


## ASSESSMENT COVER SHEET

<b>Student ID number</b>	28098552		Unit Name and Code: FIT3143, Parallel computing	
			Campus: Clayton	
			Assignment Title: Assignment 2: Event detection in a fully distributed wireless sensor network -WSN	
			Name of Lecturer: Christopher Watkins	
			Name of Tutor: Mr. Omar Al-boridi	
			Tutorial Day and Time:	
			Phone Number:	
			Email Address: Msie0001@student.monash.edu	
<b>Given Name</b>	Ming Shern		Has any part of this assignment been previously submitted as part of another unit/course? <input type="checkbox"/> Yes <input type="checkbox"/> No	
			Due Date: 20/10/2019      Date Submitted: 20/10/2019 All work must be submitted by the due date. If an extension of work is granted this must be specified with the signature of the lecturer/tutor. Extension granted until (date) _____ Signature of lecturer/tutor _____ Please note that it is your responsibility to retain copies of your assessments.	
			<b>Intentional plagiarism or collusion amounts to cheating under Part 7 of the Monash University (Council) Regulations</b>  <b>Plagiarism:</b> Plagiarism means taking and using another person's ideas or manner of expressing them and passing them off as one's own. For example, by failing to give appropriate acknowledgement. The material used can be from any source (staff, students or the internet, published and unpublished works).  <b>Collusion:</b> Collusion means unauthorised collaboration with another person on assessable written, oral or practical work and includes paying another person to complete all or part of the work.  Where there are reasonable grounds for believing that intentional plagiarism or collusion has occurred, this will be reported to the Associate Dean (Education) or delegate, who may disallow the work concerned by prohibiting assessment or refer the matter to the Faculty Discipline Panel for a hearing.	
<b>Family name</b>	Siew		<b>Student Statement:</b> <ul style="list-style-type: none"> <li>I have read the university's Student Academic Integrity <a href="#">Policy</a> and <a href="#">Procedures</a>.</li> <li>I understand the consequences of engaging in plagiarism and collusion as described in Part 7 of the Monash University (Council) Regulations <a href="http://adm.monash.edu/legal/legislation/statutes">http://adm.monash.edu/legal/legislation/statutes</a></li> <li>have taken proper care to safeguard this work and made all reasonable efforts to ensure it could not be copied.</li> <li>No part of this assignment has been previously submitted as part of another unit/course.</li> <li>I acknowledge and agree that the assessor of this assignment may for the purposes of assessment, reproduce the assignment and:               <ul style="list-style-type: none"> <li>provide to another member of faculty and any external marker; and/or</li> <li>submit it to a text matching software; and/or</li> <li>submit it to a text matching software which may then retain a copy of the assignment on its database for the purpose of future plagiarism checking.</li> </ul> </li> <li>I certify that I have not plagiarised the work of others or participated in unauthorised collaboration when preparing this assignment.</li> </ul> Signature .....  ..... Date ..... 20/10/2019 ..... * delete (iii) if not applicable	
			<i>The information on this form is collected for the primary purpose of assessing your assignment and ensuring the academic integrity requirements of the University are met. Other purposes of collection include recording your plagiarism and collusion declaration, attending to course and administrative matters and statistical analyses. If you choose not to complete all the questions on this form it may not be possible for Monash University to assess your assignment. You have a right to access personal information that Monash University holds about you, subject to any exceptions in relevant legislation. If you wish to seek access to your personal information or inquire about the handling of your personal information, please contact the University Privacy Officer: <a href="mailto:privacyofficer@adm.monash.edu.au">privacyofficer@adm.monash.edu.au</a></i>	

# Assignment 2: Event detection in a fully distributed wireless sensor network -WSN

Parallel Computing

Siew Ming Shern  
Students, School of IT  
Monash University  
Melbourne, Australia  
msie0001@student.monash.edu

**Abstract**—Event detection in a wireless sensor network is one of the useful techniques for developer to detect anomaly and supervise environment. This article illustrates innovative method to implement event detection in a wireless sensor network with parallel algorithm using **Message Passing Interface (MPI)** and encryption algorithm to secure messages between communicator and computer nodes with **Open Multi-processing (OpenMP)**. The innovative method is composed of cartesian topology which is a new communicator that split from MPI communicator to provide an architecture to all computer nodes, sliding window algorithm is used to factors in activation values from past iterations to increase rate of event detection and Caesar cipher which is form of substitution cipher is used to encrypt the message passed between computer nodes. Moreover, Caesar cipher is very effective encryption algorithm because it is easy to implement, secured and is an efficient parallel encryption algorithm. This can be evidenced that it speeds up encryption algorithm gets 24% boost and decryptions algorithm gets 12% boost speed up when 8 threads provided for parallel processing.

**Keywords**—Event detection in a wireless sensor network, MPI, OpenMP, Cartesian topology, sliding window algorithm and Caesar cipher.

