Article

Binary-Addition Tree Algorithm-Based Resilience Assessment for Binary-State Network Problems

Yi-Zhu Su * and Wei-Chang Yeh

Department of Industrial Engineering and Engineering Management, National Tsing Hua University, Hsinchu 30013, Taiwan; yeh@ieee.org

* Correspondence: u102034007@gmail.com

Received: 4 July 2020; Accepted: 24 July 2020; Published: 27 July 2020

Abstract: Applications in real life are composed of different kinds of network systems; these networks may be interfered by uncontrollable or unpredictable disruptive events involving natural disasters, human errors, evil-intentioned attacks, or other disturbances. Any of these disruptive events will cause networks to malfunction and possibly result in large economic losses. As a result, it is important to assess network resilience which is a measure to describe how a network system recovers its performance and functionality to a satisfactory level from a disruptive event. Inspired by the measures of reliability evaluation used in binary-state networks, this paper proposes a binary-addition tree algorithm-based resilience assessment for binary-state networks and applies it on a wildfire network with wireless sensors. Considering the stochastic nature of disruptive events, the proposed binary-addition tree algorithm-based resilience assessment comprehensively enumerates all the possible disruptive events and all the corresponding recovery strategies, and then calculate the network resilience. Furthermore, recovery cost limit is concerned in this paper for decision makers who choose the recovery strategies with their recovery cost limit and resilience requirement.

Keywords: network resilience; component importance measures; binary-state network; binary addition; network reliability

1. Introduction

Several infrastructure networks such as transportation networks [1–5], telecommunication networks [6–8], and power grids [9] are subject to uncontrollable disruptive events [10]. Uncontrollable or unpredictable disruptive events [10], such as natural disasters [1–3,9], human errors, evil-intentioned attacks [4], or other disturbances [4,5], inevitably influence the functions of networks in our real lives. Since network components are always high connective and interdependent, the entire network may be affected once one or more components fail such as link disconnections or node broken because of these disruptive events [1–5,9,10].

Since not all these disruptive events can be accurately predicted and completely prevented, resilience of network systems has become a crucial measure recently [1–3,5,9–15]. There are two major issues in resilience assessment of a network: network resilience and component importance measures. Network resilience is applied to assess the ability of a network system to recover its performance and functionality to a satisfactory level from a disruptive event [10–15]. Component importance measures are to identify the components which have great significant impact on the system performance once they failed in a network system [10,16,17].

In this paper, we focus on the resilience assessment of the binary-state network because the binary-state network is always the fundamental structure and has been increasingly researched and applied in the various real-life application phases for planning, design, execution, management, and control [18] involving signal transmission [19], grid or cloud computing [20], data mining [21], Internet of Things [22], etc. To evaluate the performance of a binary-state network, reliability is the common measurement in the engineering domain [18–22]. The reliability of the binary-state network is the probability that at least one directed path from the source node to the sink node exists in the network [18–22].

Calculating the binary-state network reliability is an NP-hard problem [18,23–25]. There are two ways to estimate the binary-state network reliability: direct algorithms [18,26] and indirect algorithms [27–29]. Direct algorithms find the network reliability directly, but indirect algorithms find the network reliability indirectly based on the minimal cuts (MCs) [18,27,28] or minimal paths (MPs) [18,29,30]. MPs and MCs are the arc sets with no redundant arcs and minimal number of arcs to connected and disconnected the network from the source node to the sink node [18,28,29].

However, indirect algorithms are more complicated than direct algorithms and not easily understood by decision makers who really care about the final resilience assessment and do not focus on technically calculating the network reliability which is the intermediate product of the resilience calculation process. Therefore, directed algorithms are utilized to compute the network reliability in this paper.

The direct algorithms including the binary-decision diagram (BDD) [31], the state-space algorithm [26,32], and the binary-addition tree algorithm (BAT) [18]. Coding skill with complex data structures is required and there is no open source for users and researchers to download for the BDD [31]. The state-space algorithm and the BAT both find all the possible state vectors and then calculate network reliability [18,26], but the efficient of the BAT is higher than that of the state-space algorithm due to the reduction methods contained in the BAT. Moreover, the logic and coding of the BAT is easier to be understood and performed than that of the state-space algorithm and other existing algorithms [18] and can be further applied to resilience assessment in this study. Thereafter, the BAT-based resilience assessment method is proposed in this study to provide the decision makers to systematically assess the network resilience.

In this paper, our proposed BAT-based resilience assessment is implemented on a case study about a wildfire detection sensor network which is the domain of wireless sensor networks (WSNs) in the internet of things (IoT) [6,8]. WSNs comprise autonomous sensor devices (named motes in the WSN domain or called nodes in this paper) distributed in space [33,34]. There is at least a node as a sink node in a WSN to retrieve the data gathered by other nodes and connect with other networks [33,34]. These sensors are major in monitoring the surrounding environment (e.g., temperature, humidity, vibration, motion, and voices [34], etc.), collecting raw data and transmitting it to the sink node [33,34]. They may have limited energy supply, processing unit with limited computational capabilities, wireless communications with limited bandwidth, or limited receiving range [33,34]. To achieve long-range communications in a WSN, Felici-Castell et al. suggest placing the receiver (the sink node) at a low Earth orbit (LEO) satellite [33,34]. Based on these techniques for WSNs, wildfire detection sensors can collect the required data and send to the sink node to alert the users.

The rest of the study is arranged as follows: Section 2 lists relevant acronyms, notations, nomenclatures, and assumptions used in this paper. Section 3 mainly introduces the methodological background related to this paper including network resilience and resilience-based component importance measures. Binary-addition tree algorithm (BAT) is illustrated in Section 4. Section 5 details the proposed BAT-based resilience assessment method in this study and introduces its operational steps with an example. Section 6 is a case study of a wildfire detection sensor network using proposed BAT-based resilience assessment. The conclusions are summarized in Section 7.

2. Acronyms, Notations, Nomenclatures, and Assumptions

Relevant acronyms, notations, nomenclatures, and assumptions listed in this section.

2.1. Acronyms

MP/MC:	minimal path/cut
BAT:	binary-addition tree algorithm
PLSA:	path-based layered-search algorithm

2.2. Notations

n, m :	numbers of nodes and arcs, respectively
V :	node set $V = \{1, 2, \dots, n\}$
E :	arc set $E = \{a_1, a_2, \dots, a_m\}$
e_{ij} :	directed arc between nodes i and j
a_k :	k th directed arc in E
$\mathbf{D}$:	binary-state network with the success probability of each arc
$G(V, E)$:	A graph with the source node 1, the sink node n , V , and E . For example, Figure 1 shows a graph with the source node 1, the sink node 4, $V = \{1, 2, 3, 4\}$ and $E = \{a_1 = e_{1,2}, a_2 = e_{1,3}, a_3 = e_{2,3}, a_4 = e_{2,4}, a_5 = e_{3,4}, a_6 = e_{2,1}, a_7 = e_{3,1}, a_8 = e_{4,2}, a_9 = e_{4,3}\}$. Note that all the graphs in this study are directed, e.g., $a_3 = e_{2,3}$ represents that the arc $e_{2,3}$ is from node 2 to node 3 with an arrow. Since there is no $e_{3,2}$, there is no arc $e_{3,2}$ from node 3 to node 2. And the arcs with no arrow represents that the arcs are bidirectional, e.g., $a_1 = e_{1,2}$ and $a_6 = e_{2,1}$.
$G(V, E, \mathbf{D})$:	A graph $G(V, E)$ becomes a binary-state network with the success probability of each arc, represented as $G(V, E, \mathbf{D})$.
t_0 :	time point for original network
t_e :	time point for disruptive event happened
t_d :	time point for disruption event done
t_r :	time point for recovery done
$\text{Pr}(a), \text{Pr}'(a)$:	success probability of a_k of the original network status and the better network status, respectively.
$C(a), C'(a)$:	full recovery cost for the network recovery to the original network status, and partial recovery cost for the network recovery to the better network status, respectively. For example, Figure 1 is a binary-state network $G(V, E, \mathbf{D})$ with $\text{Pr}(a), \text{Pr}'(a), C(a)$, and $C'(a)$ which are presented in Table 1.
R_0 :	reliability of original binary-state network $G(V, E, \mathbf{D})$
R_d :	reliability of binary-state network $G(V, E, \mathbf{D})$ when the disruptive event finishes
R_r :	reliability of binary-state network $G(V, E, \mathbf{D})$ after recovery strategies
A_d :	failed arc sets when the disruptive event finishes
A_r :	recovered arc sets after recovery strategies
S_i :	failed scenario $i, i = 0, 1, \dots, 2^m - 1$
O_{ij} :	recovery strategies O_{ij} , where i stands for the scenario S_i, j stands for the j th recovery strategy in $S_i, i = 0, 1, \dots, 2^m - 1, j = 0, 1, \dots, 2^{ A_d(S_i) } - 1$.
I_1 :	the ratio of the disruptive effect on the performance to original state's performance of the failed arc set
I_2 :	the ratio of the recovery effect on the performance to original state's performance of the failed arc set
I :	component importance, where component can be the failed arc set
φ :	resilience of binary-state network $G(V, E, \mathbf{D})$ after the network experiences disruptions and recovery strategies with recovery cost limit c .
c :	recovery cost limit
X :	state vector whose i th element represents the state of a_k for $k = 1, 2, \dots, m$. i.e., $X = \{X(a_1), X(a_2), \dots, X(a_m)\}$
$X(a_i)$:	state (value) of the a_i (the i th element) in X
$\text{Pr}(X(a_i))$:	occurrence probability of a_i when its state is $X(a_i)$
$\text{Pr}(X)$:	$\text{Pr}(X) = \text{Pr}(X(a_1)) \cdot \text{Pr}(X(a_2)) \cdot \dots \cdot \text{Pr}(X(a_m))$
$G(X)$:	The subgraph corresponding to state vector X such that $G(X) = G(V, \{a \in E \mid \text{for all } a \text{ with } X(a) = 1\})$; e.g., $X = (1, 1, 1, 1, 1, 0, 0, 0, 0)$, the graph of $G(X)$ is depicted in Figure 2.
n_p :	The minimal number of arcs that connects the graph from node 1 to node n ; e.g., $\{a_1, a_4\}$ and $\{a_2, a_5\}$, so $n_p = 2$.
n_c :	The minimal number of arcs that disconnects the graph from node 1 to node n ; e.g., $\{a_1, a_2\}$ and $\{a_4, a_5\}$, so $n_c = 2$.

