Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

FS 2013

Dr. T. Locher, Dr. Y.A. Pignolet
Prof. Dr. Roger Wattenhofer
P. Bissig, P. Brandes, S. Frischknecht,
S. Holzer, M. König, J. Uitto,

Exam
Principles of Distributed Computing

Wednesday, August 7, 2013
9:00 – 11:00

Do not open or turn until told to by the supervisor!

The exam lasts 120 minutes, and there is a total of 120 points. The maximal number of points for each question is indicated in parentheses. Your answers may be in English or in German. Be sure to always justify your answers. Algorithms can be specified in high-level pseudocode or as a verbal description. You do not need to give every last detail, but the main aspects need to be there. Big-O notation is acceptable when giving algorithmic complexities.

Please write down your name and Legi number (your student ID) in the following fields.

Name	Legi-Nr.

Points

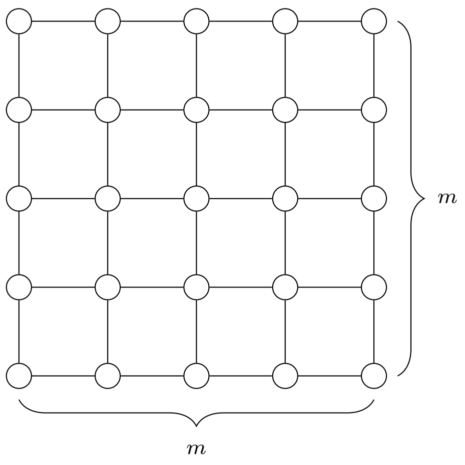
Question Nr.	Achieved Points	Maximal Points
1		11
2		30
3		27
4		27
5		25
Total		120

1 Multiple Choice (11 Points)

Evaluate each of the following statements in terms of correctness. Indicate whether a statement is true or not by ticking the corresponding box. Each correct answer gets 1 point, **each wrong answer gets -1 point**. An unanswered statement gets 0 points. If the sum of collected points is negative, then you get 0 points for this question set.

Statement	**true**	**false**
Having a leader in a wireless network can add collision detection with an overhead of $O(\log n)$.	☐	☐
An event happens with high probability if it happens at least 99% of the time.	☐	☐
Synchronizer α does not add an asymptotic overhead to the message complexity.	☐	☐
Coloring an arbitrary tree with 2 colors takes time $\Omega(n)$.	☐	☐
In a synchronous system, the flooding algorithm can replace the Dijkstra algorithm to construct a BFS tree.	☐	☐
An augmented grid with parameter $\alpha = 0$ has a larger diameter than an augmented grid with parameter $\alpha = 2$, w.h.p..	☐	☐
One can deconstruct a d-dimensional hypercube into two $(d-1)$-dimensional hypercubes by removing 2^d edges.	☐	☐
Test-and-set has a larger consensus number than compare-and-swap.	☐	☐
Orienting an unoriented binary tree in all-to-all communication takes time $O(1)$, when node IDs are unique.	☐	☐
Computing a spanning tree in all-to-all communication takes time $\Omega(\log n)$ w.h.p..	☐	☐
Given any correct sorting network, adding an incorrect sorting network at the end destroys the sorting property.	☐	☐

2 Maximal Independent Set and Coloring (30 Points)



An independent set is a subset W of all nodes V such that no two nodes in W are connected, i.e., $(w_1, w_2) \notin E$ for all $w_1, w_2 \in W$.
A **maximum** independent set is a largest possible independent set.
A **maximal** independent set is an independent set that cannot be extended by adding any node $v \in V \setminus W$.

A) [3] Mark the nodes of a **maximum** independent set in the grid shown above.

B) [5] Assume that the graph is *anonymous*, i.e., the nodes do not have unique IDs. Assume further that m is odd and known. Give a deterministic algorithm that computes a **maximum** independent set. More points are awarded for faster algorithms.

C) [2] What are the asymptotic time and message complexities of your algorithm?

D) [8] Can you extend your algorithm for even values of m? Justify your answer.

E) [12] Assume that the nodes have unique IDs. Give a deterministic algorithm with a time complexity $o(\log n)$ that computes a **maximal** independent set on a grid for arbitrary m.

3 Leader Election (27 Points)

In the lecture we learnt that leader election can be a rather difficult problem to solve. The goal of this task is to gain some more insights into the feasibility of solving leader election.

We assume nodes communicate in synchronous rounds. The graph is again *anonymous*, i.e., the nodes do not have unique IDs.

A node that terminates may no longer send any messages or change its internal state. The whole algorithm terminates when all nodes have terminated. Once the algorithm terminates, there must be exactly one node that is in the "leader" state. All other nodes must know that there is a unique leader.

A) Let us first consider an anonymous tree with n nodes. All nodes are initially unaware of their function (root, leave, inner node) in the tree and are in the same state (undecided). Your task is to come up with an algorithm such that:

- exactly one node becomes the root
- all nodes with only one neighbor become leaves
- all the remaining nodes become inner nodes

1) [4] Prove that there is no deterministic algorithm that can solve the given task.

2) [7] Can you find such an algorithm if we require the underlying graph to have an odd number of nodes? Either prove that no such algorithm exists, or state an algorithm that solves the problem. In the latter case, also provide the time complexity and prove the correctness of the algorithm.

B) Now let us consider the torus graph. The torus graph consists of a grid of m by n nodes. Each node has four neighbors (north, east, south, west). The nodes on the border of the grid are connected to their counterpart of the opposing border (see Figure 1).

