

Birkbeck
(University of London)

BSc/FD EXAMINATION

Department of Computer Science and Information Systems

Cloud Computing Concepts (BUCI028H6)

CREDIT VALUE: 15 credits

Date of examination: Tuesday, 27th May 2014
Duration of paper: 2:30pm – 4:30pm (2 hours)

RUBRIC

- 1. This paper contains 5 questions for a total of 100 marks.*
- 2. Students should attempt to answer **all** of them.*
- 3. This paper is not prior-disclosed.*
- 4. The use of non-programmable electronic calculators is permitted.*

1. **(20 marks)**

Please give brief answers to the following questions.

- (a) What is utility computing? Which are its main benefits? (5 marks)
- (b) What are the major service models of cloud computing? Which one is the most basic (i.e., closest to the hardware level)? Give an example of that service model. (5 marks)
- (c) What kinds of applications are most suitable to be put in the cloud? (5 marks)
- (d) What kinds of applications are not very suitable to be put in the cloud? (5 marks)

2. **(20 marks)**

Please give brief answers to the following questions.

- (a) In Amazon Web Services (AWS), what do AMI, EC2, S3, EBS, EMR stand for respectively? (5 marks)
- (b) In RESTful APIs, what HTTP methods should be nullipotent and what HTTP methods should be idempotent? (5 marks)
- (c) In RESTful APIs, what do the constraints “stateless” and “cacheable” mean respectively? (5 marks)
- (d) What are the possible disadvantages of using a private cloud instead of a public cloud? (5 marks)

3. **(20 marks)**

Please give brief answers to the following questions.

- (a) What are the 5Vs of Big Data? (5 marks)
- (b) Which three properties cannot be provided by a distributed system simultaneously, according to the CAP theorem? Which one of them do NoSQL databases usually compromise in favour of the other two? (5 marks)
- (c) What are the underlying assumptions of the Google File System (GFS)? (5 marks)
- (d) Why is the standard hashing technique unsuitable for distributed indexing? How does consistent hashing solve this problem? (5 marks)

4.

(20 marks)

Consider the following MapReduce program `word-count` which counts the total number of occurrences for each distinct word in a very large document collection.

```
class MAPPER
  method MAP(docid i, doc d)
    for all term t ∈ doc d do
      EMIT(term t, count 1)

class REDUCER
  method REDUCE(term t, counts [c1, c2, . . .])
    sum ← 0
    for all count c ∈ counts [c1, c2, . . .] do
      sum ← sum + c
    EMIT(term t, count sum)
```

- (a) How can we modify the mapper function so that the program computes the document frequency for each distinct word (i.e., the number of documents containing that word)?
Please try to make minimal changes to the existing code. (5 marks)
- (b) How can we modify the mapper function and also the reducer function so that the program computes the average document length in this collection?
Please try to make minimal changes to the existing code. (5 marks)
- (c) How can we use combiners to accelerate the above program for average document length? (5 marks)
- (d) How can we use the “in-mapper combining” design pattern instead of combiners to accelerate the above program for average document length? (5 marks)

5.

(20 marks)

Consider the following MapReduce program that counts the number of co-occurrences for each word-pair (i.e., the number of times those two words both occur within a certain neighbourhood such as a sentence) in a very large document collection. The “pairs” design pattern has been used.

```

class MAPPER
  method MAP(docid  $i$ , doc  $d$ )
    for all term  $w \in \text{doc } d$  do
      for all term  $u \in \text{NEIGHBOURS}(w)$  do
        EMIT(pair ( $w, u$ ), count 1)      ▷ Emit count for each co-occurrence

class REDUCER
  method REDUCE(pair  $p$ , counts [ $c_1, c_2, \dots$ ])
     $s \leftarrow 0$ 
    for all count  $c \in \text{counts } [c_1, c_2, \dots]$  do
       $s \leftarrow s + c$                     ▷ Sum co-occurrence counts
    EMIT(pair  $p$ , count  $s$ )

```

- Are we able to use the reducer function directly as the combiner function? Why? Is it usually better to set the number of map tasks larger than the number of computer nodes in the cluster? Why? (5 marks)
- How can we use the “stripes” design pattern to re-write the above program? Please try to make minimal changes to the existing code. (5 marks)
- What are the pros and cons of the “stripes” design pattern in comparison with the “pairs” design pattern? (5 marks)
- Explain in words how the given program with the “pairs” design pattern can be extended to compute the relative frequencies, i.e., conditional probabilities

$$P(b|a) = \frac{\text{count}(a, b)}{\text{count}(a)} = \frac{\text{count}(a, b)}{\sum_{b'} \text{count}(a, b')}.$$

Hint: Make use of the “order inversion” design pattern. (5 marks)