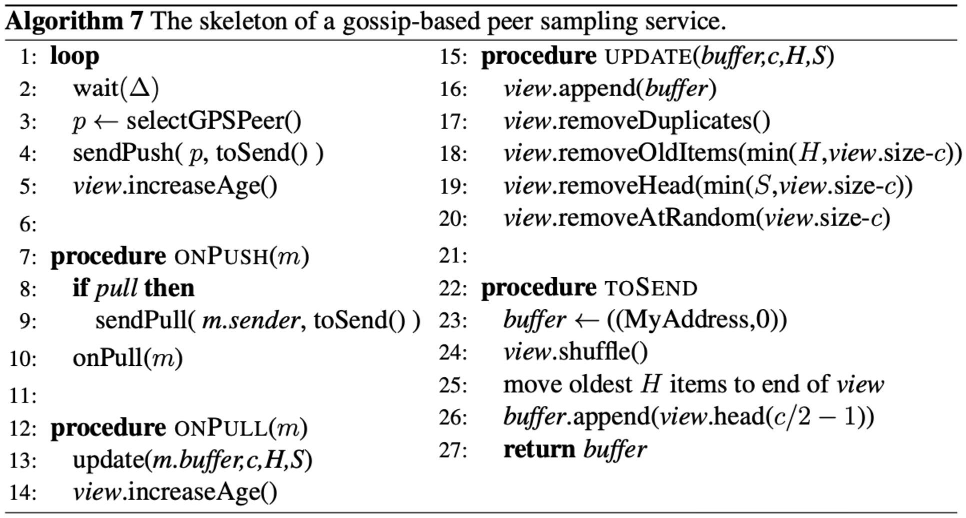
20.02.2023 Week 1 exercises: Gossip

Exercise 1:

What are the 3 ways of propagating information in a graph? What are the advantages and disadvantages of each one of them?

Exercise 2:

Recall from the lecture the skeleton for gossip-based peer sampling service. Three system parameters: c, H and S play an important role in the dynamics of the algorithm. c controls the view size while H and S represent the *healing* and *swapping* parameters.



Algorithm 7 The skeleton of a gossip-based peer sampling service.

```
loop
    wait(Δ)
    p ← selectGPSPeer()
    sendPush( p, toSend() )
    view.increaseAge()

procedure onPush(m)
    if pull then
        sendPull( m.sender, toSend() )
    onPull(m)

procedure onPull(m)
    update(m.buffer,c,H,S)
    view.increaseAge()

procedure update(buffer,c,H,S)
    view.append(buffer)
    view.removeDuplicates()
    view.removeOldItems(min(H,view.size-c))
    view.removeHead(min(S,view.size-c))
    view.removeAtRandom(view.size-c)

procedure toSend
    buffer ← ((MyAddress,0))
    view.shuffle()
    move oldest H items to end of view
    buffer.append(view.head($c/2 - 1$))
    return buffer
```

Figure 1: Gossip based peer sampling [1].

1. What is the effect of increasing or decreasing H ?
2. Can you explain similarly the effect of increasing or decreasing S ?

Exercise 3:

Gossip can be used not only for information dissemination but also for information processing. In the lecture we saw how to compute mean of values in a network in a Gossip-style protocol. Let us recall the algorithm below. For an in-depth discussion, refer to [1].

```
loop
    wait Δ
    p ← Random Peer
    sendPush(p, x)

```

```
6: procedure OnPush(m)
7:     sendPull($m.sender, x$)
8:     $x \leftarrow (m.x + x)/2$
9: procedure OnPull(m)
10:    $x \leftarrow (m.x + x)/2$
```

The algorithm can be shown to converge to the global average in the network. Now that we have established that we can compute the average, can we also compute other means ? Design a Gossip-style algorithm to compute geometric mean of values in the network.

Exercise 4 (Gossip-based resource assignment)**:**

Resource assignment to services and applications is crucial in many distributed settings. At one end of the spectrum, a centralized server can maintain a database of resource availability for the whole network. At the other end, one can think of a fully decentralized solution that automatically partitions the available nodes into "slices", taking into account specific attributes of the nodes (such as available bandwidth, storage capacity or number of processing units). The goal of this exercise is to design a gossip-based protocol that can achieve such a decentralized slicing.

We assume:

- The network is composed of a set of n nodes. Nodes are uniquely identified by their IP address.
- There are no failures, node arrivals nor departures.
- Each node is connected to a set of $k = log(n)$ other nodes called neighbors. This means that each node can communicate with this set of neighbors.
- We assume that the neighbors of the node change periodically (every t milliseconds), meaning every node periodically gets a new set of random neighbors (picked uniformly).
- A peer-sampling service provides the current view of k neighbors.
- All nodes execute the same code.
- **Hint**: attribute a random number to each node. Assume that the random number generator is uniform and there are no collisions.

1. [7 points] Design a gossip-based algorithm that arbitrarily divides the system into 10 slices of approximately equal size. Write the pseudocode of your protocol, so that by the end each node knows the other approximately $(n/10) - 1$ nodes in its slice.

2. [8 points] Assume now that each node i has an attribute x_i that measures the availability of some resource (*e.g.*, bandwidth) on node i. We assume that there exists a total ordering over the domain of the attribute values, so that the values in the network $(x_1, ..., x_n)$ can be ordered. Let us also assume that there is a slice specification that defines an ordered partitioning of the nodes. That is, the slice specification is a list of positive real numbers $s_1, ..., s_m$ such that for all $u < v, i \in S_u$ and $j \in S_v$ we have $x_i \leq x_j$. As opposed to question 1., a node does not need to know the other nodes in its slice.

   Design a gossip-based algorithm that assigns each node to one of 10 slices (m = 10) in such a way that it satisfies the slice specification, using only local message exchange with currently known neighbors. That is, we want each node to find out which slice (*i.e.*, which 10th-percentile) it belongs to. For example, each node can then know, assuming x represents the available bandwidth, if it belongs to the 10% of nodes with the highest available bandwidth.

All solutions will be in Moodle on 27.02.2023.

References

[1] Márk Jelasity. *Gossip-based protocols for large-scale distributed systems*. PhD thesis, mi, 2013.