Figure 1. Example network [18].

Table 1. Arc success probability and recovery cost in Figure 1.

a	$\text{Pr}(a)$	$C(a)$	$\text{Pr}'(a)$	$C'(a)$	a	$\text{Pr}(a)$	$C(a)$	$\text{Pr}'(a)$	$C'(a)$
$a_1 = e_{1,2}$	0.8	30	0.6	20	$a_6 = e_{2,1}$	0.9	40	0.5	20
$a_2 = e_{1,3}$	0.9	40	0.7	30	$a_7 = e_{3,1}$	0.8	50	0.6	30
$a_3 = e_{2,3}$	0.7	30	0.5	20	-	-	-	-	-
$a_4 = e_{2,4}$	0.8	40	0.6	20	$a_8 = e_{4,2}$	0.7	30	0.6	20
$a_5 = e_{3,4}$	0.9	50	0.7	40	$a_9 = e_{4,3}$	0.8	40	0.7	30

Figure 2. $G(X)$ and $X = (1, 1, 1, 1, 1, 0, 0, 0, 0)$ in Figure 1 [18].

2.3. Nomenclatures

Reliability:	The success probability that there is one direct path from node 1 to node n in a binary-state network
Vulnerability:	A kind of reliability that considers the network with disruptive events
Recoverability:	A kind of reliability that considers the network with recovery strategies
Resilience:	The capability for the network to recover from disruptions.
Connected vector:	A state vector X is connected if nodes 1 and n are connected and there is at least one path without redundant arcs in $G(X)$.

2.4. Assumptions

- (1) There is no situation with failed nodes.
- (2) Each arc only has two situations in the binary-state network.
- (3) The connected network has no parallel arcs or loops.
- (4) The success probability of each arc is independent according to the given distribution.
- (5) The network can recover to the original network status (recover perfectly) with full recovery cost or to the better network status (recover imperfectly) with partial recovery cost

3. Methodological Background

In Section 3, this study introduces the research most related topics including network resilience and resilience-based component importance measures.

3.1. Network Resilience

There are different definitions for resilience with different domains [12], such as “the inherent ability and adaptive response that enables firms and regions to avoid maximum potential losses” given by Rose for the economic domain [14], or “the sum of passive survival rate (reliability) and proactive survival rate (restoration) of a system” by Youn et al. for the engineering domain [15]. In engineering domain, reliability is always considered to be an important measure of resilience [12].

For distinct systems, the general definition and the critical part of resilience are the ability for a system to recover from disruptions, that is, to “absorb” disruptive events, and to “recover” to original or functional system state [12]. From these definitions of resilience, we elaborate the absorbing ability as “vulnerability”, and the recovery ability as “recoverability”. Concepts for vulnerability and recoverability in the resilience issue are also adapted by Pant et al. in the time-dependent systems [11,13] and applied to other studies such as infrastructure systems and component importance measure [11].

However, the specific definition for the resilience depends on the problem we are solving. In a binary-state network we are considering, reliability is the success probability that there is one direct path from node 1 to node n . Vulnerability and recoverability are also kinds of reliability which consider the network with disruptive events and recovery strategies, respectively. Moreover, based on the resilience definitions of Pant et al. [13], Rocco et al. [11] and our needs, resilience φ in Equation (1) is the average ratio of recovery to loss under the maximum recovery reliability requirement and the recovery cost limit c :

$$\varphi = \sum_i^s \varphi(S_i, c) = \sum_i^s \frac{R_r(S_i) - R_d(S_i)}{R_0 - R_d(S_i)} / s, \quad (1)$$

where S_i , s , R_0 , R_d , and R_r represent the failed scenario, total number of the failed scenario, the time points t_0 , t_d , and t_r we calculate the reliability, vulnerability, and recoverability, respectively.

Figure 3 summarizes different states in a resilience network system over time [11,13]. At time point t_0 , performance of the original system state is reliability R_0 . A disruptive event occurs at time point t_e , and ends at time point t_d . Vulnerability R_d is the ability of the system withstanding the disruption. After disruption, the system now is in the disrupted state and maybe needs to wait for recovery until time point t_s . Finally, at time point t_r , the system recovers to the recovered state with recoverability R_r .

3.2. Resilience-Based Component Importance Measures

In a resilient network, the issues of the resilience-based component importance measures are also emphasized recently [10]. There are three aspects to evaluate the component importance based on resilience concepts: one aspect is from the disruptive effect of failed components such as Ramirez-Marquez [16] and Barker et al. [17], another aspect is from the recovery effect of failed components, the other is from both disruptive and recovery effect of failed components such as Xu et al. [10]. Based on three aspects, we list Equation (2), Equation (3) and Equation (4) for failed arc set A , respectively. In Equation (2), $I_1(A_d(S_i))$ is the ratio of the disruptive effect on the performance to original state's performance of failed arc set A_d in the corresponding scenario S_i . In Equation (3), $I_2(A_d(S_i), A_r(S_i))$ is the ratio of the recovery effect on the performance to original state's performance of failed arc set A_d and recovered arc set $A_r(S_i)$ in the corresponding scenario S_i , where $A_r(S_i) \subseteq A_d(S_i)$.

Figure 3. Different states in a resilient network over time [11].

However, if we consider Equations (2) and (3) at same time as Equation (4), we can find Equation (4) is similar to Equation (1) which is the definition of the resilience. The only different between Equation (1) and Equation (4) is their target of evaluation are not the same. In Equation (4), component importance I evaluate the specific failed components (i.e., failed arc sets) in a scenario. In Equation (1), resilience φ evaluate the whole system. It is understandable that if the whole system can be assessed by resilience φ , component importance in the system also can be measured. Thus, we apply Equation (4) as our resilience-based component importance measure:

$$I_1 = I_1(A_d(S_i)) = \frac{R_0 - R_d(A_d(S_i))}{R_0}, \quad (2)$$

$$I_2 = I_2(A_d(S_i), A_r(S_i)) = \frac{R_r(A_r(S_i)) - R_d(A_d(S_i))}{R_0}, \quad A_r(S_i) \subseteq A_d(S_i), \quad (3)$$

$$I = I(A_d(S_i), A_r(S_i)) = \frac{I_2(A_d(S_i), A_r(S_i))}{I_1(A_d(S_i))} = \frac{R_r(A_r(S_i)) - R_d(A_d(S_i))}{R_0 - R_d(A_d(S_i))}, \quad A_r(S_i) \subseteq A_d(S_i), \quad (4)$$

Furthermore, aforementioned resilience φ in Equation (1) can be written as Equation (5):

$$\varphi = \sum_i^s \varphi(S_i, c) = \frac{\sum_i^s \frac{R_r(S_i) - R_d(S_i)}{R_0 - R_d(S_i)}}{s} = \sum_i^s \frac{I(A_d(S_i), A_r(S_i))}{s}, \quad (5)$$

4. BAT

The binary-addition tree algorithm (BAT) proposed by Yeh in 2020 [18] aims to directly compute the network reliability of a binary-state network. BAT is proved that its time complexity, $O(n^2 2^{n^2})$, is less than that of the current popular indirect algorithms, $O(n^2 2^{n-n})$ [18]. Furthermore, BAT is coded more easily than the other existing algorithms [18], including indirect algorithms such as depth-first search (DFS) [29], the heuristic method [35], and the universal generating function method [25,36,37], and direct algorithms such as the state-space algorithm and the binary-decision diagram (BDD) [26,31,32].

