UNIVERSITY OF MORATUWA

Faculty of Engineering

Department of Computer Science & Engineering

MSc in Computer Science
2016 Intake Semester 6 / 2017 Intake Semester 3 Examination

CS5429 DISTRIBUTED COMPUTING

Time allowed: 2 Hours December 2017

ADDITIONAL MATERIAL: *None*

INSTRUCTIONS TO CANDIDATES:

1. This paper contains **5** questions on **5** pages.
2. This examination accounts for 50% of the module assessment.
3. This is an open book examination.
4. Answer any **4** questions.
5. Start answering each of the main questions on a new page.
6. The maximum attainable mark for each question is given in brackets.
7. Only calculators approved by the Faculty of Engineering are permitted.
8. Assume reasonable values for any data not given in or with the examination paper. Clearly state such assumptions made on the script.
9. If you have any doubt as to the interpretation of the wording of a question, make your own decision, but clearly state it on the script.
10. Electronic/Communication devices are not permitted. Only equipment allowed is a calculator approved and labelled by the Faculty of Engineering.
11. This paper should be answered only in English.

Question 1 (25 marks) **[25 marks]**

(i) Using an application of your choice explain how replication transparency provides reliability and performance, while potentially loosing consistency. [5]

(ii) "*Publisher/Subscriber systems decouple publishers and subscribers in terms of time, space, and synchronization*"

Briefly explain how Message Queues can support the decoupling of publishers and subscribers. [4]

(iii) A distributed system has 2 nodes each with an availability of 99%. What is the overall availability if 2 nodes are placed as in

a) 2-tier architecture (i.e., in series)? [3]

b) singe-tier architecture with high availability (i.e., parallel)? [3]

(iv) "*Not every node in a peer-to-peer network should become a superpeer*"

Propose a superpeer election mechanism for an unstructured P2P solution. Your mechanism should propose a suitable metric to elect a leader based on characteristics of peers in a public network, as well as order and scope of messages. Justify why your solution would work. [10]

Question 2 (25 marks)

Figure 1 illustrates an IoT solution deployed to detect movements of elephants in a jungle. A ZigBee-based (i.e., low power and low bandwidth wireless technology with a range of ~100m) collar is attached to an elephant. When an elephant comes near a ZigBee receiver, a message is triggered and send to the Cloud-based backend using a 3G connection. Typical message related to an elephant detection event includes (*receiver ID*, *elephant ID*) pair. Location of receivers are known and configured on the backend.

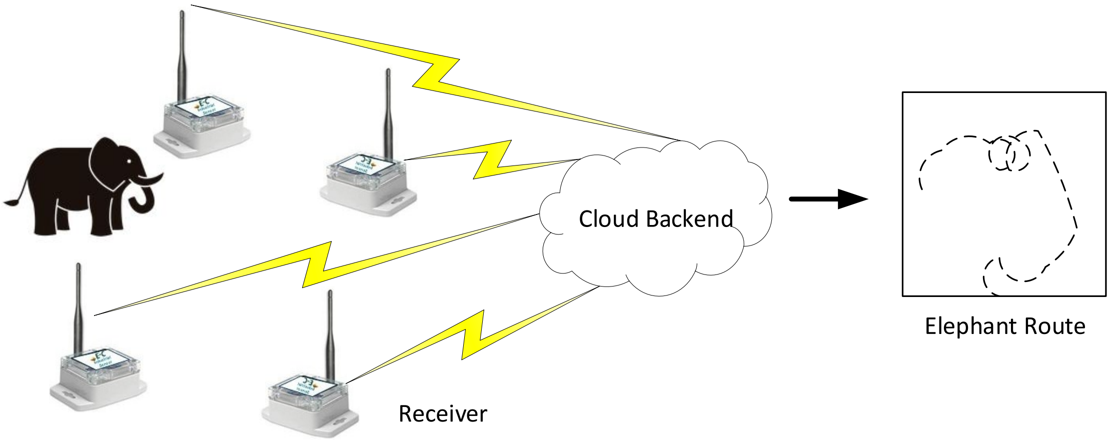


Figure 1 – Overview of elephant movement tracking solution.

