An Approach of Concurrent Communications for Distributed Sensing Networks

Tuyen P. Truong[1] and Bernard Pottier[2]
[1] Can Tho University, Can Tho 900000, Vietnam
[2] University of Brest, Brest 29283, France
Email: tptuyen@ctu.edu.vn; pottier@univ-brest.fr

***Abstract*—**This article describes research on modeling and simulating concurrent communications of distributed wireless sensor networks for improving available network capacity. The workflow with software tools was proposed that allows producing automatically Occam-Pi code to feature a concurrency model inspired by Communicating Sequential Processes (CSP) paradigm. Wireless networks are presented based on a graph abstraction model for simulating the complex behaviors of systems. Concurrent communications of distributed sensing networks are handled by the well-known message-passing model used to program parallel and distributed applications. Parallel and distributed algorithms were employed to describe local node's behaviors to build up the networks and manage communications of distributed sensing systems in environmental monitoring. Finally, code synthesis to port on executing hardware was carried out on an actual LoRa wireless network in order to assess the feasibility and usefulness of our approach for concurrent communication.

***Index Terms*—**Concurrent computing, distributed sensing networks, message-passing model, NetGen tool, Occam-Pi

I. Introduction

In recent years, wide-area wireless networks have emerged as an attractive solution for remote sensing applications, thanks to the current innovation solutions such as Low Power Wide Area Networks (LPWAN) and sensor technology progresses [1]-[3]. Semtech's LoRa is an innovative wireless telecommunication technology with Chirp Spread Spectrum (CSS) modulation being used to provide an ultra-long-range spread spectrum communication with high interference immunity. LoRa modulation uses the frequency of the carrier, applying continuous linear increases (up-chirp) and decreases (down-chirp) over time while encoding information. Based on CSS being widely used in radar applications, LoRa technology trades off low data rate for long-range communications, and low-power consumption. The LoRa transceiver chip has different configuration parameters such as Carrier Frequency (CF), Spreading Factor (SF), bandwidth (BW) and Coding Rate (CR). The non-orthogonal combinations of LoRa signals can be achieved by selecting proper spreading factors (SFs) and/or bandwidths (BWs). In Fig. 1, cells are highlighted in red color that denotes the non-orthogonal combinations. These combinations cannot be used for multiple spread signals to be transmitted on the same channel at the same time [3]-[5].

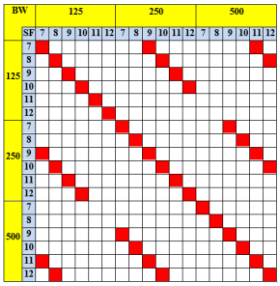

Fig. 1. A summarization of orthogonal and non-orthogonal combinations of LoRa signals.

Occam is a concurrent programming language designed for parallel computers, notably Transputers [6], [7] by adopting the well-known Communicating Sequential Processes (CSP) paradigm. Occam-Pi, the derivation of Occam, was inspired by binding Pi-calculus and Hoare's CSP. The compiler KRoC (Kent Retargetable Occam-Pi Compiler) was developed at Kent University in the United Kingdom aiming at producing native bytecode of the Transputer for different platforms [7]-[14].

In literature, many studies have been carried out to facilitate modeling and simulating large-scale distributed sensing systems based on concurrent simulation [1]-[3]. In addition, some authors conduct their research on concurrent transmission in order to improve the capability of networks [15]-[24]. In [25], a framework is proposed to facilitate expressing parallelism, communication, and reconfigurability. Distributed algorithms were developed to describe the local behavior of the node. The data for simulating in Occam-Pi programming language are produced automatically by the NetGen tool [26]. Simulators help to validate and evaluate the algorithms. These Occam-Pi programs are then transformed to execute on real hardware. This is feasible thanks to a virtual machine, namely Transputer Virtual Machine (TVM), and KRoC compiler [6], [8], which allow running extended transputer bytecode on a small hardware resource of sensing nodes.

