6 Conclusion

NTS – Network Topology Simulation is a success. NTS simulates a store and forward network topology and is able to run PVM programs with only minor modifications. NTS implements standard communications over the simulated network (which can be as simple or as complicated as the user wishes) and much more.

A simple distribution is provided for download and installation from the VUB Parallel Laboratory website. The proposed overloaded PVM communication library superimposes over the standard PVM with ease. Converting PVM programs (which meet the restrictions imposed on using NTS) is fast and simple. No configuration of NTS is needed within the Client code, all configurations happen via the separate configuration program. The ntsMaster configuration program provides an easy but complete Graphical User Interface to access all facets of NTS with ease, and provides many helpful tools to run and analyze results from experiments run on the simulated network. The integrated control provided by the Master melds NTS's working modes seamlessly into one, always providing access to all the options.

The underlying simulated network uses Routing Principles from RIP and Parallel Processing networks to be stable, and dynamic, adapting to new constraints automatically. The Master program is designed for extensibility: most of its primary functions are in the form of plugged-in subsystems.

While the main aims of NTS have been reached, the limitations imposed for its use, limit the parallel algorithms that can be executed on it. The main limitation is the inability to listen to any incoming message due to sequencing problems. The Master being designed for easy extensibility only provides a limited set of analytical tools. More tools were originally planned but due to synchronization problems between Routers and the Master program, which caused a delay in development these packages were never included. The underlying network being robust once running, it is the setting up of the network that is prone to disruption. A single problem in instantiation of Routers can compromise the whole network structure during the setting up stages. These problems do not create "crashes" however they may cause the network to become unstable. The only way to return to a stable condition is to reset PVM and restart the simulation.

It is believed that even though NTS has some restrictions, it will be a helpful tool for programmers to test the effect of various network topologies on their parallel algorithms.

6.1 Further Development

6.1.1 Listening To Any Incoming Message

One of the main restrictions to using NTS is the fact that the "-1" tag cannot be used as a wild card for receiving the first incoming message, which meets the requirements. In PVM, both the source and the message tag can have wild tags. This restriction is explained in detail within the Design section of the ntsLibrary.

This restriction was already present in the first incarnation of the AOM project. An attempt at bypassing this restriction was tried in the early development stages of NTS using a separate listening thread to "log" all incoming messages into a database which could be queried by the overloaded communication functions to retrieve the desired message for the Client. This technique though promising has one fundamental flaw. PVM receive functions automatically change the received message buffer to the default buffer. As a separate thread would be used to "log" incoming messages separate to the client line of calculation, there would be no guarantee that Client code would not be working with the default buffer when this would untimely be changed because of an incoming message. PVM also has proven that it is not very robust while being used in various threads (refer to section 4.7 Known Bugs).

An alternative approach, though a bit complex is proposed: When the client asks to receive a message, NTS listens for the first incoming message ("-1" tag in both source and message tag). This message is then open and compared with the current technique of combined parameters to see if it is the required message. If it is not, it is saved in a log and a new message is listened for. The log should rearrange messages by sequence and order of arrival. Once the requested message arrives, the message is returned. The next time a message is requested by the client, NTS will first check whether the message has not been logged first.

For listening with wild cards, the log would first be checked for any correspondents, the first match would be returned (as the messages are ordered). If no

match were found, NTS would listen to incoming messages, logging them until a match occurs. A match of course compromises the next sequence required.

This technique requires lots of book keeping, and multiple checks at all times to make sure the required message is given back. Problems would start to happen if a client were to receive more messages then the sequence limit. These would have to be logged according to time of sending as well as order of arrival. This would require extra data to have to be included in the forwarded message.

At the time of writing, no easy solution to the problem has been found.

6.1.2 Efficiency

During Design, the main utility conceived for the Master would be the user adding and removing network components, either during the initial building phase, or dynamic editing. The structure representing the network was therefore designed with ease of adding and removing elements from it. Linked lists were therefore used to keep the elements of the structure.

As it turns out, in a Graphical User Interface, it is the various views which access the data structure a lot more often, as an example, while selecting an item, each view accesses the Data in turn to get the attributes of the newly selected item it requires for a repaint. The views rarely need to add or remove components from the data structure; they just need to access it.

The views also hold more information about the elements they would like to access, more then a user would. While a user would have to input a name that would be searched through the data structure, the views have access to the identities of the elements given to them on creation, which are unique to the session. Because of the underlying structure being linked lists, views have got to access elements through searches through attributes like a normal user would, dropping the efficiency of the program.

A rewrite of the underlying structure to use a “map” with the identity of the elements used as the key would cut the searches from views and greatly enhance the overall efficiency of the program.

6.1.3 Robustness

As explained within the Unfinished Business, the aim of having a robust simulated network was not completely reached. The simulated network while not "crashing" as such for most errors, gets into a state where a restart must be done for the system to get back into a stable system. This happens mostly from bad use of PVM functionalities due to limitations. The program should be revisited with the considerations of the limitations of PVM to make the system more stable.

6.1.4 Extensions

More analytical tools were planned for the ntsMaster program. These due to lack of time were never included with the package. The basic tools and data for extra analytical subsystems are present within the package. With the design for extensibility of the Master program designing and including these tools should be simple.

The main project of extension was to include a graph render of various aspects of the network simulation that were recorded. Parameters of the current simulated topology could then be calculated, such as the **Time per Hop** by graphing the average time taken for the number of hops.