The indirect algorithms also called the MP-based algorithms since they find MP first. However, different from the indirect algorithms, BAT uses a simple binary-addition tree to list all the possible solutions which also called state vectors instead find all the MPs first. Then, the connected vectors are screened out from these state vectors by the PLSA. Reduction methods in BAT are applied to reduce unnecessary computations. The network reliability is thus achieved by summing up the probabilities of all the events composed of the connected state vectors. Details of the binary-addition tree, the Path-based

Layered-Search Algorithm (PLSA), the reduction methods, and the method to calculate the probabilities of the connected vectors are presented in this section.

4.1. Binary-Addition Tree

In the BAT, an easy understanding binary-addition tree is used to develop all the possible state vectors at first. Elements of a state vector equals either 0 or 1 and stands for the state of the related arc, e.g., X is a state vector in the original graph G and the value of its i th element is $X(a_i)$, i.e., $X = \{X(a_1), X(a_2), \dots\}$. For example, if $X(a_i)$ equals one, arc a_i exists in subgraph $G(X)$. All the state vectors are generated by adding 1 based on the first state vector, and stop generating until all the elements are 1. There are no duplicated state vectors since the sequence of the arcs in all the state vectors are identical. Moreover, each arc has two states 1 or 0 and there are at most m arcs in a state vector, and thus the time complexity of the binary-addition tree to list all possible state vectors is $O(2^m) = O(2^{n^2})$, where $m = n^2$.

The Algorithm 1 is the pseudocode to describe the abovementioned procedures for a binary-addition tree.

Algorithm 1. Pseudocode for binary-addition tree [18]

Input: $G(V, E)$.

Output: All possible state vectors X without duplications.

STEP a. Let $SUM = 0$, $k = 1$, and $X_1 = X$ be a zero vector with m elements.

STEP b. Let $i = m$.

STEP c. If $X(a_i) = 0$, let $X(a_i) = 1$, $k = k + 1$, $X_k = X$, $SUM = SUM + 1$, and proceed to STEP e.

STEP d. Let $X(a_i) = 0$. If $i > 1$, let $i = i - 1$ and proceed to STEP c.

STEP e. If $SUM = m$, stop and $X_1, X_2, \dots, X_k$ are all possible state vectors. Otherwise, proceed to STEP b.

4.2. PLSA

The path-based layered-search algorithm (PLSA) is used to verify the connectivity of each state vector before computing the network reliability. In the PLAS, nodes are verified layer-by-layer, and nodes reached from node 1 to the current layer L_i store in an ordered vertex subset V_l . $V_l = L_1 \cup L_2 \cup \dots \cup L_k$, where the initial layer $L_1 = \{1\}$ and the current layer $L_i = \{v \in [V - (L_1 \cup L_2 \cup \dots \cup L_{i-1})] \mid \text{for all } e_{\alpha,v} \in E \text{ and } \alpha \in L_{i-1}\}$ for $i = 2, 3, \dots, k$. The state vector is connected if node n belongs to the current layer L_i ; otherwise, the state vector is disconnected if node n does not belong to the current layer L_i . In verifying whether a state vector is connected, there are at most n nodes in V_l , and thus the time complexity of the PLSA is $O(n)$. The Algorithm 2 is the pseudocode to describe the abovementioned procedures for PLSA.

Algorithm 2. Pseudocode for PLAS [18]

Input: A state vector X obtain from the binary-addition tree

Output: Whether X is connected or disconnected

STEP a. Let $i = 2$ and $V_l = L_1 = \{1\}$

STEP b. Let $L_i = \{v \notin V_l \mid \text{for all } e_{\alpha,v} \in E \text{ and } \alpha \in V_l\}$.

STEP c. If $n \in L_i$, there is a directed $(1, n)$ -path in $G(X)$. Stop.

STEP d. If $L_i = \emptyset$, there is a $(1, n)$ -cut in $G(X)$. Halt.

STEP e. Let $V_l = V_l \cup L_i$, $i = i + 1$, and go to STEP b

4.3. Reduction Methods

Computation efficiency of the BAT is improved by reduction methods. Reduction methods are applied in the network to remove the impossible solutions by reducing the number of arcs

and to reduce number of connectivity verifications by applying the minimal number of connected or disconnected arcs.

- (a) **Reduce Number of Impossible Solutions** The arc from the sink node (node n) to the source node (node 1) does not exist in a directed $(1, n)$ -path. Therefore, these arcs impossible to be used are removed from the original network, without effect on the reliability results.
- (b) **Reduce Number of Connectivity Verifications** A directed connected or disconnected network has a number of arcs larger than or equal to n_p and n_c , respectively. Hence, if the numbers of state 1 s and 0 s in the state vector X are smaller than n_p and n_c , nodes 1 and n are disconnected and connected in the network, respectively.

For both two parts of reduction, since it is only to scan one time for each state vector, the time complexity of the reduction methods for a state vector is $O(1)$.

4.4. Calculation of Binary-State Network Reliability Using Connected State Vectors

In the BAT, the final step is to calculate the binary-state network reliability according to all the state vectors acquired from previous steps.

The product of the reliabilities of the working arcs, $\Pr(X_i(a_j) = 1)$, and the un-reliabilities of the failed arcs, $\Pr(X_i(a_j) = 0)$, is the probability of the state vector X , i.e.:

$$\Pr(X_i) = \prod_{j=1}^m \Pr(X_i(a_j)), \quad (6)$$

For all the possible state vectors (connected or disconnected), we have:

$$\sum_{\forall i} \Pr(X_i) = \sum_{\forall i} \prod_{j=1}^m \Pr(X_i(a_j)) = 1, \quad (7)$$

R is the reliability of the binary-state network which sums all the probabilities of connected state vectors X , which represents nodes 1 and n are connected in $G(X)$:

$$R = \sum_{\forall X} \Pr(X), \quad (8)$$

Moreover, there are at most m arcs in a state vector, and thus the time complexity of the reliability computation for a state vector is $O(m) = O(n^2)$, where $m = n^2$. In summary, from the four main procedure of the BAT in this section, the time complexity for the BAT to achieve the binary-state reliability is $O((1 + n + m)2^m) = O((1 + n + n^2)2^{n^2}) = O(n^2 2^{n^2})$.

5. Proposed BAT-Based Resilience Assessment

The procedure for the proposed BAT-based resilience assessment for evaluating the reliability of original network, the vulnerability of disruptive network, the recoverability of the recovered network, the component importance measure, and the resilience of the overall network is presented in Section 5.1. A demonstrated example is presented in Section 5.2.

5.1. BAT-based Resilience Assessment

The step-by-step procedure of the proposed BAT-based resilience assessment based on the BAT in Section 4 is presented as Algorithm 3.

Algorithm 3: Procedure of the BAT-based Resilience Assessment: to evaluate the binary-state network resilience.

Input: $G(V, E, D)$, recovery cost, and the recovery cost limit c .

Output: Original network reliability and assessment for network resilience with vulnerability, recoverability, resilience-based component importance measure, and resilience.

STEP A. Apply BAT algorithm to find the original network reliability R_0 .

STEP A1. Reduce number of impossible solutions.

STEP A2. Apply binary-addition tree to find all state vectors.

STEP A3. Reduce number of connectivity verifications.

STEP A4. Apply PLAS Algorithm to verify connectivity of state vectors.

STEP A5. Calculate reliability of each found connected vector and the original network.

STEP B. Apply BAT-based resilience assessment to obtain network reliability (vulnerability) R_d .

STEP B1. Apply binary-addition tree to find all scenarios with different failed arcs.

STEP B2. For each scenario, utilize the product of the connected state vectors found from step A5 to calculate network reliability (vulnerability) R_d .

STEP C. Apply BAT-based resilience assessment to obtain network reliability (recoverability) R_r .

STEP C1. Apply binary-addition tree to find all possible recovery strategies for different arcs failed.

STEP C2. For the recovery strategies, utilize the product of connected vectors found from STEP A5, calculate network reliability (recoverability) R_r , and calculate related recovery cost.

STEP D. Component importance measure by Equation (2) to (4).

STEP E. According to the maximum recovery reliability requirement and the recovery cost limit c , make decision among all recovery strategies and calculate resilience φ by Equation. (5).

STEP A is based on Section 4 and is for calculating the reliability of original network. There are five steps, STEP A1 to STEP A5, in the STEP A. STEP A1 and STEP A3 are based on Section 4.3 for reducing number of impossible solutions and connectivity verifications, respectively. STEP A2 implements the binary-addition tree introduced in Section 4.1. STEP A4 applies PLSA introduced in Section 4.2. STEP A5 from Section 4.4 is the final step in the STEP A to compute probability product of the connected state vectors and the reliability of original network.

STEP B to STEP E are developed according to the logic of binary-addition tree and the results obtained from STEP A. STEP B is for calculating vulnerability of disruptive network, and STEP C is for calculating the recoverability of the recovered network. They both contain two steps. Both STEP B1 and STEP C1 apply binary-addition tree to list all possible failed scenarios and related recovery strategies, respectively. And both STEP B2 and STEP C2 utilize the results from STEP A5 to obtain network vulnerability and recoverability, respectively. STEP D and STEP E use the vulnerability and recoverability acquired from STEP B2 and STEP C2 to further calculate the component importance and the network resilience.

