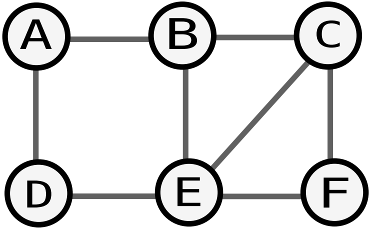
Synchronous single initiator spanning tree algorithm using flooding

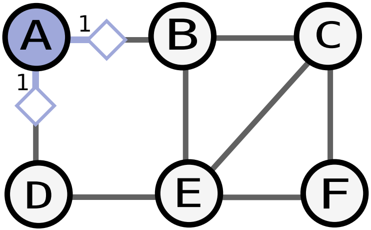
Siddharth Bhat, Anurag Chaturvedi, Hitesh Kaushik

March 13, 2020

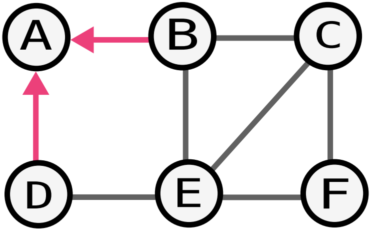
Example



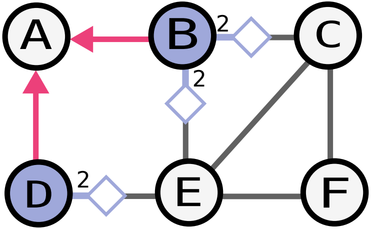
Example



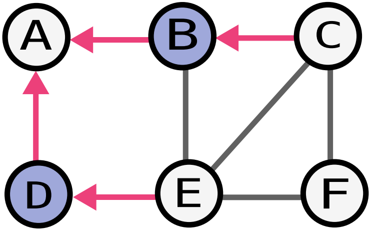
Example



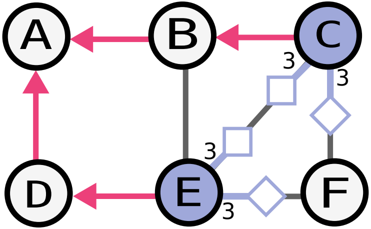
Example



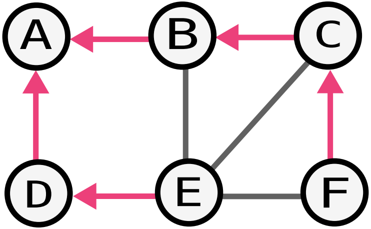
Example



Example



Example



Synchronous BFS (Pseudocode)

- Assume root begins computation.
- Algorithm is synchronous.

Synchronous BFS (Pseudocode)

- Assume root begins computation.
- Algorithm is synchronous.

```
def bfs_spanning_tree(self):
```

Synchronous BFS (Pseudocode)

- Assume root begins computation.
- Algorithm is synchronous.

```
def bfs_spanning_tree(self):
  if self.id == ROOT_ID:
    self.visited = True; self.depth = 0;
    for n in self.neighbours: n.send(self.id)
```

Synchronous BFS (Pseudocode)

- Assume root begins computation.
- Algorithm is synchronous.

```
def bfs_spanning_tree(self):
  if self.id == ROOT_ID:
    self.visited = True; self.depth = 0;
    for n in self.neighbours: n.send(self.id)
  for round in range(1, DIAMETER+1):
    if self.visited: # if visited, skip
```

Synchronous BFS (Pseudocode)

- Assume root begins computation.
- Algorithm is synchronous.

```
def bfs_spanning_tree(self):
  if self.id == ROOT_ID:
    self.visited = True; self.depth = 0;
    for n in self.neighbours: n.send(self.id)
  for round in range(1, DIAMETER+1):
    if self.visited: # if visited, skip
      if self.queries: # if we have a query
        # randomly choose from queries
        parent = random.choice(self.query)
        self.visited = True
        self.depth = round
```

Synchronous BFS (Pseudocode)

- Assume root begins computation.
- Algorithm is synchronous.

```
def bfs_spanning_tree(self):
  if self.id == ROOT_ID:
    self.visited = True; self.depth = 0;
    for n in self.neighbours: n.send(self.id)
  for round in range(1, DIAMETER+1):
    if self.visited: # if visited, skip
      if self.queries: # if we have a query
        # randomly choose from queries
        parent = random.choice(self.query)
        self.visited = True
        self.depth = round

        # synchronous
        for n in self.neighbours: n.send(self.id)
    self.queries = [];
```