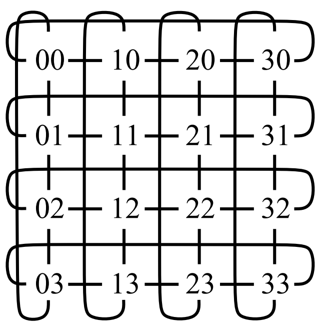


Figure 1: Sample of a torus graph with $m = n = 4$ nodes

1) [3] Is there a deterministic, non-uniform[1], and anonymous algorithm to elect a leader in such a graph? Prove impossibility or give an algorithm.

2) [3] If the size of the grid (m, n) is unknown, is it possible to compute the size of the grid? Prove impossibility or give an (uniform) algorithm.

3) [10] Now assume that an arbitrary node is removed from the graph and its neighbors are interconnected (north-south and east-west). Is anonymous leader election possible in this graph? Prove impossibility or give an algorithm.
Hint: Can a node determine if it is lying in the same row or column as the node that has been removed?

[1]An algorithm is called uniform if the number of nodes is **not** known to the algorithm.

4 Colored Hypercube (27 Points)

Let $G = (V, E)$ be a hypercube with $|V| = n$ with $n = 2^d$ with d being the dimension of the hypercube. Let the vertices have unique IDs from $0, \ldots, n-1$. An edge between two vertices exists if and only if the binary representation or their ID differs by exactly one bit. Note that the nodes do *not* know in which bit the ID of a specific neighbor differs from their own ID, i.e., they do not which dimension an edge crosses. Examples are shown in Figures 2 and 3.

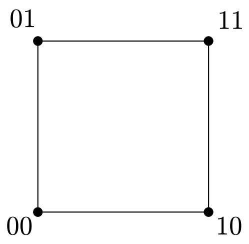


Figure 2: Hypercube H_2 of dimension 2

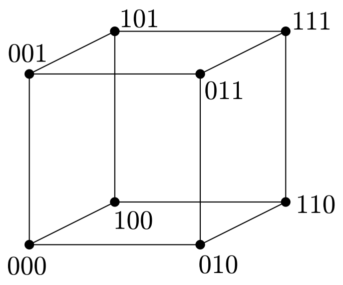


Figure 3: Hypercube H_3 of dimension 3

A) [6] First assume that the nodes do *not* know their IDs but there is a unique leader u. Propose an algorithm to color the nodes correctly. Analyze its time and message complexity and prove correctness. Use as few colors as possible. More points are awarded to faster solutions.

B) [4] Now assume that the nodes are aware of their IDs. Can you do better? Propose an algorithm to color the nodes correctly. Again analyze its time and message complexity, and prove correctness. Use as few colors as possible.

An edge coloring is valid if no two adjacent edges share the same color. Formally, for all edges $e_1 = (u, v) \in E, e_2 = (v, w) \in E$, we have that $c_{e_1} \neq c_{e_2}$.

C) [4] Assume that the nodes know their IDs. Propose an algorithm to color the edges correctly. Analyze its time and message complexity and prove correctness. Use as few colors as possible.

D) [8] Assume that the nodes do *not* know their IDs, but there is a unique leader u. Propose an algorithm to color the edges correctly. Analyze its correctness, running time, and message complexity. Use as few colors as possible.

E) [5] Does an optimal edge coloring always use at least as many colors as an optimal vertex coloring on hypercubes with $d \geq 2$? Argue why this is true or provide a counterexample. What about general graphs? Prove this statement or provide a graph class that violates this property and argue why it violates this property.

5 Peer-to-Peer Computing (25 Points)

Recall the butterfly graph seen in the lecture:

Definition (Butterfly)**.** *Let $d \in \mathbb{N}$. The d-*dimensional butterfly *$BF(d)$ is a graph with node set $V = [d+1] \times [2]^d$ and edge set $E = E_1 \cup E_2$ with*

$$\begin{aligned} E_1 &= \{\{(i,\alpha),(i+1,\alpha)\} \mid i \in [d],\ \alpha \in [2]^d\} \text{ and} \\ E_2 &= \{\{(i,\alpha),(i+1,\beta)\} \mid i \in [d],\ \alpha,\beta \in [2]^d,\ \alpha \text{ and } \beta \text{ differ only at the } i^{th} \text{ position}\}\ . \end{aligned}$$

You may ignore churn for this question.

A) [5] Draw $BF(2)$.

B) [3] Which vertices have which degrees in $BF(d)$? What is the diameter of $BF(d)$? How many vertices does $BF(d)$ have?

C) [4] How many edges need to "fail" (i.e., be removed) for $BF(d)$ to become disconnected? How many need to fail to isolate all (i,α) for a given α from the rest of the graph?

D) [3] What is an advantage and a disadvantage of using a butterfly graph for a peer-to-peer network compared to using a hypercube?

E) [10] How can the butterfly graph be used to implement a *distributed hash table* (DHT)? In other words, where could files, indexed by bit strings of a certain length b, be stored in the butterfly graph, and how would these files be looked up (given the corresponding bit string)? Propose a solution where a file is stored at one node and another solution where a file is stored redundantly at multiple nodes. Briefly describe their pros and cons. You may assume $2^b \geq |V|$ and $d = 2^k - 1$ for a $k \in \mathbb{N}$.