Scheduling

Set of tasks for baking cookies:

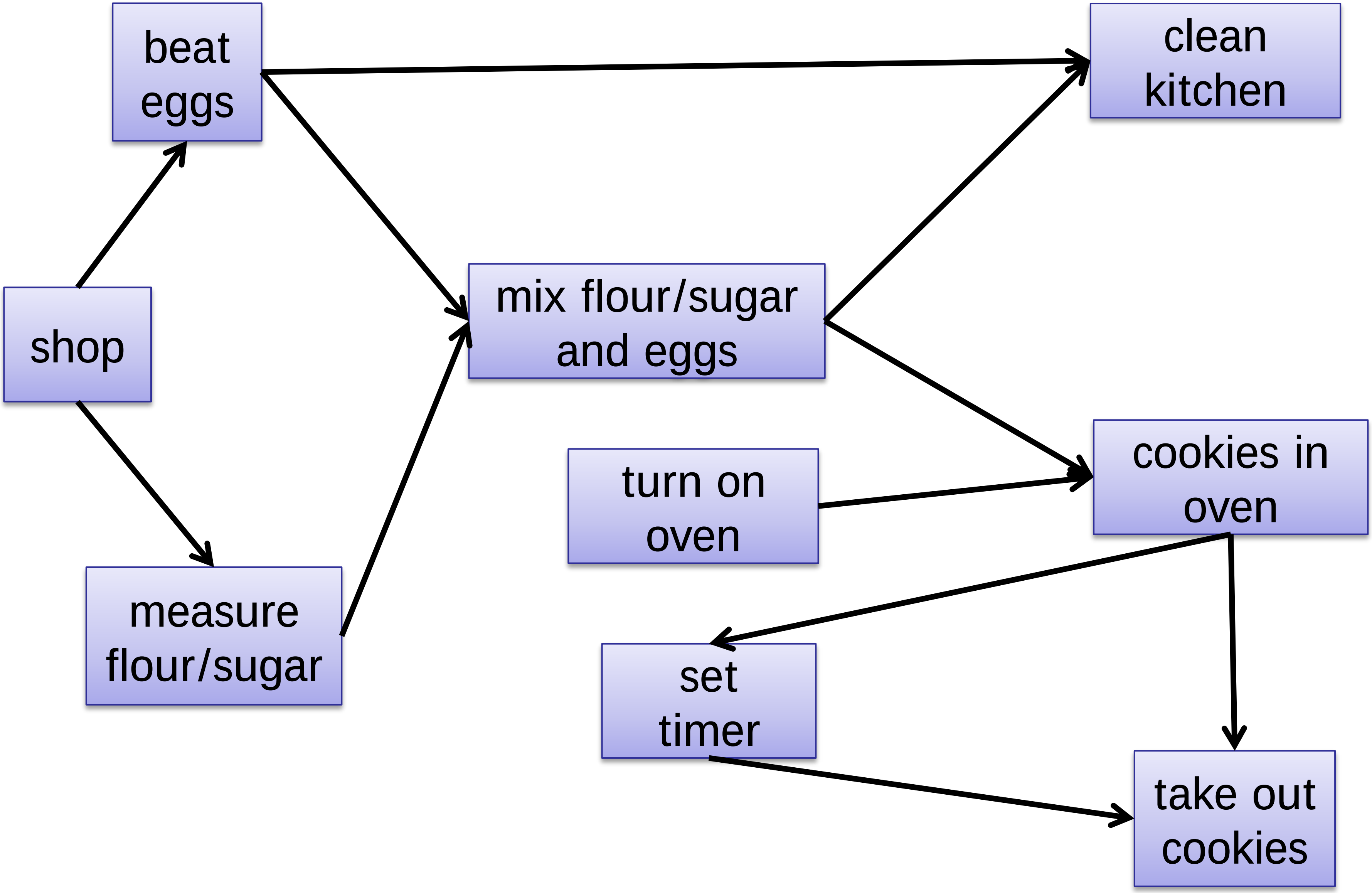
- Shop for groceries
- Put the cookies in the oven
- Clean the kitchen
- Beat the eggs in a bowl
- Measure the flour and sugar in a bowl
- Mix the eggs with the flour and sugar
- Turn on the oven
- Set the timer
- Take out the cookies