## I. INTRODUCTION

Wireless sensor networks (WSNs) is referred as a network of sensor nodes that transfer the data assembled from sensor node to base station through remote connections.[1] Wireless sensor networks (WSNs) has various topologies that can act as infrastructure.[1] Message Passing

Interface (MPI) took these topologies in another level by allowing topology to function as both inter-communicator and infrastructure of wireless sensor networks which will help us conducting these assignment.

Event detection in a fully distributed wireless sensor network in the context of this assignment is each sensor nodes that has received activation values will need to notify base station nodes and every message that is communicated between nodes will have to be encrypted to secure message from intruders.

Cartesian topology is used as Inter Process Communication for Wireless sensor networks (WSNs) because MPI provided this functionality with ease to retrieve neighbor node and able to provide infrastructure for wireless sensor networks.[2] Cartesian topology in MPI will creates a Cartesian with (N x M) dimensions which composed of sensor nodes arranged in N x M cartesian and then return a new communicator which consist of sensor nodes in a cartesian topology.[2] Sliding window algorithm is employed to increase rate of event detection and Caesar cipher is also utilized to secure messages between sensor nodes. Caesar cipher will likely be able to obtains speed up boost due to its simplicity implementation.

## II. THEORETICAL SPEED UP ANALYSIS

### A. Amdahl's law

Sequential processing for encryption and decryption algorithm using Caesar cipher is analyzed to discover the theoretical speed up of the of encryption and decryption algorithm using Caesar cipher with multiple thread from OpenMP. Nonetheless, Amdahl's law which can be represented with equation (2) is used to calculate

the theoretical speed up factor when using multi thread Caesar cipher algorithm.

$$S(p) = \frac{1}{r_s + \frac{r_p}{p}} \quad (2)$$

Where

$r_p$  : Parallel portion

$r_s$  : Serial portion

$p$  : number of Threads

TABLE I THEORETICAL SPEED UP FACTOR FOR ENCRYPTION USING AMDAHL'S LAW

Number of Thread, p	2	4	6	8
Speed Up Factor, S(p)	1.9989	3.9978	5.9972	7.9938

TABLE II THEORETICAL SPEED UP FACTOR FOR DECRYPTION USING AMDAHL'S LAW

Number of Thread, p	2	4	6	8
Speed Up Factor, S(p)	1.9985	3.9963	5.9959	7.9991

Amdahl's law shows that the potential speed up from the Caesar cipher for encryption and decryption is expected to have a speed up almost equivalent to number of threads that is used to parallelized. The more threads in OpenMP is used, the faster the computational time of encryption and decryption.

### III. DESIGN OF EVENT DETECTION

Program will run in parallel and will continue only if number of processors provided is 21 but will abort if the condition is not met due to insufficient processor to continue. If condition is met, program will then continue to initialize the MPI environment which eventually creates MPI communicator for all sensor nodes. Cartesian communicator is then created on that MPI communicator and it will consist of only 20 sensor nodes because only 20 nodes is needed for 4 x 5 grids leaving 21<sup>th</sup> node being elected as base station. All sensor node in cartesian communicator

will sent and receive at same time from all its adjacent nodes. They will store all random values of neighbor nodes to window buffer and will use window sliding algorithm to play its role to detect past iteration values obtained and find all activation values in buffer. Once all activation values retrieved from buffer, it will be sent over to base station so that base station will receive the message and write message to a log file. The message passing between all nodes comes with prices because it will need to be encrypted with Caesar cipher to secure message between sensor nodes.

#### A. Cartesian topology

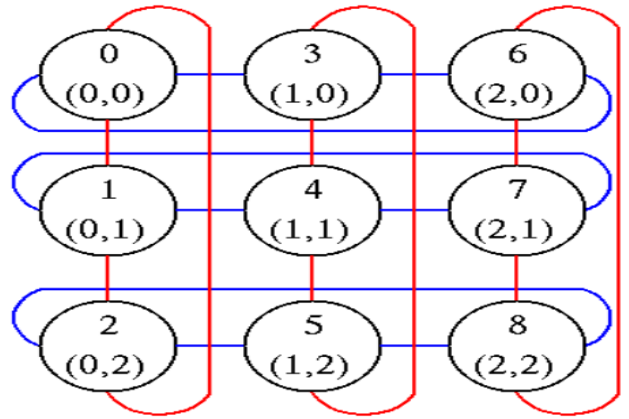


Figure 2: Cartesian topology [4]

Cartesian topology will be able to helps to ensure sensor nodes are arranged in a 4 x 5 (rectangular-shaped) grid in a communicator and the base station will likely be excluded from Cartesian communicator but will be a node in MPI communicator and not in cartesian communicator.

