Packet switching computer network: An Empirical approach

1058561

Abstract

This paper discusses the use of Empirical Modelling to demonstrate the computation of shortest path in a packet switching computer network. The shortest path between two stations depends on the congestion of the transmission links and the path with the minimum congestion is selected for the transmission of data. The model can be used by students of a networking module as they can change value of observables and see the effect of it in the model.

1 Introduction

A Computer network is a collection of computers interconnected by communication channels that facilitate the sharing of data. Understanding the algorithms behind the network is a challenge to the students of a networking course. Empirical Modelling with its emphasis on constructivist stance makes it a good tool for "Learning by Doing" which is essential for the beginners to comprehend the intricacies of the algorithms that power a computer network.

This paper will first discuss about computer networks, empirical modelling and its role in constructionist learning. The subsequent sections will describe the model and an analysis of it and how well it meets the desired objectives.

2 Overview

This section will briefly introduce the concepts in computer network and empirical modelling and its application to constructionist learning.

2.1 Computer networks

A computer network is made up of devices interconnected by a network of intermediate switching nodes. The main objective of the switching nodes is the transmission of the data from the sender to receiver. In order to fulfil this requirement they provide a switching facility that will move data from the sender through several nodes till it reaches its destination. The nodes are connected to each other in some topology by transmission links. The end devices that wish to communicate are sometimes referred to as stations and they can be computers, terminals, telephones or other communicating devices. In this project the sender is a client computer and the receiver is a server machine[2].

There are two traditional approaches to connect stations, namely circuit switching and packet switching. Circuit switching network is generally used when transmission of data between the two stations require a dedicated communication path. The model described in this project is based on packet switching networks which are generally used in internet.

Packet switching networks are more responsive, robust and flexible as compared to circuit switching network. The client machine breaks the message into packets and sends these packets, one at a time, to the network. There are two approaches used in contemporary packet switching networks to route the packets from sender to destination machine. These are virtual circuit and datagram. Both these approaches are illustrated in the model developed in this project. In the virtual circuit approach, a pre-planned route is established before any packets are sent. Once the route is established, all the packets between a pair of communicating parties follow this same route through the network. While in the datagram approach each packet is treated independently, with no reference to the packets that have gone before through the node. In both the approaches the route is determined by using Djkistra's shortest path first algorithm which takes into account the weights between the nodes and returns the path with the minimum overall weight. In case of virtual circuit the shortest path is calculated before sending the data from the client machine to the server, while in datagram the shortest path is calculated at each node. Therefore datagram technique is more responsive to any change in the weight of the transmission links as compared to virtual circuits. In the model weight of a transmission link corresponds to the congestion and latency of that link in the real network.

2.2 Constructionist learning and Empirical Modelling

One of the most interesting aspects of using computers in the field of educational technology is with respect to constructionist learning – the idea of "learning by doing". By actively participating in a field on their own terms, rather than being taught its principles directly, students construct their own knowledge of the field based on experience. The style of constructionist learning is getting more and more acceptance with the learners because it promotes a more thorough understanding.

Empirical modelling paradigm focuses on the building of construals rather than programs. A construal is an artefact with which the user can interact in order to understand a phenomenon. Empirical Modelling promotes the use of "thinking with computers," using the computer to aid thought development and explore a modelled universe. Thus as a field of research empirical modelling offers clear benefits in constructionist learning. The main advantage of learning with a construal is that a property of the construal can be defined and redefined to study its effect on the phenomenon and this approach fits well with the "what would happen if I..." approach to experimenting with a model. Modelling a packet switching network in empirical modelling will help the students in understanding the two main switching techniques and also the shortest path first algorithm.

Empirical modelling encourages the use of programming models using observables, agents and dependencies between the observables. An observable is some feature of a situation to which a value or status can be attributed. Empirical procedures and conventions are involved in identifying a particular observable and assigning its value. An agent is defined as a family of observables whose presence and absence in a situation is correlated in time, which is typically deemed to be responsible for particular changes to observables. All changes to the values of observables in a situation are typically construed as due to actions on the part of agents. A dependency is defined as a relationship between observables that pertains in the view of a particular agent. It expresses the observation that when the value of a particular observable *x* is changed, other observables (the dependents of *x*) are of necessity changed in a predictable manner as if in one and the same action.