Scheduling

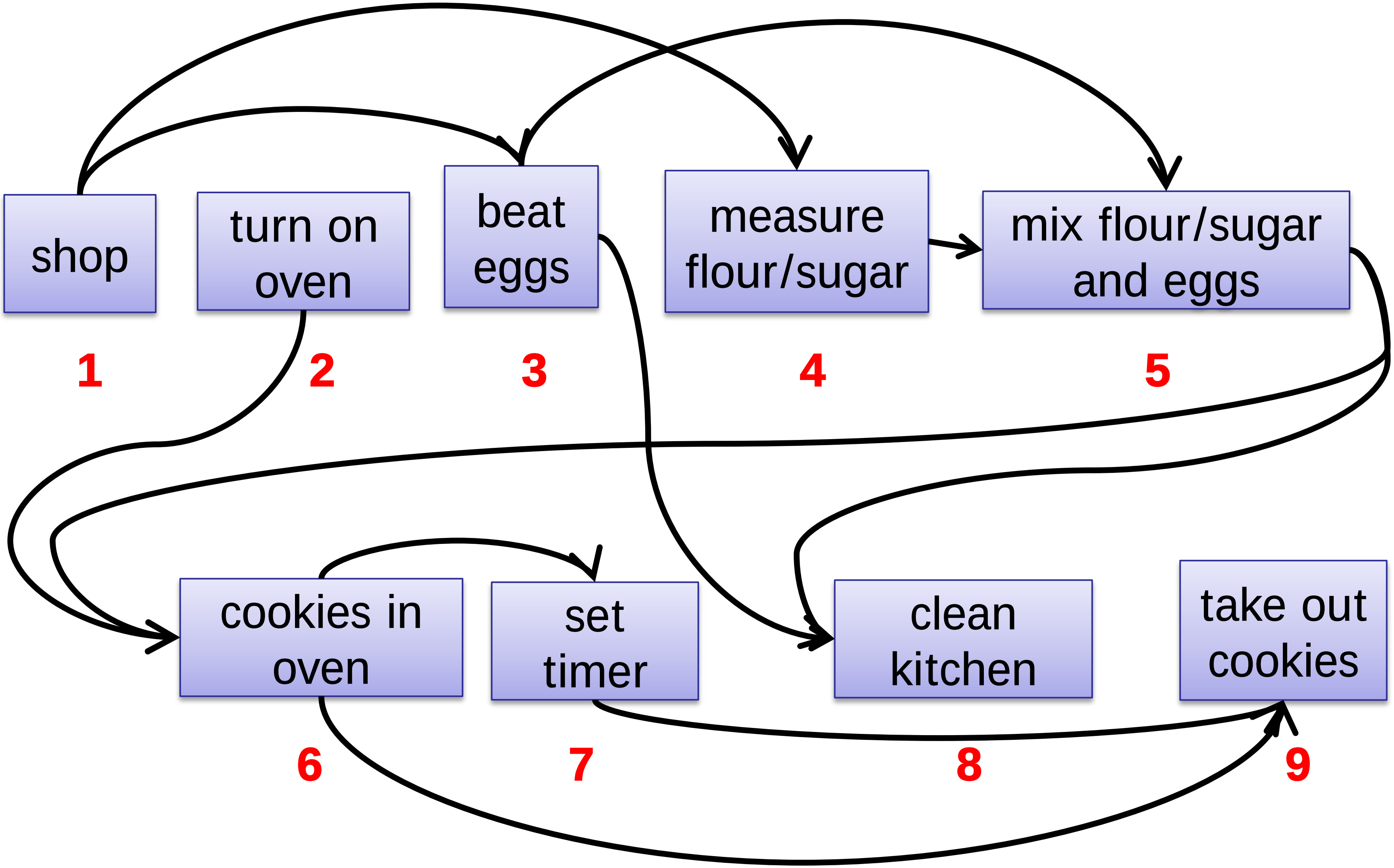
Ordering:

- Shop for groceries before beat the eggs
- Shop for groceries before measure the flour
- Turn on the oven before put the cookies in the oven
- Beat the eggs before mix the eggs with the flour
- Measure the flour before mix the eggs with the flour
- Put the cookies in the oven before set the timer
- Measure the flour before clean the kitchen
- Beat the eggs before clean the kitchen
- Mix the flour and the eggs before clean the kitchen

Scheduling


beat eggs
clean kitchen
shop
mix flour/sugar and eggs
turn on oven
cookies in oven
measure flour/sugar
set timer
take out cookies

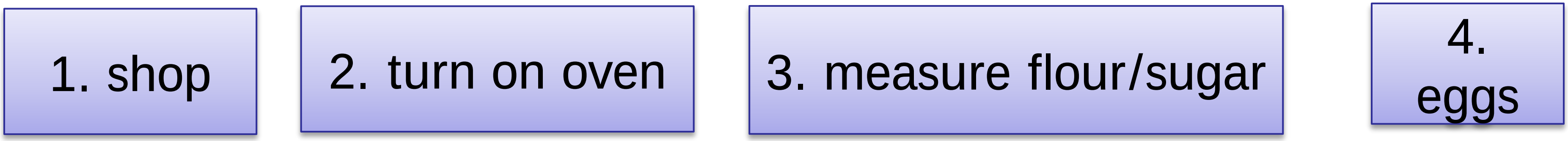
Topological Ordering



Topological Order

Properties:

1. Sequential total ordering of all nodes



1. shop

2. turn on oven

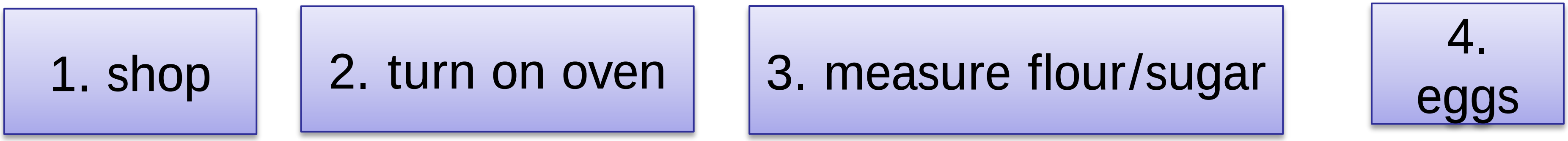
3. measure flour/sugar

4. eggs

Topological Order

Properties:

1. Sequential total ordering of all nodes



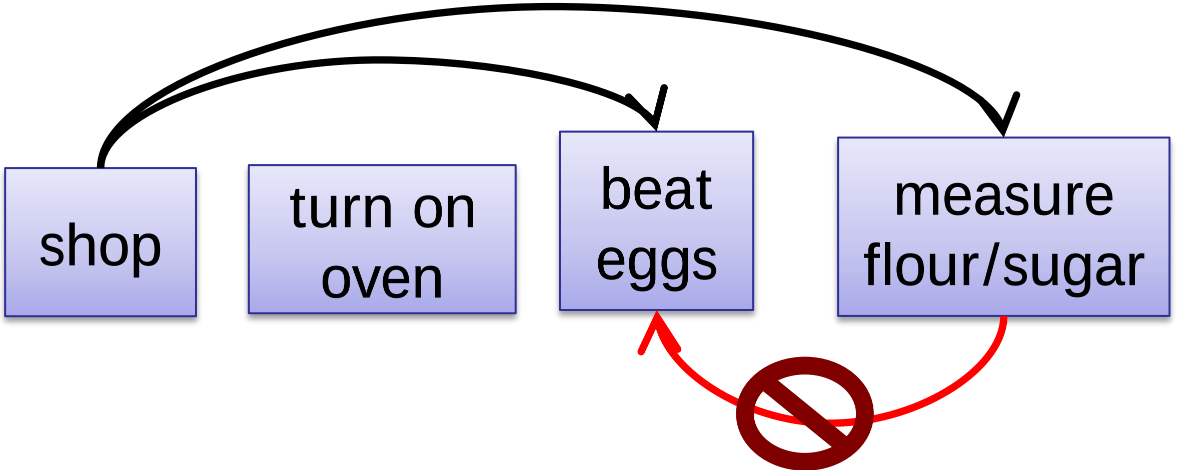
1. shop

2. turn on oven

3. measure flour/sugar

4. eggs

2. Edges only point forward



shop

turn on oven

beat eggs

measure flour/sugar

Topological Sort

Input:

- Directed Acyclic Graph (DAG)

Output:

- Total ordering of nodes, where all edges point forwards.

Algorithm:

- Post-order Depth-First Search
- O(V + E) time complexity

Topological Sort

Alternative algorithm:

Input: directed graph G

Repeat:

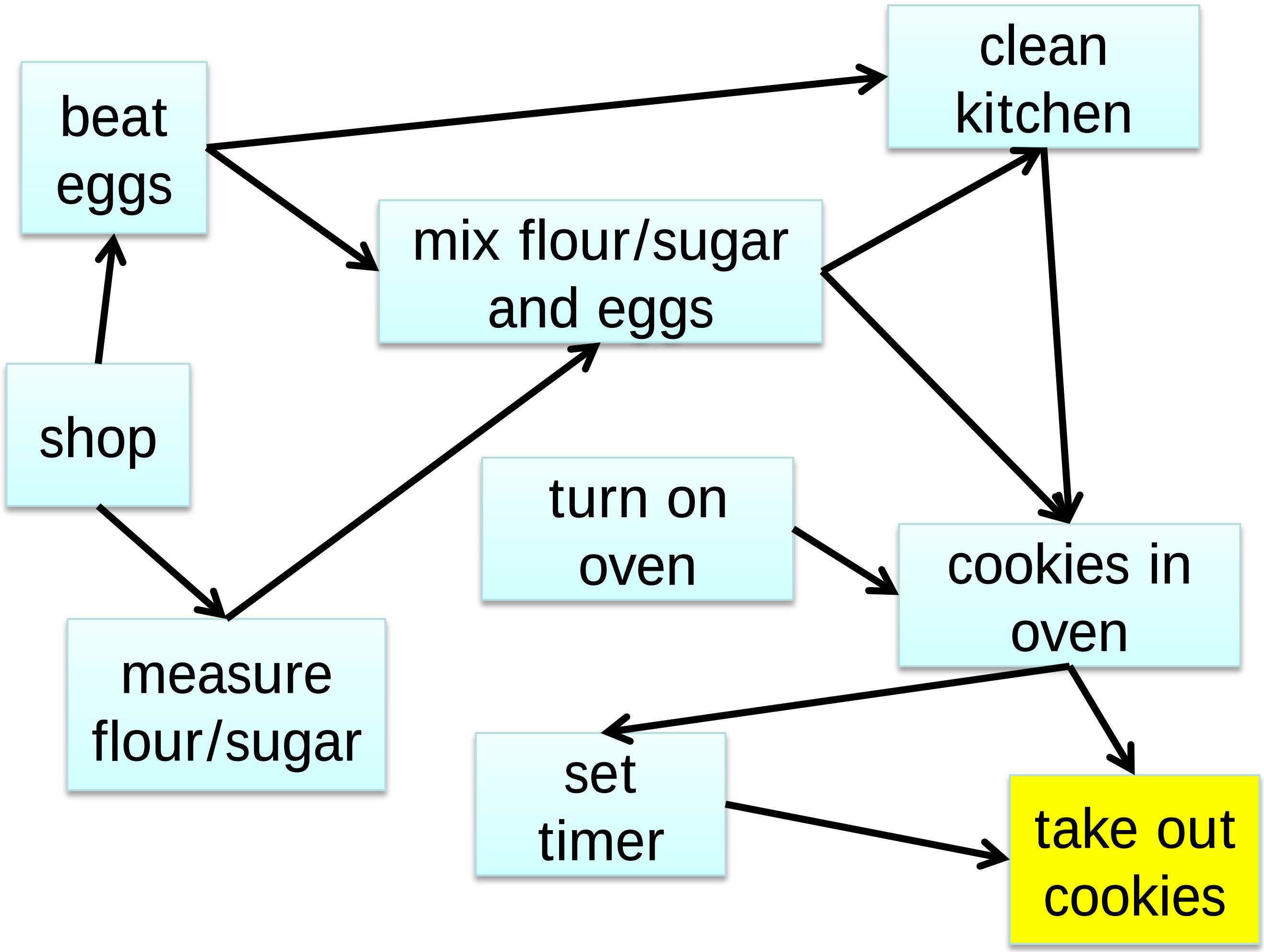
- S = all nodes in G that have *no* incoming edges.
- Add nodes in S to the topo-order
- Remove all edges adjacent to nodes in S
- Remove nodes in S from the graph