This is the first time that the BAT-based methods used for the assessment of network vulnerability, recoverability, resilience, and component importance measure.

5.2. Example

To illustrate the detailed procedure of the proposed BAT-based methods for assessing the network resilience, we use the binary-state network shown in Figure 1. Figure 1 is a bridge problem which belongs to one of benchmark problems that are used as a standard with which other algorithms can be compared for practical usage [18]. Following is the procedure of the solution.

Solution:

STEP A. Apply BAT algorithm to find the original network reliability R_0 . STEP A includes STEP A1 to STEP A5 as following.

STEP A1. Reduce number of impossible solutions.

In Figure 1, there are directed arcs from the sink node to the source node, and these arcs are impossible to be used. Thus, Figure 1 is reduced to Figure 2 after removing arcs a_6, a_7, a_8 , and a_9 and the reliabilities of Figures 1 and 2 are the same. Consequently, the runtime can theoretically be reduced from 2^9 to 2^5 because the number of directed arcs is reduced from nine to five.

STEP A2. Apply Binary-addition tree to find all state vectors.

For STEP A2, we use Figure 2 obtained from STEP A1 to replace Figure 1 as the original network graph G for following procedures. In Figure 2, X_i is a state vector with five zero elements, where $i = 1, 2, \dots, 2^5$. By applying the binary-addition tree, all the possible state vectors as shown in Table 7 can be found without duplications, i.e., $X_1 = (0, 0, 0, 0, 0)$, $X_2 = (0, 0, 0, 0, 1)$, $X_3 = (0, 0, 0, 1, 0)$, $X_4 = (0, 0, 0, 1, 1)$, $X_5 = (0, 0, 1, 0, 0)$, and until to $X_{32} = (1, 1, 1, 1, 1)$. Note that $G(X_{32})$ represented Figure 2.

STEP A3. Reduce number of connectivity verifications.

To disconnected or connected the network in Figure 2, the minimal number of arcs equals two, i.e., $n_c = n_p = 2$. For instance, $\{a_1, a_2\}$ and $\{a_1, a_4\}$ are one of the minimal sets to disconnected or connected the network, respectively. So that in Table 7, those whose numbers of state 1 s are smaller than n_p , said X_1, X_2, X_3, X_5, X_9 , and X_{17} , are disconnected. And those whose numbers of state 0 s are smaller than n_c , said $X_{24}, X_{28}, X_{30}, X_{31}$, and X_{32} , are connected. The connectivity of these 11 aforementioned state vectors do not need to be verified, and thus reduce number of connectivity verifications from $2^5 = 32$ to 21.

STEP A4. Apply PLAS Algorithm to verify connectivity of state vectors.

Table 7 shows the procedure of the PLSA to determine whether $X = (1, 1, 1, 1, 1)$ is a connected state vector in Figure 2. Whether the corresponding state vector X in $G(X)$ is connected (Y) or disconnected (N) is summarized in the columns named "Connected?" of Table 4. For example, as shown in Figure 4, $X_{10} = (0, 1, 0, 0, 1)$ is connected implies that nodes 1 and n in $G(X_{10})$ are connected.

STEP A5. Calculate reliability of each found connected vector and the original network.

In Table 4, assume the success and failure probabilities of a_i are p_i and $q_i = 1 - p_i$, respectively. From Equations. (5) to (7), we have $\Pr(X_1) = q_1q_2q_3q_4q_5$ and $\Pr(X_2) = q_1q_2q_3q_4p_5$, $\Pr(X_1) + \Pr(X_2) + \dots + \Pr(X_{32}) = 1$, and $R_0 = \Pr(X_{10}) + \Pr(X_{12}) + \Pr(X_{14}) + \Pr(X_{16}) + \Pr(X_{19}) + \Pr(X_{20}) + \Pr(X_{22}) + \Pr(X_{23}) + \Pr(X_{24}) + \Pr(X_{26}) + \Pr(X_{27}) + \Pr(X_{28}) + \Pr(X_{30}) + \Pr(X_{31}) + \Pr(X_{32}) = 0.94168$. Therefore, we obtain the reliability of each connected vector and the original network.

STEP B. Apply BAT-based resilience assessment to obtain network reliability (vulnerability) R_d .**STEP B1.** Apply binary-addition tree to find all scenarios with different failed arcs.

From Figure 2, all possible scenarios S_i with different arcs failed are found by the logic of binary-addition tree and list in Table 2, where $i = 0, 1, \dots, 2^5 - 1$. That is, let S_i be a state vector with five zero elements and $S(a_k)$ be the state value 1 or 0 for arc a_k failed or not failed, respectively, i.e., $S_i = \{S_i(a_1), S_i(a_2), \dots\}$. Thus, $S_0 = (0, 0, 0, 0, 0)$ represents no arcs failed, $S_1 = (0, 0, 0, 0, 1)$ represents a_5 failed, $S_2 = (0, 0, 0, 1, 0)$ represents a_4 failed, $S_3 = (0, 0, 0, 1, 1)$ represents a_4 and a_5 failed, and until to $S_{31} = (1, 1, 1, 1, 1)$ represents all arcs failed. Note that the index i of S_i starts from 0 because S_0 have no arcs failed and the reliability of S_0 equals to original network.

So, actually, there are $2^5 - 1$ scenarios with the different failed arcs in the network in Figure 2.

STEP B2. For each scenario, utilize the product of the connected state vectors found from STEP A5 to calculate network reliability (vulnerability) R_d .

From STEP A5, we have the product of each connected state vectors shown in Table 4. The network vulnerability for the scenario S_i , $R_d(S_i)$, is just the sum of the product of the remaining connected state vectors in S_i . Following provides a few examples with one, two, three arcs failed, respectively. For the scenario S_{16} with a_1 failed, we only need to consider the connected state vectors with $X(a_1) = 0$ in Table 4. Then the network vulnerability $R_d(S_{16}) = \Pr(B_{10}) + \Pr(B_{12}) + \Pr(B_{14}) + \Pr(B_{16}) = 0.162$. For the scenario S_{24} with a_1, a_2 failed, we only need to consider the connected state vectors with $X(a_1) = 0$ and $X(a_2) = 0$ in Table 4. Since there is no connected state vector in S_{24} , the network vulnerability $R_d(S_{24})$ equals zero. And For the scenario S_{28} with a_1, a_2, a_3 failed, we only need to consider the connected state vectors with $X(a_1) = 0$, $X(a_2) = 0$, and $X(a_3) = 0$ in Table 4. Hence, there is no connected state vector in S_{28} , the network vulnerability $R_d(S_{28})$ equals zero. S_i and $R_d(S_i)$ obtain from STEP B1 and B2 are summarized in Table 2. In Table 2, columns $A_d(S_i)$ is an arc set contains the failed arcs in the scenario S_i and $R_d(A_d(S_i)) = R_d(S_i)$ is the network vulnerability of the scenario S_i . Note that scenarios S_i and failed arc sets $A_d(S_i)$ (or following mentioned $A_r(S_i)$) have one-to-one relationship, that is, a S_i corresponds to a specific $A_d(S_i)$, so does $A_d(S_i)$. Thus, we sometimes do not write A_d for some notations in this article, such as $R_d(A_d(S_i)) = R_d(S_i)$, $R_r(S_i) = R_r(A_r(S_i))$, and $C(S_i) = C(A_r(S_i))$.

STEP C. Apply BAT-based resilience assessment to obtain network reliability (recoverability) R_r .

STEP C1. Apply binary-addition tree to find all possible recovery strategies for different arcs failed.

For a scenario S_i with failed arcs, there are different strategies to recover the network to better or original state. Theses recovery strategies $O_{i,j}$ are found by the logic of binary-addition tree and list in Table 3, where i stands for the scenario S_i , j stands for the j th recovery strategy in S_i , $i = 0, 1, \dots, 2^5 - 1$, $j = 0, 1, \dots, 2^{|A_d(S_i)|} - 1$. That is, let $O_{i,j}$ be a state vector with the number of zero elements which equals the number of failed arcs in S_i and $O_{i,j}(a_k)$ be the state value 1 or 0 for arc recovered or not recovered, respectively, i.e., $O_{i,j} = \{O_{i,j}(a_1), O_{i,j}(a_2), \dots\}$. For example, as shown in Table 3, $O_{28,0} = (0, 0, 0)$ represents no arcs recovered in the S_{28} , $O_{28,1} = (0, 0, 1)$ represents arc a_3 recovered in the S_{28} , $O_{28,2} = (0, 1, 0)$ represents arc a_2 recovered in the S_{28} , and until to $O_{28,7} = (1, 1, 1)$ represents all arcs recovered in the S_{28} . Note that the index i and index j of $O_{i,j}$ both start from 0 because $O_{0,0}$ have no arcs failed, no arcs recovered, and the reliability of $O_{0,0}$ equals to original network. So, actually, there are $2^{|A_d(S_i)|} - 1$ recovery strategies with the different recovered arcs for the scenario S_i in the network.

