The course

- **Class**: Wednesdays 10:30-12:30, Taub 4
- **Office hours**: Wednesdays 9:30-10:30, Taub 516
- **TA**: Seri Khoury

- **Prerequisites**: Data Structures, Introduction to Algorithms, Probability Theory

- **Grade structure**: 3 home assignments, final project

- **Book**: Distributed Computing: A Locality-Sensitive Approach
  David Peleg, SIAM 2000
What is distributed computing?

• Multiple *computing components* which *communicate* in order to perform some computation
  – Communication networks
  – Multicore computers

• Characterized by the absence of central control
Computing models

• First task: **model** the system

• Describe the computation as it happens in reality, but avoid details that are specific
  – Finite state machines
  – Turing machines
Distributed graph algorithms

- $n$ computing components
- **Communication**: sending messages along links

**Model:**
- Communication links described as a graph $G$
- The topology of $G$ is **unknown** to the nodes
- Typical knowledge of a node $v$:
  - A polynomial bound on the number of nodes $n$
  - The identities of its neighbors $N(v)$
Assumptions

- **Message length**: unbounded/bounded
- **Timing**: synchronous/asynchronous
- **Knowledge**:
  - The number of nodes $n$
  - The maximal degree in the graph $\Delta$
Complexity measures

• Basic underlying assumption:
  Communication is more expensive than local computation

• We typically measure communication
  – Number of communication rounds (synch)
  – Number of messages sent
  – Number of bits sent
Uncertainty

• Various additional aspects of distributed computing to be discussed depending on time
  – Faulty nodes or links
  – Dynamic changes to the network topology
  – Noise

• These characterize real systems and show the limited control we have over the computation
Model

• **Synchronous** system. **Complexity**: number of rounds
• In each round:

  - **LOCAL**

    node $v$ can send messages to every node in $N(v)$

  - **CONGEST**

    node $v$ can send messages of $O(\log n)$ bits to every node in $N(v)$
    
    – Motivation: sending an ID in a single message
A BFS tree

• Breadth-First Search Tree
  – Input: a root $r$
  – Output: a tree of shortest paths to $r$

• Distributed setting:
  – Input to node $v$: indication whether $v$ is $r$
  – Output: $d(v, r)$ and parent in the BFS tree
  • No knowledge of the entire tree is required
A BFS algorithm

Variables for node $v$:
\begin{itemize}
    \item \texttt{state} $\in \{\text{activated, deactivated}\}$, initially deactivated, except for $v = r$
    \item \texttt{parent}, initially $v$
    \item \texttt{dist}, initially $\infty$, except for $v = r$, initially 0
\end{itemize}

\begin{algorithm}
1 for $i=1, \ldots$ do
2 \hspace{1em} if \texttt{state} = activated then
3 \hspace{2em} send $i$ to all neighbors
4 \hspace{2em} \texttt{state} $\leftarrow$ deactivated
5 \hspace{1em} else
6 \hspace{2em} if receive message for first time (from node $w$) then
7 \hspace{3em} \texttt{parent} $\leftarrow$ $w$
8 \hspace{3em} \texttt{dist} $\leftarrow$ $i$
9 \hspace{3em} \texttt{state} $\leftarrow$ activated
\end{algorithm}
Correctness

• **Claim 1**: There is a BFS tree $T$ rooted at $r$, such that after the algorithm runs, for each node $v$:
  – $\text{parent}(v)$ is set to the parent of $v$ in $T$
  – $\text{dist}(v)$ is set to $d(v,r)$

• **Observation**: a node gets activated at most once
  – Because it becomes activated only when the condition in line 6 holds (the first message is received), which can happen only once.
**Claim 1:** parent(v) is set to the parent of v in T
dist(v) is set to d(v,r)

**Proof:**
- We prove the following by induction over the rounds. We show that each node that is **activated** in round i produces the correct **distance** and **parent** in round i − 1, and that all nodes in distance i − 1 are indeed **activated** in round i.
Claim 1: parent(v) is set to the parent of v in T
dist(v) is set to d(v,r)

Proof:
• The base case: this is for round 1, where only r is activated. Its parent does not change and it outputs distance 0 before the execution begins. Its message reaches all nodes within distance 1 and only them, and so they become activated for the next round.
Claim 1: parent(v) is set to the parent of v in T
\[ \text{dist}(v) \text{ is set to } d(v,r) \]

Proof:

- **Induction hypothesis**: Assume that every node in distance \( i - 1 \) from \( r \) is activated in round \( i \), and that for every node that is activated in round \( i \), its distance from \( r \) is \( i - 1 \) and its parent points to a neighbor in distance \( i - 2 \).
Claim 1: parent(v) is set to the parent of v in T
        dist(v) is set to d(v,r)

Proof:
• Induction step: Let v be a node in distance i, then v is activated in round i + 1 because it has a neighbor w within distance i – 1. By the induction hypothesis w is activated in round i and so v receives its message and becomes activated for round i + 1.
Claim 1: parent(v) is set to the parent of v in T
   dist(v) is set to d(v, r)

Proof:
• Further, it means that v updates its parent to
   some node within distance i – 1, and outputs
   distance i in the previous round.
• It remains to show that the other direction also
   holds, that is, if v is activated in round i + 1 then
   indeed its distance is i. This is because if it is
   activated then it is because it receives a message
   from some w in round i, and by the induction
   hypothesis the distance of w is i – 1.
Complexity

• **Claim 2**: The number of rounds is $O(D)$

• **Notation**: $D$ is the diameter of the graph

• **Proof**: By Claim 1, a node in distance $i$ from $r$ completes after $i$ rounds, and hence the number of rounds is the depth of $T$, which is $O(D)$. 
Lower bound

• **Claim 3**: Every synchronous BFS algorithm requires $\Omega(D)$ rounds.

• Intuitively, this is because a node within distance D from the root r can find it out only after D rounds.