This article presents research on modeling and simulating concurrent communication in distributed wireless networks. LoRa systems have been chosen as a study case. The key idea is to enable concurrent

Manuscript received August 14, 2019; revised March 6, 2020.
doi:10.12720/jcm.15.4.372-378

communication by establishing orthogonal radio links. Every node is equipped with several radio connections for sending and receiving messages simultaneously with almost no interference. In order to facilitate the development of distributed algorithms, the wireless communication of distributed sensing network is modeled as a directed graph. By utilizing both advantages of KRoC and TVM, Occam-Pi programs for concurrent processes of nodes after being developed and turning can execute on real WSN hardware.

The rest of this article is structured as follows. After summarizing the related works in Section 1, Section 2 presents a methodological approach for generating a graph model of networks and then modeling distributed synchronous behaviors following the message-passing model. Section 3 describes the workflow of simulating synchronous behaviors of concurrent communication in distributed wireless sensor networks. For the valuation and evaluation of our proposed approach, experiments were carried out on an actual LoRa wireless network for validation. Finally, Section 4 provides a discussion before concluding the article in Section 5.

II. Modeling Distributed Sensing Networks

A. Abstract Network Layout

QuickMap is a specific map browser allows selecting various kind of tile server for retrieving tile map from free GIS database such as OpenStreetMap (OSM). In Fig. 2, QuickMap is employed to select an area in the north of Phu Quoc island 574 km^2 in southwest Vietnam where the National park Phu Quoc for biological and ecological conservation is located. Next, the PickCell tool provides the feature of layout an abstract network over a concerned zone for sensing purposes.

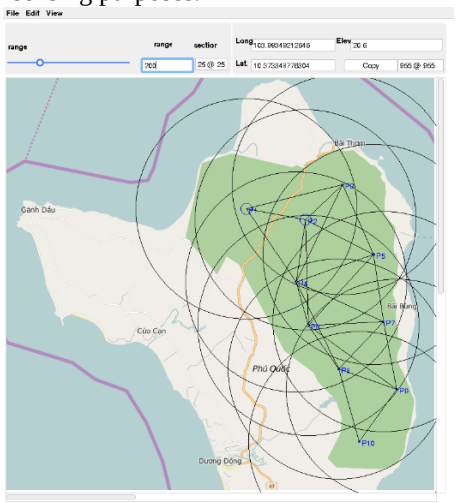

Fig. 2. QuickMap and PickCell tool allows users to deploy abstract networks stacked on raster tile maps such as OpenStreetMap, ArcGIS maps, and so on.

B. Abstract Network Presentation and Generation

The NetGen tool produces an automatically abstract network model in Occam-Pi for simulators. In Fig. 3, a directed graph is presented from the .dot file, which was generated automatically by NetGen to describe the topology of the abstract network. In this graph, sensor nodes are denoted by vertices and wireless connections are represented as arcs. Fig. 4 presents the data model of the abstract network. The structural description shown above is graphically represented.

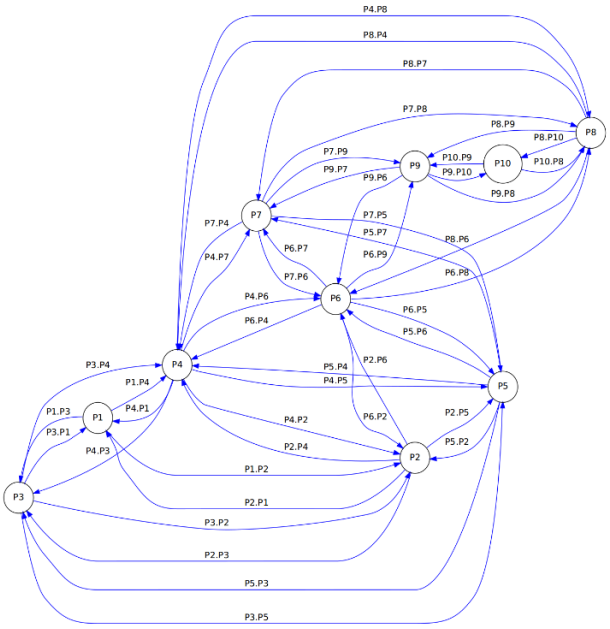

Fig. 3. Graphviz tool [27] is used to present a directed graph generated for the network in Fig. 2 from a .dot file.

