NS2 Leach Implementation

Jason A. Pamplin

Purpose

This document outlines what is necessary to install and run the LEACH protocol on version 2.27 of ns2. At the time of this writing, this is the newest version of ns2. The LEACH implementation was written as a stand-alone application. Thus, in the past a version compiled for LEACH may or may not work for other protocols. In addition, the original version of LEACH was compiled for version 2.5b which is an outdated version of ns2. The following goals have been achieved:

- merged and compiled ns2.27 to support LEACH protocol.
- modified code to allow for support of all protocols including LEACH without recompilation.
- validated running of all demos using the ns2.27 validation script.
- Executed LEACH simulation included in the uAMPS changes package (`tcl/ex/wireless.tcl`)

Methodology

The basic method for integration of the leach code was to make all of the changes as specified in the uAMPS changes package to the `ns-allinone-2.27.tar.gz` package. This consisted of using `grep` to find all the code segments marked with `#ifdef MIT_uAMPS` and including them in the proper place in the most ns codebase. This was done incrementally with compilation and testing done after individual segments were converted. This was a very manual process by necessity. In some places the code had to be modified significantly due to changes in the way the latest version 2.27 of ns2 works verses 2.5b of which the original implementation of LEACH was built for.

My test cases during conversion were the `wireless.tcl` and `wireless-demo-csci694.tcl`. Both of these simulations are in the `tcl/ex` directory located in the ns-2.27 base directory. This allowed me to test effects on other wireless implementations as well as the LEACH implementation. Surprisingly, the uAMPS changes assume that you will not be running any other wireless protocols. Thus, straight implementation of the LEACH changes results in Segmentation Faults for other wireless simulations. This has been fixed.

Changes

Adding LEACH support

Below is a list of all the line numbers that were changed in existing ns2-27 code. This does not include the addition of the `mit` directory with additional code.

```
apps/app.cc:53:#ifdef MIT_uAMPS
apps/app.cc:126:#ifdef MIT_uAMPS
apps/app.h:53:#ifdef MIT_uAMPS
common/mobilenode.cc:341:#ifdef MIT_uAMPS
common/packet.cc:49:#ifdef MIT_uAMPS
common/packet.cc:127:#ifdef MIT_uAMPS
common/packet.h:63:#ifdef MIT_uAMPS
common/packet.h:167:#ifdef MIT_uAMPS
common/packet.h:223:#ifdef MIT_uAMPS
common/packet.h:404:#ifdef MIT_uAMPS
common/packet.h:515:#ifdef MIT_uAMPS
common/packet.h:587:#ifdef MIT_uAMPS
```

```
common/packet.h:610:#ifdef MIT_uAMPS
common/packet.h:677:#ifdef MIT_uAMPS
mac/channel.cc:117:#ifdef MIT_uAMPS
mac/ll.h:55:#ifdef MIT_uAMPS
mac/ll.h:110:#ifdef MIT_uAMPS
mac/mac-sensor-timers.cc:5:#ifdef MIT_uAMPS
mac/mac-sensor-timers.h:5:#ifdef MIT_uAMPS
mac/mac-sensor.cc:5:#ifdef MIT_uAMPS
mac/mac-sensor.h:5:#ifdef MIT_uAMPS
mac/mac.cc:95:#ifdef MIT_uAMPS
mac/phy.cc:60:#ifdef MIT_uAMPS
mac/phy.h:93:#ifdef MIT_uAMPS
mac/phy.h:146:#ifdef MIT_uAMPS
mac/wireless-phy.cc:94:#ifdef MIT_uAMPS
mac/wireless-phy.cc:114:#ifdef MIT_uAMPS
mac/wireless-phy.cc:133:#ifdef MIT_uAMPS
mac/wireless-phy.cc:220:#ifdef MIT_uAMPS
mac/wireless-phy.cc:237:#ifdef MIT_uAMPS
mac/wireless-phy.cc:373:#ifdef MIT_uAMPS
mac/wireless-phy.cc:433:#ifdef MIT_uAMPS
mac/wireless-phy.cc:593:#ifdef MIT_uAMPS
mac/wireless-phy.h:52:#ifdef MIT_uAMPS
mac/wireless-phy.h:116:#ifdef MIT_uAMPS
mit/rca/rca-ll.cc:7:#ifdef MIT_uAMPS
mit/rca/rcagent.cc:7:#ifdef MIT_uAMPS
mit/uAMPS/bsagent.cc:8:#ifdef MIT_uAMPS
mit/uAMPS/bsagent.h:8:#ifdef MIT_uAMPS
trace/cmu-trace.cc:59:#ifdef MIT_uAMPS
trace/cmu-trace.cc:1026:#ifdef MIT_uAMPS
trace/cmu-trace.cc:1063:#ifdef MIT_uAMPS
trace/cmu-trace.cc:1171:#ifdef MIT_uAMPS
trace/cmu-trace.h:60:#ifdef MIT_uAMPS
trace/cmu-trace.h:132:#ifdef MIT_uAMPS
```

All of the code blocks added is marked with the `#ifdef MIT_uAMPS` preprocessor directive. Thus, compilation with LEACH support can be done by adding the `–DMIT_uAMPS` flag to the Makefile. These code segments are not necessarily exactly as they appear in the original LEACH implementation. Some of them have been modified significantly to account for changes in the underlying ns2 code for such services as logging and tracefiles. The changes to the code blocks from the original are too numerous to list in the document. The reader may compare each code block with its corresponding file in the uAMPS_changes package. It is important to note that not all code blocks have a one-to-one correlation with the original as some were merged and deleted during the conversion process.