STEP C2. For the recovery strategies, utilize the product of connected vectors found from STEP A5, calculate network reliability (recoverability) R_r , and calculate related recovery cost.

From STEP A5, we have the product of each connected state vectors shown in Table 4. The network recoverability for the recovery strategy $O_{i,j}$, $R_r(A_r(O_{i,j})) = R_r(O_{i,j})$, is just the sum of the product of the remaining connected state vectors in $O_{i,j}$. Note the recovery strategy $O_{i,j}$ and recovered arc sets $A_r(O_{i,j})$ have one-to-one relationship, that is, an $O_{i,j}$ corresponds to a specific $A_r(O_{i,j})$, so does

$A_r(O_{i,j})$. Thus, we sometimes do not write A_r for some notations in this article, such as $R_r(A_r(O_{i,j})) = R_r(O_{i,j})$, $C(A_r(O_{i,j})) = C(O_{i,j})$, $C'(A_r(O_{i,j})) = C'(O_{i,j})$. For example, for the recovery strategy $O_{28,1}$ in scenario S_{28} with only a_3 recovered and a_1, a_2 still failed, we only need to consider the connected state vectors with $X(a_1) = 0$ and $X(a_2) = 0$ in Table 4. Since there is no connected state vector in $O_{28,1}$, the network recoverability $R_r(O_{28,1})$ which the network recover to the original state equals zero. $O_{28,j}$ and $R_r(O_{28,j})$ obtain from STEP C1 and C2 are summarized in Table 3. In Table 3, columns $A_r(O_{i,j})$ is an arc set contains the recovered arcs in the recovery strategy $O_{i,j}$, $A_d(S_i) - A_r(O_{i,j})$ is an arc set contains the still not recovered arcs. Note there are full recovery cost $C(A_r(O_{i,j})) = C(O_{i,j})$ which recover to the original network, or partial recovery cost $C'(A_r(O_{i,j})) = C'(O_{i,j})$ which recover to the functional network for different recovery strategies $O_{i,j}$. And the corresponding network recoverability of recovery strategy $O_{i,j}$ are $R_r(O_{i,j})$ and $R_r'(O_{i,j})$, respectively. Furthermore, if there is no recovery reliability requirement and no recovery cost limit, it is necessary to choose the best recovery strategies in terms of the recovery efficiency, that is, the largest ratio of recovery reliability to cost. Therefore, the most efficient recovery strategies for scenario S_i among all the related recovery strategies $O_{i,j}$ is $R_r(S_i) = R_r(A_r(S_i)) = R_r\{A_r(S_i) | \max\{R_r(O_{i,j})/C(O_{i,j}), R_r'(O_{i,j})/C'(O_{i,j})\} \text{ for } j = 1, 2, \dots, 2^{|A_d(S_i)|} - 1\}$ for $i = 1, 2, \dots, 2^5 - 1$, e.g., $R_r(S_{28}) = 0.6988$.

STEP D. Component importance measure by Equations (2)–(4).

Components can be nodes, arcs, or communities, and in our paper is “arc sets”. According to STEP C2, we can have all the best recovery strategies $R_r(S_i)$ and thus compute component importance I for arc sets by Equations (2)–(4) as shown in Table 5 if not considering recovery reliability requirement and recovery cost limit. Here index i are listed based on the number and the sequence of the arc sets $A_d(S_i)$ in Table 5 for more easily to compare the importance for different number of arcs. From Table 5, $\{a_2\}$ and $\{a_5\}$ are the most important among arc sets with one arc. $\{a_1, a_5\}$, $\{a_2, a_5\}$, and $\{a_3, a_5\}$ are the most important among arc sets with two arcs. $\{a_2, a_3, a_5\}$ is the most important among arc sets with three arcs. Component importance of these arc sets, $\{a_2\}$, $\{a_5\}$, $\{a_1, a_5\}$, $\{a_2, a_5\}$, $\{a_3, a_5\}$, $\{a_2, a_3, a_5\}$, are equals one, which represents the disruptive network recover to the original state and it is the most efficient to recover these failed arcs. Moreover, arc sets with four or five arcs are equally important. However, if we only use I_1 or I_2 , the results of the component importance might vary widely, e.g., for $\{a_1\}$, $I_1 = 0.8280$ and $I_2 = 0.5700$. According to I_1 , the importance of $\{a_1\}$ is apt to be illustrates as important. Otherwise, according to I_2 , the importance of $\{a_1\}$ is apt to be illustrates as not important. Note that in this example, $A_d(S_i)$ and $A_r(S_i)$ are identical for each S_i . This phenomenon illustrates that there is a significant effect if there is one arcs failed in this small network.

STEP E. According to the maximum recovery reliability requirement and the recovery cost limit c , make decision among all recovery strategies and calculate resilience φ by Equation (5).

To calculate the resilience φ by Equation (5), the recovery strategies are chosen if the network can recover to its maximum reliability under different recovery cost limit c . That is, here the chosen recovery strategies for scenario S_i among all the related recovery strategies $O_{i,j}$ is $R_r(S_i) = R_r(A_r(S_i)) = R_r\{A_r(S_i) | \max\{R_r(O_{i,j}), R_r'(O_{i,j})\} \text{ for } C(O_{i,j}) \text{ or } C'(O_{i,j}) < c, j = 1, 2, \dots, 2^{|A_d(S_i)|} - 1\}$ for $i = 1, 2, \dots, 2^5 - 1$. And the cost of the chosen recovery strategies $R_r(S_i)$ is $C(S_i) = C(A_r(S_i))$. There are two issues for the network resilience within the recovery cost limit c : (1) to make decisions under c , and (2) to determine the proper c . Initially, we discuss the different network resilience under different c . Secondly, we focus on finding appropriate c . Finally, within these appropriate c , we chosen the most appropriate recovery strategies and obtain the corresponding network resilience. Following is the discussion. If $c = 200$,

Table 4. All the X_i obtained in the BAT.

i	X_i	Connected?	i	X_i	Connected?
1	(0, 0, 0, 0, 0)	N	17	(1, 0, 0, 0, 0)	N
2	(0, 0, 0, 0, 1)	N	18	(1, 0, 0, 0, 1)	N
3	(0, 0, 0, 1, 0)	N	19	(1, 0, 0, 1, 0)	Y
4	(0, 0, 0, 1, 1)	N	20	(1, 0, 0, 1, 1)	Y
5	(0, 0, 1, 0, 0)	N	21	(1, 0, 1, 0, 0)	N
6	(0, 0, 1, 0, 1)	N	22	(1, 0, 1, 0, 1)	Y
7	(0, 0, 1, 1, 0)	N	23	(1, 0, 1, 1, 0)	Y
8	(0, 0, 1, 1, 1)	N	24	(1, 0, 1, 1, 1)	Y
9	(0, 1, 0, 0, 0)	N	25	(1, 1, 0, 0, 0)	N
10	(0, 1, 0, 0, 1)	Y	26	(1, 1, 0, 0, 1)	Y
11	(0, 1, 0, 1, 0)	N	27	(1, 1, 0, 1, 0)	Y
12	(0, 1, 0, 1, 1)	Y	28	(1, 1, 0, 1, 1)	Y
13	(0, 1, 1, 0, 0)	N	29	(1, 1, 1, 0, 0)	N
14	(0, 1, 1, 0, 1)	Y	30	(1, 1, 1, 0, 1)	Y
15	(0, 1, 1, 1, 0)	N	31	(1, 1, 1, 1, 0)	Y
16	(0, 1, 1, 1, 1)	Y	32	(1, 1, 1, 1, 1)	Y

Table 5. All the component importance from Figure 2.

i	$A_d(S_i)$	$I_1(A_d(S_i))$	$I_2(A_d(S_i), A_r(S_i))$	$I(A_d(S_i), A_r(S_i))$	i	$A_d(S_i)$	$I_1(A_d(S_i))$	$I_2(A_d(S_i), A_r(S_i))$	$I(A_d(S_i), A_r(S_i))$
0	-	-	-	-	28	a_1, a_2, a_3	1.0000	0.7421	0.7421
16	a_1	0.8280	0.5700	0.6885	26	a_1, a_2, a_4	1.0000	0.7421	0.7421
8	a_2	0.9213	0.9213	1	25	a_1, a_2, a_5	1.0000	0.7421	0.7421
4	a_3	0.7032	0.4453	0.6332	22	a_1, a_3, a_4	0.9897	0.7318	0.7394
2	a_4	0.8173	0.5593	0.6844	21	a_1, a_3, a_5	1.0000	0.7421	0.7421
1	a_5	0.9320	0.9320	1	19	a_1, a_4, a_5	1.0000	0.7421	0.7421
24	a_1, a_2	1.0000	0.7421	0.7421	14	a_2, a_3, a_4	1.0000	0.7421	0.7421
20	a_1, a_3	0.9484	0.6905	0.7280	13	a_2, a_3, a_5	0.9980	0.9980	1
18	a_1, a_4	0.9656	0.7077	0.7329	11	a_2, a_4, a_5	1.0000	0.7421	0.7421
17	a_1, a_5	1.0000	1.0000	1	7	a_3, a_4, a_5	1.0000	0.7421	0.7421
12	a_2, a_3	0.9796	0.7217	0.7367	30	a_1, a_2, a_3, a_4	1.0000	0.7421	0.7421
10	a_2, a_4	0.9893	0.7314	0.7393	29	a_1, a_2, a_3, a_5	1.0000	0.7421	0.7421
9	a_2, a_5	0.9932	0.9932	1	27	a_1, a_2, a_4, a_5	1.0000	0.7421	0.7421
6	a_3, a_4	0.8931	0.6502	0.7280	23	a_1, a_3, a_4, a_5	0.9417	0.6988	0.7421
5	a_3, a_5	0.9225	0.9225	1	15	a_2, a_3, a_4, a_5	0.9417	0.6988	0.7421
3	a_4, a_5	0.9417	0.6988	0.7421	31	a_1, a_2, a_3, a_4, a_5	0.9417	0.6988	0.7421

Table 6. Range of c for the distinct number of failed arcs from Table 1.