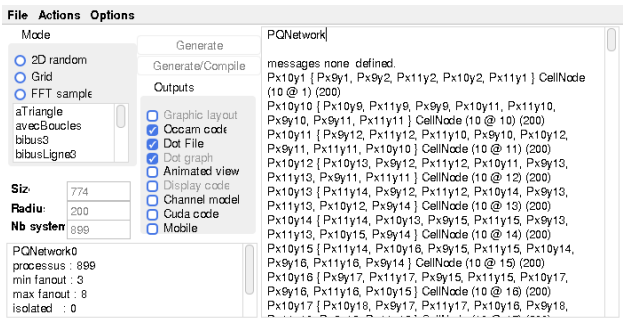

Fig. 4. NetGen window shows the data model of the abstract network. The list within braces is the connectivity of the named process.

C. Modeling Distributed Synchronous Behaviors of Sensing Node Using Occam-Pi

Occam-Pi provides a complete set of primitives for the expression of concurrency communication of distributed sensing systems:

- PARallel constructs for the development of structural parallelism controlled by barriers. The processes at sensor nodes can be specified as PAR constructs with an arbitrary number of components and arbitrary connectivity. PAR construct terminates with the end of barrier detection within the kernel primitives. It can denote either local or distributed activities.
- CHANnels as a single primitive for synchronization and communication. Channels are implemented as physical radio links. These links are operated by low-

level software binding circuitries access to the local run-time Occam-Pi kernels. Kernels exist on each system and handle the relation between process synchronization and termination of radio communications. In practice, we have implemented support to extend an existing virtual machine for the Arduino with a primitive software operating LoRa communications on a transceiver addresses by SPI bus.

- ALTernatives specifying non-deterministic behaviors. Alternatives appear when a set of channels is activated without enforcing a particular order in synchronization. This is the case for processors with several peripheral circuits working concurrently to acquire or process information.
- PROTOCOLs specify the type of communications that circulate on channels. The compiler will verify the conformity between local data structures and network messages. Protocols will appear on the red channels to denote neighborhood communications and control operations.
- TIMERs allow producing delays based on real-time clocks existing in the nodes. An obvious interest for timers is to sequence operations according to general scheduling such as time-division multiple access (TDMA), or scheduling table produced for LoRa networks.

III. Simulating Concurrent Communications of Distributed Wireless Sensor Networks

A. Simulating Concurrent Communications

In the following sections, we present distributed algorithms, which describe the local behavior of nodes based on the proposed framework in [24].

- **Exploring Neighbor Nodes**

To explore the topology of network properly every node must be executed the process for information exchange relied on message passing at least *MaxNodes* rounds, where *MaxNodes* is a number of nodes of the network (see Fig. 5). Apart from being obtained knowledge about the attributes of surrounding neighbors such as nodes' latitude, longitude, and elevation, the diameter of the network may be another target at the convergent time of processes. This diameter is always less than n and afterward, it becomes a new efficient value of round for the following processes.

```
PROC explorationNeighbors()
  SEQ
    initLocalVal()
    SEQ i=0 FOR (SIZE outChannels)
      outMessages[i] ! localValues
    SEQ turns=0 FOR MaxNodes
      SEQ
        PAR
          PAR i=0 FOR (SIZE inChannels)
            inChannels[i] ? inMessages[i]
          PAR i=0 FOR (SIZE outMessages)
            outChannels[i] ! outMessages[i]
        SEQ i=0 FOR (SIZE inChannels)
          upDateLocalValues(localValues, inMessages[i])
        SEQ i=0 FOR (SIZE outChannels)
          outMessages[i] ! localValues
:
```

Fig. 5. Cell behavior is defined based upon a message-passing model [11].