3 Description of the model

Designing a model in empirical paradigm encourages a method of abstraction away from the core real world system which is to be modelled in order to create a finite set of controllable objects within an effectively closed world. This abstraction allows the system to be simplistically modelled while still allowing for a realistic exploration of the state space. Abstraction can best be shown through the use of agentification techniques designed to find the core agents within a complex system, i.e. those objects within the system with the power to alter the state of the model[3]. In order to emulate the real world scenario like non-determinism it is important to introduce some randomness in the model. In this model the weight between the nodes changes randomly with time signifying the change in network parameters like network congestion and latency between two nodes. The weight between the nodes also acts as an agent in the datagram packet switching network. This can be proved by the fact that change in weight between two nodes affects the state of the model in the sense that the shortest path which was initially determined between the client and the server changes so that the total weight throughout the path is minimum.

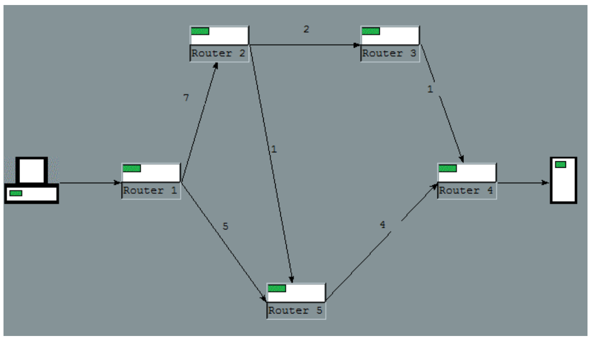
One of the important points in the use of empirical techniques is the recognition that the system modeller has a view point on the system, and thus an unconscious bias towards artefacts and agents which make sense in the context of their experiences. Thus a target observer might have different opinion about the model as compared to the modeller. The user can define and redefine observables based upon his observation.

3.1 Use of EM notations

In developing the EM model, the EDEN, DoNaLD and SCOUT notations are used extensively. EDEN, which stands for the Engine of DEfinitive Notations, is the primary tool of use for building EM models. EDEN implements a variety of notations, which includes DoNaLD and SCOUT. DoNaLD (Definitive Notation for Line Drawing enables the development of 2D graphics within the EDEN environment. DoNaLD is used in order to develop the EM model to be visual and make use of coordinates for placement of objects. The transmission links between two nodes are lines drawn in DoNaLD. SCOUT, the definitive notation for SCreen LayOUT, is used to define the screen layout and also defining artefacts to enable interaction within the model[1]. This includes the various buttons for selecting nodes and updating weights between the two nodes, buttons for sending request to server and also the button for auto-generating requests.

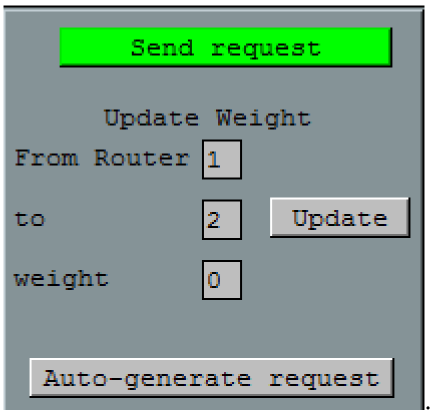
3.2 The model in detail

The model created consists of a client machine connected to a server machine through a network of routers. The routers are connected through a transmission link. Currently there is connection between following pairs of routers : (1,2), (1,5), (2,3), (2,5), (3,4) and (5,4). A proposal to add a feature for extending the network by adding new routers and new transmission links has been mentioned in the proposed extension of the model.



The numbers shown besides a transmission link is the value of the weight of that transmission link. The weight signifies congestion or failure of a transmission link. The higher the value of the weight the more will be the congestion. Whereas failure of a transmission link is specified by assigning a weight equal to zero to that link.

Weights can be assigned manually by the user through the input boxes provided in the control panel. Apart from manual assignment, the values of weight can also change automatically with time. This signifies the dynamic behaviour of the network



To simulate continuous flow of data from the client to the server there is a provision of sending automatic generated requests which sends requests every 3 seconds.

The type of the packet switching network can be changed by using the buttons at the top of the model. The difference between these two modes is that in virtual circuit once the shortest path is calculated the packets flow through that path, independent of any later changes in the network like increase in congestion in a transmission link in the path. While in datagram mode shortest path is computed at each node, therefore depending upon the congestion in the network the shortest path can change from the initial shortest path computed between router 1 and router 4. Information regarding what's happening when user selects datagram mode is shown in the console. It would be best to switch of auto-generation of requests when in datagram mode, so that the user can understand easily what's going on in the computation of shortest path at each router.

