Distributed Computing - II: CS5320

Programming Assignment 2: Termination Detection

Assignment Report

Sagar Jain - CS17BTECH11034

March 15, 2020

Contents

Program Design 2
Spanning Tree Algorithm 2
Message Optimal Algorithm 3
Program Output 4
Spanning Tree Algorithm 4
Message Optimal Algorithm 4
Results & Graphs 5
Spanning Tree Algorithm 5
Explaination of Results 6
Message Optimal Algorithm 7
Explaination of Results 7

Program Design

Spanning Tree Algorithm

Cell Struct

The cell struct represents each node in the distributed network. The following constitutes the cell.

1. A vector of all the children nodes of the node i.e. **children**.
2. A boolean array which represents if the tokens were received from a particular node i.e **tokenReceived**.
3. A boolean which represents if a node has a token or not i.e. **haveToken**.
4. **cellColour**, the colour of the cell.
5. **nodeColour**, the colout of the node, which also ends up becoming the colour of the token which the node sends to its parent.
6. A mutex lock which is used to achieve mutual exclusion every time data of this cell is being accessed i.e **lock**.

Cell Process

This is the function which every cell runs when it is launched, it also launches another process which is used to receive messages from other threads. The following are the main tasks of this function.

1. Launch the receiving threads.
2. Wait for the listening threads to begin listening before beginning to send messages. This is done using **pthread_cond_wait**.
3. Start sending encoded messages of the colour of the cell (red or blue) to a corresponding number of the neighbours of the cell.
4. After each round of sending messages, check if the cell has received tokens from all the children and if so, if the cell itself is currently blue, if both theses condition are satisfied then send a token of the colour of the node (**node colour and cell colour are different things**) to the parent.

5. Once a token is sent to the parent, the **receivedToken** vector has all its entries sent to false as we must wait for all the childrent to send tokens once again before this cell can send a token to its parent,

Receiver Process

This function is responsible for handling the receiving of messages from other cells, and taking appropriate action on the receipt of said messages. The main tasks of this function are highlighted below:

1. The receiving process first sets up the listening server, and after each server is set up it increments a variable in the critical section, using a **pthread_conditional_variable** we can broadcast to all the sending threads that they can begin sending messages once the value of this variable is equal to the number of cells.

2. On receiving a message the receiving thread decodes the contents of the message, i.e. the sender id and the encoded digit which signifies the colour of the node cell or the colour of the token. A message can either be one which contains a token or one which contains the colour of the sender.

3. After decoding the contents of the message a given cell, changes or doesn't change its cell colour and/or node colour based on the contents of the received message.

Miscellaneous

The following is the message encoding format and meaning:
Every message begins with the sender id, this is followed by a delimiting #, that is followed by one of the following numbers,
1 - White Cell
2 - Red Cell
3 - Blue Cell
4 - White Token
5 - Black Token

Message Optimal Algorithm

Program Output

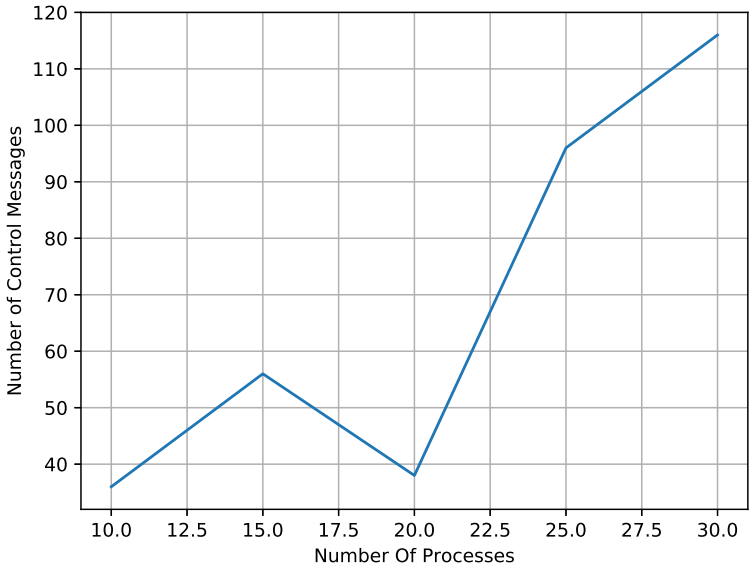
Spanning Tree Algorithm

Message Optimal Algorithm

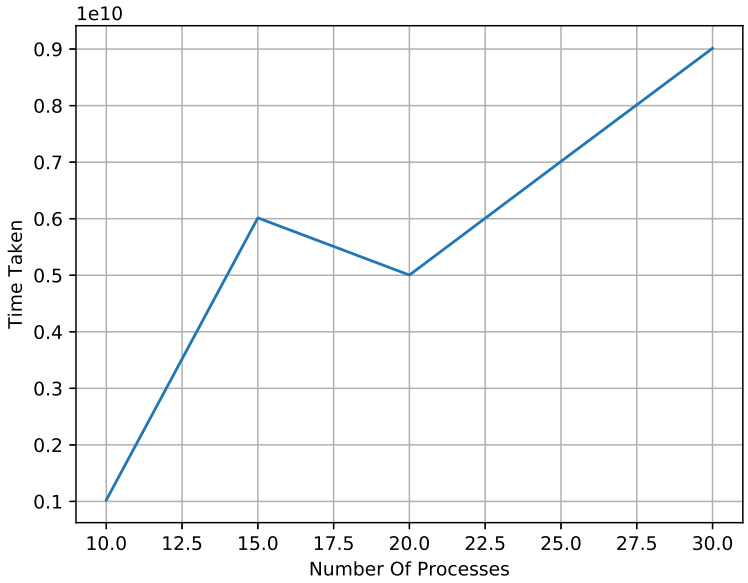
Results & Graphs

Spanning Tree Algorithm

Number of Processes Vs Number of Control Message



Number of Processes Vs Time Taken



Explaination of Results

The following are a few points to note / observations about the above graphs:

1.

Message Optimal Algorithm

G1

G2

Explaination of Results

The following are a few points to note / observations about the above graphs:

1.