- **Base Station Election**

```
PROC SelectBSNode(INT selectedLinkIndex)
  Location currentLocation:
  REAL32 localDistance, newDistance:
  SEQ
    localDistance:= 0.0
    selectedLinkIndex:= -1

    --Seek all neighbors
    SEQ i=1 FOR myTable[limit] - 1
      SEQ
        IF
          (myTable[tabNodes][i][numHop]=1)
            SEQ
              -- calculate Euclidean distance
              newDistance:= SQRT(PowerTwo(REAL32 ROUND (myTable[tabNodes][0][myLocation][xLoc] -
                myTable[tabNodes][i][myLocation][xLoc])) +
                                 PowerTwo(REAL32 ROUND (myTable[tabNodes][0][myLocation][xLoc] -
                                   myTable[tabNodes][i][myLocation][xLoc])))
              IF
                (newDistance > localDistance)
                  SEQ
                    localDistance:= newDistance
                    selectedLinkIndex:= myTable[tabNodes][i][linkIndex]
                TRUE
                  SKIP
          TRUE
            SKIP
```

Fig. 6. A procedure in Occam-Pi describes the voting process for the base station election based on the global maximum distance metric of the network.

Environmental data collected from sensors should be gathered to some certain nodes, namely base stations, and then relay to the data center. In a network whose nodes are distributed randomly, distributed processes executing locally on every node vote one or several nodes becoming a base station (see Fig. 6). The metric used in voting can be geographic distance, a number of hops of forwarding messages as an example.

- **Building up Routing Table**

```
PROC RoutingTableBuildUp()
    [MaxFanOut] BOOL Makers:
    BYTE nullByte:
    SEQ
      ourId := identity
      ResetFile(ourFile)
      AddIdInFile(ourId, ourFile)
      ResetTab(ourTab)
      AddIdInTab(ourId, 0, -1, ourTab)
      --Buffers initilize
      SEQ i=0 FOR SIZE out
        outBuff[i] := ourFile
      --Synchronous loop
      SEQ i=0 FOR (MaxNodes - 1)
        SEQ
          --Communications
          PAR
            PAR j=0 FOR SIZE in
              in[j] ? CASE
                file ; inBuff[j]
                  Makers[j] := TRUE
                null ; nullByte
                  Makers[j] := FALSE
            PAR j=0 FOR SIZE out
              out[j] ! file ; outBuff[j]
          --Input analysis
          tour := i
          ResetFile(ourFile)
          SEQ index=0 FOR SIZE in
            SEQ
              thisFile := inBuff[index]
              thisLimit := thisFile[limit]
              SEQ entry=0 FOR thisLimit
                SEQ
                  anInputId := thisFile[newFound][entry]
                  FoundIdInTab(anInputId, ourTab, found)
                  IF
                    (NOT found)
                      SEQ
                        AddIdInFile(anInputId, ourFile)
                        AddIdInTab(anInputId, i+1, index, ourTab)
                    TRUE
                      SKIP
          --Update the output
          SEQ m=0 FOR SIZE out
            outBuff[m] := ourFile
    :
```

Fig. 7. The Occam-Pi program to create a routing table using synchronous communicating processes.

For sending or forwarding messages within a network, at every node a routing table needs to be built up from obtained knowledge by executing the process as can be seen in Fig. 7.

- **Scheduling Communication**

Although each node can establish several separated radio links for sending and receiving tasks simultaneously by utilizing TVM and Occam-Pi concurrent processing,

scheduling schemes relied on the time-sharing principle. It allows optimizing network performance and extending the system's lifetime.

```
process |<nodeId,numHop,timeSlot>|...
6 |<2,1,1>|<4,1,2>|<5,1,3>|<7,1,4>|<8,1,5>|<9,1,6>|<0,2,1>|<3,2,2>|<1,2,3>
0
1
2
3
4 |<0,1,1>|<2,1,2>|<3,1,3>|<5,1,4>|<6,1,5>|<7,1,6>|<8,1,7>|<9,2,1>|<1,2,2>
5
7
8
9
```

Fig. 8. Time slot generation at BS nodes (P4, P6) for scheduling communications to gather data from all sensor nodes to BS.

A distributed coloring algorithm is developed to depict the TDMA scheme for synchronous networks. Two BS nodes are highlighted by solid yellow-colored circles. Various colors being used to indicate time slots for synchronous communications within the network. Fig. 8 shows the time slot being generated at BS nodes.