$ A_d(S_i) $	Max	Min
1	50	20
2	90	40
3	130	60
4	160	90
5	190	130

Table 7. Procedure from the PLSA for Figure 2.

i	L_i	L_{i+1}	V_l	Remark
1	{1}	{2, 3}	{1, 2, 3}	-
2	{2, 3}	{4}	{1, 2, 3, 4}	a directed $(1, n)$ -path found

Figure 4. $G(X_{10})$ and $X_{10} = (0, 1, 0, 0, 1)$.

6. Case Study

Most countries have forests and may suffer from wildfires (also called forest fires) [6,8]. There are various kinds of sensors (i.e., temperature, wind speed, wind direction, images and so on) that can detect wildfires [6,8]. These sensors are typically distributed in the forest and collect context information that provides the user (i.e., forest manager) to determine sequential emergency treatments in order to minor the possible losses [6]. Thus, to detect the wildfire and successfully transmit related information to the forest manager is an important task for a wildfire detection sensor network which is an application of internet of things (IoT) [6,8].

However, if one of the components in the network is damaged, it may cause the message transmission delay or even be interrupted. Therefore, systematically assessing the network resilience and repair strategies are important issues. This section considers the real problem of a wildfire detection sensor network. The network is illustrated in Section 6.1. Section 6.2 utilizes the proposed BAT-based resilience assessment on this network.

6.1. Wildfire Detection Sensor Network

A wildfire detection sensor network with eight nodes and 11 arcs is shown in Figure 5, where source node 1 represents the sensor which is the closest to the location of the wildfire, and sink node 8 represents the forest manager who obtains the information sent from sensors via his equipment (i.e., computer, mobile phone, or server). Other nodes, node 2 to node 7, are all wireless sensors similar to node 1 to detect wildfires but maybe with different functions. Arcs between these nodes are the directions of information when the sensor in node 1 detects wildfires. To simplify the problem, here we only consider the most possible transmission directions among these wireless sensors when node 1 detects wildfires though wireless links between sensors are bidirectional actually. The information transmission about wildfires may be disrupted by unpredictable or irresistible factors in the forest. Once the disruption of information transmission happens, every arc a has two recovery choices: full recovery to original success probability $\Pr(a)$ with cost $C(a)$, or partial recovery to better success probability $\Pr'(a)$ with cost $C'(a)$. The corresponding success probability to transmit information and recovery cost (unit: 10^3 US dollar) of the arcs are listed in Table 8. Moreover, the recovery cost limit c equals 10, 20, or 30 (unit: 10^2 US dollar).

Figure 5. The wildfire detection sensor network.

Table 8. Arc success probability and recovery cost in Figure 5.

a.	Pr(a)	C(a)	Pr'(a)	C'(a)	a	Pr(a)	C(a)	Pr'(a)	C'(a)
$a_1 = e_{1,2}$	0.98	3	0.80	2.5	$a_7 = e_{4,5}$	0.88	3.2	0.70	2.7
$a_2 = e_{1,4}$	0.95	6	0.75	5	$a_8 = e_{4,6}$	0.92	3.6	0.85	3.4
$a_3 = e_{2,3}$	0.90	2.4	0.70	1.8	$a_9 = e_{5,8}$	0.90	2.9	0.85	2.7
$a_4 = e_{2,7}$	0.99	4.5	0.85	4	$a_{10} = e_{6,7}$	0.98	6.5	0.82	6
$a_5 = e_{3,4}$	0.85	3	0.75	2.6	$a_{11} = e_{7,8}$	0.95	2.8	0.88	2.4
$a_6 = e_{3,5}$	0.85	2.5	0.70	2	-	-	-	-	-

6.2. Resilience Assessment

The proposed BAT-based resilience assessment is applied to analyze the network resilience. Following is the analysis procedure.

Input: $G(V, E, D)$, recovery cost in Figure 5 and Table 8, and the recovery cost limit $c = 10, 20$, or 30 .

Output: Original network reliability and assessment resilience for the binary-state wildfire detection sensor network with vulnerability, recoverability, resilience-based component importance measure, and resilience.

STEP A. Apply BAT algorithm to find the original network reliability R_0 .

STEP A1. Reduce number of impossible solutions.

In Figure 5, there are no directed arcs from the sink node to the source node and thus proceed to STEP A2.

STEP A2. Apply Binary-addition tree to find all state vectors.

In Figure 5, X_i is a state vector with eleven zero elements since there are eleven arcs, where $i = 1, 2, \dots, 2^{11}$. By applying the binary-addition tree, all the possible state vectors can be found without duplications, i.e., $X_1 = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$, $X_2 = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$, $X_3 = (0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)$ and until to $X_{2048} = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$.

STEP A3. Reduce number of connectivity verifications.

To connected and disconnected the network in Figure 5, n_p equals 3 and n_c equals 2, respectively. For example, $\{a_1, a_2\}$ and $\{a_1, a_4, a_{11}\}$ are one of the minimal sets to disconnected or connected the network, respectively. For X_i , the numbers of state 1 s smaller than n_p is disconnected, and the numbers of state 0 s smaller than n_c is connected. Thus, number of connectivity verifications in the STEP A4 is reduced.

STEP A4. Apply PLAS Algorithm to verify connectivity of state vectors X_i . There are 639 connected X_i .

STEP A5. Calculate reliability of each found connected vector and the original network $R_0 = 0.99188$.

STEP B. Apply BAT-based resilience assessment to obtain network reliability (vulnerability) R_d .

STEP B1. Apply binary-addition tree to find all scenarios S_i with different failed arcs, where $i = 0, 1, \dots, 2^{11} - 1$. There are $2^{11} - 1$ S_i in Figure 5.

STEP B2. For each scenario S_i , utilize the product of the connected state vectors found from STEP A5 to calculate network reliability (vulnerability) $R_d(S_i)$. S_i , $R_d(S_i)$, and related failed arc set $A_d(S_i)$ are summarized in Table 9.

Table 9. All the S_i from Figure 5 without recovery cost limit c .

i.	S_i	$A_d(S_i)$	$R_d(S_i)$	$R_r(S_i)$	I
0	(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)	-	0.99188	-	-
1	(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1)	a_{11}	0.04337	0.87713	0.87901
2	(0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)	a_{10}	0.01974	0.99188	1
3	(0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1)	a_{10}, a_{11}	0.00086	0.99188	1
4	(0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0)	a_9	0.09462	0.99188	1
5	(0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1)	a_9, a_{11}	0	0.99188	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$		
2047	(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)	$a_1, a_2, a_3, a_4, a_5,$ $a_6, a_7, a_8, a_9, a_{10},$ a_{11}	0	0.87713	0.88430

STEP C. Apply BAT-based resilience assessment to obtain network reliability (recoverability) R_r .

STEP C1. Apply binary-addition tree to find all possible recovery strategies for different arcs failed. There are $(2^{11} - 1) \times (2^{|A_d(S_i)|} - 1)$ recovery strategies O_{ij} , where i stands for the arc failed scenario S_i , j stands for the j th recovery strategy in S_i ; $i = 0, 1, \dots, 2^{11} - 1$, $j = 0, 1, \dots, 2^{|A_d(S_i)|} - 1$.

STEP C2. For the recovery strategies O_{ij} , utilize the product of connected vectors found from STEP A5, calculate network reliability (recoverability) R_r , and calculate related recovery cost.

Here, the largest ratio of recovery reliability to cost is considered. That is, the most efficient recovery strategies for scenario S_i among all the related recovery strategies O_{ij} is $R_r(S_i) = R_r(A_r(S_i)) = R_r\{A_r(S_i) | \max\{R_r(O_{ij})/C(O_{ij}), R_r'(O_{ij})/C'(O_{ij})\} \text{ for } j = 1, 2, \dots, 2^{|A_d(S_i)|} - 1\}$ for $i = 1, 2, \dots, 2^{11} - 1$. $R_r(S_i)$ without considering recovery limit c are summarized in Table 9.

