NetTopo: Beyond Simulator and Visualizer for Wireless Sensor Networks

Lei Shu[1], Chun Wu[1], Yan Zhang[2], Jiming Chen[3], Lei Wang[4], Manfred Hauswirth[1]
[1]*Digital Enterprise Research Institute, National University of Ireland, Galway*
[1]*{lei.shu, chun.wu, manfred.hauswirth}@deri.org*
[2] *Simula Research Laboratory, Norway email: yanzhang@ieee.org*
[3] *Zhejiang University, China email: jmchen@ieee.org*
[4] *Dalian University of Technology, China email: lei.wang@dlut.edu.cn*

Abstract

Deploying real WSN testbed provides a realistic testing environment, and allows users to get more accurate test results. However, deploying real testbed is highly constrained by the available budget when the test needs a large scale WSN environment. By leveraging the advantages of both simulators and real testbed, an approach that integrates simulation environment and testbed can effectively solve both scalability and accuracy issues. In this paper, we present NetTopo for providing both simulation and visualization functions to assist the investigation of algorithms in WSNs. One case study is described to prove the effectiveness of NetTopo.

1. Introduction

Using testbeds allows rigorous and replicable testing. However, there are two serious limitations on this approach in the following two conditions: 1) Large scale. Until today, it is still very expensive to buy a large number of sensor nodes for a large scale testbed. Especially, for most academic researches the cost for building a large scale testbed is not acceptable. 2) Not replicable environment. For some specific applications, e.g., monitoring an erupting volcano, deploying a testbed is unwanted since the devices are exposed to dangerous conditions which can cause serious damage.

Due to the complementary properties of simulators and testbeds, a better solution can be the integration of simulation environment and physical testbed. Having this integrated framework, applications can run partially in a simulation environment and partially in a physical WSN testbed which can solve both scalability and accuracy issues for the evaluation of algorithms in WSNs. This integration is specially motivated by the following two concrete scenarios: 1) Researchers want to compare the performance of running a same algorithm in both simulator and real testbed. The comparison can guide researchers to improve the algorithm design and incorporate more realistic conditions. A good example is the applying of face routing algorithm in GPSR [1], which is proved to be loop free in theory but actually is not loop free in realist situations, due to the irregular radio coverage [2]. 2) A budget limitation prevents researchers from buying enough real sensor nodes but the research work has to base on a large scale WSN. For example, to evaluate the performance of sensor middleware [3], a large scale sensor network is needed. Researchers can actually do the research work by integrating a small number of real sensor nodes and a large number of virtual sensor nodes generated from the simulator.

The integration of simulation environment and physical testbed brings three major challenges: 1) *Sensor node simulation*. Normally, a number of heterogeneous sensor devices can be used for building a WSN testbed. The integrated platform should not simulate only a specific sensor device, which means that the heterogeneous problem requires the integrated platform to be flexible enough to simulate any new sensor device. 2) *Testbed visualization*. Sensor nodes are small in size and do not have user interfaces as displays or keyboards, which is difficult to track the testbed communication status. On the other hand, the communication topology in testbed is invisible, but researchers usually need to see the topology to analyze their algorithms. For example, when implementing a routing algorithm in the testbed, the actual routing path is expected to be visible. 3) *Interaction between the simulated WSN and testbed*. The simulated WSN and the real testbed need to exchange information, e.g., routing packet. Their horizontal interconnection, communication, interaction, and collaboration are all emerging difficult problems that need to be addressed.

In this paper, we present an extensible integrated framework of simulation and visualization called NetTopo to assist investigation of algorithms in WSNs. With respect to the simulation module, users can easily

978-0-7695-3431-2/08 $25.00 © 2008 IEEE
DOI 10.1109/FGCN.2008.18

define a large number of on-demand initial parameters for sensor nodes, e.g., residential energy, transmission bandwidth, radio radius. Users also can define and extend the internal processing behavior of sensor nodes, such as energy consumption, bandwidth management. It allows users to simulate an extremely large scale heterogeneous WSN. For the visualization module, it works as a plug-in component to visualize testbed's connection status, topology, sensed data, etc. These two modules paint the virtual sensor nodes and links on the same canvas which is an integration point for centralized visualization. Since the node attributes and internal operations are user definable, it guarantees the simulated virtual nodes to have the same properties with those of real nodes. The sensed data captured from the real sensor nodes can drive the simulation in a pre-deployed virtual WSN. Topology layouts and algorithms of virtual WSN are customizable and work as user defined plug-ins, both of which can easily match the corresponding topology and algorithms of real WSN testbed. As a major contribution of this research work, NetTopo is released as open source software on the SourceForge. Currently, it has more than eighty java classes and 11,000 Java lines source codes. Users can freely download the latest version of NetTopo by accessing the NetTopo website [4].

