ECTI Transactions on Computer and Information Technology
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

Optimizing Incident Detection Thresholds Using the A* Algorithm: An Enhanced Approach for the California Algorithm

Korn Puangnak[1] and Manthana Tiawongsuwan[2]

ABSTRACT

This paper presents an improved version of the California Algorithm (CA), focusing on threshold selection criteria. The CA is a widely recognized incidence detection algorithm used as a benchmark for comparison with newly developed incident detection algorithms. This study proposes criteria for threshold selection in CA based on the A* algorithm, which aims to find optimal thresholds using a Performance Index (PI) as a cost function. Our proposed method reduces processing time by optimizing resource utilization and establishes a standard for threshold selection in CA for comparison and evaluation purposes. Experimental results from our proposed method demonstrate its effectiveness in reducing the complexity required to determine optimal thresholds. Optimization of the CA method using the A* algorithm results in a 98.68% reduction in the number of nodes searched compared to a Complete Search Tree (CST).

Article information:

Keywords: Incident Detection, California Algorithm, A* Algorithm, Performance Index

Article history:
Received: February 12, 2023
Revised: April 4, 2023
Accepted: September 14, 2023
Published: October 14, 2023 (Online)

DOI: 10.37936/ecti-cit.2023174.251643

1. INTRODUCTION

Nowadays, traffic management systems employed by various transportation authorities in Thailand have adopted technology to improve operational efficiency, reduce errors, and decrease reliance on human personnel. This technological integration is commonly known as the Intelligent Transport System (ITS), with Automatic Incident Detection (AID) playing a pivotal role. AID algorithms have undergone significant development to achieve superior performance, each bearing a name specific to the methodology employed. In the 1960s, when the first incident detection algorithm, CA, was developed [1-2], researchers traced back the origins of AID algorithms. Initially designed for implementation in the Los Angeles freeway and surveillance control center, CA underwent subsequent adaptations and refinements, resulting in sequentially numbered iterations. Today, this algorithm's latest version is called CA10 [1]. In 1978, Arulampalam et al. introduced the Bayesian algorithm [3], which employs probability values to identify changes in incident signals, utilizing the same incident detection data as CA. In 1979, Collins presented the High Occupancy Incident Detection Algorithm (HIOCC) [4], which utilizes signal filtering techniques to preprocess data before incident determination. In 1983, Collins introduced the Pattern Recognition Algorithm (PATREG), incorporating considerations for variations in vehicle speeds across traffic lanes. In 1989, Persaud and Hall proposed an algorithm [4] that determined thresholds (TH) based on traffic flow, occupancy rates, and vehicle speeds, graphically representing these relationships.

Further innovations occurred in 2013 with the application of fuzzy logic techniques [5] and in 2014 with the adoption of the Discrete Wavelet Transform (DWT) [6] by Jaraspat, along with various modern approaches. Specific environmental conditions and limitations often dictate the choice of AID algorithm [7]. Based on extensive studies and a comprehensive literature review, CA emerges as a preferred candidate for use as a benchmark in comparing performance with new algorithms [8-9]. In 2020, Evans et al. [10] introduced the Road Cast Incident Detection (RCID) system, aimed at enhancing the efficiency of incident detection methods by categorizing incident patterns according to contextual factors, such as weather conditions like rain or smog.

In employing the CA algorithm for incident detection, the decision-making process's efficacy is con-

[1,2] The authors are with Faculty of Engineering, Rajamangala University of Technology Phra Nakhon, Thailand, E-mail: Korn.p@rmutp.ac.th and Manthana.t@rmutp.ac.th

tingent upon utilizing a threshold value of three distinct levels. These threshold values can be obtained from the analysis of occupancy data, comparing data between the upstream and downstream sections conducted by specialists. It is necessary to perform tests on all cases with the learning dataset to obtain precise and reliable threshold values. Various factors, including environmental conditions, lane configurations, traffic volume variations, and local driver behavior, determine these thresholds. Consequently, testing all possible scenarios on the continuously expanding learning dataset requires significant time and resources.

As mentioned, one can effectively test CA's threshold values using the A* algorithm. This algorithm was first introduced in 1968 by Peter et al. [11] at Stanford. The algorithm employs heuristics to guide its search and achieve optimal performance. The search algorithm is the method used to establish the cost function and estimate and select paths for modifying the threshold values, including deciding how to access the target to confirm the effectiveness of the findings, which will be discussed in detail afterward.

2. CALIFORNIA ALGORITHM INCIDENT DETECTION

Among the Comparative algorithms based on occupancy data (OCC), CA [1-2] serves as an incident detection algorithm, utilizing the traffic sensor's measurements at the upstream $OCC_{up}(t)$ and downstream $OCC_{dw}(t)$ levels. Let t be the occupancy of the traffic data measurement, and $t-1$ be the occupancy of the traffic quantity two minutes before [12]. When the traffic measurement ratio is evaluated from the average value every minute, OCC_{DF} is the difference between the upstream and downstream occupancy ratio at time t used to check against the first condition calculated as in Equation 1. OCC_{RDF} is the quotient between Equation 1 and the ratio of the upstream occupancy at time t used to check against the second condition calculated as in Equation 2. OCC_{TD} is the quotient between the difference of the ratio of the downstream occupancy at time $t-1$ and t and the ratio of the upstream occupancy at time t used to check against the third condition calculated as in Equation 3.

$$OCC_{DF} = OCC_{up}(t) - OCC_{dw}(t) \tag{1}$$

$$OCC_{RDF} = \frac{OCC_{DF}}{OCC_{up}(t)} \tag{2}$$

$$OCC_{TD} = \frac{OCC_{dw}(t-1) - OCC_{dw}(t)}{OCC_{dw}(t-1)} \tag{3}$$

The three steps for the incident consideration conditions are Step 1: Compare the result value between OCC_{DF} and T_1, where T_1 is the threshold. If the result of Equation 1 exceeds T_1, then proceed to step 2. If the condition is false, the result is typical. Step 2: Compare the value of the results between OCC_{RDF} and T_2, where T_2 is the threshold. If the result of Equation 2 exceeds T_2, then proceed to step 3. If the condition is false, the result is typical. Step 3: If the results of Equation 3 exceed T_3, the incident happens. The result is typical if the condition is false, as shown in Fig.1.