Time:

– O(V + E) time complexity

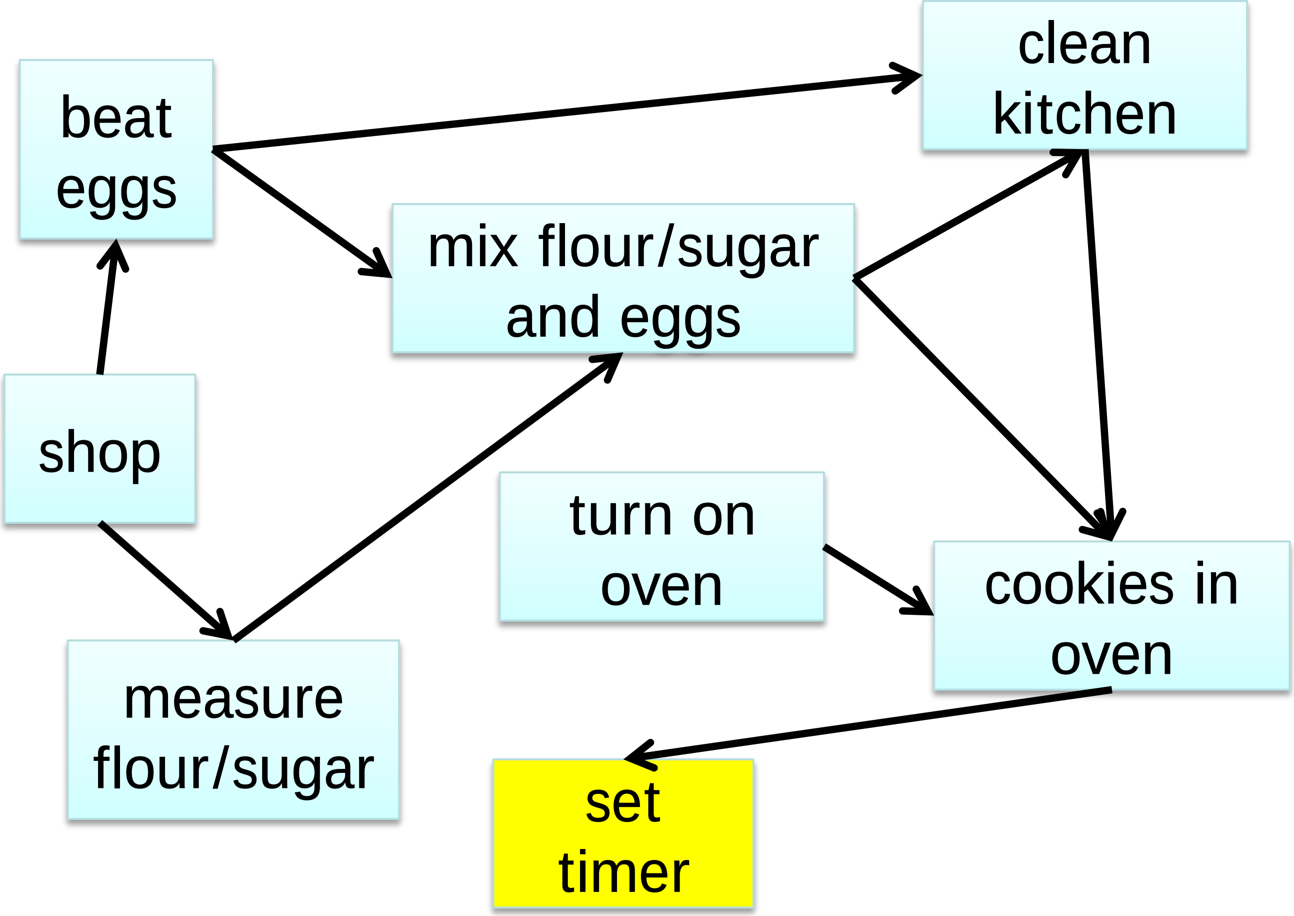
Alternative Algorithm

1.
2.
3.
4.
5.
6.
7.
8.
9.