One of the large implementation differences between *tkeden* and various other modelling solutions is the concept of dependency. This dependency allows for system wide change from the alteration of a single variable state. The model has dependency between the weights and the shortest path computed by the Shortest Path First algorithm.

4 Evaluation

4.1 Empirical modelling principles and tools

The division of a model into observables, dependencies and agents provide a realistic and powerful method for studying a real world phenomenon. Defining dependency between observables helps a programmer to reduce effort invested in synchronising dependent variables.

But the lack of creating an object as possible in object oriented languages like Java, C++ introduces a huge drawback. There is also no scope of creating a template, based upon which new instances can be created. It must be possible in an object to define variables, methods acting upon these variables and also dependencies between different variables. For the creation of new instances a programmer has to copy many lines of code with just some minor changes. This reduces the modularity and reusability of the code.

One of the major cause of frustration while developing the model was the difference of syntax in different notations. Some notations like EDEN have loosely typed variables while others like SCOUT have strongly typed variables. SCOUT has a very different syntax and appearance as compared to other notations.

Another problem encountered was with integrating variables in different notations in Eden. In Eden, all observables share one global namespace and when introducing new notations, some use this namespace directly (e.g. Scout), while for accessing some a prefix needs to be added to all observable names (e.g. Donald).

4.2 Limitations and proposed extensions of the model

The model developed in this project is based on a huge simplification of the actual processes involved in a packet switching computer network.

The most significant simplification is the absence of handshake process that occurs between station and nodes and also between nodes while establishing the communication channel. In the TCP/IP protocol suite TCP uses three-way handshake to establish a connection:

1. **SYN**: The active open is performed by the client sending a SYN to the server. It sets the segment's sequence number to a random value A.
2. **SYN-ACK**: In response, the server replies with a SYN-ACK. The acknowledgment number is set to one more than the received sequence number (A + 1), and the sequence number that the server chooses for the packet is another random number, B.
3. **ACK**: Finally, the client sends an ACK back to the server. The sequence number is set to the received acknowledgement value i.e. A + 1, and the acknowledgement number is set to one more than the received sequence number i.e. B + 1.

At this point both the client and server have received an acknowledgement of the connection and the connection is open for data transfer[2]. The model built in this project assumes an open connection when animating the transfer of data.

Another simplification is the fact that in datagram packet switching network all the packets do not flow through the same transmission link. Different packets might take different route and it is the responsibility of the exit node (Router 4 in the model) to re-arrange the packets according to their sequence number and if there is any packet missing then the node requests a copy of the packet from the client[2].

The model also lacks the ability of adding transmission links between existing nodes and also a feature for adding new nodes in the network. It would be helpful if a drag and drop feature could be added to the model through which new routers could be introduced in the network.

A principle extension of the model will be breaking the data from the client into packets and then transmitting these packets through the transmission links. The size of each packet will depend on the Maximum Transmission Unit (MTU) of the shortest path calculated. The path MTU is the largest packet size that can traverse the path without suffering fragmentation. Details of how path MTU can be calculated can be seen in RFC 1191 (for IPv4) [4] and RFC 1981 (for IPv6) [5].

The interface of the model also needs to be enhanced so that the output of the algorithm is displayed on screen. Currently all the information related to shortest path computed and the errors in input is shown in the output console.

Acknowledgements

I am highly thankful and grateful to Meurig Beynon for helping whenever I faced problem during the implementation of the model and also happily discussing the idea of packet switching network from the empirical modelling perspective.

References

[1] Beynon, M. EDEN - the Engine for DEfinitive Notations. Retrieved February 2009, from Computer Science > Empirical Modelling: *http://www2.warwick.ac.uk/fac/sci/dcs/research/em/software/eden/*

[2] Stallings W. Computer Networking with Internet Protocols and Techonology, 2009

[3] O'Gorman. Grid Computing : An Empirical Perspective, WEB-EM -01, 2005,(*http://www2.warwick.ac.uk/fac/sci/dcs/research/em/publications/web-em/01/grid.pdf*)

[4] RFC 1191 (*http://tools.ietf.org/html/rfc1191*)

[5] RFC 1981 (*http://tools.ietf.org/html/rfc1981*)

[6] Yung, S. (1991). *Room viewer*

[7] O'Gorman (2005). *Grid computing*