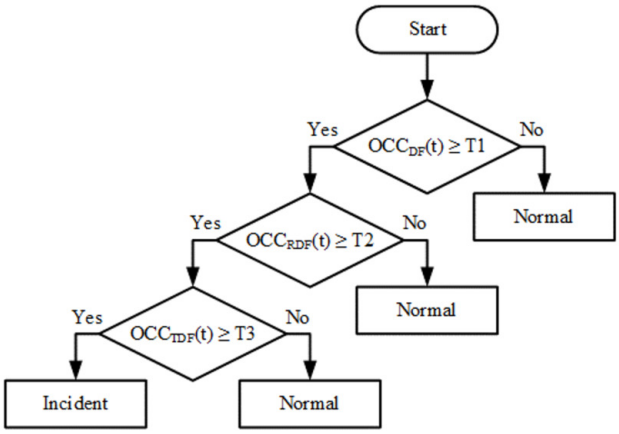

Fig.1: *CA Condition of the CA.*

Later, CA underwent significant improvements to enhance its task performance. Researchers named the algorithm and its versions sequentially as they developed them. Currently, the latest iteration of this algorithm is known as CA#10 [12]. Based on previous research findings, CA#7 has demonstrated its capability to exhibit optimal incident detection performance compared to its predecessors, as illustrated in Fig.2.

They assigned the term D_{OCC}, representing the occupancy ratio at the time 't' equal to $OCC_{dw}(t)$. CA#7 categorizes incident patterns based on the system's state. Specifically, when the state is 0, it indicates a normal condition. If the state is 1, it suggests an incident might occur. When the state is 2, it signifies that an incident has occurred. Finally, when the state is 3, the incident is ongoing.

The effectiveness of the algorithm's performance depends significantly on the predefined threshold used to determine specific conditions [13]. To ensure optimal performance, the algorithm responsible for establishing the threshold value must be capable of thoroughly examining the threshold under various scenarios. The predefined threshold value encompasses a range from the minimum to the maximum values that could occur within any condition interval at each step, as illustrated in Fig.3.

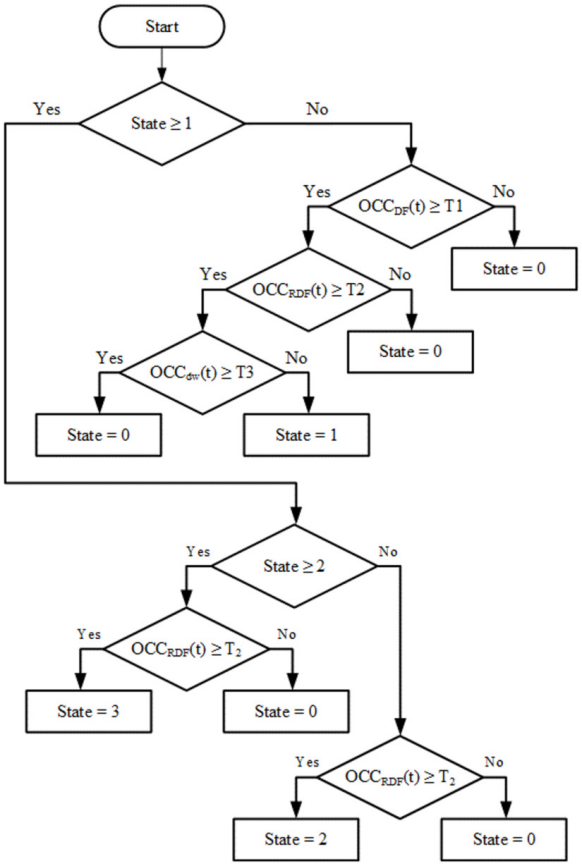

Fig.2: *Conditions of the CA#7.*

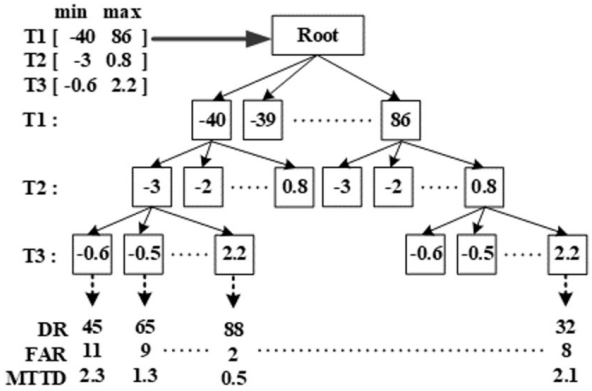

Fig.3: *Considering the incident detection algorithm based on the threshold value adjusting test in any cases.*