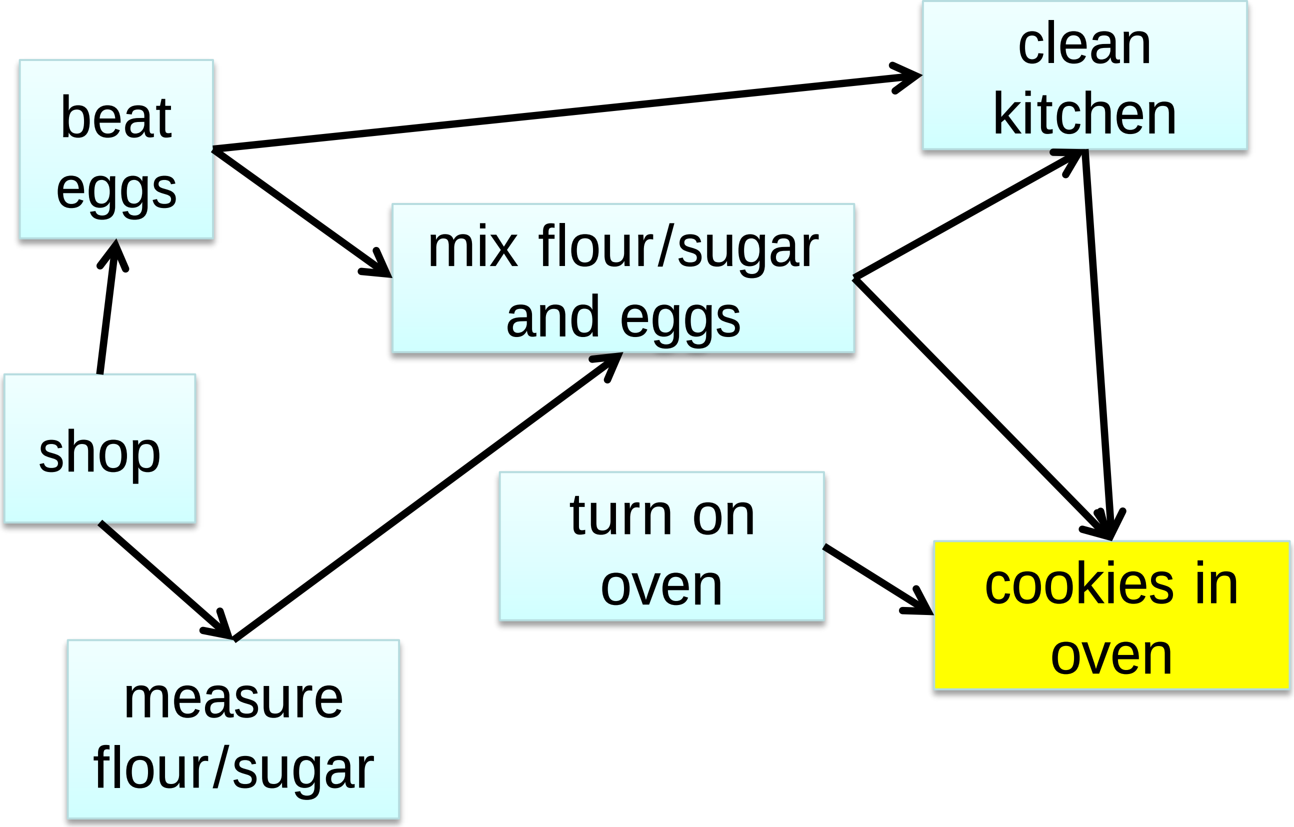
Alternative Algorithm

1.
2.
3.
4.
5.
6.
7.
8.
9. Take out



