Infrastructure Optimizer Modeling of Networks

HyPerformix, Inc.
4301 Westbank Drive
Building A, Suite 300
Austin, TX 78746
www.hyperformix.com

Abstract
This whitepaper presents an overview of the HyPerformix Infrastructure Optimizer™ network model.

Introduction

The Optimizer GUI allows computers to be connected by network devices. Two computers exchange messages across these network devices. This paper presents an overview of the Optimizer network model. We discuss some of the general issues that arise in simulating data flow across a network and how Optimizer addresses these issues.

Network Devices

Optimizer classifies network devices into two categories: datalinks and interconnects. Datalinks include: Ethernet, token ring, FDDI, generic datalink, point-to-point, frame relay and basic WAN. Interconnects include: routers, switches, and bridges – although the functionality of these devices is significantly blurred in contemporary products. Switched Ethernet belongs to both the categories as it is an abstraction that encapsulates an Ethernet switch and its dedicated point-to-point Ethernet segments. A computer can be directly connected to a datalink, but it cannot be directly connected to an interconnect. A datalink can be directly connected to an interconnect. Two datalinks cannot be directly connected; two interconnects cannot be directly connected. (A connection between a computer and a switched Ethernet or between a switched Ethernet and an interconnect implies an Ethernet datalink.)

Network Messages

A message transferred between ADN (HyPerformix's modeling language) processes and threads running on two different computers by a '*Send*' statement or a '*Service Reply*', leads to a transfer of the message across the network. Another source of network messages is a read or write of data on a File or Volume assigned to mass storage on a computer other than the one where the Read or Write is executed. This is also referred to as "remote I/O".

Message Routing

For two computers to be able to transfer messages they should be connected by a sequence of network devices. The types of these devices will be in the sequence datalink - interconnect - datalink - ... - interconnect - datalink. At the beginning of simulation, Optimizer processes the topology diagram of the model and constructs a routing table that contains the least time route between any two datalinks. The time of traversing a route is computed as the sum of the time of traversing each network device in the route. For a datalink, the time equals the time required to transfer a 1500 byte packet across the datalink in absence of any other dynamic load. For an interconnect, the cost equals the time taken to transfer a packet across the interconnect in absence of any other dynamic load. The following formulas detail the cost calculations:

Component	*Cost Formula*	*Description*
Datalink (except for basic WAN and Frame Relay)	1500/(BW * (1 – b)	1500 = a typical 1500 byte-packet, BW = bandwidth, b = background load
Basic WAN and Frame Relay	L	L = latency
Interconnects	1/(T * (1 – b))	T = Throughput, b = background load

During simulation, when a message needs to be transferred between two computers, the routing table is used to select which network route is to be taken for transferring the message. This is the default route. The user may override this route selection through the use of network policies.

Network Packets

Network messages are divided into packets. These packets are transferred between computers through datalinks and interconnects. Packet size is set according to the network protocols in effect. The maximum size of a packet is determined as the smallest of the maximum payload size permitted in the datalinks in the default route from the source to the destination. This is in accordance with the path MTU protocol used in the Internet. Optimizer simulates the flow of a packet through the route and computes the delay suffered by the packet at each network device in the route. The delay at a device depends on the presence of other network traffic in the device and is explained in detail in our documentation on network devices. By default, all packets of a message traverse the same route through the network. The user may specify different routes for packets within the same, non-TCP/IP message through the use of network policies.

Network Policies

Network policies enable the user to override default network routing behavior for selected packets and to study their effect on end-to-end performance. This customization typically begins with the definitions of policies. These policies are analogous to policies found in many Policy Based Management Tools (PBMT) like Checkpoint Floodgate or HP Policy Expert. The policies state conditions for selecting packets and special network processing actions for these selected packets. The special processing could include customized routing path specification, load balancing and prioritized routing. These policies may be applied to individual packets or to entire messages.

Simulation of Transport Layer Features

In simulating the flow of a packet along its route, Optimizer takes into account standard transport layer features that impact network performance. However, it ignores features that have insignificant impact on network performance. For example, packet loss detection and retransmission of lost packets in TCP are not simulated. While packet loss does happen in practice, it's very rare during normal operation of a well-designed network. Thus, it has insignificant impact on performance. It is appropriate to state that Optimizer model outputs include packet population statistics of network devices which can be used to detect the build-up of congestion in a network. Steps taken to avoid congestion in the network will result in avoidance of packet loss.