The blue line in figure 2 represent the ability of node to retrieve all its neighbor horizontally via a displacement setting and red line in figure 2 represent the ability of node to retrieve all its neighbor vertically via a displacement setting. Although all nodes in cartesian topology have their owns coordinate, it will likely introduce code complexity if it is used because it needs to know all neighbor coordinate and then retrieves their node but MPI\_Cart\_Shift don't need to knows its neighbor coordinate and just a specify k value displacement from a node and it can search vertically and horizontally easily.

### B. Sliding window algorithm

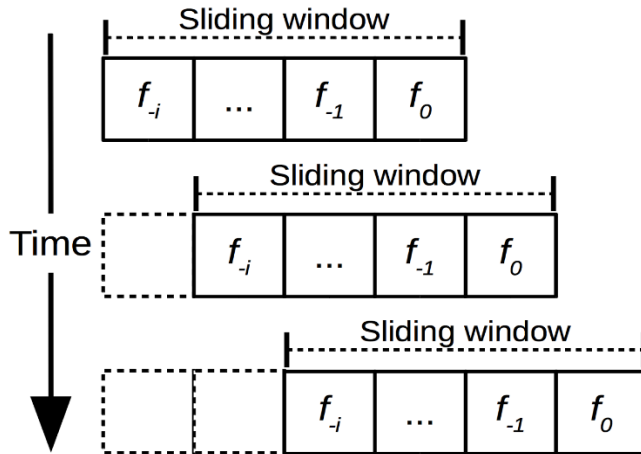


Figure 3: Sliding window algorithm [5]

Sliding window algorithm is an algorithm that make use of a buffer as window and keep tracks of important value from past iterations for decision making.[3] Thus, sliding window algorithm can be used to keep track of previous iteration for generating event with few settings needed: window have size of 3 and maximum number of neighbors of each node is 4. This explains why window buffer has size 12 because each window has at most 4 random number which are received from neighbor node. Any values that is appears at least 3 times on buffer will trigger an event. Moreover, sliding window algorithm will eventually cause event to occur more often than usual in a single iteration which results in rise of frequency of event detection.

### C. Caesar cipher

Digit	Replacement Digit	Aplhabet Ascii code
0	A	65
1	B	66
2	C	67
3	D	68
4	E	69
5	F	70
6	G	71
7	H	72
8	I	73
9	J	74

Figure 4 Caesar cipher for each digit number

The Caesar Cipher is one of the most seasoned and easiest types of encoding a message. It involves substitution cipher on each letter in the plaintext is replaced with another letter by shifting original letter either up or down based on ordinal order of Alphabet. Therefore, Caesar Cipher is regarded shift cipher. [6] Caesar Cipher in this assignment is used for numeric digits where each numeric digit is substitute with Alphabet. Based on ascii code, each digits in number will need to be shifted by 65 forward and modification has been made to output such that all replacement digits are sorted in reverse which further improve the security of message.

For example,

Encryption:

123 => BCD => DCB => 686766.

Cipher text is 686766.

Decryption:

686766 => 321 => 123

Plain text is 123.

For each thread in OpenMP:

A is ascii code of Alphabet A

Encryption mathematical expression:

message = (message\*100) + (digit % 10 + A);

Decryption mathematical expression:

message = (message\*10) + (digit % 100 - A);

When encrypting, number is read one digit at a time and message will needs two extra digits to fit new ascii code which is ranged from 65 to 74.

When decrypting, number is read two digits at a time and message will needs one extra digit to fit new integer which is ranged from 0 to 9.

Caesar cipher is used because it is a form of encryption that is difficult to read on casual inspection and integers don't normally have space like string which can make it a good secured message.[6] Beside, Caesar cipher is fairly easy to implement and no sweats break needed to implement Caesar cipher.[6] OpenMP has functionality that divides each iteration of respective threads to retrieving digits from number which is needed by Caesar cipher and change digits to Alphabet. Then, sort it in reverse order.