- **Data dissemination**

```
1  SEQ tours = 0 FOR MaxNodes
2    SEQ
3      -- Concurrent communications
4      PAR
5        PAR n=0 FOR SIZE in
6          in[n] ? CASE
7            packet ; inBuff[n]
8              SKIP
9
10       PAR m=0 FOR SIZE out
11         out[m] ! packet ; outBuff[m]
```

Fig. 9. Synchronous communicating processes occur concurrently on both input and output channels for data dissemination.

In Fig. 9, SEQ structure provides a synchronous loop in MaxNodes round which specifies the number of turns to communicate with all neighbor nodes. Concurrent communications occur in both output channels (?) and input channels (!) [7], [9]. By adopting CSP with message-passing model, the communications of each node are taken place on safe channels with guard barriers, which are provided by KRoC compiler. Each sensor node of the network sends its own data to the base station at a given time slot strictly. Once a node receiving a package from one of its neighbors, the node unpacks that package to get data if the package is for it, arrive status. Otherwise, the node forwards, indicated as transfer status, the packet to the appropriate next hop toward the destination. A proper link index corresponding to the destination address in the packet is figured out by seeking on its local routing table.

A directed graph for a TDMA scheme to aggregate data in wireless sensor networks is depicted in Fig. 10. Vertices are represented for nodes and arcs are known as direct connections between nodes. Two isolated connections for sending and receiving from a node to another neighbor node are represented separately by two arrows. Each directed link between a couple of nodes is represented by colored arrows from sending the node to the receiving node. Different colors are utilized to give a visual demonstration of the timing attribute of links. Fig. 11 represents a piece of communication transactions in the experimental network.

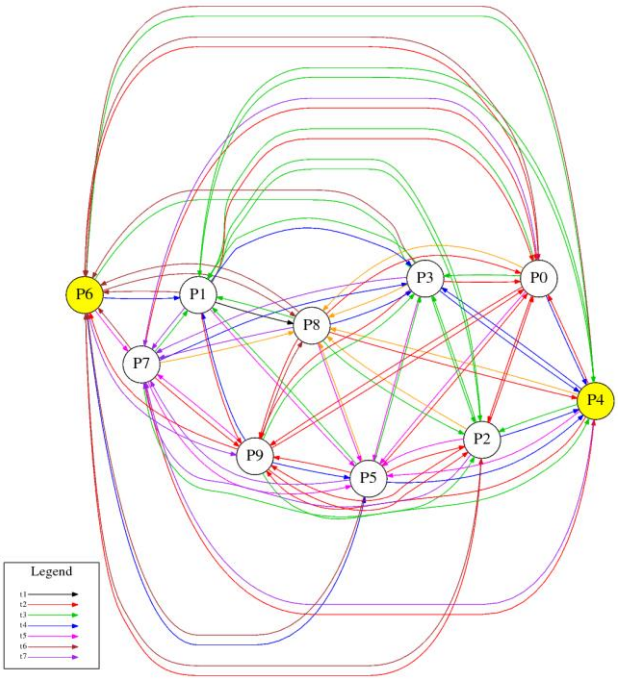

Fig. 10. A graph coloring for TDMA scheduling in wireless sensor networks.

```
<processID status round desId srcId>
9  transfer round:3 destId:4 srcId:2
8  transfer round:3 destId:4 srcId:0
4  arrive   round:3 destId:4 srcId:7
3  transfer round:3 destId:6 srcId:0
```

Fig. 11. A piece of communication transactions, it shows that node P4 received a package from node P7. Node P3, P8, and P9 forwarded packages towards corresponding destination nodes.

B. Experimental Validation

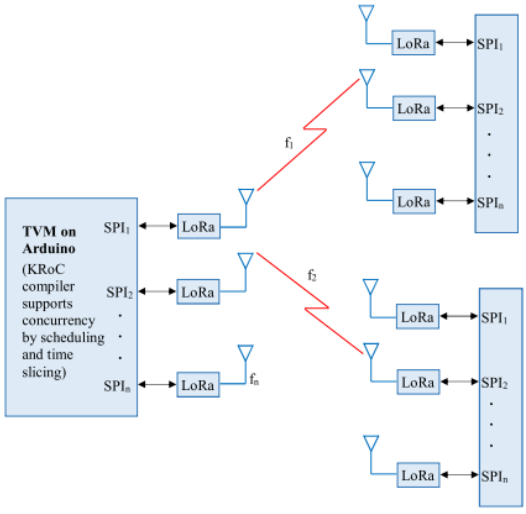

Fig. 12. Block diagram of the system for several LoRa communications simultaneously on separated channels.

