Description of OPNET Usage in MURI Research

The Multidisciplinary University Research Initiative (MURI) project that is undertaken at the University of Michigan is to investigate the various algorithms and protocols at different layers of future army communication systems from a low energy perspective. The ultimate goal of MURI research is the global optimization and simulation of the overall communication systems that may involve different layers altogether at the same time and the demonstration of the difficulty and attack of the research problems of such large scale. The MURI project involves eleven faculty and over twenty graduate and undergraduate students.

The communication system that we are investigating may be partitioned into three different layers, namely, the network layer, the process layer, and the device layer. The global optimization and simulation of the overall communication system usually involves too huge dimension, which is hardly tractable for the initial approach. As a consequence, an integration of optimization and simulation has to be taken at each different layer at the moment in order to achieve some idea of performance versus energy consumption. As we have seen, even after such decomposition, the optimization and simulation problem is still not easy at all to deal with.

At the network layer, we have set up a simple scenario which can be defined as a position estimation task. Multiple radio nodes are initially deployed in a drop zone and moves towards the same destination according to a drift model. The drift model is at present defined as follows: each node calculates a direction vector from its current position to the destination and its next position is its current position plus a distance along this direction vector and another random vector whose entries are uniformly distributed. Each node broadcasts packets containing its position information periodically while traveling towards the goal. During each broadcasting, a node retransmits the same packet for a certain number of times based on the predetermined retransmission probability. The transmission power is a fixed value for each packet transmission regardless of distance between the nodes during the course of a single simulation.

Each node updates its estimate of each of the other nodes' positions and uncertainty regions periodically. When a node receives a packet from another node, it updates its position estimate of the other node whose packet has just been received and the corresponding uncertainty region collapses into a single point, which is equal to the other node's current position. On the other end, if a node does not receive a packet from another node, it updates its estimate of the other node's position based on the direction vector from its estimated position of the other node toward the destination and the corresponding uncertainty region of the other node grows under the worst case assumptions.

Each node has a fixed battery capacity used purely for transmission. If a node exhausts its battery energy before reaching the destination, it can not broadcast its packet to inform the other nodes of its position information. The condition for terminating a simulation is when any one of the nodes reaches the destination.

The system performance is measured by the average mean squared error between a node's real

position and its estimated position as seen by another node. The average is taken over all nodes in the communication network and over the simulation time. At the moment, there are three parameters considered in determining our performance measure, namely, T, the transmission interval, q, the retransmission probability, and P, the power of each packet transmission. Of course, we will later take in more parameters in our decision space, which may be incorporated from other layers. We want to optimize performance with respect to these many variables. However, even with the current three parameters, it is very hard to come up with a closed form expression of performance versus parameters, which is usually required by a optimization program.

Our optimization program utilizes the Hide-and-Seek simulated annealing method, which requires a well defined objective function. We want to provide the optimization program an illusion that there exists an objective function and we decided to achieve this via simulation. Since OPNET is becoming a defacto standard in industry and is very powerful and flexible, it is certainly our first choice to choose as our simulation tool.

We first finished simulation of our basic network model in OPNET for its own testing purpose, after which we interfaced the OPNET simulator with the optimization program. The communication of the optimization program and OPNET simulator works as follows: for each parameter set generated by the optimization program, OPNET simulator runs our network model over this parameter set many times with a different random seed at each time and returns the performance measure, in our case, the average mean squared error of estimated position and real position. These average mean squared errors are averaged again over the number of times that OPNET runs over the same parameter set and the resultant value is returned to the optimization program as the value of the objective function evaluated at the parameter set provided by the optimization program. Upon receiving the value of the objective function, the optimization program generates another set of parameters based on its own internal algorithm. The above process keeps going until the number of iterations in the optimization program is reached. The best parameter set and the corresponding function value up to the current number of iterations is then given by the optimization program.

Since our simulation in OPNET is quite computationally intensive and we need to take the performance measure for the same parameter set but over different random seeds, it is natural to parallelize the simulation in OPNET for the same set of parameters on different machines, with each machine running an OPNET simulation with a different random seed. The performance measure returned by each machine is then averaged and returned to the optimization program, upon which a new parameter set is decided by the optimization program for evaluation. By parallelly running OPNET simulation on different machines, the time span to get the final optimization result is greatly reduced.

We are going to build more complicated network models that may involve many parameters from different layers and we will certainly further investigate the sensitivity of performance measure with respect to different parameters, all of which will in turn require more simulation and optimization process.