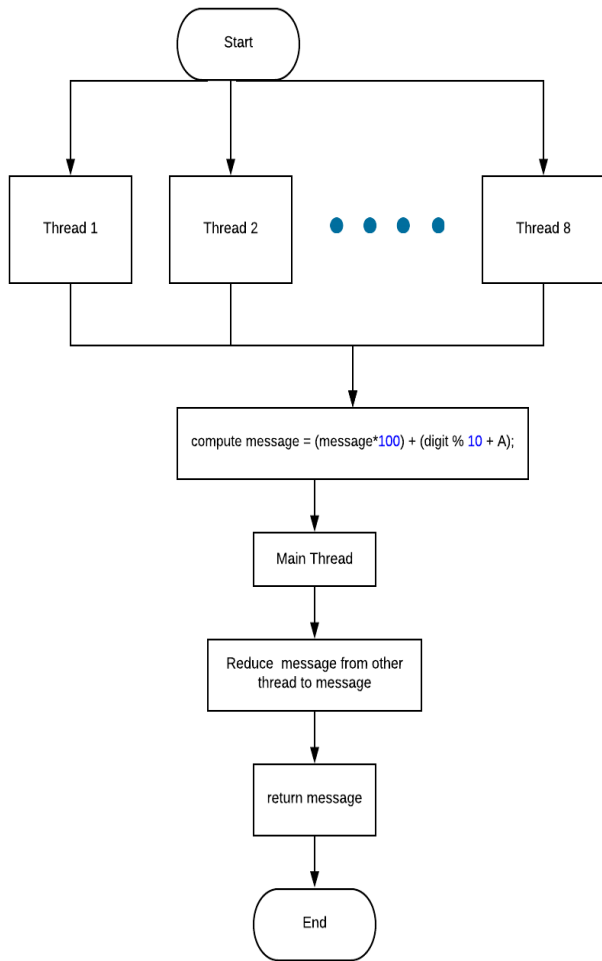


Figure 6 Flowchart for Caesar cipher with Encryption algorithm

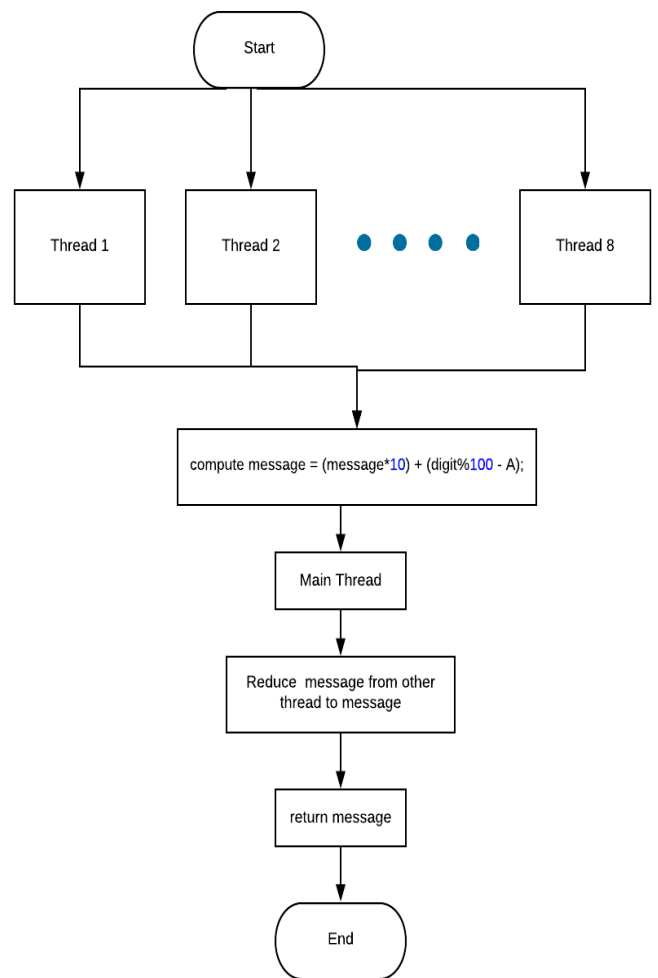


Figure 7 Flowchart for Caesar cipher with Decryption algorithm