Transputer Virtual Machine (TVM) is a portable and lightweight virtual machine for Transputer bytecode, which enables to execute transputer instruction set [6], [7]. TVM can operate on top of small microcontroller platforms. In our experiments, Occam-Pi programs for remote sensing applications can interpret on the target hardware platform by using KRoC [9]. Arduino boards

[28], as an example, which are equipped 8-bit AVR microcontrollers with a small capacity of memories. We did experiments on Lora technology with special attention to concurrent communication ability. The concept for experiments is demonstrated by the block diagram in Fig. 12. Three inAir9b, wireless SX1276 LoRa modules, are manufactured by Modtronix were used in this experiment [29]. These long-range transceivers can interface with the microcontroller via SPI bus (see Fig. 13).

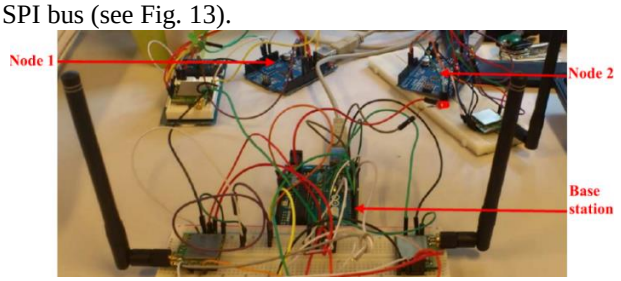

Fig. 13. System arrangement for the experiment on two separated wireless communications simultaneously. It is enabled thanks to the CSS modulation of LoRa technology, SPI master-slaves mechanism, and Occam-Pi concurrent processes.

```
1  #INCLUDE "wiring.module"
2
3  PROC main ()
4    PLACED [MAX.PORT]BYTE ports 0:
5    SEQ
6      -- Set pin2 and 3 (INT0 and INT1) as inputs
7      pinMode (2, INPUT)
8      pinMode (3, INPUT)
9
10     -- Set pin8 and 9 (red LED and green LED) as outputs, then turn off
11     pinMode (8, OUTPUT)
12     pinMode (9, OUTPUT)
13
14     digitalWrite(8, LOW)
15     digitalWrite(9, LOW)
16
17     -- Set interrupts sense
18     ports[EICRA] := #06
19
20     -- Enable INT0 and INT1
21     ports[EIMSK] := #03
22
23     beginSerial (9600)
24     serialWrite ("Starting ...*n")
25
26     CHAN BOOL c1, c2:
27     INT i0, i1:
28     BOOL b:
29
30     PAR
31       -- For sending processes
32       WHILE TRUE
33         SEQ
34           wait.for.interrupt (VINTR.INT0, i0)
35           c1 ! TRUE
36           serialWriteInt (i0)
37           serialWrite ("*n")
38
39       WHILE TRUE
40         SEQ
41           wait.for.interrupt (VINTR.INT1, i1)
42           c2 ! TRUE
43           serialWriteInt (i1)
44           serialWrite ("*n")
45
46       -- For receiving processes, blink LEDs accordingly
47       WHILE TRUE
48         ALT
49           c1 ? b
50             SEQ
51               serialWrite ("blinking red LED*n")
52               digitalWrite(8, HIGH)
53               delay (100)
54               digitalWrite(8, LOW)
55               delay (100)
56           c2 ? b
57             SEQ
58               serialWrite ("blinking green LED*n")
59               digitalWrite(9, HIGH)
60               delay (100)
61               digitalWrite(9, LOW)
62               delay (100)
63 :
```

Fig. 14. An Occam-Pi program being executed on TVM for concurrent LoRa wireless communications (see the PAR structure from line 30).

