A Framework for Fast Congestion Detection in Wireless Sensor Networks Using Clustering and Petri Net-based Verification

Khanh Le¹, Thang Bui¹, Tho Quan¹, and Laure Petrucci²

 ¹ Ho Chi Minh City University of Technology, Vietnam {lnkkhanh,thang,qttho}@cse.hcmut.edu.vn
² LIPN, CNRS UMR 7030, Université Paris 13, Sorbonne Paris Cité, France Laure.Petrucci@lipn.univ-paris13.fr

Abstract. Applications of Wireless Sensor Networks (WSN) in harsh conditions usually cover a vast area with sensors randomly deployed by an uncontrolled method, *e.g.* dropped by helicopters. Thus the actual topology is unpredictable and can suffer from possible congestion. We propose the FCD framework for congestion detection, based on clustering techniques combined with Petri nets modelling and verification.

1 Introduction

The congestion problem in Wireless Sensor Networks (WSNs): A WSN is a collection of hundreds or thousands of sensors. Sensors are cheap, low energy consuming devices, with limited memory and processing capabilities [1]. They can communicate with one another using WiFi. Depending on the targeted application, a WSN is deployed in a *dense* or sparse mode. Environment monitoring WSNs are usually implemented on a dense network topology [6] whereas some applications require sparsely spreading sensors over a large geographical area, *e.g.* for tracking transportation in a city [8]. The limited processing capacity and energy of sensors has some disadvantages, hence the necessity of some QoS (Quality of Service) constraints such as delay, security or congestion. We here focus on congestion detection.

Following [11], congestion depends on the network topology and the transmission rate. Indeed, sensors in a dense network are deployed at very close distances, thus some packets transmitted over the same paths may collide. Also, the processing rate of sensors may be smaller than their receiving rate: if the transmission rate is too high, the sensor's buffer overload can induce congestion. Similarly, the congestion also occurs during transmission in sparse deployment networks if a huge number of packets are sent within a short time. Packets loss and retransmission may cause the overload at some sensors.

Tools for Congestion Detection: The main approaches to congestion detection are *simulation* or *model* based. In the first case, a simulator is used to mimic the operations of the WSN, measure the performance, and check whether a certain anomaly like congestion occurs or not. Widely used simulators include ns2 [7] and Omnet++ [10]. In these, a WSN is considered as a network with sensors, channels and their activity (protocols). Hence, users must program their models according to the protocol used. The model-based approach enjoys two immediate advantages over simulator approaches: (1) the WSN is modelled at a higher level of abstraction, only including sensors and channels, thus it is independent of the framework used; (2) the model defines all scenarios and allows for exhaustively model-checking desired properties.

Petri Nets (PNs) are well-suited for modelling WSNs. To the best of our knowledge, WSN-PN [5] is the sole framework so far to model a WSN by a PN. WSN-PN allows users to model a WSN (using a domain specific input for WSNs), which is then translated into a PN; then WSN-PN verifies congestion on the PN model by means of model-checking. In WSN-PN, users do not need to work with the details of the PN model. Instead, they only need to specify the topology and parameters setting of a WSN; the corresponding PN is automatically generated.

The proposed FCD framework: Congestion detection becomes intractable due to the state space explosion when the number of sensors increases. Thus, FCD combines WSN-PN [5] for modelling and verification, and COCA (Congestion-Oriented Congestion Algorithm for WSNs) clustering algorithm [3] in order to reduce the state space explosion problem. COCA detects groups of sensors that have a high chance of congestion and that can be verified individually. If these are congestion free, they are abstracted and combined with the remaining sensors to introduce a new abstracted WSN whose size is significantly reduced compared to the original oneand that can in turn be verified.

2 Petri Net-Based Verification of WSNs

We adopt a Component-based PNs approach for modelling, which allows convenient abstraction of components [4] for congestion detection.

Petri net generation for a WSN: A WSN is defined as $WSN = \{S, C\}$, where S is the set of sensors and C is the set of channels. Sensors can be *source*, *sink* or *intermediate nodes*. A *channel* is established between two communicating sensors. Information on sensors and channels forms the topology of the WSN.

To build the corresponding PN model, sensors and channels are first modelled individually as *Component PNs*. Then, these Component PNs are combined together, forming the global model. For example, Fig. 1a models a source sensor.

It is also necessary to attach to transitions some code that manipulates the quantitative values: *sensor buffer size*, *sending rate* and *processing rate* [5].