3. PERFORMANCE MEASUREMENT

The performance measurement of the incident detection algorithms can be considered from the result data of the three tests [8]. The three tests consist of the Detection Rate (DR), the False Alarm Rate (FAR), and the Mean Time To Detect (MTTD). DR is the number of detected incidents, defined as a percentage of accuracy per the sampling time interval calculated as in Equation 4. FAR value is the number of error detection intervals. It is a percentage of daily errors [14], calculated as in Equation 5. MTTD is the average time to detect over 'n' incidents. Equation 6 defines MTTD as a second calculation.

$$DR = \frac{D_n}{D_t} \times 100 \tag{4}$$

Equation 4: Given DR, the number of detected incidents. The definition of DR involves expressing accuracy as a percentage per sampling time interval. The D_n is the number of correct incident detections defined as times. The D_t is the total number of actual incidents defined as times.

$$FAR = \frac{N_f \times N_h}{N_t} \times 24 \times 100 \tag{5}$$

Equation 5: Given FAR, the number of error detection intervals. The definition of FAR expresses it as a percentage. The N_f is the number of the FAR defined as times. The N_h is the number of declared incident alarms (including all correct and false alarms) defined as times per hour. The N_t is the total number of actual incidents defined as times.

$$MTTD = \frac{1}{N} \sum_{i=1}^{N} (T_{IA} - T_{IO}) \tag{6}$$

Equation 6: Given MTTD, the average time to detect over 'n' incidents. The definition of mean time to detect designates it as a second. N is the number of the actual incidents defined as times. T_{IA} is the incident detection time interval defined as a second. T_{IO} is the time interval of the actual incident defined as a second.

To effectively assess the algorithm's performance above, it is imperative to conduct evaluations across a broad spectrum of scenarios. This comprehensive testing approach underscores the algorithm's threshold value search process, which demands substantial time and resources. As the quest for a precise threshold value and the need to fortify the incident dataset for diverse learning purposes continue, we expect a significant growth in the number of learning datasets without constraints on their future proliferation.

The outcome of the algorithm performance evaluation testing yields three distinct parameters. These parameters are interrelated and address various facets of the algorithm's effectiveness. Consequently, comparing the overall algorithm's performance can take time and effort. In response to this challenge, Chung introduced the concept of evaluating overall algorithm performance in 1998, known as PI [14], calculated as outlined in Equation 7.

$$PI = \left[\frac{100 - DR}{100}\right]^m \times FAR^n \times MTTD^p \tag{7}$$

Equation 7: Given PI, the incident detection performance index. A lower PI value indicates better performance than a higher PI. The value of the best

performance is 0.012, 'm' is an exponent to limit the importance of the DR consideration values = 0 - 1, and 'n' is an exponent to limit the importance of the FAR consideration values = 0 - 1, and 'p' is an exponent to limit the importance of the FAR consideration values = 0 - 1. Due to the initial consideration, let m, n, and p be one based on the extra examined condition before the calculation. Let DR equal 99 if it is greater than 99. Let FAR equal to 0.01 if it is less than 0.01. Let MTTD equal to 1-time interval in data editing if it equals 0. The PI presented in the research will be used to obtain the PI measurement result based on the same limitation, and the PI value evaluation result can be shown and tested by any threshold criteria based on the learning data model, as shown in Fig.4.

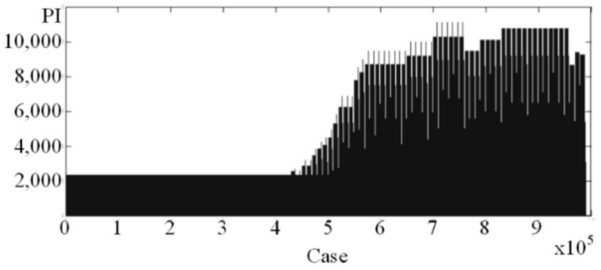

Fig.4: *PI Testing at any threshold criteria based on Learning Data Model.*

The correlation between DR, FAR, and MTTD affecting the PI value variation, as shown in Fig.5(a), reflects that when the DR value changes by 1% or the FAR value changes by 0.01% at a time, they both affect the PI value variation to the same degree. For MTTD, the effects on the PI value variation are much more precise than for DR and FAR, as shown in Fig.5(b).

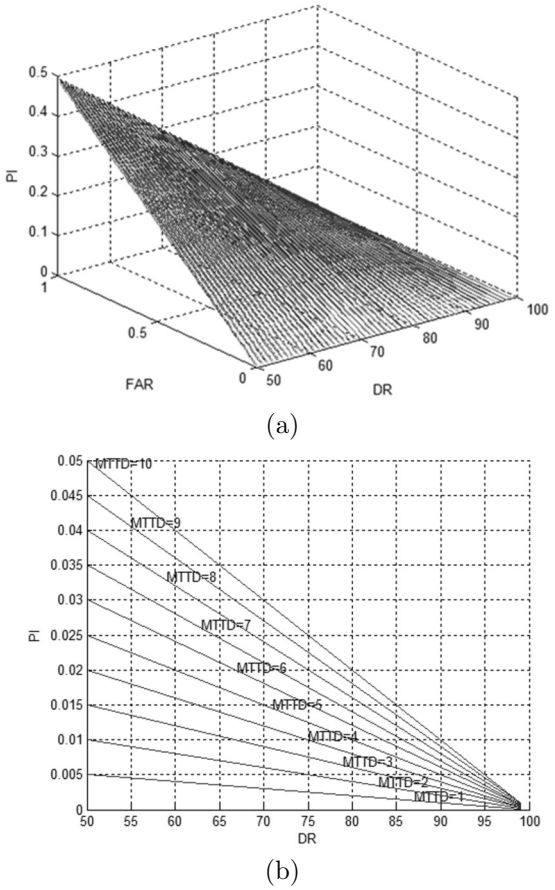

Fig.5: *Consideration of the effect of DR, FAR, and MTTD variation on PI: (a) The relation between DR & FAR when the MTTD value is steady (b) MTTD effect on PI when the FAR value is steady.*