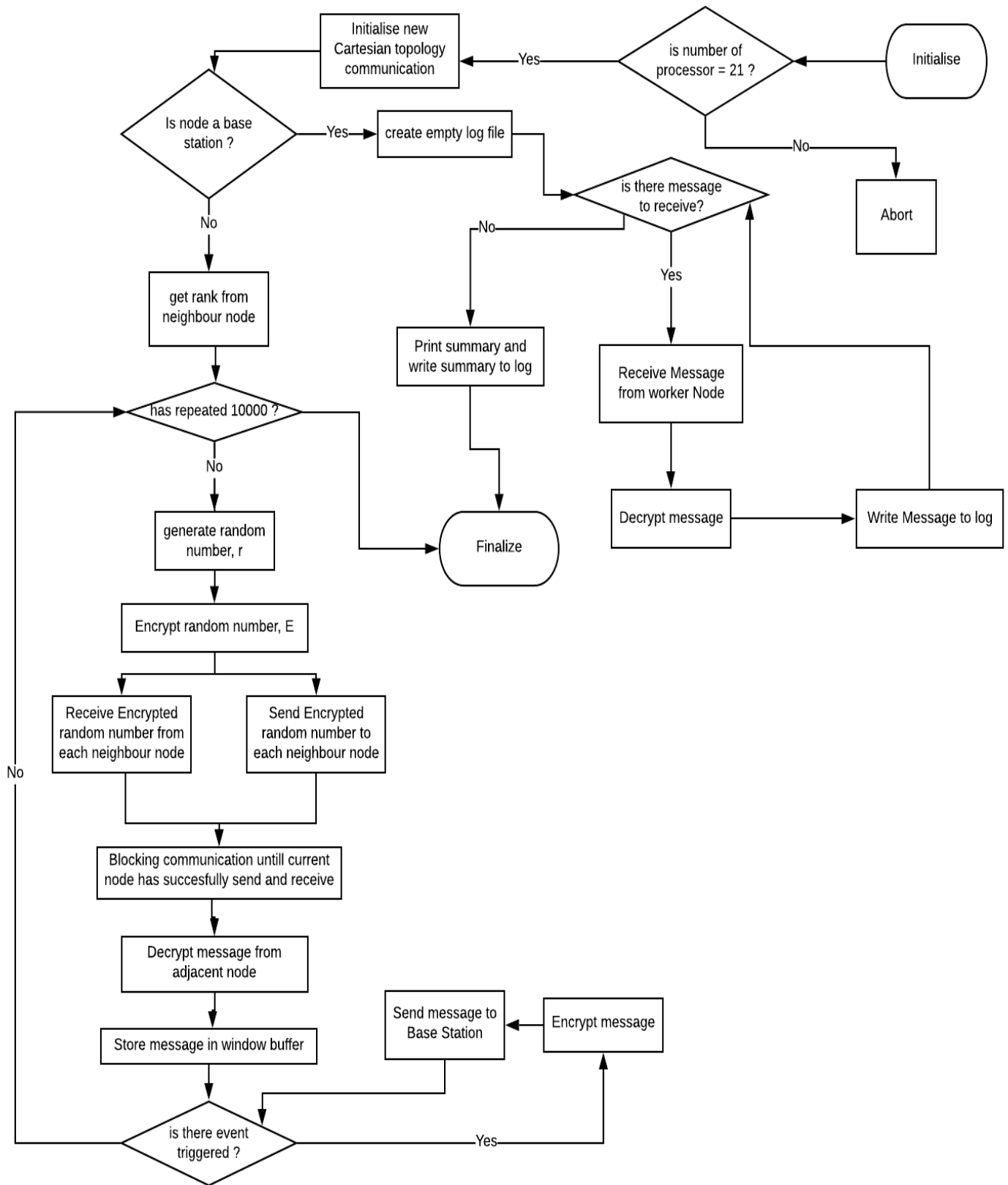
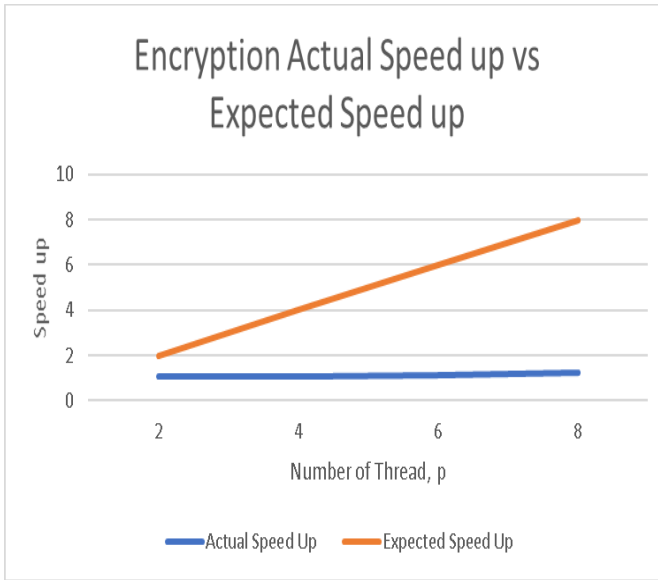


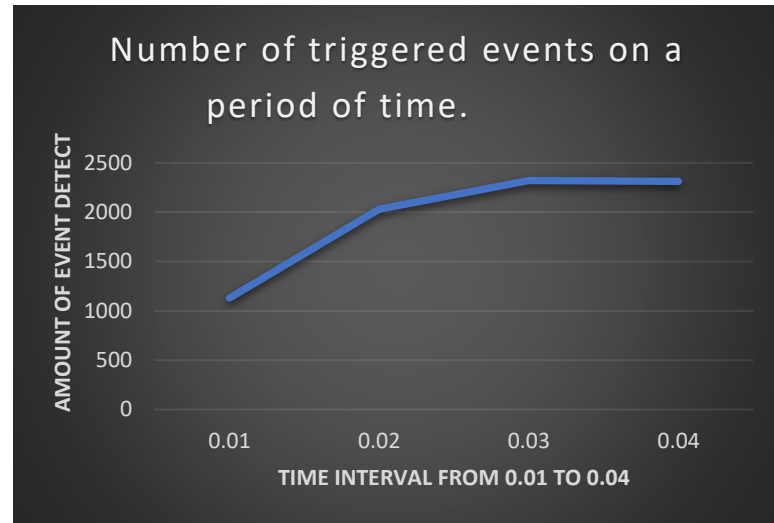
Figure 8 Flowchart for WSN Event detection Implementation.

