

## Problems for Week 5

### (a) Recognizing bipartite graphs

A *bipartite* graph is an undirected graph  $G = (V, E)$  in which  $V$  can be partitioned into two sets  $V_1$  and  $V_2$  such that all edges go between the two sets  $V_1$  and  $V_2$ ; that is,  $(u, v) \in E$  implies either  $u \in V_1$  and  $v \in V_2$  or  $u \in V_2$  and  $v \in V_1$ .

- (i) Prove that an undirected graph is bipartite if and only if it has no cycles of odd length.
- (ii) Using breadth-first search, design and analyze an algorithm to determine whether a given undirected graph is bipartite.

### (b) Two stacks and a queue

This is a simple exercise in amortized analysis. A stack is a dynamic set that implements a *last-in first-out* (LIFO) policy. It supports two operations: PUSH and POP. The operation PUSH( $x$ ) inserts new element  $x$  into the stack, and the operation POP deletes (and returns) the last element inserted into the stack.

A queue is a dynamic set that implements a *first-in first-out* (FIFO) policy. It supports two operations: ENQUEUE and DEQUEUE. The operation ENQUEUE( $x$ ) inserts new element  $x$  into the queue, and the operation DEQUEUE deletes (and returns) the element that has been in the queue for the longest time.

Show how to implement a queue with two stacks so that the amortized cost of each ENQUEUE and DEQUEUE operation is  $O(1)$ .