
6. Chithaluru, P.; Al-Turjman, F.; Kumar, M.; Stephan, T. Energy-balanced neuro-fuzzy dynamic clustering scheme for green & sustainable IoT based smart cities. *Sustain. Cities Soc.* **2023**, *90*, 104366.
7. Alahari, H.P.; Yalavarthi, S.B. A survey on network routing protocols in internet of things (IOT). *Int. J. Comput. Appl.* **2017**, *160*, 18–22.
8. Khan, T.; Singh, K.; Hasan, M.H.; Ahmad, K.; Reddy, G.T.; Mohan, S.; Ahmadian, A. ETERS: A comprehensive energy aware trust-based efficient routing scheme for adversarial WSNs. *Future Gener. Comput. Syst.* **2021**, *125*, 921–943. [CrossRef]
9. Elappila, M.; Chinara, S.; Parhi, D.R. Survivable path routing in WSN for IoT applications. *Pervasive Mob. Comput.* **2018**, *43*, 49–63. [CrossRef]
10. Guo, W.; Yan, C.; Lu, T. Optimizing the lifetime of wireless sensor networks via reinforcement learning-based routing. *Int. J. Distrib. Sens. Netw.* **2019**, *15*, 1550147719833541. [CrossRef]
11. Younus, M.U.; Khan, M.K.; Bhatti, A.R. Improving the Software-Defined Wireless Sensor Networks Routing Performance Using Reinforcement Learning. *IEEE Internet Things J.* **2022**, *9*, 3495–3508. [CrossRef]
12. Moy, J.T. *OSPF: Anatomy of an Internet Routing Protocol*; Addison-Wesley Professional: Boston, MA, USA, 1998.
13. Moy, J.T. *OSPF Complete Implementation*; Pearson Education: London, UK, 2008.
14. Biradar, A.G. A comparative study on routing protocols: RIP, OSPF and EIGRP and their analysis using GNS-3. In Proceedings of the 2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE), Jaipur, India, 1–3 December 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–5.
15. *RFC 5880*; Bidirectional Forwarding Detection (BFD). Proposed Standard. Internet Engineering Task Force: Fremont, CA, USA, 2010.
16. *RFC 9355*; OSPF Bidirectional Forwarding Detection (BFD) Strict-Mode. RFC Editor: Marina del Rey, CA, USA, 2023.
17. Tsegaye, Y.; Geberehana, T. Ospf Convergence Times. Master's Thesis, Computer Science and Engineering, Chalmers University Of Technology, Göteborg, Sweden, 2012.
18. Goyal, M.; Soperi, M.; Baccelli, E.; Choudhury, G.; Shaikh, A.; Hosseini, H. Improving convergence speed and scalability in OSPF: A survey. *IEEE Commun. Surv. Tutorials* **2011**, *14*, 443–463. [CrossRef]
19. Manousakis, K.; McAuley, A.J. Using stochastic approximation to design OSPF routing areas that satisfy multiple and diverse end-to-end performance requirements. In Proceedings of the 2008 6th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks and Workshops, Berlin, Germany, 1–3 April 2008.
20. *RFC 5449*; OSPF Multipoint Relay (MPR) Extension for ad Hoc Networks. Internet Engineering Task Force, Request For Comments (Experimental). Internet Engineering Task Force: Fremont, CA, USA, 2009.
21. Baccelli, E.; Cordero, J.; Jacquet, P. Multi-point relaying techniques with OSPF on ad hoc networks. In Proceedings of the IEEE International Conference on Systems and Networks Communications (ICSNC), Porto, Portugal, 20–25 September 2009.
22. *RFC 5820*; Extensions to OSPF to Support Mobile Ad Hoc Networking. Internet Engineering Task Force: Fremont, CA, USA, 2010.
23. *RFC 7038*; Use of OSPF-MDR in Single-Hop Broadcast Networks. Internet Engineering Task Force: Fremont, CA, USA, 2013.