## Encryption Speed Up

Number of Thread, p	2	4	6	8
Speed Up Factor, S(p)	1.091466	1.075331	1.14507	1.240518

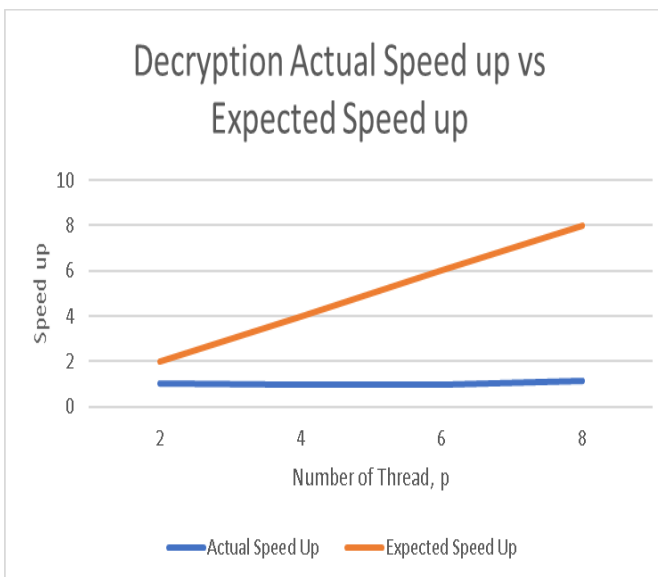


Attempt runs	number of events	Total Time taken for Base station and node	Total Time taken for adjacent node
1	7855	0.015779	44.435861
2	7676	0.012823	43.052471
3	7853	0.014508	36.426833
4	7501	0.013073	36.604699
5	7732	0.014137	42.153317
6	7794	0.015910	38.632458



## Decryption Speed Up

Number of Thread, p	2	4	6	8
Speed Up Factor, S(p)	1.014131088	0.997152	0.979732707	1.129749461



## IV. RESULTS AND DISCUSSIONS

The result of speed up factor with Amdahl's law shows that encryption and decryption with Caesar cipher with parallel processing can be faster than the sequential processing of Caesar cipher. However, the speed up boost may not meet up to expected speed up factor and this can be due to output dependency which slow down the computational time from reaching beyond limits. The result of this experiment contradicts to earlier hypothesis that Amdahl's Law stated that as number of threads increased, the speed up factor should be approximately close to number of threads. Fortunately, there are some number of threads which helps in slight speed up boost.

The computational time for communication between nodes in Cartesian topology is shown to be much slower than the computational time for communication nodes in Cartesian topology and Base station. This is likely due to either inefficiency of cartesian topology in MPI or Simultaneous Blocking Send and Receive which have communication overhead that the program has been dealing

As number of event increases, the time taken for communication between all nodes in MPI Communicator also increases. This can be evidenced that more frequency of event in table leads more time taken to communicate between nodes. The best explanations for this are every event takes time to send over the message before it can continue running and the message are actually store in buffer which explain the delay of messages.

Decryption algorithm of Caesar cipher is much slower than Encryption algorithm of Caesar cipher and this can be effect of trying to retrieve back original message of Caesar cipher. The encryption algorithm seems to have a slight boost in computational time compared to decryption algorithm where speed up which has irregular speed up boost. Nonetheless, a total of 8 threads is adequate to get speed up boost. This can be evidenced that encryptions algorithm gets 24% boost and decryptions algorithm gets 12% boost speed up.

Therefore, the result of the experiment shows that Encryption and Decryption with Caesar cipher can be equip with speed up boost if number of threads is chosen appropriately and 8 should be an ideal choice to have the Encryption and Decryption algorithm speed up but it will not likely meet Amdahl's Law speed up expectation. The computational time for communication between nodes in Cartesian topology is either inefficiency of cartesian topology in MPI or Simultaneous Blocking Send and Receive that delays the communication.

## VI. CONCLUSIONS

Cartesian topology is an infrastructure that arrange all sensor node in a communicator which helps abstract data type much easier when neighbor node is needed, sliding window algorithm will enhance the amount of event and Caesar cipher can proves to be a secured and efficient encryption and decryption algorithm.

The theoretical speed up factors may not consider the effect of delay caused by communication between threads. Therefore, it will be developer role to ensure effect that is not consider in theoretical speed up to be dealt

drastically to allow program to run more efficiently. Performance degradation is almost certainly happened if developer ignores potential communication overhead,

Nonetheless, Caesar cipher may be desired to replace with a faster and probably more secured encryption algorithm because experiment has shown it may enjoy the speed boost from parallelism, but it may at some point hits its limit and no longer being efficient. At this point, an alternative encryption such as AES with ECB mode or any other parallelizable encryption algorithm may be required. Although sliding window method is desirable method to perform event detection, a considerable single sliding window to compare with more than single sliding windows allow investigation for the impact made when window size is increased.