STEP D. Component importance measure by Equations (2)–(4).

According to STEP C2, we can have all the best recovery strategies $R_r(S_i)$ and thus compute component importance I for arc sets by Equation (2) to (4) as shown in Table 9 if not considering recovery reliability requirement and recovery cost limit. Moreover, rather important arc sets which contain one, two and three arcs are shown in Table 10, where $|A_d(S_i)|$ denotes the number of failed arcs in arc sets.

Table 10. $I = 1$ for arc sets with one, two and three arcs without recovery cost limit c .

$ A_d(S_i) $	$A_d(S_i)$
1	$\{a_4\}, \{a_8\}, \{a_9\}, \{a_{10}\}$
2	$\{a_{10}, a_{11}\}, \{a_9, a_{11}\}, \{a_9, a_{10}\}, \{a_8, a_{11}\}, \{a_8, a_{10}\}, \{a_8, a_9\}, \{a_7, a_{10}\}, \{a_7, a_9\},$ $\{a_7, a_8\}, \{a_6, a_{10}\}, \{a_6, a_8\}, \{a_5, a_{10}\}, \{a_5, a_9\}, \{a_5, a_8\}, \{a_4, a_{10}\}, \{a_4, a_9\}, \{a_4, a_8\},$ $\{a_1, a_{10}\}, \{a_1, a_8\}$
3	$\{a_9, a_{10}, a_{11}\}, \{a_8, a_{10}, a_{11}\}, \{a_8, a_9, a_{11}\}, \{a_8, a_9, a_{10}\}, \{a_7, a_{10}, a_{11}\},$ $\{a_7, a_9, a_{10}\}, \{a_7, a_8, a_{11}\}, \{a_7, a_8, a_{10}\}, \{a_7, a_8, a_9\}, \{a_6, a_9, a_{10}\}, \{a_6, a_8, a_{10}\},$ $\{a_6, a_8, a_9\}, \{a_5, a_{10}, a_{11}\}, \{a_5, a_9, a_{11}\}, \{a_5, a_9, a_{10}\}, \{a_5, a_8, a_{11}\}, \{a_5, a_8, a_{10}\},$ $\{a_5, a_8, a_9\}, \{a_5, a_7, a_{10}\}, \{a_5, a_7, a_8\}, \{a_4, a_{10}, a_{11}\}, \{a_4, a_9, a_{11}\}, \{a_4, a_9, a_{10}\},$ $\{a_4, a_8, a_{11}\}, \{a_4, a_8, a_{10}\}, \{a_4, a_8, a_9\}, \{a_4, a_7, a_{10}\}, \{a_4, a_7, a_9\}, \{a_4, a_7, a_8\},$ $\{a_4, a_6, a_{10}\}, \{a_4, a_6, a_8\}, \{a_4, a_5, a_{10}\}, \{a_4, a_5, a_9\}, \{a_4, a_5, a_8\}, \{a_3, a_9, a_{10}\},$ $\{a_3, a_8, a_{10}\}, \{a_3, a_8, a_9\}, \{a_2, a_9, a_{10}\}, \{a_2, a_8, a_{10}\}, \{a_2, a_8, a_9\}, \{a_1, a_{10}, a_{11}\},$ $\{a_1, a_9, a_{10}\}, \{a_1, a_8, a_{10}\}, \{a_1, a_8, a_9\}, \{a_1, a_5, a_{10}\}, \{a_1, a_5, a_8\}, \{a_1, a_4, a_{10}\},$ $\{a_1, a_4, a_9\}, \{a_1, a_4, a_8\}$

STEPE. According to the maximum recovery reliability requirement and the recovery cost limit c , make decision among all recovery strategies and calculate resilience φ by Equation (5).

With recovery cost limit c , $R_r(S_i) = R_r(A_r(S_i)) = R_r\{A_r(S_i)\} \max\{R_r(O_{i,j}), R_r'(O_{i,j})\}$ for $C(O_{i,j})$ or $C'(O_{i,j}) < c$, $j = 1, 2, \dots, 2^{|A_d(S_i)|} - 1$ for $i = 1, 2, \dots, 2^{11} - 1$. And the corresponding cost is $C(S_i)$, the resilience φ is calculated by Equation (5). Following is the discussion. If $c = 10$, the network resilience $\varphi = 0.14223$, and the recovery strategies for the network back to the original state (i.e., $I = 1$) for all arc sets are listed in Table 11. With $c = 10$, only arc sets with one, two and three arcs can have the network full recovery to the original state. And for $c = 20$ and $c = 30$, $\varphi = 0.66056$ and $\varphi = 0.89646$, respectively. Moreover, range of c for the distinct number of failed arcs and corresponding φ are listed in Table 12, Table 13, and Figure 6 at this final analysis stage.

Table 11. $I=1$ for all arc sets with recovery cost limit $c=10$.

$ A_d(S_i) $	$A_d(S_i)$
1	$\{a_4\}, \{a_8\}, \{a_9\}, \{a_{10}\}$
2	$\{a_{10}, a_{11}\}, \{a_9, a_{11}\}, \{a_9, a_{10}\}, \{a_8, a_{11}\}, \{a_8, a_{10}\}, \{a_8, a_9\},$ $\{a_7, a_{10}\}, \{a_7, a_9\}, \{a_7, a_8\}, \{a_6, a_{10}\}, \{a_6, a_8\}, \{a_5, a_{10}\}, \{a_5, a_9\},$ $\{a_5, a_8\}, \{a_4, a_{10}\}, \{a_4, a_9\}, \{a_4, a_8\}, \{a_1, a_{10}\}, \{a_1, a_8\}$
3	$\{a_8, a_9, a_{11}\}, \{a_7, a_8, a_{11}\}, \{a_7, a_8, a_9\}, \{a_6, a_8, a_9\}, \{a_5, a_9, a_{11}\},$ $\{a_5, a_8, a_{11}\}, \{a_5, a_8, a_9\}, \{a_5, a_7, a_8\}, \{a_4, a_9, a_{11}\}, \{a_4, a_8, a_{11}\},$ $\{a_4, a_7, a_9\}, \{a_4, a_6, a_8\}, \{a_4, a_5, a_9\}, \{a_4, a_5, a_8\}, \{a_3, a_8, a_9\},$ $\{a_1, a_8, a_9\}, \{a_1, a_5, a_8\}, \{a_1, a_4, a_9\}, \{a_1, a_4, a_8\}$

Table 12. Range of c for the distinct number of failed arcs from Table 8.

$ A_d(S_i) $	Max	Min	$ A_d(S_i) $	Max	Min
1	6.5	1.8	7	29.8	16.7
2	12.5	3.8	8	32.7	20.1
3	17.0	6.2	9	35.5	24.1
4	20.6	8.7	10	38.0	29.1
5	23.8	11.3	11	40.4	35.1
6	26.8	14.0	-	-	-

Table 13. Corresponding φ for c in Table 12.

c	φ	c	φ	c	φ
1.8	0.00152	16.7	0.48092	29.8	0.89457
3.8	0.01506	17.0	0.49948	32.7	0.90306
6.2	0.04241	20.1	0.66486	35.1	0.90368
6.5	0.04841	20.6	0.68922	35.5	0.90399
8.7	0.10213	23.8	0.80596	38.0	0.90399
11.3	0.19840	24.1	0.81389	40.4	0.90399
12.5	0.25163	26.8	0.86993	-	-
14.0	0.33117	29.1	0.89136	-	-

Figure 6. φ for different c from Table 13.

In Table 13 and Figure 6, φ is smaller than 0.2 when c is between 0 and 10; φ has obviously growth from 0.2 to 0.8 when c is increased from 10 to 25; φ grows slowly and approach to 0.9 when c is greater than 25. For this network, our opinion is to choose the recovery strategy that c equals around 30 to obtain the high network resilience and have the efficient recovery cost.

7. Conclusions

This paper presents a novel resilience assessment method called the BAT-based resilience assessment that employs a binary-addition tree to find all the state vectors of a network with its original, disrupted, and recovered status. In the proposed BAT-based resilience assessment, the PLSA is applied to verify the connected state vectors, reduction methods are utilized to release the computational load. According to these connected vectors, the original network reliability is calculated and the network vulnerability, network recoverability, resilience-based component importance measure, and network resilience are assessed comprehensively.

The network resilience of an example and a wildfire network with wireless sensors are evaluated in this study. The process of resilience assessment demonstrates the different network status from the original, the disrupted, to the recovered one with different evaluation phases. Resilience-based component importance measure and network resilience, two important issues in resilience, are considered in this paper. For the resilience-based component importance measure, for example, refer to the importance of arc sets with different number of failed arcs, the decision makers can formulate relevant strategies such as having more inventory of or more attentions on these vulnerable components. Moreover, tuning routing protocols or adapting these vulnerable components also could help the decision makers to overcome all these related issues from a practical point of view.