4. A* ALGORITHM

Nowadays, the search process for alternative routes that are not limited or cannot be limited can be solved using a Heuristic search algorithm [15]. These searching algorithm groups have various types of performances, such as backtracking, beam, Bellman-Ford, best-first, bidirectional, depth-limited, Dijkstra, and the A* algorithm [16]. A* is one of the algorithms suitable for the pattern and searching goal to find the most optimal threshold value. Define this threshold value in the evaluation equation of the weight cost for each alternative path derived from the result calculated in Equation 8.

$$f(n) = g(n) + h(n) \tag{8}$$

Let $f(n)$ be the function of the total performance measurement used for the selecting search threshold process value $g(n)$ be the function of the previous searching path performance evaluation. Using the actual cost function measurement value of the previous search paths is common. Let $h(n)$ be the function of the heuristic performance evaluation value used to estimate the cost function prediction of the present path to the goal. These total performance measurement functions mentioned above are related to the search, as shown in Fig.6.

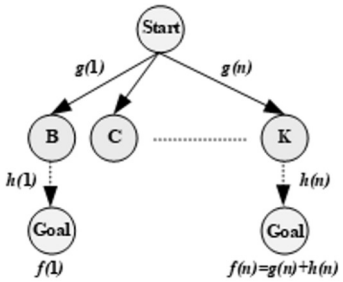

Fig.6: *The determination and the assessment process for the best search algorithm by A*.*

Considering the performance process to search for the goal, A* is profitable by estimating the PI value at present instead of finding the actual search goal

value and the minimum of the PI dataset at the initial threshold instead of the predicted route searching value that can provide the optimal threshold value of CA, as mentioned in the next section.

5. THE PROPOSED ICA (IMPROVED CA)

5.1 Relationship between Thresholds and Performance

To improve the CA threshold, we have to consider the result of the PI, which consists of threshold levels 1, 2, and 3. After the adjusted threshold test in any case, together with the traffic data in the 2-lane blocking incident model, change the data value of each level from the minimum to the maximum per every 100 number intervals. This change will likely provide 100,000,000 cases, as shown in Fig.4. To consider the changes more clearly, one must test the variation in the threshold value at each level. The threshold value of other levels not considered at that time must be equal to the minimum threshold level, as shown in Fig.7.

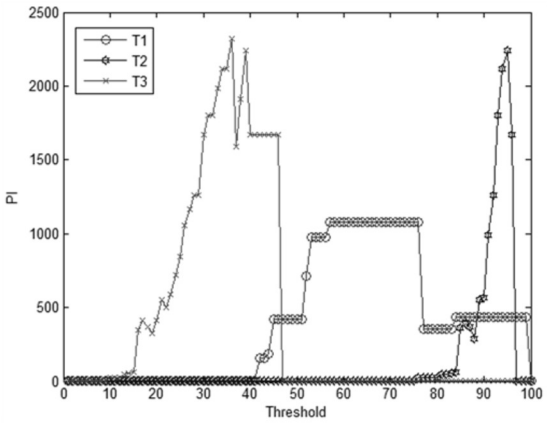

Fig.7: *The 3-level PI comparison of the test result.*

5.2 The Threshold Search Process based on A*

To get the best threshold for CA to use for incident detection, first, let the determination interval of the threshold value be 'n' for using it to create a possible path to search for the threshold value, as shown in Fig.8. We calculated the threshold value used to determine the PI from Equations 9, 10, and 11. Let min_{T1}, min_{T2}, and min_{T3} be the possible minimum threshold values for each level of the determination. Let max_{T1}, max_{T2}, and max_{T3} be the maximum threshold values for each determination.

$$T_1 = min_{T1} + \frac{i \times (max_{T1} - min_{T1})}{n} \tag{9}$$

$$T_2 = min_{T2} + \frac{j \times (max_{T2} - min_{T2})}{n} \tag{10}$$

$$T_3 = min_{T3} + \frac{k \times (max_{T3} - min_{T3})}{n} \tag{11}$$

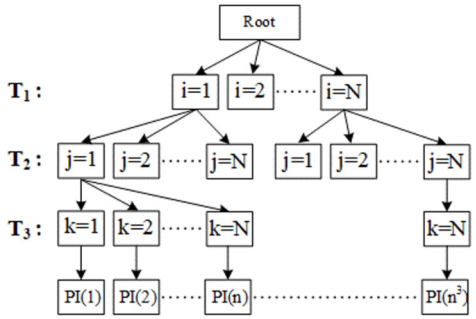

Fig.8: *Watermarked Images.*

To generate the possible threshold by presenting a range of determination 'n', there will always be an alternative to the possible determination value, including the determination index value n^3 path.

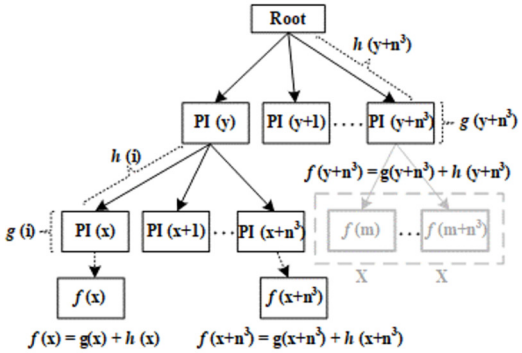

Fig.9: *Plotting graph from the new path from the threshold search value at a specific case.*

To create a path from any specific case's threshold value searching process, one can build a path for the best threshold value searching process on the graph, as shown in Fig.9. Start by setting the threshold boundary, which will be the initial data for the threshold value searching process for the specific case. When considering the node, we will construct the total paths of each round to become a baby node of the graph. For each round, the algorithm will select the last node with the most negligible $f(x)$ value to construct a baby node in order as a recursion, as shown in Fig.9.