## VII. REFERENCES

- [1] Wireless Sensor Networks and their Applications. (n.d.). Retrieved from Elprocus: <https://www.elprocus.com/introduction-to-wireless-sensor-networks-types-and-applications/>
- [2] MPI topic: Topologies. (n.d.). Retrieved from Texas Advanced Computing Center, The University of Texas at Austin: <http://pages.tacc.utexas.edu/~eijkhout/pcse/html/mpi-topo.html>
- [3] Sliding Window Algorithm Basic Information. (n.d.). Retrieved from RIP Tutorial: <https://riptutorial.com/algorithm/example/25071/sliding-window-algorithm-basic-information>
- [4] Rotating information around rows or columns of 2D cartesian process topology. (n.d.). Retrieved from University of Jyväskylä: <http://users.jyu.fi/~tro/rhl00/kotit/demo5/rings.html>
- [5] Jair Ferreira Júnior, E. C. (2017). Driver behavior profiling: An investigation with different smartphone sensors and machine learning.
- [6] Ellis, S. R. (2009). Computer and Information Security Handbook.



## APPENDIX

### Encryption Message Time

Computer specifications		<i>a) Intel Core i7-9700k</i> <i>b) 8</i> <i>c) 8gb</i> <i>d) 70 Gb/s</i>			
Value of <u>Min</u> Random Number		950(default)			
Value of Max Random Number		1,000 (default)			
Value of <u>IterationMax</u>		10,000 (default)			
	Serial program	Parallel Program			
		OpenMP			
		2 Threads	4 Threads	6 Threads	8 Threads
Run #1	0.001574	0.001599	0.001781	0.001543	0.001446
Run #2	0.001891	0.001782	0.001533	0.001378	0.001491
Run #3	0.001711	0.001667	0.001992	0.001669	0.001392
Run #4	0.001762	0.001451	0.001473	0.001799	0.001524
Run #5	0.002155	0.001832	0.001677	0.001552	0.001477
Average time	0.0018186	0.0016662	0.0016912	0.0015882	0.001466

# Decryption Message Time

Computer specifications		<i>a) Intel Core i7-9700k</i> <i>b) 8</i> <i>c) 8gb</i> <i>d) 70 Gb/s</i>			
Value of <u>Min</u> Random Number		950(default)			
Value of Max Random Number		1,000 (default)			
Value of <u>IterationMax</u>		10,000 (default)			
	Serial program	Parallel Program			
		OpenMP			
		2 Threads	4 Threads	6 Threads	8 Threads
Run #1	0.001338	0.001522	0.001432	0.001334	0.001312
Run #2	0.001431	0.001245	0.001223	0.001561	0.001224
Run #3	0.001348	0.001441	0.001345	0.001299	0.001241
Run #4	0.001344	0.001321	0.001215	0.001422	0.001119
Run #5	0.001285	0.001123	0.001456	0.001193	0.001131
Average time	0.0013492	0.0013304	0.0013342	0.0013618	0.0012054

### Before Encryption and Decryption

```
=====
Current local time and date: Sun Oct 20 17:11:28 2019

Communication time between node and base-station: 0.000070
Reference Node: 0
Random Number: 971
Adjacent Nodes: 5
Adjacent Nodes: 1
=====
Current local time and date: Sun Oct 20 17:11:28 2019

Communication time between node and base-station: 0.000008
Reference Node: 0
Random Number: 952
Adjacent Nodes: 5
Adjacent Nodes: 1
=====
Current local time and date: Sun Oct 20 17:11:28 2019

Communication time between node and base-station: 0.000001
Reference Node: 0
Random Number: 952
Adjacent Nodes: 5
Adjacent Nodes: 1
=====
Current local time and date: Sun Oct 20 17:11:28 2019

Communication time between node and base-station: 0.000002
Reference Node: 0
Random Number: 987
Adjacent Nodes: 5
Adjacent Nodes: 1
=====
Current local time and date: Sun Oct 20 17:11:28 2019

Communication time between node and base-station: 0.000002
Reference Node: 0
Random Number: 992
Adjacent Nodes: 5
Adjacent Nodes: 1
```

=====  
Total number of event occurred: 7987  
Total Time of communication: 0.014690  
Average Time of communication: 0.000002  
=====