(i) Would you recommend persistent-asynchronous or transient-asynchronous communication for the data transfer between receivers and the Cloud-backend? Briefly Discuss. [4]

(ii) Cloud-based server maintains an in-memory table with the list of received events. Is this server *stateful* or *stateless*? Briefly discuss. [3]

(iii) Following 3 options are suggested to order the elephant detection events such that the route taken by an elephant can be estimated:

(a) Receiver's clock is set to physical time at the time of deployment, and is used to timestamp each event.
(b) Events are to be timestamped only at the server based on arrival time.
(c) Each event is to be tagged with an event ID.

Recommend the most suitable solution while considering pros and cons of each of the solution. [7]

(iv) Can Totally Ordered Multicast among receivers be used to order the elephant detection events? Discuss. [3]

(v) Propose a protocol for the messages exchanged between a receiver and Cloud-based server.

Your proposal should contain message formats, order of messages, data types, and lengths. [8]

Question 3 (25 marks)

Spotify is a music, podcast, and video streaming service. Spotify is a freemium service; basic features are free with advertisements or limitations, while additional features such as improved quality and music downloads are offered via paid subscriptions. Figure 2 illustrates a high-level architecture of Spotify services. Services are partitioned based on features, e.g., music, podcast, and video streaming. If one feature fails, other features are independent and will continue to work. When there is a weak dependency between features, failure of one feature may sometimes lead to degradation of service of another feature. Feature partitioning gives scalability, reliability, and an efficient way of focusing on feature improvements. Spotify runs several server instances across the world to achieve geographic scalability.

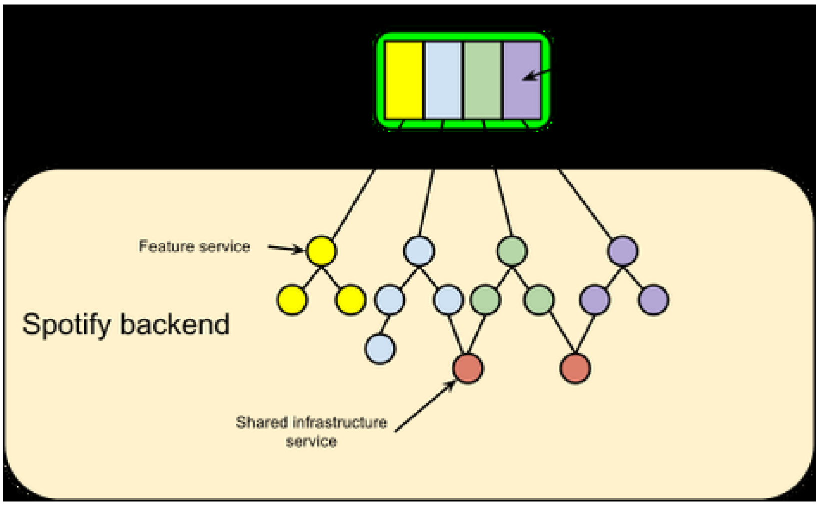


Figure 2 – High-level architecture of Spotify. Source: https://labs.spotify.com

(i) Briefly explain how feature partitioning could enhance the scalability of the system. [3]

(ii) "*When bandwidth between Spotify client and backend drops, one solution is to degrade quality of the stream, e.g., downgrade from MP3 to MP4.*"

Do you agree or disagree with this statement? Discuss. [4]

(iii) What load balancing technique would you recommend for backend feature services in Spotify? Briefly discuss. [4]

(iv) What do you recommend, RPC, RMI, REST API, or Web services to communicate between Spotify clients and backend? Justify. [4]

(v) What active CDN technology would you recommend to scale Spotify over having their own servers? Discuss. [4]

(vi) Given that Spotify uses geographically distributed set of servers (with or without CDN), how can Spotify ensure transactions related to paid subscribers will satisfy ACID properties? Discuss. [6]

Question 4 (25 marks)

SETI is one of the largest volunteer computing platforms that remotely executes jobs using idle computing resources. These jobs include analyzing images/data from optical and radio telescopes for the presence of extra-terrestrial life. Over 200,000 SETI volunteer nodes are active at any given time. Each node contacts the SETI server and asks for a new job based on its computing capabilities. The same job is submitted only to two nodes to increase the reliability while maintaining better resource utilization (volunteer nodes may fail at any time). Once the job is completed, node submits the answer and asks for another job. You are required to design a job dispatching solution that allocates only two copies of the same job to volunteer nodes.