Optimizer simulates a window-based congestion control algorithm. It is similar to the congestion control algorithm used in TCP. The Optimizer algorithm permits a source (an ADN process/thread) to send out

a maximum of 16000 bytes of data in multiple packets at a time. After these packets are sent any other source of data in the same computer gets a chance to send a maximum of 16000 bytes. This is transparent to the ADN application and simulated in the Optimizer network model.

The result of running a congestion/flow control algorithm is pacing or a pause between data segments. These pauses prevent a slow receiver from getting overwhelmed, mitigate congestion in the network, and allow multiple sources from the same computer to transfer data without one of them dominating the network. The Optimizer practice of letting a source send a maximum of 16000 bytes before yielding to other sources in the computer simulates the congestion control behavior of a TCP/IP network. Congestion control plays a role in network performance only when large messages are exchanged between computers. For client-server applications exchanging short messages, network traffic is naturally paced by application behavior. So network performance is not significantly impacted by an explicit congestion control.

Simulation of Protocol Overhead

As a message is sent from the source to the destination, Optimizer takes into account the overhead associated with the protocols roughly corresponding to OSI protocol layers 2 through 4. For example, for sending a message over TCP/IP, appropriate TCP header overhead bytes and IP header overhead bytes are included in the total data to be transmitted. When a packet is transferred over a datalink, the frame overhead bytes of the datalink are included in the transmission.

Simulation of Packet Construction Time

Some CPU time is needed at the source and destination computers to construct and deconstruct a network packet to transfer data. This time is used to allocate memory for the packet, determine/compute values for header and trailer fields, and assign to the fields. For example, to transfer data using TCP/IP across an Ethernet, the source computer has to construct a TCP header, an IP header, and an Ethernet header.

Optimizer does not currently account for this time in simulation of a packet flow. The primary effects of the packet construction/deconstruction time are modest increases in CPU utilization and end-to-end delay of a packet. Normally in model construction the associated CPU time is subsumed in the measured CPU utilization of the application. If these effects are believed to be significant for a model and are not accounted for in other measurements, the execution of an appropriate number of CPU instructions as the packet construction/deconstruction time can be added to the model with an ADN Execute statement associated with the message Send and Receive. The packet construction time is independent of network topology.

Simulation of TCP Connection Set Up Time

If you are simulating a TCP connection, then it may be necessary to account for some time for setting up a TCP connection. This time falls into two categories: local setup processing and TCP connection establishment. Local setup processing normally includes mapping a host name to an Internet address, allocating a socket, binding to a port and exchange of a pair of short messages between the originator and the message destination. Optimizer does not specifically account for this time. If the effect of either or both of these is believed to be significant for a model, they can be modeled by the following:

setup	at the beginning of a TCP/IP session add an execution of an appropriate number of CPU instructions with an ‘Execute’ statement

connection	at the beginning of a TCP/IP session, exchange a short (e. g., 40 bytes) message and reply between the originator and the destination application/service.

A related issue is TCP "slow start." This occurs when a new TCP connection is established and an appropriate packet "window" size is not established. The originator will send two packets and wait for an acknowledgement of receipt (ACK) and then send four packets and wait for an ACK and continue doubling the number of outstanding packets until packet loss occurs or until the originator packet window size is achieved. Optimizer does not model TCP slow-start. In most applications, TCP connections are sustained for sufficient time that neither connection setup nor slow-start are significant contributors to perceived system performance.

Finding Answers to Network-related Questions

Optimizer provides easy access to a wide variety of statistics: utilization, throughput, response time, and population. Packet population statistics of network devices can be used to detect build-up of congestion in a network. Utilization will report whether the network component is operating outside the nominal range. Data throughput reports throughput for data bytes. Protocol headers are not included. Total throughput metrics includes both data and protocol overhead bytes. These statistics are reported for each network component and may be further subdivided into per-client or per-process basis to help isolate the performance over the network of a specific application.

Further Information

Two other documents that supplement this paper are available. They give details of how network devices are modeled in Optimizer:

1. Infrastructure Optimizer Modeling of Data Link Components
   http://www.hyperformix.com/whitepapers/modeling-datalinks-whitepaper.pdf
2. Infrastructure Optimizer Modeling of Interconnect Components
   http://www.hyperformix.com/whitepapers/modeling-interconnects-whitepaper.pdf