Component-based abstraction: Depending on the topology and transmission rate, it is often the case that congestion detection only requires considering sensors or channels but not both. The Component-based PN approach supports the abstraction of components for more efficient verification. For example, in Fig. 1b, sensors are abstracted as individual places, depicted larger and dashed, in case only channels are needed.



Fig. 1: Component PN and Component Based Abstraction models of a sensor

Congestion detection: WSN-PN uses the PAT model-checker [9] to verify the following LTL congestion property: **#assert WSN()** |= []<> Congestion

3 FCD Framework Processes

The overall process has three main parts.

Congestion-oriented clustering: The clustering step groups the sensors with a high chance of congestion into clusters, using COCA. It operates according to two metrics: physical distance and imbalance of transmission rate of sensors, which are major congestion factors. The clusters generated by COCA are subnetworks with a high chance of congestion. The sensors not included in clusters are named abandoned sensors. For instance, Fig. 2a illustrates the clustering of a simple WSN. The three clusters C_1, C_2, C_3 are pictured by ellipses.

PN models of clusters and local verification: The clusters are considered as sub-WSNs and modelled using the Petri Net-based technique presented in Section 2. However, they miss important information such as source and sink sensors, *e.g.* in Fig. 2a, cluster C1 misses both source and sink.

If a cluster misses source/sink, external sensors are chosen to serve as auxiliary ones, such that: (1) the source is an external sensor that sends incoming packets to the cluster; (2) the sink is an external sensor that receives outgoing packets from the cluster. The original cluster C1 in Fig. 2b is replaced by Fig. 2c, *i.e.* both sensors S17 and S18 become sources while S20 becomes sink.

Each sensor being modelled by a PN, the size of whole PN model increase with multiple sources/sinks. However, according to [2], their number can be reduced: (1) all sinks can be merged since congestion only occurs in intermediate nodes; (2) sources are merged when they send to the same sensor and the new sending rate is the sum of previous ones.

Abstracted clustered global models and verification: To limit state space explosion, clusters are abstracted, then composed together with the abandoned sensors. Congestion-less clusters are abstracted as "virtual" sensors. *Dummy channels* are created to mimic real channels. Finally, the PN model is generated and the congestion property verified.



(c) Re-establishing Source and Sink for cluster C1



(d) Abstraction of cluster C1

Fig. 2: Illustration of FCD Processes



Fig. 3: Abstracted network topology

Clusters are abstracted as follows: their intermediate sensors are removed while sensors with incoming/outgoing channels to/from the cluster are kept as well as sources and sinks. For example, abstraction of cluster C1 is shown in Fig. 2d. The inner arcs in the cluster are computed according to transmission rates, thus creating *dummy channels* that mimic the original behaviour.

Dummy channels depend on the incoming and outgoing packet rates. The *incoming packet rate of a sensor* s_i , denoted by $In(s_i)$, is the total number of packets that are sent to sensor s_i . Its *out*-

going packet rate, denoted by $Out(s_i)$, is the minimum value of the incoming packets rate of s_i and its processing rate, *i.e.* $Out(s_i) = \min(In(s_i), pr(s_i))$.

A dummy channel c_{ij} is created between sensors s_i and s_j if and only if there exists a path from s_i to s_j . Its transfer rate, $tr_d(c_{ij})$ is the minimum value of



Fig. 4: Illustration of dummy channel creation

outgoing packets of sensor s_k and transfer rate $tr(c_{kj})$, for all sensors s_k along the path, *i.e.*: $tr_d(c_{ij}) = \min_{s_k \in path(s_i, s_j), s_{k'} = succ(s_k)} \min(Out(s_k), tr(c_{kk'}))$.

Consider cluster C1 in Fig. 4a where the red numbers are the processing rates of sensors, and the blue ones the transfer rates of channels. Figure Fig. 4b illustrates the dummy channels creation when removing S7 and S11. The abstracted clusters and the remaining *abandoned sensors* lead to an abstracted network, as shown in Fig. 3. WSN-PN is used again to verify congestion on this new network.

4 Experiments

FCD was experimented with WSNs modelled by WSN-PN under random topologies having 70 to 10,000 sensors, as shown in Table 1.

