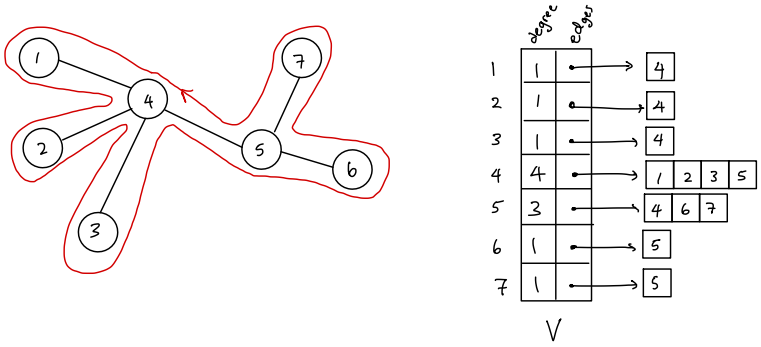
Homework Assignment #2
Due: September 19, 2025 at 5:00 p.m.

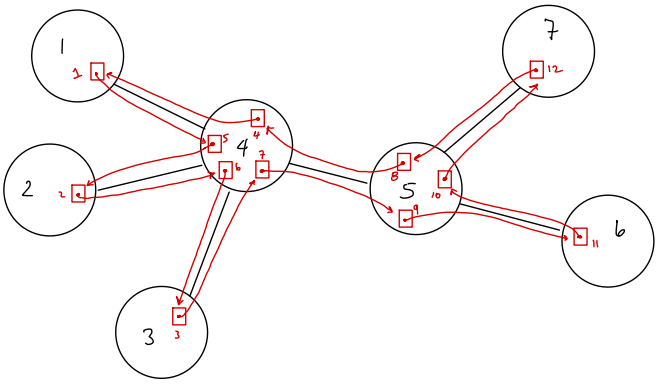
1. In class, we discussed the Euler tour technique and mentioned that it could be applied to an arbitrary unrooted tree (not just binary rooted trees). Here we discuss some details of doing this.

Suppose that the tree is represented using adjacency arrays as follows. Assume the tree vertices are numbered 1 to n. There is an array $V[1..n]$ of Vertex objects. Each Vertex object v has two fields: *degree*, which stores the degree of the vertex, and *edges*, which is a pointer to an array of vertices adjacent to v. Assume the array $edges[i]$ is sorted in increasing order, for each i. An example tree is shown below, with an Euler tour in red and the adjacency array representation of the tree.



Let $degree(v)$ be the number of edges incident with a tree vertex v. Recall that a tree with n nodes has exactly $n-1$ edges. Since each edge is counted in the degree of its two endpoints, we have $\sum_v degree(v) = 2(n-1)$.

The Euler tour technique creates $2(n-1)$ nodes (shown as red boxes in the figure below). It associates $degree(v)$ nodes with each tree vertex v and arranges the nodes into a circular singly-linked list so that, for each tree edge (u, v), there is one list edge from a node associated with vertex u to a node associated with vertex v and one list edge from a node associated with vertex v to a node associated with vertex u.



vertex	1	2	3	4	4	4	4	5	5	5	6	7
next Vertex	4	4	4	1	2	3	5	4	6	7	5	5
next Node	5	6	7	1	2	3	9	4	11	12	10	8
	1	2	3	4	5	6	7	8	9	10	11	12

E

We represent the Euler tour list using an array E of Node objects. Each Node has three fields:

- *vertex* stores the vertex that the Node is associated with,
- $nextVertex$ stores the vertex that the next Node in the linked list is associated with, and
- $nextNode$ stores the index in E of the next Node.

In the drawing of the tree, the red numbers beside the nodes are the indices of the nodes in E.

[4] **(a)** Give detailed pseudocode to compute E from V on a CREW PRAM using $O(n)$ processors. (If given the example array V shown above, it should produce the array E shown above.) You can use subroutines like prefix sum or list ranking described in class without giving the pseudocode for them. How much time does your computation take?

[4] **(b)** We can *root* the tree by picking one vertex r and calling it the root. In the Euler tour, we pick any one of the list nodes associated with r and change its next pointer to nil. Describe how to use (unweighted) list ranking to determine the parent of each vertex in the rooted tree.

[2] **(c)** Illustrate your answer to part (b) with the example tree shown above by breaking the link from node 3 to node 7 (thereby making vertex 3 the root of the tree) and explaining how your algorithm determines the parent of vertex 5.

2. Page 46–47 of the textbook describes how to find the maximum value in an array $A[1..n]$ of integers in $O(1)$ time using n^2 processors. Each of the n^2 integers is assigned to compare a pair of values $A[i]$ and $A[j]$. Then, for each i, one processor checks if *all* of the comparisons of the form $A[i] \geq A[j]$ returned true (using your answer to part (a), below). If so, $A[i]$ is the maximum element.

In this question, you will find the maximum in $O(1)$ time using fewer than n^2 processors. All of your answers should work on a COMMON CRCW PRAM.

[2] **(a)** Suppose you are given an array $B[1..n]$ of boolean values. Describe how to compute the logical AND of all the values, $B[1] \wedge B[2] \wedge \ldots \wedge B[n]$, in a single step using n processors.

[1] **(b)** How many processors do you need to find the maximum of $\sqrt{n}$ integers in $O(1)$ time using the textbook's maximum algorithm?

[4] **(c)** Describe how to find the maximum value in an array $A[1..n]$ of integers in $O(1)$ time using $O(n^{1.5})$ processors.

Hint: divide the array A into blocks and first find the maximum within each block.