For the network resilience considered with the recovery cost limit, the decision makers can decide the level of network resilience and budget the reasonable corresponding recovery cost.

In this work, all nodes are completely reliable and only arcs have possibility to fail in the binary-state network. However, there are more complicated network systems in the world than the binary-state network with perfect nodes and these complex systems might need to utilize advanced models such as hidden Markov models or in a more complex and robust approach: percolation theory. It is necessary to improve the computational efficiency and extend our proposed BAT-based resilience assessment for more complex networks.

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

Funding: This research received no external funding.

Acknowledgments: We acknowledge the editor and the referees for their constructive comments and recommendations and the support of National Tsing Hua University.

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

References

1. Hsieh, C.-H.; Feng, C.-M. The highway resilience and vulnerability in Taiwan. *Transp. Policy* **2018**. [\[CrossRef\]](#)
2. Markolf, S.A.; Hoehne, C.; Fraser, A.; Chester, M.V.; Underwood, B.S. Transportation resilience to climate change and extreme weather events—Beyond risk and robustness. *Transp. Policy* **2019**, *74*, 174–186. [\[CrossRef\]](#)
3. Trotter, M.J.; Ivory, V. A systems-based framework as an engagement tool: Adaptation for insight on transport network resilience. *Case Stud. Transp. Policy* **2019**, *7*, 167–177. [\[CrossRef\]](#)
4. Avci, O.; Ozbulut, O. Threat and vulnerability risk assessment for existing subway stations: A simplified approach. *Case Stud. Transp. Policy* **2018**, *6*, 663–673. [\[CrossRef\]](#)
5. Calvert, S.C.; Snelder, M. A methodology for road traffic resilience analysis and review of related concepts. *Transp. A Transp. Sci.* **2018**, *14*, 130–154. [\[CrossRef\]](#)
6. Perera, C.; Liu, C.H.; Jayawardena, S. The emerging internet of things marketplace from an industrial perspective: A survey. *IEEE Trans. Emerg. Top. Comput.* **2015**, *3*, 585–598. [\[CrossRef\]](#)
7. Kabadurmus, O.; Smith, A.E. Evaluating reliability/survivability of capacitated wireless networks. *IEEE Trans. Reliab.* **2017**, *67*, 26–40. [\[CrossRef\]](#)
8. Hajian, M.; Melachrinoudis, E.; Kubat, P. Modeling wildfire propagation with the stochastic shortest path: A fast simulation approach. *Environ. Model. Softw.* **2016**, *82*, 73–88. [\[CrossRef\]](#)
9. Liu, X.; Shahidehpour, M.; Li, Z.; Liu, X.; Cao, Y.; Bie, Z. Microgrids for enhancing the power grid resilience in extreme conditions. *IEEE Trans. Smart Grid* **2016**, *8*, 589–597.
10. Xu, Z.; Ramirez-Marquez, J.E.; Liu, Y.; Xiahou, T. A new resilience-based component importance measure for multi-state networks. *Reliab. Eng. Syst. Saf.* **2020**, *193*, 106591. [\[CrossRef\]](#)
11. Rocco, C.M.; Barker, K.; Moronta, J.; Ramirez-Marquez, J.E. Community detection and resilience in multi-source, multi-terminal networks. *Proc. Inst. Mech. Eng. Part O J. Risk Reliab.* **2018**, *232*, 616–626. [\[CrossRef\]](#)
12. Hosseini, S.; Barker, K.; Ramirez-Marquez, J.E. A review of definitions and measures of system resilience. *Reliab. Eng. Syst. Saf.* **2016**, *145*, 47–61. [\[CrossRef\]](#)
13. Pant, R.; Barker, K.; Ramirez-Marquez, J.E.; Rocco, C.M. Stochastic measures of resilience and their application to container terminals. *Comput. Ind. Eng.* **2014**, *70*, 183–194. [\[CrossRef\]](#)
14. Rose, A. Economic resilience to natural and man-made disasters: Multidisciplinary origins and contextual dimensions. *Environ. Hazards* **2007**, *7*, 383–398. [\[CrossRef\]](#)
15. Youn, B.D.; Hu, C.; Wang, P. Resilience-driven system design of complex engineered systems. *J. Mech. Des.* **2011**, *133*, 101011. [\[CrossRef\]](#)
16. Whitton, J.C.; Ramirez-Marquez, J.E. Resiliency as a component importance measure in network reliability. *Reliab. Eng. Syst. Saf.* **2009**, *94*, 1685–1693. [\[CrossRef\]](#)

17. Barker, K.; Ramirez-Marquez, J.E.; Rocco, C.M. Resilience-based network component importance measures. *Reliab. Eng. Syst. Saf.* **2013**, *117*, 89–97. [[CrossRef](#)]
18. Yeh, W.-C. Novel Binary-Addition Tree Algorithm (BAT) for Binary-State Network Reliability Problem. *arXiv* **2020**, arXiv:2004.08238.
19. Wang, Y.; Xing, L.; Wang, H.; Coit, D.W. System reliability modeling considering correlated probabilistic competing failures. *IEEE Trans. Reliab.* **2017**, *67*, 416–431. [[CrossRef](#)]
20. Yeh, W.-C.; Wei, S.-C. Economic-based resource allocation for reliable Grid-computing service based on Grid Bank. *Future Gener. Comput. Syst.* **2012**, *28*, 989–1002. [[CrossRef](#)]
21. Yeh, W.-C. A Novel Generalized Artificial Neural Network for Mining Two-Class Datasets. *arXiv* **2019**, arXiv:1910.10461.
22. Yeh, W.-C.; Lin, J.-S. New parallel swarm algorithm for smart sensor systems redundancy allocation problems in the Internet of Things. *J. Supercomput.* **2018**, *74*, 4358–4384. [[CrossRef](#)]
23. Aven, T. Some considerations on reliability theory and its applications. *Reliab. Eng. Syst. Saf.* **1988**, *21*, 215–223. [[CrossRef](#)]
24. Colbourn, C.J. *The Combinatorics of Network Reliability*; Oxford University Press, Inc.: Oxford, UK, 1987.
25. Levitin, G. *The Universal Generating Function in Reliability Analysis and Optimization*; Springer: New York, NY, USA, 2005; Volume 6.
26. Lee, C.-Y. Representation of switching circuits by binary-decision programs. *Bell Syst. Tech. J.* **1959**, *38*, 985–999. [[CrossRef](#)]
27. Forghani-Elahabad, M.; Kagan, N. Assessing reliability of multistate flow networks under cost constraint in terms of minimal cuts. *Int. J. Reliab. Qual. Saf. Eng.* **2019**, *26*, 1950025. [[CrossRef](#)]
28. Yeh, W.-C. A simple algorithm to search for all MCs in networks. *Eur. J. Oper. Res.* **2006**, *174*, 1694–1705. [[CrossRef](#)]
29. Yeh, W.-C. New method in searching for all minimal paths for the directed acyclic network reliability problem. *IEEE Trans. Reliab.* **2016**, *65*, 1263–1270. [[CrossRef](#)]
30. Niu, Y.; Gao, Z.; Sun, H. An improved algorithm for solving all d -MPs in multi-state networks. *J. Syst. Sci. Syst. Eng.* **2017**, *26*, 711–731. [[CrossRef](#)]
31. Hao, Z.; Yeh, W.-C.; Wang, J.; Wang, G.-G.; Sun, B. A quick inclusion-exclusion technique. *Inf. Sci.* **2019**, *486*, 20–30. [[CrossRef](#)]
32. Bryant, R.E. Graph-based algorithms for boolean function manipulation. *Comput. IEEE Trans.* **1986**, *100*, 677–691. [[CrossRef](#)]
33. Felici-Castell, S.; Navarro, E.A.; Pérez-Solano, J.J.; Segura-García, J.; García-Pineda, M. Practical considerations in the implementation of collaborative beamforming on wireless sensor networks. *Sensors* **2017**, *17*, 237. [[CrossRef](#)] [[PubMed](#)]
34. Noriega-Linares, J.E.; Rodriguez-Mayol, A.; Cobos, M.; Segura-Garcia, J.; Felici-Castell, S.; Navarro, J.M. A wireless acoustic array system for binaural loudness evaluation in cities. *IEEE Sens. J.* **2017**, *17*, 7043–7052. [[CrossRef](#)]
35. Yeh, W.-C. A simple heuristic algorithm for generating all minimal paths. *IEEE Trans. Reliab.* **2007**, *56*, 488–494. [[CrossRef](#)]
36. Levitin, G.; Xing, L.; Dai, Y. Optimal Spot-Checking for Collusion Tolerance in Computer Grids. *IEEE Trans. Dependable Secur. Comput.* **2017**, *16*, 301–312. [[CrossRef](#)]
37. Yeh, W.-C. A simple universal generating function method to search for all minimal paths in networks. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2009**, *39*, 1247–1254.