Let $g(x)$ be the present PI value equal to the PI value of the present mentioned node calculated as Equation 12 when x is the present node, and $h(x)$ is the result change tendency for the prediction of the PI value for the PI value calculated as in Equation 13 when y is the parent node spot of the considered path. If the difference between $PI(x)$ and $PI(y)$ value is less than or equal to zero, let $h(x)$ be equal to 1. After completing the selection process of the least

$f(x)$, repeat the selection process until the search result matches the later-mentioned goal condition. Due to expanding the repeated path each time, it will stop constructing further paths. When the PI value of the total paths is equivalent to those depicted in the dotted lines, it indicates that all the pursued paths are the most optimal ones we could further explore.

$$g(x) = PI(x) \tag{12}$$

$$h(x) = \frac{1}{PI(y) - PI(x)} \tag{13}$$

The threshold search process based on the A* algorithm will generate alternative routes according to Equations 9-11, calculate the cost function according to Equations 12 and 13 of all routes, and select the route with the minimum cost function to take the range of parameters to divide into sub-paths according to Equations 9, 11, and calculate the cost function according to Equations 12, 13 repeatedly until the cost function at the same level is equal and is the lowest value.

5.3 Initial Parameter, Goal, and Related Parameter

5.3.1 Initial Parameter

To start the algorithm, set the evaluation limitation equal to the minimum and maximum value obtained from the total threshold value estimated for each rank to be a frame of the recursion task to work on a limitation that affects the algorithm performance. Then, let the initial PI value for the root reference equal the most negligible PI value from the first path construction to be the reference value for the determination of the changing tendency of the algorithm performance.

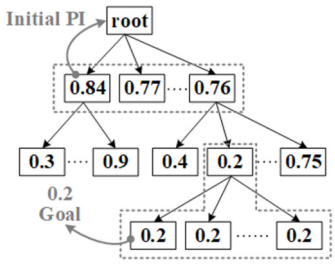

Fig.10: *The best threshold from the search tree.*

5.3.2 Initial Parameter

The threshold value searching process, particularly within the presented recursion case, concludes when the PI value, or $f(x)$, of the child nodes generated under the same parent node reaches its minimum value across all searching paths. This condition, along with their correlation, is depicted in Fig.10 and serves as the result that the algorithm provides, effectively concluding the repetitive task and delivering the algorithm's answer.

5.4 The effectiveness of an algorithm

The task of each step of the presented algorithm can explain the whole task in the pseudo-code, as shown in Fig.8. Presenting the above algorithm and the BigO algorithm performance is evaluated BigO as $O(logn)$ [15], [17], which shows better task performance than the complete search algorithm or the specific search, which has to search all the possible paths to test the finding of the desired paths and has the algorithm performance as $BigO(n^3)$. The following section shows the performance mentioned in the following experiment.

6. THE RESULT OF THE EXPERIMENT

The experiment to evaluate the task performance of the development of the threshold performance value of the CA utilizing the presented A* consists of three parts: Assumption and Initial Parameter is a definition: to specify the data detail for the experiment and the data source to make the experiment result the same standard as the previous research. The presentation of experimental results involves evaluating the presented data compared to the search for a general threshold algorithm value. Presenting the analysis results regarding the performance of the provided algorithm constitutes the completion of the experiment. These results include detailed information as outlined below:

6.1 The Limitation and Initial Parameter

Currently, the experiments to compare the effectiveness of the incident detection algorithm have many assumptions [7]. This article uses traffic data generated from the AIMSUN application in the experiments. The conditions to create a traffic simulation model according to the parameters are in Table 1. The parameter is according to the research of Binglei et al. [18]. This research presents a comparison of the effectiveness of many incident detection algorithms. Fig.11 shows the traffic data. Finally, we will test the traffic data for the effectiveness of the algorithms proposed in MATLAB.

6.2 The Experiment & Evaluation

The previously stated limitations and initial conditions provide the foundation for conducting experiments with the method provided to compare it with the complete search algorithm or threshold value searching in different scenarios. In the initial stage, we utilize two sets of data samples to ascertain the threshold value in various scenarios to identify the number of paths that result in the maximum PI value