The rest of the paper is: Section 2 illustrates NetTopo architecture and Section 3 describes features of NetTopo. Section 4 presents one case study provided in NetTopo as examples. Section 5 concludes this paper and describes the future work.

2. NetTopo architecture

From the high-level point of view, NetTopo consists of both simulation and visualization functions. These two functions need to interact with each other and access/manipulate some common resources. For focusing on the integration issues of them, we use component based NetTopo architecture, which is flexible enough for adding new components in the future. The basic architecture is illustrated in Figure 1.

Main Control and *Utility* are two components involved in all layers. *Main Control* is the core component working as a coordinator in charge of the interactions of other components. It can be regarded as an adaptor between input and output interfaces of other components and enables them to work smoothly. *Utility* provides some basic services, e.g., defined application exceptions, format verification, number transforms, and dialogue wrappers.

File Manager is for the purpose of data persistence, e.g., logging runtime information, recording statistical results, keeping references of virtual sensor nodes.

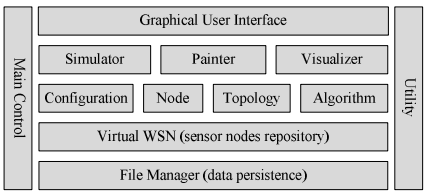


Figure 1. NetTopo Architecture

Log information and statistical results are recorded as character streams into human readable format. References of virtual sensor nodes are stored as serialized format for easy recovery and reuse. All these references are encapsulated in *Virtual WSN*, which works like a runtime sensor nodes repository and also declares interface to allow other components to add new virtual nodes, delete particular nodes, retrieve the same type of nodes and their derived children, etc.

Node, *Topology* and *Algorithm* components are designed as highly extensible modules that can be regarded as plug-ins. *Node* represents a virtual sensor node. Virtual sensor nodes do not have fixed properties or structures. For example, sensor nodes can have very different sensing attributes: temperature, humidity, vibration, pressure, etc. To allow users to create their own virtual sensor nodes, an abstract interface named *VNode* is declared to define several basic methods representing actions of a real sensor node. Any user desired node that wishes to run on the simulator must implement the *VNode* interface. *Topology* stands for the topology to be deployed in *Virtual WSN*. Network topology can be various shapes, e.g., line, circle, triangle, tree. Users can flexibly implement any needed network topology. *Algorithm* represents an algorithm to be applied in the *Virtual WSN*. The algorithm can be any routing, clustering, scheduling, controlling algorithm, etc. Users can freely implement their needed algorithms for their specific studies.

The graphical user interface (*GUI*) in Figure 2 consists of three major components: a display canvas (on the upper left), which can be dragged in case of viewing a large scale WSN, a property tab for displaying node properties (on the upper right), and a display console for logging and debugging information. *Painter* is separated from the main *GUI* due to the frequent paining tasks. The painter is also designed as an abstract interface for various painting requirements, e.g., 2D or 3D. The specific painter used in Figure 2 is Painter_2D. Additionally, the painter encapsulates the lower painting API, interacts with the *Virtual WSN* and main *GUI* and provides advanced painting methods, e.g., it can paint a link between any two nodes by just using their ID information.

