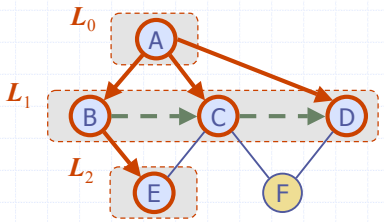
Breadth-First Search



Breadth-First Search (§ 12.3.3)

- Breadth-first search (BFS) is a general technique for traversing a graph
- A BFS traversal of a graph G
  - Visits all the vertices and edges of G
  - Determines whether G is connected
  - Computes the connected components of G
  - Computes a spanning forest of G
- BFS on a graph with n vertices and m edges takes $O(n + m)$ time
- BFS can be further extended to solve other graph problems
  - Find and report a path with the minimum number of edges between two given vertices
  - Find a simple cycle, if there is one

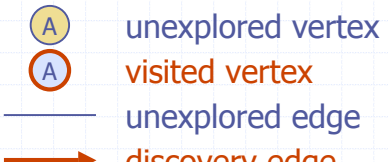
BFS Algorithm

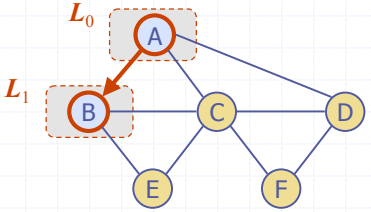
- The algorithm uses a mechanism for setting and getting "labels" of vertices and edges

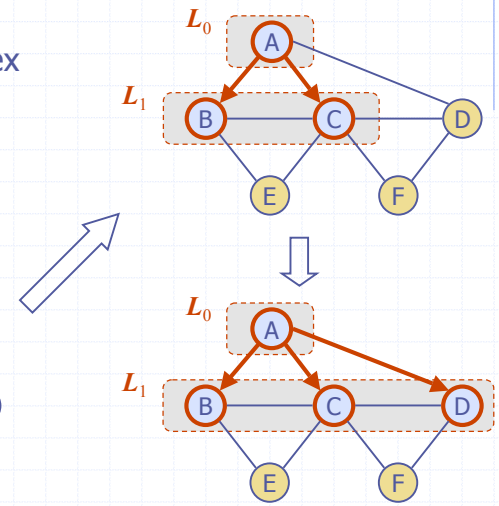
```
Algorithm BFS(G)
    Input graph G
    Output labeling of the edges
        and partition of the
        vertices of G
    for all u ∈ G.vertices()
        setLabel(u, UNEXPLORED)
    for all e ∈ G.edges()
        setLabel(e, UNEXPLORED)
    for all v ∈ G.vertices()
        if getLabel(v) = UNEXPLORED
            BFS(G, v)
```

```
Algorithm BFS(G, s)
    L0 ← new empty sequence
    L0.insertLast(s)
    setLabel(s, VISITED)
    i ← 0
    while ¬Li.isEmpty()
        Li+1 ← new empty sequence
        for all v ∈ Li.elements()
            for all e ∈ G.incidentEdges(v)
                if getLabel(e) = UNEXPLORED
                    w ← opposite(v,e)
                    if getLabel(w) = UNEXPLORED
                        setLabel(e, DISCOVERY)
                        setLabel(w, VISITED)
                        Li+1.insertLast(w)
                    else
                        setLabel(e, CROSS)
        i ← i +1
```

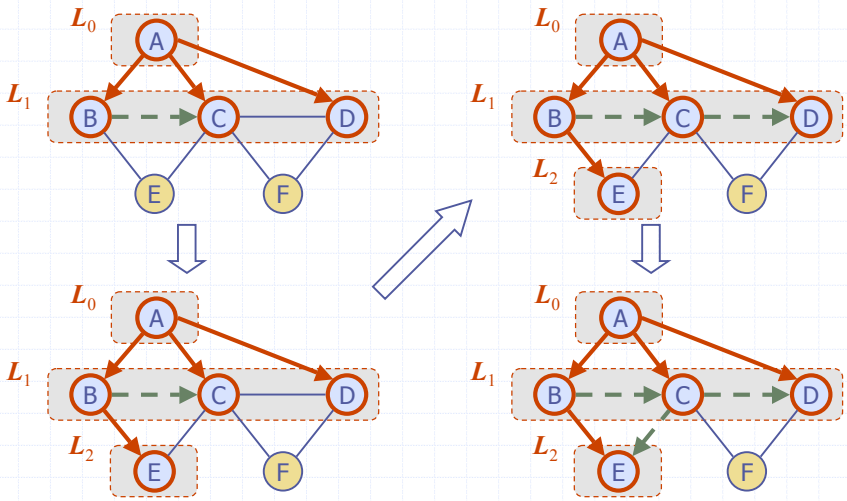

Example







Example (cont.)



Example (cont.)