Distributed algorithms to describe the local behavior of the sensor node after being developed and evaluated by simulators, the final code synthesis was ported on executing hardware for testing in reality. Two different LoRa channels corresponding to two center frequencies were used for separated radio links in this experiment.

Occam-Pi libraries were developed for LoRa communication. Arduino Mega 2560 board is connected with two LoRa modules through two different SPI ports [28], [29]. Two external interrupt services are exploited to stimulate the corresponding routine for sending and receiving messages. In Fig. 14, Occam-Pi program, the ALTernative construction (line 48) is utilized to handle incoming messages allowing non-deterministic concurrent processes.

IV. Discussions

Part of the motivation is to use our proposed framework for rapid and secure distributed sensing systems. By providing a clean separation between networks and behaviors. The designer just needs to concentrate on developing robust distributed algorithms for local behaviors. Fine-tuning algorithms were archived utilizing a graph model for simulating such abstract networks. The simulators were developed in Occam-Pi programming language based upon the CSP synchronized message-passing model for highly concurrent computation. By adopting CSP paradigm communication in Occam-Pi is handled via channels using message passing that guarantees to isolate between the processes. The radio connections of the network, it is enabled by handling simultaneous communications of a distributed sensing network on orthogonal LoRa channels. It helps to avoid collision occurs when the transmissions overlap in time.

V. Conclusions

We have described a developing method to model wireless communication of sensing networks on top of a concurrent language inheriting from CSP. In addition, we did case studies on the buildup of an actual network by providing abstract services of distributed processes in which communication links are established on long-range (LoRa) wireless technology. Experiments on concurrent communications were done on a real LoRa wireless network to prove the feasibility and efficiency of our proposed approach. Instead of a microcontroller-based control system, Programmable System-on-Chip (PSoC) can be a promising hardware platform for increasing the performance and flexibility of the system utilizing mixed-signal arrays of configurable integrated analog and digital peripherals and dynamic configuration ability. It is critical to do further studies on LoRa communications about interference in a dense radio network, performance constraint as well as energy consumption.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Tuyen P. Truong and Bernard Pottier conducted the research; Tuyen P. Truong wrote the paper; Bernard

Pottier provided vital feedback and a number of full revisions of the manuscript; all authors had approved the final version.

ACKNOWLEDGMENT

The author would like to thank the Vietnam Ministry of Education and Training (MOET), SAMES[1] project and Can Tho University, Vietnam for funding different phases of this work.