(i) To enhance the scalability and availability of the system, it is proposed to run 2 SETI servers within the SETI server room. How would spit-brain syndrome apply in the context of this proposal and how do you plan to overcome it? [6]

(ii) Another proposal is to have 2 geographically distributed SETI servers. How would CAP theorem apply in the context of this proposal and how do you plan to overcome it? [6]

(iii) SETI gives points to volunteers once they submit an answer for a job. However, to encourage faster job completion, SETI proposed to give full points to the first volunteer that submits the answer, while second volunteer that submit the same answer will be given only 60% of the points.

How do you propose to differentiate the first volunteer from the second? [4]

(iv) SETI servers are bandwidth heavy as each job file is 2-4 MB. Size of a typically result file is 10% of the job file. Therefore, it is proposed to build a P2P solution to distribute the SETI jobs among volunteers.

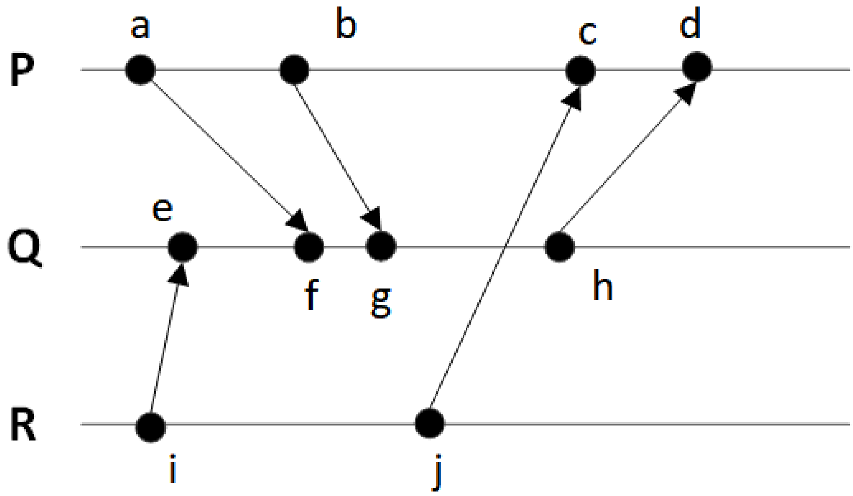
a) How much server bandwidth can be saved using a P2P solution?

Hint: *Give a rough calculation based on how many files the server need to push in new design compared to old design.* [3]

b) Describe how you would map an already developed P2P file sharing solution to distribute SETI jobs.
Clearly state how a job file to be named, locate/search the file, and deliver. [6]

Question 5 (25 marks)

(i) a) Label the following diagram with Lamport's Timestamps.



[4]

b) Identify 4 event pairs with the same Lamport time stamp. [2]

c) Discuss how one of the event pairs you identified in i(b) above can (or cannot) be ordered using Vector Clocks. [4]

(ii) 'Next word suggester' tool suggests the highly likely next word to be typed given the previous word. Next word suggester is a useful tool for many applications including information retrieval, grammar checkers, and style checkers. These applications benefit by estimating the likelihood of a word following another word. Frequency of a sequence 'A B' is seen in a large corpus is directly correlated to the likelihood of word 'B' occurring after word 'A'.

Given a huge text collection composed of a large set of files, where each file consists of one paragraph. Design a solution using MapReduce to estimate the sequence counts of various word pairs in order.

Clearly state all assumptions and give pseudo code of mappers and reducers. [15]

------------------------- END OF THE PAPER -------------------------