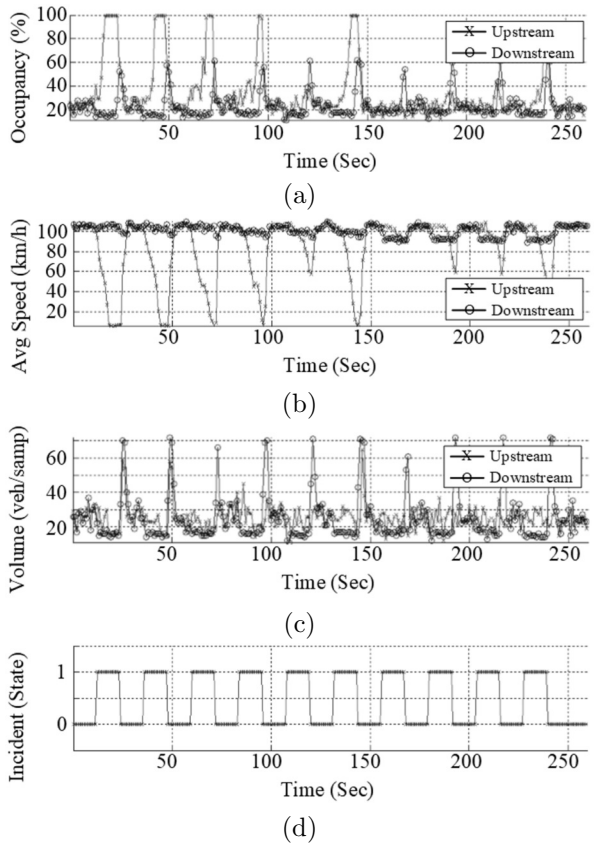

Fig.11: *The traffic data of 1 lane blocking: (a) occupancy data, (b) average speed data, (c) volume data, (d) incident event.*

Table 1: *The parameter of traffic simulation for the experiment.*

Parameter	Value	Unit
Scenario and Environment		
Roadways	3	Lane
Road Length	500	m
Width of Lane	3.5	m
Traffic Condition		
Average Speed	105	km/h
Flow Rate	3,000	veh/h
Sampling Rate	30	s
Event Base		
Incidents Occur	10	times
Incident Start	360, 1080, 1800, 2520, 3240, 3960, 4680, 5400, 6420, 6840	min
Incident Duration	360	s
Location (Upstream)	150, 150, 150, 270, 270, 270, 270, 340, 340, 340	m
Simulation Time	7,800	s
Amount of Events	8	cases
Incident Pattern	1, 2, 3 (Closure)	block

for the CA. The primary aim is to attain the maximum DR, FAR below 25%, and MTTD below 5 minutes. Let 'n' represent the number of paths in each layer, whereas n3 represents the overall number of paths generated. The findings obtained from the experiment are accurately presented in Table 2, revealing significant insight into the algorithm's efficacy across different conditions.

Table 2: *The experiment result of the path constructing for the threshold searching at any cases on a sample data set.*

n	n^3 (Node)	PI	DR	FAR	MTTD
Data Set Sample 1					
10	1,000	31.92	97.50	38.42	3.00
20	8,000	28.78	97.50	34.64	3.00
30	27,000	28.40	96.67	25.64	3.00
40	64,000	19.96	97.50	24.03	3.00
50	**125,000**	**19.54**	**97.50**	**23.53**	**3.00**
100	1,000,000	19.54	97.50	23.53	3.00
Data Set Sample 2					
10	1,000	43.56	97.50	52.43	3.00
20	8,000	43.40	97.50	52.24	3.00
30	27,000	43.56	97.50	52.43	3.00
40	64,000	43.07	97.50	51.85	3.00
50	125,000	43.40	97.50	52.24	3.00
100	**1,000,000**	**42.91**	**97.50**	**51.65**	**3.00**

The experiment involving path construction for threshold determination within dataset Sample 1 underscores the significance of interval division in achieving the desired threshold value. In this particular experiment, the dataset comprises 50 intervals per rank. We initially divide each rank into ten intervals to assess the experiment's outcomes. As we increment the number of intervals, the results demonstrate a consistent decrease in the PI value. This trend suggests that a finer division of intervals leads to a more precise and valid threshold. The PI value stabilizes after reaching a 50-interval division per rank, yielding results equivalent to those obtained with a 100-interval division per rank.

Conversely, the experiment conducted with dataset sample 2 reveals that the most effective interval division for this dataset sample comprises 100 intervals per rank. Thus, determining the optimal interval division level for the desired threshold value in each learning dataset necessitates varying interval divisions based on comparative experiment results, tailoring the approach to the specific characteristics of the dataset and algorithm in question.

The experiment for the presented algorithm performance evaluations, if 'n' equals 5, compared to the complete search algorithms for the best threshold value searching for the incident detection CA and CA#7, can present the experiment result as shown in Table 3.

The experiment result shown in Table 3 can be considered in a graph, as shown in Fig.12. This graph

Table 3: *The experiment result to compare the threshold value between the complete search algorithm and the A* algorithm..*

Set	CA(Node)			CA #7 (Node)		
	PI	CST	A*	PI	CST	A*
1-lane blocking incident						
1	29	8 K	0.2 K	43	125 K	2.5 K
2	24	125 K	0.3 K	14	27 K	1.2 K
Set	CA(Node)			CA #7 (Node)		
	PI	CST	A*	PI	CST	A*
3	79	125 K	0.8 K	1	64 K	1.8 K
4	42	1 M	3 K	2	1 K	0.2 K
5	191	1 M	17 K	2	1 K	0.1 K
6	58	1 K	1.5 K	1	64 K	0.5 K
7	44	64 K	0.3 K	2	1 K	0.8 K
8	127	1 K	17 K	1	27 K	1.2 K
avg	**-**	**290 K**	**3 K**	**1**	**38 K**	**1.0 K**
2-lane blocking incident						
1	19	125 K	2 K	8	64 K	4 K
2	19	1 M	1.2 K	7	125 K	2.8 K
3	15	1 M	1 K	9	64 K	1.2 K
4	46	1 M	4.8 K	46	1 M	0.1 K
5	313	1 K	0.5 K	5	64 K	2.6 K
6	16	125 K	2.6 K	10	8 K	1.8 K
7	17	1 M	8.6 K	9	1 M	12 K
8	15	1 M	3.2 K	6	8 K	0.2 K
avg	**-**	**656 K**	**3 K**	**-**	**291 K**	**3.1 K**
Lane-closure incident						
1	56	1 K	0.1 K	1	8 K	2.2 K
2	56	1 K	0.1 K	0	1 K	0.3 K
3	85	1 K	0.1 K	1	64 K	0.2 K
4	85	1 K	0.1 K	2	1 K	0.5 K
5	56	1 K	0.1 K	2	1 K	0.1 K
6	56	1 K	0.1 K	1	64 K	0.3 K
7	56	1 K	0.1 K	2	1 K	1.5 K
8	56	1 K	0.1 K	1	27 K	16.2 K
avg	**-**	**1 K**	**0.1 K**	**-**	**20 K**	**2.7 K**
total	**-**	**315 K**	**2.1 K**	**-**	**117 K**	**2.3 K**