24. Waqas, M.; Malik, S.U.R.; Akbar, S.; Adeel, A.; Naveed, A. Convergence time analysis of OSPF routing protocol using social network metrics. *Future Gener. Comput. Syst.* **2019**, *94*, 62–71. [CrossRef]

25. Jain, N.; Payal, A.; Jain, A. Effect of data packet size on the performance of RIP and OSPF routing protocols in hybrid networks. *Int. J. Pervasive Comput. Commun.* **2021**, *17*, 361–376. [CrossRef]
26. Maccari, L.; Cigno, R.L. Improving routing convergence with centrality: Theory and implementation of pop-routing. *IEEE/ACM Trans. Netw.* **2018**, *26*, 2216–2229. [CrossRef]
27. Liu, G.; Deng, Y.; Cheong, K.H. Network immunization strategy by eliminating fringe nodes: A percolation perspective. *IEEE Trans. Syst. Man, Cybern. Syst.* **2022**, *53*, 1862–1871. [CrossRef]
28. Maccari, L.; Ghiro, L.; Guerrieri, A.; Montresor, A.; Cigno, R.L. Exact distributed load centrality computation: Algorithms, convergence, and applications to distance vector routing. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *31*, 1693–1706. [CrossRef]
29. Ghiro, L.; Restuccia, F.; D'Oro, S.; Basagni, S.; Melodia, T.; Maccari, L.; Lo Cigno, R. What is a Blockchain? A Definition to Clarify the Role of the Blockchain in the Internet of Things. *arXiv* **2021**, arXiv:2102.03750.
30. Chethan Raj, C.; Hanumanthappa, J.; Sharath Kumar, G.N.; Inchara, G.P. Intelligent Trust Classification for Social Internet of Things Using Centrality Feed Forward Networks. *SN Comput. Sci.* **2024**, *5*, 803. [CrossRef]
31. Bertsimas, D.; Tsitsiklis, J. Simulated annealing. *Stat. Sci.* **1993**, *8*, 10–15. [CrossRef]
32. Nadimi-Shahraki, M.H.; Zamani, H.; Asghari Varzaneh, Z.; Mirjalili, S. A systematic review of the whale optimization algorithm: theoretical foundation, improvements, and hybridizations. *Arch. Comput. Methods Eng.* **2023**, *30*, 4113–4159. [CrossRef] [PubMed]
33. Zhao, W.; Wang, L.; Zhang, Z.; Mirjalili, S.; Khodadadi, N.; Qiang, G. Quadratic Interpolation Optimization (QIO): A new optimization algorithm based on generalized quadratic interpolation and its applications to real-world engineering problems. *Comput. Methods Appl. Mech. Eng.* **2023**, *417*, 116446. [CrossRef]
34. Zhao, S.; Zhang, T.; Cai, L.; Yang, R. Triangulation topology aggregation optimizer: A novel mathematics-based meta-heuristic algorithm for continuous optimization and engineering applications. *Expert Syst. Appl.* **2024**, *238*, 121744. [CrossRef]
35. Isermann, R. Process fault detection based on modeling and estimation methods—A survey. *Automatica* **1984**, *20*, 387–404. [CrossRef]
36. Cerdà-Alabern, L.; Iuhasz, G.; Gemmi, G. Anomaly detection for fault detection in wireless community networks using machine learning. *Comput. Commun.* **2023**, *202*, 191–203. [CrossRef]
37. Raja Basha, A. A review on wireless sensor networks: Routing. *Wirel. Pers. Commun.* **2022**, *125*, 897–937. [CrossRef]
38. Kumar, S.; Raw, R.S.; Bansal, A. Minimize the routing overhead through 3D cone shaped location-aided routing protocol for FANETs. *Int. J. Inf. Technol.* **2021**, *13*, 89–95. [CrossRef]
39. Li, R.; Makhijani, K.; Dong, L. New IP: A data packet framework to evolve the Internet. In Proceedings of the 2020 IEEE 21st International Conference on High Performance Switching and Routing (HPSR), Newark, NJ, USA, 11–14 May 2020.
40. Vijayalakshmi, S.; Bose, S.; Logeswari, G.; Anitha, T.J.C.S. Hybrid defense mechanism against malicious packet drop attack for MANET using game theory. *Cyber Secur. Appl.* **2023**, *1*, 100011. [CrossRef]