Research Proposal

Sumitbhai Jansari
jansas01@student.uwa.edu.au
School of Computer Science and Software Engineering
The University of Western Australia

**Title:** Revised FlexiTP for Wireless Sensor Network

**Supervisor:** Professor Amitava Datta

**Background**

Wireless sensor network consists of many small, low-power and intelligent sensor nodes. These sensor nodes are placed in a physical area of interest for example natural ecosystems, battlefields and man-made environments to gather relative information. These nodes need MAC layer protocols to communicate with each other.

There are several Mac Layer protocol has been implemented based on TDMA, or CSMA, or a hybrid concept of TDMA & CSMA. Such protocols (TRAMA, FLAMA, PMAC, ZMAC) can be adopted at the expense of high-energy consumption for sharing traffic information.[1] It also does idle listening. FlexiTP attempts to address these disadvantages.

**Previous Work**

FlexiTP, designed by Winnie Louis Lee under supervision of Prof Amitava Datta, is a TDMA based protocol with the loose slot structure. Nodes wake up at their scheduled slots, otherwise be in a power saver mode to reduce the idle listening. It is highly energy efficient. Even the designer of the protocol has conducted several testing and proved FlexiTP as a better protocol than the ZMAC protocol.[2]
Problem

FlexiTP requires the data gathering cycle to stabilize the network against the topology changes and network contention. This time synchronization scheme to stabilize the network depends on the length of the fault-tolerance slots.[3] Current FlexiTP protocol has fewer fault-tolerance slots. As we increase the size of fault tolerance slot, it makes the reduction in the length of the communication slots. This will result into high network contention due to less communication slots. So, we have to come with redesigned slot structure to overcome this problem without affecting the network communication channel.

Aim

The aim of this project is to make FlexiTP wireless MAC protocol, which is more robust to fault tolerance and complete. This can be achieved by implementing extensive fault tolerance support in existing FlexiTP. We will implement either of following solutions with

- Equally distributed CSMA & TDMA slots, or
- Might be achieved by giving more preference to CSMA slots, or
- Applying collaboration efforts of CSMA/TDMA slot.

After implementing the algorithm, we will compare it with the existing FlexiTP protocol. Result Analysis and Project outcomes will be discussed at the end.

Method

The following tasks are required to achieve a satisfactory outcome for this project:

1. Initial Phase: In this phase, we will gain the basic understanding of different MAC protocols and FlexiTP protocol. This also involves research on how to make the network more robust and energy efficient and time efficient.

2. Setting-Up the Environment: Set up and configure the NS-2 network simulator environment to review the build the implementation of the FlexiTP protocol.
3. Learning Techniques: Detailed study and understanding of the FlexiTP code are required to produce the revised version. And also get used to with the NS-2 network simulator environment.

4. Designing the Skeleton: We will divide the entire protocol into smaller modules and each module is designed in details. This detailed design document will help us when the theme of the project is lost. So, the aim of the project is achieved.

5. Coding: We will implement the detailed design document with the coding standard.

6. Testing: We will evaluate the test runs of the existing FlexiTP protocol and revised FlexiTP protocol under the same environment.

7. Result Analysis: Compare all the test results and determine the success or failure of the project. In case of failure, analyse the area which went wrong, so future work can be improved.

**Planning**

The task to achieve the goal of this project include

1. Initial Phase – 3 Weeks
2. Setting Up the Environment — 0.5 Week
3. Learning Techniques – 3.5 Weeks
4. Designing the Skeleton – 4 Weeks
5. Coding – 4 Weeks
6. Testing – 2 Weeks
7. Result Analysis – 2 Weeks
8. Documentation/Dissertation – 4 Weeks
9. Poster & Seminar – 2 Weeks
Software Requirements

- NS-2 Network Simulator
- GNU C++ Compiler
- OTcl/Tk GUI Toolkit

References