Table 1. Experimental results							
Ν	b Nb of	Nb of	Nb of	Verification	Nb of Sensors in	Time (s)	Note
	Sensors	Channels	Clusters	Result	Congested Cluster		
1	40	60	3	Congested at Cluster 1	17	21.96	
2	: 70	100	17	Congested at Cluster 1	19	32.32	
3	100	135	11	Congested at Cluster 2	36	264.08	Takes more than 145s to verify
							Cluster 1 which contains 43 sensors
4	800	931	467	No Cong	estion	330.00	
5	800	1000	568	Congested at Cluster 12	52	399.91	
6	1,000	1590	793	Congested at Cluster 88	16	1,168.01	
7	100	100	77	Congested at Cluster 5	5	38.13	
8	350	130	233	Congested at Cluster 17	44	320.28	Takes time to verify Cluster 12
9	800	500	713	Congestion on new	network topology	7,240.80	
1	0 800	600	624	Congested at Cluster 3	7	107.41	
1	1 5,000	2000	4,845	Congested at Cluster 19	11	176.31	
1	2 10,000	6000	9,444	Congested at Cluster 71	5	1,379.34	
13	3 800	920	722	Timeout aft	er 8 hours 7 minutes		Cannot verify Cluster 127
							which contains 73 sensors
1	4 8000	3000	78	Timeout after 12 hours		Cannot verify Cluster 5	
						which contains 111 sensors	
1	5 10,000	12,000	8,975	Timeout after 9 hours 25 minutes		Cannot verify new network	
							topology which contains 4,969 sensors

Table 1: Experimental results

Even though time is spent for clustering and local verification, the total verification time of FCD is significantly reduced compared to WSN-PN. Topologies

334 PNSE'16 – Petri Nets and Software Engineering

1–6 have a dense deployment, and most sensors are grouped into clusters. The more the sensors, the longer the verification. In topologies 7–12, the WSNs are sparsely deployed. Even though most clusters are very small, congestion occurs in large ones. The last three cases do not allow for getting a result, due to the limitations of the WSN-PN tool, since the networks to verify are too large.

5 Conclusion

This paper presented FCD, a framework combining clustering technique and formal verification in order to efficiently find possible congestion in WSNs. WSNs are clustered based on the congestion-oriented measurement first. Then, the verification process is performed on each individual cluster. Congestion is detected earlier if it exists within a cluster. Otherwise, the verification process is repeated on a new abstracted network obtained from abstracted clusters and abandoned sensors in case clusters are confirmed congestion-free in the previous step. Experiments show that in most cases, congestion is detected on clusters, which significantly decreases the verification time.

References

- Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless sensor networks: a survey. Computer Networks 38(4), 393–422 (2002)
- Kaynar, D.K., Lynch, N.A., Segala, R., Vaandrager, F.W.: The Theory of Timed I/O Automata, Second Edition. Synthesis Lectures on Distributed Computing Theory, Morgan & Claypool Publishers (2010), http://dx.doi.org/10.2200/ S00310ED1V01Y201011DCT005
- 3. Le, K., Bui, T., Quan, T., Petrucci, L.: COCA: Congestion-oriented clustering algorithm for wireless sensor networks. In: ICCSN, Beijing, China (Jun 2016)
- Le, K., Bui, T., Quan, T., Petrucci, L., André, E.: Component-based abstraction of Petri net models: An application for congestion verification of wireless sensor networks. In: SoICT, Hue, Vietnam. pp. 342–349 (Dec 2015), http://doi.acm. org/10.1145/2833258.2833298
- Le, K., Bui, T., Quan, T., Petrucci, L., André, E.: Congestion verification on abstracted wireless sensor networks with the WSN-PN tool. Advances in Computer Networks 4(1), 33–40 (2016)
- Moon, S.H., Lee, S., Cha, H.: A congestion control technique for the near-sink nodes in wireless sensor networks. In: UIC, Wuhan, China. pp. 488–497 (Sep 2006), http://dx.doi.org/10.1007/11833529_50
- 7. The Network Simulator NS-2. http://www.isi.edu/nsnam/ns/
- Shah, R.C., Roy, S., Jain, S., Brunette, W.: Data MULEs: modeling and analysis of a three-tier architecture for sparse sensor networks. Ad Hoc Networks 1(2-3), 215-233 (2003), http://dx.doi.org/10.1016/S1570-8705(03)00003-9
- Si, Y., Sun, J., Liu, Y., Dong, J.S., Pang, J., Zhang, S.J., Yang, X.: Model checking with fairness assumptions using PAT. Frontiers of Computer Science 8(1), 1–16 (2014), http://dx.doi.org/10.1007/s11704-013-3091-5
- Varga, A., Hornig, R.: An overview of the OMNeT++ simulation environment. In: SimuTools, Marseille, France (march 2008), http://dx.doi.org/10.4108/ICST. SIMUTOOLS2008.3027
- Wan, C., Eisenman, S.B., Campbell, A.T.: CODA: congestion detection and avoidance in sensor networks. In: SenSys. pp. 266-279. ACM (2003), http://doi.acm. org/10.1145/958491.958523