University of Glasgow | School of Computing Science

Assessed Coursework

Course Name	ANC (H)			
Coursework Number	1			
Deadline	Time:	4:30pm	Date:	8th March 2019
% Contribution to final course mark	20%			
Solo or Group ✓	Solo	✓	Group	
Anticipated Hours				
Submission Instructions	See “Deliverables”			
Please Note: This Coursework cannot be Re-Assessed				

You must complete an “Own Work” form via https://studentltc.dcs.gla.ac.uk/ for all coursework

ANC (H): Assessed Exercise

Description

Consider a simplified point-point network description consisting of a set of nodes {N1, N2, ….} and a set of links with single (bidirectional) costs, of the form (N1, N2, C). In addition, assume every node has a routing table with 3 columns: destination node, distance and outgoing link.

*With a programming language of your choice** implement an application which will read a file containing a simple human-readable (and writable) description of such a network and then simulate distance-vector routing with *synchronised periodic exchanges* and updating of node routing tables, over any selected number of iterations. With synchronised periodic exchanges, all nodes exchange their vectors simultaneously and then update tables, repeating this process at regular intervals.

A user should be able to:

- compute routing tables for any preset number of exchanges or until stability is achieved;
- preset any link to change cost or fail after any chosen exchange (you may assume for simplicity that neighbours notice cost changes or unreachable neighbours immediately);
- view the best route between any source and destination after any chosen iteration
- trace the routing tables of any set of nodes for any specified number of iterations in a way that can be easily viewed;
- engage, on request, a *split-horizon* capability to help combat slow convergence
- run the application under Windows 10 x64.

Once your simulator is working, document a walkthrough of **two** simple examples with appropriate input networks to illustrate how some exemplar routes and tables evolve under:

- normal convergence of the distance-vector algorithm;
- slow convergence both **with and without** the split horizon facility switched on.

***Describe clearly but briefly, with the help of output from your application, what is going on as the routes evolve.* Use diagrams where useful to aid your descriptions.**

Deliverables.

Submit the following in a zip file attachment via email to Lewis.Mackenzie@glasgow.ac.uk by the deadline above. The subject heading in your email must be: "ANC4 assessed exercise".

1. A short (< 1 page) informal overview of your high-level application design and a short user manual (< 1 page) including instructions to build, install and run the program*. These instructions should be adequate to allow a user to prepare and test network description files of his/her own. (~20%)
2. A well-commented source listing (informative commenting is what counts). Instructions on how to build the executable from this source must be included in 1 above. (~10%)
3. A functioning executable file which MUST be derivable from 2. Marks are gained for implementing the required functionalities and for an application which is easy to use. A graphical interface is not necessary (although you may provide one if you wish) but ***usability*** is important. (~*30%*)
4. For your two example networks, give input description files and, using words, diagrams and output from your application, a documented "walkthrough" of how some exemplar tables and routes evolve as each iteration completes. Marks here will be awarded for ***clarity and brevity***. (~*30%*)
5. A (0.5-page max) status report discussing any limitations of your program and how you would improve it given the time. (~10%)

All documents must be in ***Acrobat PDF format***. Marks will be awarded for ***clarity and brevity*** of presentation. Inadequate instructions for building, running or using your application will lose marks.

Items 1, 4 and 5 should also be submitted in hard copy in the box provided in Lilybank Gardens.

*** NOTE: Although you may use any programming language you like, you must ensure your executable will *run* without requiring the user to install any additional commercial software. If anything else (e.g. a runtime environment) is needed, which would not be present on a freshly installed copy of Windows 10 x64, you MUST include instructions about how to get it and install it!**