compares the task result for the best incident detection threshold between the complete search algorithm and the presented algorithm of CA. The graph shown in Fig.13 compares the task result for the best incident detection threshold of CA#7 between the complete search algorithm and the presented algorithm.

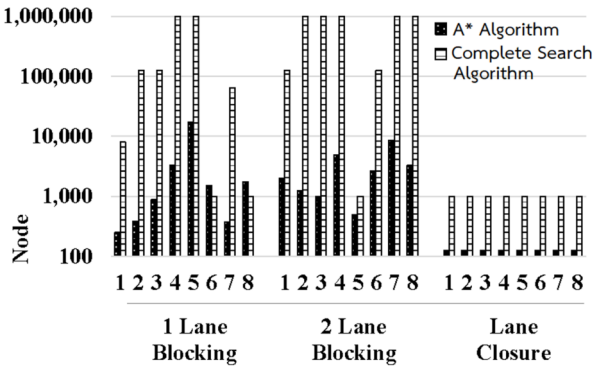

Fig.12: *The number of the paths of the threshold searching by CA.*

When comparing the presented algorithm to the entire search algorithm, the amount of data is less than average for the CA when running on the 1-lane blocking incident data set, which equals 98.9% of the data in the whole search algorithm data set. When performing on the 2-lane blocking incident data set, it equals 99.54% of the complete search algorithm data set quantity. When performing on the lane-closure incident data set, it equals 87.30% of the complete search algorithm data set quantity. The A* algorithm can reduce the number of nodes by an average of 99.33% for testing with CA methods.

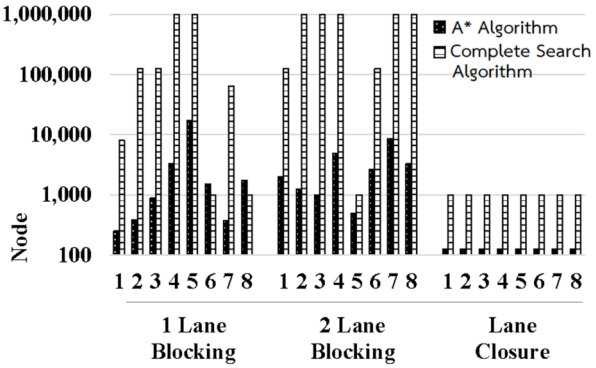

Fig.13: *Number of paths for threshold searching by CA#7.*

For CA#7, comparing the presented algorithm to less than average when performing on the 1-lane blocking incident dataset equals 97.22% of the complete search algorithm data quantity. When performing on the 2-lane blocking incident data set, it equals 98.92% of the complete search algorithm data quantity. Performing on the lane-closure incident data set equals 87.05% of the complete search algorithm data quantity. The A* algorithm can reduce the number of nodes by an average of 98.03% for testing with CA methods.

Fig.14 shows that for CA and CA#7, the number of nodes needed to find the optimal threshold is the same for both the A* and complete search algorithms. This result is accurate for all three test situations from the simulated situation data.

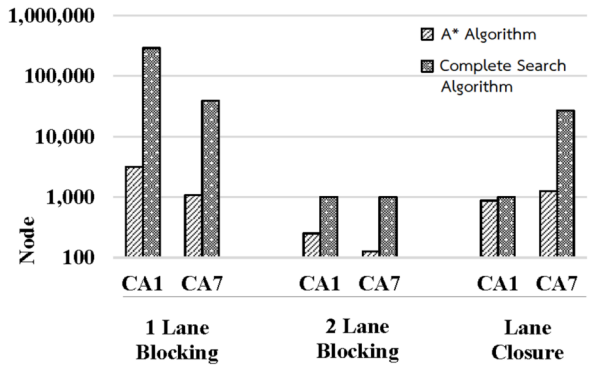

Fig.14: *Comparison of the numbers of the paths on average for the threshold searching for each situation.*

6.3 The Analysis

The experimental study result indicates that the best threshold algorithm for searching for the CA using the presented A* algorithm can help reduce the number of inefficient paths in the search process. This algorithm helps to reduce the time and the size of the memory unit of the task during searching. The incident detection algorithm CA method that uses the A* algorithm to adjust the threshold can reduce the number of nodes needed to find the best answer by 98.68% compared to a complete search tree. The A* algorithm reconstructs paths to find only the most likely to have the best value. This algorithm generates searching paths fewer than the method that generates all subroutes with the best results.

When employing the complete search approach, the optimal threshold search algorithm for incident detection with CA cannot accurately determine the number of interval divisions for each evaluation level. When the intervals for each level are sufficiently large to enhance efficiency and resource conservation in the determination value search, the resulting threshold value may prove inefficient, unreliable, or inappropriate. Conversely, excessive intervals yield an efficient and dependable threshold value but demand significantly more time and resources.