Rank 0	has	Random Number: 950	occured	1
Rank 0	has	Random Number: 951	occured	2
Rank 0	has	Random Number: 952	occured	5
Rank 0	has	Random Number: 953	occured	5
Rank 0	has	Random Number: 954	occured	1
Rank 0	has	Random Number: 956	occured	1
Rank 0	has	Random Number: 957	occured	4
Rank 0	has	Random Number: 959	occured	1
Rank 0	has	Random Number: 960	occured	4
Rank 0	has	Random Number: 961	occured	2
Rank 0	has	Random Number: 963	occured	3
Rank 0	has	Random Number: 964	occured	1
Rank 0	has	Random Number: 966	occured	3
Rank 0	has	Random Number: 967	occured	3
Rank 0	has	Random Number: 968	occured	3
Rank 0	has	Random Number: 970	occured	2
Rank 0	has	Random Number: 971	occured	1
Rank 0	has	Random Number: 972	occured	3
Rank 0	has	Random Number: 974	occured	3
Rank 0	has	Random Number: 975	occured	2
Rank 0	has	Random Number: 976	occured	4
Rank 0	has	Random Number: 977	occured	1
Rank 0	has	Random Number: 979	occured	2
Rank 0	has	Random Number: 981	occured	2
Rank 0	has	Random Number: 982	occured	1
Rank 0	has	Random Number: 983	occured	2
Rank 0	has	Random Number: 985	occured	1
Rank 0	has	Random Number: 986	occured	2
Rank 0	has	Random Number: 987	occured	2
Rank 0	has	Random Number: 989	occured	3
Rank 0	has	Random Number: 991	occured	2
Rank 0	has	Random Number: 992	occured	6
Rank 0	has	Random Number: 993	occured	3
Rank 0	has	Random Number: 994	occured	4
Rank 0	has	Random Number: 995	occured	2
Rank 0	has	Random Number: 996	occured	1
Rank 0	has	Random Number: 999	occured	2

## After Encryption and Decryption

```
|=====
Current local time and date: Sun Oct 20 17:21:09 2019

Communication time between node and base-station: 0.000082
Reference Node: 0
Random Number: 999
Adjacent Nodes: 5
Adjacent Nodes: 1
=====
Current local time and date: Sun Oct 20 17:21:09 2019

Communication time between node and base-station: 0.000005
Reference Node: 0
Random Number: 993
Adjacent Nodes: 5
Adjacent Nodes: 1
=====
Current local time and date: Sun Oct 20 17:21:09 2019

Communication time between node and base-station: 0.000002
Reference Node: 0
Random Number: 988
Adjacent Nodes: 5
Adjacent Nodes: 1
=====
Current local time and date: Sun Oct 20 17:21:09 2019

Communication time between node and base-station: 0.000002
Reference Node: 0
Random Number: 988
Adjacent Nodes: 5
Adjacent Nodes: 1
=====
Current local time and date: Sun Oct 20 17:21:09 2019

Communication time between node and base-station: 0.000002
Reference Node: 0
Random Number: 982
Adjacent Nodes: 5
Adjacent Nodes: 1
-----
```

```
=====
Total number of event occurred: 7544
Total Time of communication: 0.014819
Average Time of communication: 0.000002
Total Time to encrypt message: 0.001463
Total Time to decrypt message: 0.001164
Average Time to encrypt message: 0.000000
Average Time to decrypt message: 0.000000
=====
```

```
Rank 0 has Random Number: 950 occurred 3
Rank 0 has Random Number: 952 occurred 2
Rank 0 has Random Number: 954 occurred 2
Rank 0 has Random Number: 955 occurred 2
Rank 0 has Random Number: 957 occurred 1
Rank 0 has Random Number: 958 occurred 2
Rank 0 has Random Number: 959 occurred 6
Rank 0 has Random Number: 960 occurred 2
Rank 0 has Random Number: 961 occurred 2
Rank 0 has Random Number: 963 occurred 2
Rank 0 has Random Number: 964 occurred 4
Rank 0 has Random Number: 966 occurred 2
Rank 0 has Random Number: 967 occurred 1
Rank 0 has Random Number: 968 occurred 1
Rank 0 has Random Number: 972 occurred 2
Rank 0 has Random Number: 973 occurred 1
Rank 0 has Random Number: 974 occurred 2
Rank 0 has Random Number: 975 occurred 1
Rank 0 has Random Number: 976 occurred 1
Rank 0 has Random Number: 977 occurred 2
Rank 0 has Random Number: 978 occurred 2
Rank 0 has Random Number: 979 occurred 4
Rank 0 has Random Number: 981 occurred 2
Rank 0 has Random Number: 982 occurred 2
Rank 0 has Random Number: 983 occurred 1
Rank 0 has Random Number: 986 occurred 1
Rank 0 has Random Number: 987 occurred 2
Rank 0 has Random Number: 988 occurred 2
Rank 0 has Random Number: 992 occurred 1
Rank 0 has Random Number: 993 occurred 1
Rank 0 has Random Number: 995 occurred 1
Rank 0 has Random Number: 996 occurred 1
Rank 0 has Random Number: 997 occurred 1
```

---