## 1. Leader election problem

### Definition

Leader election is the simple idea of giving one thing (a process, host, thread, object, or human) in a distributed system some special powers. Those special powers could include the ability to assign work, the ability to modify a piece of data, or even the responsibility of handling all requests in the system [1]. Leader election is a powerful tool for improving efficiency, reducing coordination, simplifying architectures, and reducing operations. On the other hand, leader election can introduce new failure modes and scaling bottlenecks. In addition, leader election may make it more difficult for you to evaluate the correctness of a system [2].

### Possible Algorithm used to solve the problem

One possible algorithm used to solve a problem is an anonymous ring, a ring is said to be anonymous if every processor is identical. More formally, the system has the same state machine for every processor. There is no deterministic algorithm to elect a leader in anonymous rings, even when the size of the network is known to the processes [2]. This is due to the fact that there is no possibility of breaking symmetry in an anonymous ring if all processes run at the same speed. The state of processors after some steps only depends on the initial state of neighboring nodes. So, because their states are identical and execute the same procedures, in every round the same messages are sent by each processor. Therefore, each processor state also changes identically and as a result if one processor is elected as a leader, so are all the others [1].

### Specific field of application and example

In radio networks, the n nodes may in every round choose to either transmit or receive a message. If no collision detection is available, then a node cannot distinguish between silence or receiving more than one message at a time [2]. Should collision detection be available, then a node may detect more than one incoming message at the same time, even though the messages itself cannot be decoded in that case. In the beeping model, nodes can only distinguish between silence or at least one message via carrier sensing [3].

## 2. Consensus problem

### Definition

In the context of distributed systems design, a consensus is often loosely used to mean some form of agreement. Consensus involves multiple servers agreeing on values. Once they reach a decision on a value, that decision is final [4]. Typical consensus algorithms make progress when any majority of their servers is available; for example, a cluster of 5 servers can continue to operate even if 2 servers fail. If more servers fail, they stop making progress but will never return an incorrect result [5].

**Possible Algorithm used to solve the problem**

A possible algorithm to solve the problem is the Byzantine fault, it is a condition of a computer system, particularly distributed computing systems, where components may fail and there is imperfect information on whether a component has failed [6]. The term takes its name from an allegory, the "Byzantine Generals Problem", developed to describe a situation in which, in order to avoid catastrophic failure of the system, the system's actors must agree on a concerted strategy, but some of these actors are unreliable [7].

**Specific field of application and example**

The Proof of Work algorithm, this consensus algorithm is used to select a miner for the next block generation. Bitcoin uses this PoW consensus algorithm. The central idea behind this algorithm is to solve a complex mathematical puzzle and easily give out a solution. This mathematical puzzle requires a lot of computational power and thus, the node who solves the puzzle as soon as possible gets to mine the next block [7].

## 3. Distributed search

**Definition**

Distributed searching is the capability to search across multiple computers. A typical searching system indexes the data on a single computer. Users will search on the computer and the search system will respond with results from a single computer. In distributed search, users still search the same way [8].

**Possible Algorithm used to solve the problem**

Tree searching or Distributed Tree Searching is of the algorithm that consists of using multiple processes, each with a node and a set of processors attached, with the goal of searching the sub-tree below the said node [9]. Each process then divides itself into multiple coordinated sub-processes which recursively divide themselves again until an optimal way to search the tree has been found based on the number of processors available to each process. Once a process finishes, DTS dynamically reassigns the processors to other processes as to keep the efficiency to a maximum through good load-balancing, especially in irregular trees. Once a process finishes searching, it recursively sends and merges a resulting signal to its parent-process, until all the different sub-answers have been merged and the entire problem has been solved [10].

**Specific field of application and example**

One major example of the everyday use of DTS is network routing. The Internet can be seen as a tree of IP addresses, and an analogy to a routing protocol could be how post offices work in the real world [11]. Society heavily depends on the time the data takes to find its way to its destination. As such, IP-routing divides the work into multiple sub-units which each have different scales of calculation capabilities and use each other's result to find the route in a very efficient manner [8].

## 4. Spanning tree generation

**Definition**

A spanning tree of that graph is a subgraph that is a tree and connects all the vertices together. A single graph can have many different spanning trees [11]. A tree is a connected undirected graph with no cycles. It is a spanning tree of a graph G if it spans G (that is, it includes every vertex of G) and is a subgraph of G (every edge in the tree belongs to G). A spanning tree of a connected graph G can also be defined as a maximal set of edges of G that contains no cycle, or as a minimal set of edges that connect all vertices [12].

**Possible Algorithm used to solve the problem**

This is one of the most important algorithms of spanning tree, Kruskal's algorithm which find the minimum cost spanning tree uses the greedy approach. This algorithm treats the graph as a forest and every node it has as an individual tree. A tree connects to another only and only if, it has the least cost among all available options and does not violate MST properties [13].

**Specific field of application and example**

Spanning tree is basically used to find a minimum path to connect all nodes in a graph. Common application of spanning trees is Civil Network Planning, Computer Network Routing, Protocol Cluster Analysis. Basically, we can consider a city network as a huge graph and now plans to deploy telephone lines in such a way that in minimum lines we can connect to all city nodes. This is where the spanning tree comes into picture [14].