Simulator and *Visualizer* represent the high level functions in NetTopo. The structure difference between

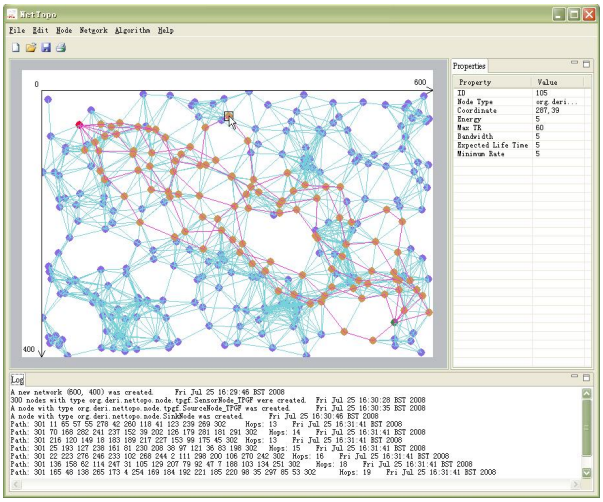


Figure 2. NetTopo main GUI (the TPGF [20] multipath routing algorithm is executed in the WSN)

these two components is that *simulator* is a built-in of NetTopo but *visualizer* is loaded as a plug-in. This is because different accessing interfaces (wrappers) are needed for different devices, e.g., the HTTP based connection is used for getting image streams from wireless camera and the socket connection is used for getting Crossbow sensor data. The common components they all utilize include *Virtual WSN*, *Painter, Node*, *Configuration* and *GUI*. Using these shared resources sometimes can cause synchronization problem, e.g., when both *Simulator* and *Visualizer* components need to add new sensor nodes in the graphical display canvas.

3. Features of NetTopo

Features of NetTopo in current version can be classified in the following four categories: 1) Extensibility. *Configurable sensor nodes with defined attributes*. Users can define their own virtual sensor nodes with expected attributes. New type of nodes will be loaded as a plug-in, which provides an extra choice when users plan to deploy a WSN. *Customizable sensor network topology layout*. Users can define their own topology based on the API described in the *Topology* component. This is helpful when users focus on studying a particular topology of the network. *User-defined algorithms and functions*. An algorithm can be composed of several functions, each of which acts for a particular purpose. User can debug a single function or add a new function without influencing others in the same algorithm. *Device based wrappers for visualization*. A wrapper is used to get information from sensor device. To visualize different hardware devices, users can create different wrappers to set up the connection for extracting information. 2) Flexibility. *Single node deployment*. Users can deploy a single node in a given location. This is useful for a slight modification to the *virtual WSN* or placement for a sink node or source node. *Random multi-node deployment*. Users can randomly deploy a specified number of sensor nodes. The random seed can be the irreproducible current time point of the running computer or any specified reproducible integer. *Specific multi-node deployment*. Users can deploy a specified number of sensor nodes based on pre-defined topologies to form some special shapes, e.g., users can deploy a circle by specifying the location of circle center, radius, and node number. *Repeated node deployment*. Users can repeatedly deploy different kind of sensor nodes in the *Virtual WSN*. This allows the deployment of heterogeneous sensor networks. 3) Practicability. *Data persistence for virtual WSN*. The network deployment state can be saved in a specific type of file using ".wsn" as the postfix. Users can base on these files to reuse the deployed *virtual WSN* or share these files with friends to discuss a common problem. *Snapshot for virtual WSN*. Users can capture a snapshot for a *virtual WSN* and save it as ".bmp" picture. This feature allows users to further analyze the simulation results and use the saved picture for sharing or writing papers. *Node manipulation*. Users can delete specified nodes, view the current properties of the nodes, modify the property values of a node before starting a simulation, search a node by its ID and disable nodes or kill nodes in a specified region to make a hole in the WSN or make an irregular WSN field. *Recording of simulation results*. NetTopo can save the simulation results in a specific type of file using ".report" as the postfix. Users can use normal text editor software to open it and read the simulation result. The simulation results are formulated into a unified format that allows users to further import them into Microsoft Office Excel to get the graphical results, e.g., curves and charts.