Special Changes

Some changes are significant enough in nature to list explicitly. This does not mean that you need to make these changes if you follow the procedure listed below. These changes could, however, affect the performance or results of the program and are thus listed here so that you can be aware of them when you run your simulations.

add-interface

The method for implementation of the add-interface method of the TCL object library was extended in the latest version of ns2. This resulted in requiring 3 additional parameters whenever a node wished to add an interface. The following code was added at line 207 of the mit/uAMPS/sims/uamps.tcl file.

```
        $node topography $topo

        if ![info exist inerrProc_] {
            set inerrProc_ ""
        }
        if ![info exist outerrProc_] {
            set outerrProc_ ""
        }
        if ![info exist FECProc_] {
            set FECProc_ ""
        }

        # Connect the node to the channel.
        $node add-interface $chan $prop $opt(ll) $opt(mac)  \
          $opt(ifq) $opt(ifqlen) $opt(netif) $opt(ant) \
          $topo $inerrProc_ $outerrProc_ $FECProc_
```

This basically initializes these new required variables to an empty string and calls the add-interface method to meet the requirement for addition of inerrProc, outerrProc and FECProc variables in the function call. This was deemed to be easier to implement than adding an overloaded function to the actual C library. In addition, such overloading would not be in keeping with the intent of the ns2 developers to require these parameters.

TDMASchedule deletions

At the beginning of the simulation, the mit/uAMPS/sims/uamps.tcl attempts to clean up previous simulation results. One of these files is the TDMA schedules. This is done in line 243 with:

```
        catch "eval exec rm [glob $opt(dirname)/TDMAschedule.*.txt]"
```

Unfortunately, this line will fail if there are not matching TDMAschedule.*.txt files in that directory. The following change corrects the problem:

```
     catch "eval exec rm [glob -nocomplain $opt(dirname)/TDMAschedule.*.txt]"
```

Energy Check

In mac/wireless-phy.cc a new object is created explicitly for the LEACH protocol. This is basically the energy aware node. However, if other wireless implementations do not call the attach-energy function for a given node, the energy_ object is never initialized. This is not a problem except that the sendUp and sendDown functions attempt to adjust the nodes' energy value. This results in an access violation. At line 108, the energy_ variable is initialized to 0 and any place this variable is accessed a check of if (energy_) is used to be sure that the energy_ variable is not access unless the energy_ object has indeed been created.

Setup Procedure

1. Obtain the ns-allinone-2.27.tar.gz package. This can be found at:
   http://www.internetworkflow.com/downloads/ns2leach/ns-allinone-2.27.tar.gz
2. Unpackage the archive
   ```
   gunzip ns-allinone-2.27.tar.gz
   tar –xvf ns-allinone-2.27.tar
   ```
3. Change directory to ns-allinone-2.27
   ```
   cd ns-allinone-2.27
   ```

4. run the install script
   ```
   ./install
   ```
5. Add the appropriate environment variables as specified at the end of installation.
6. Obtain the mit.tar.gz package at http://www.internetworkflow.com/downloads/ns2leach/mit.tar.gz .
7. Place the mit.tar.gz package into the ns-allinone-2.27/ns-2.27 directory.
8. Change directory to ns-allinone-2.27/ns-2.27
9. Unpackage the archive which will overwrite all appropriate files and set up symbolic links.
   ```
   gunzip mit.tar.gz
   tar –xvf mit.tar
   ```
10. A sample make file can be found at http://www.internetworkflow.com/downloads/ns2leach/Makefile. Edit the Makefile as follows:
    - Add –DMIT_uAMPS to the DEFINE list
    - Add –I./mit/rca –I./mit/uAMPS to the INCLUDE list
    - Add the following just prior to the line `gaf/gaf.o \`
      ```
      mit/rca/energy.o mit/rca/rcagent.o \
      mit/rca/rca-ll.o mit/rca/resource.o \
      mac/mac-sensor-timers.o mac/mac-sensor.o mit/uAMPS/bsagent.o \
      ```
11. Clean up previous build
    ```
    make clean
    ```
12. Rebuild ns2 – this can take a while so I recommend redirecting output, running in background and going to lunch.
    ```
    nohup make 2>error.log >make.log &
    ```
13. Test default wireless demo and LEACH
    ```
    ./test
    ```
14. Validate the full installation – this takes a while too so take a break
    ```
    nohup ./validate-full 2>validate.error >validate.log &
    ```

References

[1] Wendi Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. "Energy-Efficient Communication Protocol for Wireless Microsensor Networks." Proceedings of the Hawaii International Conference on System Sciences, January 4-7, 2000 © 2000 IEEE.

[2] MIT µAMPS project ns2 code extensions. http://www-mtl.mit.edu/research/icsystems/uamps/research/leach/leach_code.shtml

[3] Marc Greis' Tutorial for the UCB/LBNL/VINT Network Simulator "ns" http://www.isi.edu/nsnam/ns/tutorial/

Future Work

- Create a complete setup script for performing this installation to make this exceptionally easy. There are a lot of people out there who have tried to do this and even more who want too. It would be good publicity for the school.