**Shortest directed cycle.** Given a digraph *G*, design an efficient algorithm to find a directed cycle with the minimum number of edges (or report that the graph is acyclic). The running time of your algorithm should be at most proportional to *V*(*E*+*V*) and use space proportional to *E*+*V*, where *V* is the number of vertices and *E* is the number of edges.

**Hamiltonian path in a DAG.** Given a directed acyclic graph, design a linear-time algorithm to determine whether it has a *Hamiltonian path* (a simple path that visits every vertex), and if so, find one.

**Reachable vertex.**

* DAG: Design a linear-time algorithm to determine whether a DAG has a vertex that is reachable from every other vertex, and if so, find one.
* Digraph: Design a linear-time algorithm to determine whether a digraph has a vertex that is reachable from every other vertex, and if so, find one.