## 5. Mutual exclusion problem

**Definition**

A mutual exclusion is a property of concurrency control, which is instituted for the purpose of preventing race conditions. It is the requirement that one thread of execution never enters a critical section while a concurrent thread of execution is already accessing critical section, which refers to an interval of time during which a thread of execution accesses a shared resource, such as Shared data objects, shared resources, shared memory [15].

**Possible Algorithm used to solve the problem**

Dijkstra is by far the most common algorithm that used to solve mutual exclusion problem. The Dijkstra's algorithm finds the shortest path from a particular node, called the source node to every other node in a connected graph [16]. It produces a shortest path tree with the source node as the root. It is profoundly used in computer networks to generate optimal routes with the aim of minimizing routing costs [17].

**Specific field of application and example**

Mutual exclusion locks are a commonly used mechanism for synchronizing processes or threads that need access to some shared resource in parallel programs. They work as their name suggests. If a thread "locks" a resource, another thread that wishes to access it will need to wait till the first thread unlocks it [18].

## 6. Resource allocation problem

**Definition**

Allocation problems involve the distribution of resources among competing alternatives in order to minimize total costs or maximize total return. Such problems have the following components; a set of resources available in given amounts, a set of jobs to be done, each consuming a specified number of resources, and a set of costs or returns for each job and resource. The problem is to determine how much of each resource to allocate to each job [19].

**Possible Algorithm used to solve the problem**

The banker's algorithm can solve this resource allocation problem, because a banker's algorithm is a resource allocation and deadlock avoidance algorithm that tests for safety by simulating the allocation for predetermined maximum possible amounts of all resources, then makes an "s-state" check to test for possible activities, before deciding whether allocation should be allowed to continue [20].

**Specific field of application and example**

In algorithmic approach resource is allocated by using some computer program which is defined for a specific domain, this will automatically and dynamically distribute resources to the user [21]. Electronic devices dedicated to routing and communication is commonly used this method. For example: channel allocation in wireless communication may be decided by base transceiver using an appropriate algorithm [22].

# Resources

[1] Wikipedia, *"Leader Election"* - https://en.wikipedia.org/wiki/Leader_election

[2] Marc Brooker, *"Leader Election in Distributed Systems "* - https://aws.amazon.com/builders-library/leader-election-in-distributed-systems/
[3] Uregina, *"Distributed Algorithms"* - http://www2.cs.uregina.ca/~hamilton/courses/330/notes/distributed/distributed.html
[4] Wikipedia, *"Consensus "* - https://en.wikipedia.org/wiki/Consensus_(computer_science)
[5] Isuru Boyagane, *"What is Consensus in Distributed Systems? "*- https://medium.com/@isuruboyagane.16/what-is-consensus-in-distributed-system-6d51d0802b8c
[6] Wikipedia, *"Byzantine fault "*- https://en.wikipedia.org/wiki/Byzantine_fault
[7] TechTarget Contributor, *"Consensus algorithm"* - https://whatis.techtarget.com/definition/consensus-algorithm
[8] Wikipedia, *"Distributed Tree Search"* - https://en.wikipedia.or/wiki/Distributed_tree_search
[9] Azam Qureshi, *"What Is Distributed Searching?"* - https://www.intradyn.com/what-is-distributed-searching/
[10] IBM Docs, *"What is distributed computing"* - https://www.ibm.com/docs/en/txseries/8.1.0?topic=overview-what-is-distributed-computing
[11] TechTarget Contributor, *"Distributed Search"* - https://whatis.techtarget.com/definition/distributed-search
[12] SpringerLink, *"Algorithms for generating all possible spanning trees"* - https://link.springer.com/article/10.1007/s40747-018-0079-7
[13] TutorialsPoint, *"Data Structure & Algorithms - Spanning Tree"* - https://www.tutorialspoint.com/data_structures_algorithms/spanning_tree.htm
[14] Wikipedia, *"Kruskal Algorithm"* - https://en.wikipedia.org/wiki/Kruskal%27s_algorithm
[15] GeeksforGeeks, *"Kruskal's Minimum Spanning Tree Algorithm"* - https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/
[16] Leslie Lamport, *"The mutual exclusion problem"* - https://dl.acm.org/doi/abs/10.1145/5383.5385
[17] Wikipedia, *"Dijkstra Algorithm"* - https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm
[18] TutorialsPoint, *"Dijkstra's algorithm to compute the shortest path through a graph"* - https://www.tutorialspoint.com/dijkstra-s-algorithm-to-compute-the-shortest-path-through-a-graph
[19] Wikipedia, *"Mutual Exclusion"* - https://en.wikipedia.org/wiki/Mutual_exclusion
[20] TutorialsPoint, *"Operating System Resource Allocation"* - https://tutorialspoint.dev/computer-science/operating-systems/operating-system-resource-allocation-graph-rag
[21] Wikipedia, *"Banker's Algorithm"* - https://en.wikipedia.org/wiki/Banker%27s_algorithm
[22] Sudeshna Negi & Punya Palit, *"Resource Allocation: A Guide on How to Apply it on Project Management"* - https://www.saviom.com/blog/resource-allocation-a-guide-on-how-to-apply-it-on-project-management/