4. Case studies

To demonstrate the usability of NetTopo we present one case study on simulation and visualization as user example. In this case study, a testbed composed of Crossbow Mica2 sensor nodes is visualized. Additionally, these real sensor nodes are considered as source nodes in a pre-deployed *virtual WSN*: when the sensed temperature value of any real node exceeds a threshold, which means an event is detected, it then automatically starts a simulation for exploring one/multiple routing paths by using TPGF [5] in the integrated *virtual WSN*. Crossbow WSN testbed

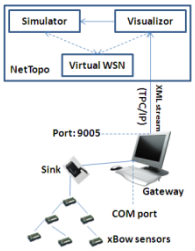


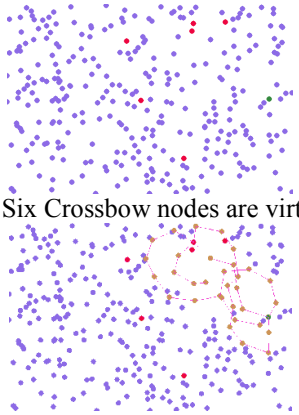
Figure 3. Crossbow WSN testbed visualization flow

consists of six Mica2 nodes. Figure 3 shows the whole network structure and flow of sensed data. The Crossbow driver called xServe is installed in gateway for converting sensed data into XML stream and providing a TCP/IP service on port 9005. NetTopo can be located on gateway or another computer that can communicate with the gateway.

The sink node collects packets sent from sensor nodes. Each packet of any node includes lots of properties, e.g., its node ID and its parent node ID. By using a wrapper to set up a TCP/IP connection on port 9005, NetTopo can read the XML stream from the gateway, extract the node ID information and draw some round circles representing virtual sensor nodes on the canvas. In addition to using *Painter* to update the GUI, this particular *Visualizer* component also creates virtual sensor nodes in the *virtual WSN*, which allows the references of these nodes to be obtained by *Simulator* for using in the simulation. Consequently, by getting the node ID and parent ID mapping information in the XML packet, NetTopo can easily draw the topology of the network connection. Nodes' latest properties and sensed data, e.g., voltage, temperature, humid, pressure, can also be periodically captured from the XML stream by setting a specific sampling rate. The values of all these properties are presented on the property tab of the main GUI and refreshed when new data arrive. Furthermore, these six virtualized Mica2 nodes are considered as source nodes in the *virtual WSN*. When the temperature reading of any Mica2 node exceeds a threshold, the *Simulator* is involved to explore multiple routing in a pre-deployed *virtual WSN*, which include many other deployed simulated virtual sensor nodes. Figure 4 (a) shows the visualization of the six Crossbow nodes in the pre-deployed *virtual WSN*. Figure 4 (b) shows that one of the Crossbow nodes explored four routing paths by using TPGF.

5. Conclusion

In this paper, we present NetTopo, an integrated framework of simulation and visualization for WSNs. The friendly GUI makes it easy to use and the modular


(a) Six Crossbow nodes are virtualized

(b) One Crossbow node explored 4 routing paths

Figure 4. An example of the integration of the testbed and the simulation environment

components enable it to be flexibly extended. NetTopo can support an extremely large scale network simulation by integrating simulated sensor networks and visualized testbed. It is very useful for a fast rapid prototyping of an algorithm.

6. Acknowledgement

This paper was supported by: (in part) the Lion project supported by Science Foundation Ireland under grant no. SFI/02/CE1/I131, (in part) by the European project CONET (Cooperating Objects NETwork of excellence) under grant no. 224053, (in part) by the Nature Science Foundation of China under Grant 60604029, (in part) by Nature Science Foundation of Zhejiang Province under Grant Y106384, and (in part) by Joint Funds of NSFC-Guangdong under Grant U0735003.

7. Reference

[1] B. Karp, H.T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks", in Proceedings of the Annual International Conference on Mobile Computing and Networking (MobiCom 2000), Boston, USA, August.

[2] K. Seada, A. Helmy, R. Govindan, "Modeling and Analyzing the Correctness of Geographic Face Routing Under Realistic Conditions. Ad Hoc Networks", doi:10.1016/j.adhoc.2007.02.008, pp. 855-871, 2007.

[3] K. Aberer, M. Hauswirth, A. Salehi, "Infrastructure for data processing in large-scale interconnected sensor network", in *8th International Conference on Mobile Data Management* Mannheim, Germany, 2007.

[4] http://lei.shu.deri.googlepages.com/nettopo

[5] L. Shu, Z. Zhou, M. Hauswirth, D. Phuoc, P. Yu, L. Zhang, "Transmitting Streaming Data in Wireless Multimedia Sensor Networks with Holes", in Proc. of the Sixth International Conference on Mobile and Ubiquitous Multimedia, Dec. 12-14, 2007. Finland.