7. CONCLUSION

The A* algorithm presented in this paper can quickly help search for the best threshold for incident detection by the CA and CA#7 and use fewer resources than the complete search algorithm, which has to generate the whole possible path for the most efficient performance. Optimization of the CA method with the A* algorithm can reduce the number of nodes in the search by 98.68% compared to the complete search tree. Nowadays, the rate of incident data for system learning has been continually growing, so the fast threshold searching process for the precision of performance is an essential solution for applying the algorithm to the actual task system in the future.

Based on the experimental results, the A* algorithm is the most optimal alternative for seeking the threshold value. However, this presented algorithm must be tested for threshold value searching about a lot of data and other practical and reliable incident detection algorithms to make the algorithm standard for considering the best threshold value for all algorithms to compare the performance of each incident detection algorithm later.

References

[1] H. J. Payne and S. C. Tignor, "Freeway incident-detection algorithms based on decision trees with states," *Transportation Research Record*, no. 682, pp. 30-37, 1978.

[2] K. N. Balke, "An Evaluation of Existing Incident Detection Algorithms," *Interim Report*, no. FHWA/TX-93/1232-20, 1993.

[3] M. S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," in *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174-188, Feb. 2002.

[4] J. A. Barria and S. Thajchayapong, "Detection and Classification of Traffic Anomalies Using Microscopic Traffic Variables," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 3, pp. 695-704, Sept. 2011.

[5] J. La-inchua, S. Chivapreecha and S. Thajchayapong, "A new system for traffic incident detection using fuzzy logic and majority voting," *2013 10th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, Krabi, Thailand, pp. 1-5, 2013.

[6] J. La-inchua, S. Chivapreecha and S. Thajchayapong, "Fuzzy logic-based traffic incident detection system with discrete wavelet transform," *2014 11th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, Nakhon Ratchasima, Thailand, pp. 1-6, 2014.

[7] K. Puangnak and S. Chivapreecha, "A Review Study of Incident Detection Algorithms with Performance Index Parameter," *2019 16th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, Pattaya, Thailand, pp. 923-926, 2019.

[8] E. Parkany and C. Xie, "A complete review of incident detection algorithms and their deployment: what works and what does not," *Transportation Research Information Services (TRIS)*, NETCR 37, NETC 00-7, 2005. [Online]. Available:`https://onlinepubs.trb.org/onlinepubs/trispdfs/00988875.pdf`

[9] Y. J. Stephanedes and C A. P. hassiakos, "Application of filtering techniques for incident detection," *Journal of Transportation Engineering*, vol. 119, no. 1, pp. 13-26, 1993.

[10] J. Evans, B. Waterson and A. Hamilton, "A random forest incident detection algorithm that incorporates contexts," *International Journal of Intelligent Transportation Systems Research*, vol. 18, pp. 230-242, 2020.

[11] P. E. Hart, N. J. Nilsson and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," in *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100-107, July 1968.

[12] H. J. Payne, E. D. Helfenbein, and H. C. Knobel, "Development and testing of incident detec-

tion algorithms, volume 2: Research methodology and detailed results," *United States. Federal Highway Administration. Office of Research and Development*, no. FHWA-RD-76-20, 1976.

[13] K. Puangnak and S. Chivapreecha, "Comparative Study of Threshold Selection for Incident Detection based on California Algorithm," *2019 16th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, Pattaya, Thailand, pp. 911-914, 2019.

[14] E. Chung, M. Kuwahara and T. Yoshii, "Comparative Study of Freeway Incident Detection Algorithms Using Real-life Incident Data," *SEISAN KENKYU*, vol. 50, no. 9, pp. 329-332, 1998.

[15] L. W. Santoso, A. Setiawan and A. K. PRAJOGO, "Performance Analysis of Dijkstra, A* and Ant Algorithm for Finding Optimal Path Case Study: Surabaya City Map," Doctoral dissertation, Petra Christian University, Indonesian, 2010. [Online]. Available:`https://core.ac.uk/download/pdf/32452834.pdf`

[16] K. Keattisak and H. Visit, "Flight Path Planning with an Unlimited Number of Goals using a Deepening A* Search," in *Proceedings of the 10th National Computer Security Conference*, Bangkok, Thailand, 2006.

[17] S. Thajchayapong and J. A. Barria, "Anomaly detection using microscopic traffic variables on freeway segments," *Transportation Research Board of the National Academies*, 10-2393, 2010.

[18] X. Binglei, H. Zheng and M. Hongwei, "Fuzzy-logic-based traffic incident detection algorithm for freeway," *2008 International Conference on Machine Learning and Cybernetics*, Kunming, pp. 1254-1259, 2008.

Korn Puangnak received his Doctoral degree from Department of Sustainable Industrial Management Engineering, Rajamangala of University Technology Phra Nakhon, Bangkok, Thailand. in 2011. Currently, he is an assistant professor at Department of Computer Engineering, Faculty of Engineering, Rajamangala University of Technology Phra Nakhon, Thailand. His research falls in the domain of machine learning, and ITS (intelligent transport system).

Manthana Tiawongsuwan received the B.S. (2009) and M.S. (2014) degrees in Electrical Engineering from King Mongkut's University of Technology, North Bangkok, Thailand. She received her Ph.D. degree in Computer Science and Systems Engineering from Kyushu Institute of Technology, Japan, in 2022. She is currently working as a lecturer at Rajamangala University of Technology, Phra Nakhon, Thailand. Her research interests include video and image processing, communication, and transportation.