REFERENCES

[1] A. Mahmood, E. Sisinni, L. Guntupalli, R. Rondón, S. Hassan, and M. Gidlund, *Scalability Analysis of a LoRa Network under Imperfect Orthogonality*, 2018.
[2] T. P. Truong, B. Pottier, and H. X. Huynh, "Cellular simulation for distributed sensing over complex terrains," in *Sensors*, 2018.
[3] T. P. Truong, H. X. Huynh, and Pottier, "Monitoring of environment: A high-performance method for radio coverage exploration," in *Proc. IEEE Radio 2016*, Réunion Island, October 2016.
[4] Semtech, Sx1276/77/78/79 -137 MHz to 1020 MHz Low Power Long Range Transceiver. Rev. 5 - August 2016. Tech. Rep. August, 2016.
[5] Semtech, LoRa Modulation Basics, Tech. Rep. May, 2015.
[6] C. L. Jacobsen and M. C. Jadud, The Transterpreter: A Transputer Interpreter, pp. 99–106, 2004.
[7] INMOS Limited, Transputer Instruction Set: A Compiler Writer's Guide, 1988.
[8] Occam 2.1 Reference Manual, SGS-Thomson Microelectronics Limited, 1995.
[9] University of Kent, Welcome to KRoC. [Online]. Available: http://projects.cs.kent.ac.uk/projects/kroc/trac
[10] C. A. R. Hoare, "Communicating sequential processes," *Commun.*, vol. 21, no. 8, pp. 666-677, August 1978.
[11] R. Milner, *Communication and Concurrency*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989.
[12] M. Raynal, *Distributed Algorithms for Message-passing Systems*, Springer-Verlag Berlin Heidelberg, 2013.
[13] P. Welch and F. Barnes, "Communicating mobile processes: Introducing occam-pi," pp. 175–210, 2005.
[14] D. May, Communicating processes and Occam, 1987.
[15] D. Son, B. Krishnamachari, and J. Heidemann, Wireless Medium Access for Concurrent Communication, 2008.
[16] D. Son, B. Krishnamachari, and J. Heidemann, "Experimental study of concurrent transmission in wireless sensor networks," in *Proc. Fourth International Conference on Embedded Networked Sensor Systems*, 2006, pp. 237-250.
[17] D. U. Son, J. Heidemann, and B. Krishnamachari, Towards Concurrent Communication in Wireless Networks, 2007.
[18] S. Thielemans, M. Bezunartea, and K. Steenhaut, "Establishing transparent IPv6 communication on LoRa based low power wide area networks (LPWANS)," in *Proc. 2017 Wireless Telecommunications Symposium (WTS)*, Chicago, IL, 2007, pp. 1-6.
[19] M. Jadud, L. Jacobsen, and J. Dimmich, Concurrency on and off the Sensor Network Node, 2006.
[20] C. Liao, G. Zhu, D. Kuwabara, M. Suzuki, and H. Morikawa, "Multi-Hop LoRa networks enabled by concurrent transmission," *IEEE Access*, vol. 5, pp. 21430-21446, 2017.
[21] C. L. Jansen, G. Amalinda, and W. Agustinus, "Mo L. known and unknown facts of LoRa: Experiences from a large-scale measurement study," in *ACM Trans. Sen. Netw.*, 2019, p. 35.
[22] H. Hafi, W. Abdou, and S. Merniz, "Impact of concurrent communications in geographical broadcasting protocols for vehicular ad hoc networks," in *Proc. IEEE 9th Latin-American Conference on Communications (LATINCOM)*, Guatemala City, 2017, pp. 1-6.
[23] V. Thiemo, B. Martin, R. Utz, and A. Juan, "Mitigating inter-network interference in LoRa networks," in *Proc. International Conference on Embedded Wireless Systems and Networks (EWSN '17)*. Junction Publishing, USA, 2017, pp. 323-328.
[24] S. Abboud, N. E. Rachkidy, and A. Guitton, Efficient Decoding of Synchronized Colliding LoRa Signals, 2019.
[25] T. P. Tuyen, P. Bernard, and X. H. Hiep, "A developing method for distributed sensing systems," in *Proc. 3rd International Conference on Machine Learning and Soft Computing (ICMLSC)*, Da Lat, Vietnam, January 25–28, 2019, pp. 210-214.
[26] B. Pottier and P. Y. Lucas, Dynamic Networks: Netgen Tools, A Guide for Simulation Software, Installation and Use, 2014.
[27] Graphviz - Graph Visualization Software. [Online]. Available: https://www.graphviz.org/
[28] Arduino Home Page. [Online]. Available: http://www.arduino.cc.
[29] Modtronix. Wireless SX1276 LoRa Module, 868MHz and 915MHz, +20dBm, 3.3V, SMA Connector. [Online]. Available: http://modtronix.com/inair9b.html

Tuyen P. Truong is a lecturer in College of Engineering Technology, Can Tho University (CTU), Vietnam. His research interests include embedded systems, Cyber-Physical Systems (CPS), parallel and distributed algorithms, modeling and simulation of physical phenomena based on cellular automata, long-range communications in wireless sensor networks, especially focusing on LoRa technology.

[1] SAMES is funded by the STIC-ASIA program from Ministry of Foreign Affairs (MAEDI), France, grouping researchers from University of Brest and Can Tho University, IRD, BPPT, and Cirela Association. http://sames.univ-brest.fr

Bernard Pottier is currently a full professor at Universite de Bretangne Occcidentale (UBO), France. His research interests include parallel processing, distributed algorithms, Cyber-Physical Systems (CPS), wireless sensor networks (WSNs). He is the leader of WSN research group at Lab-STICC, UMR 6285, CNRS. He is also the leader of Stic Asie Modeling for Environment and Simulation (SAMES) project.