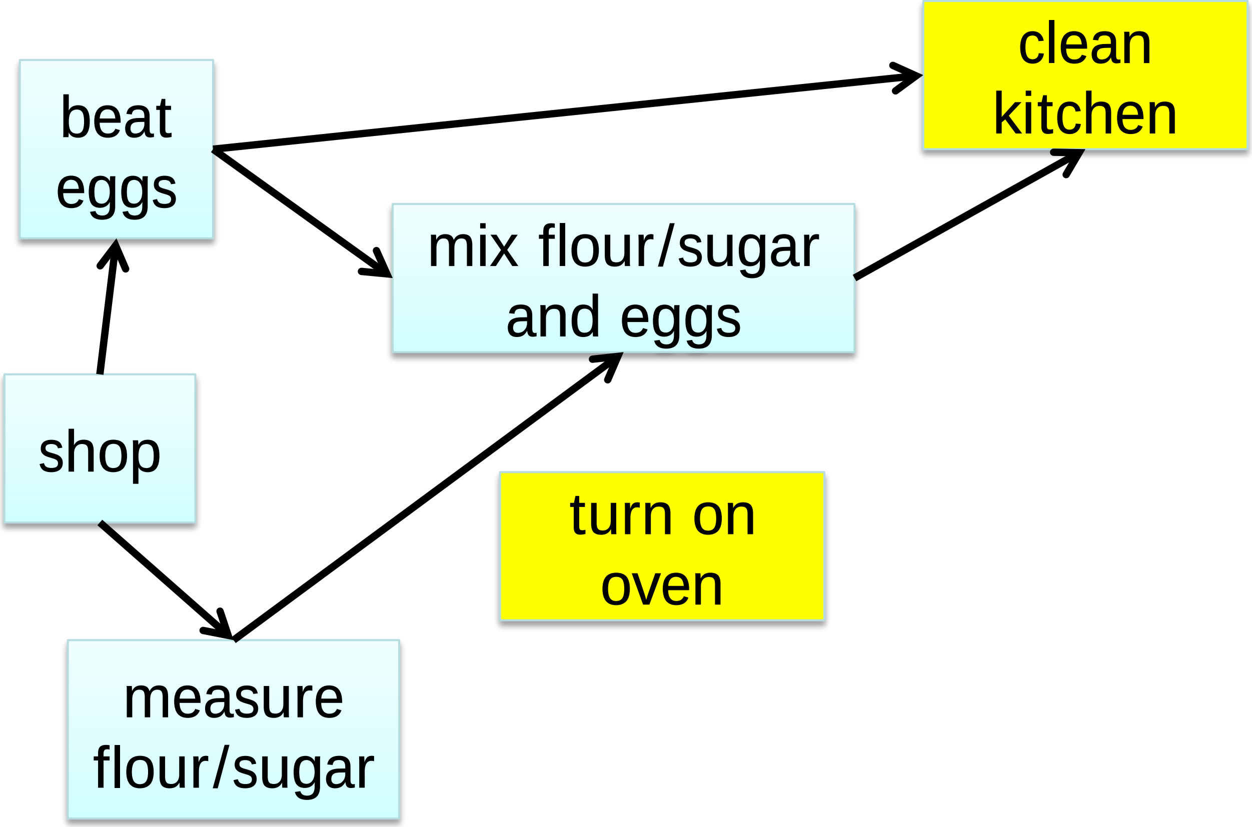
Alternative Algorithm

1.
2.
3.
4.
5.
6.
7.
8. Set Timer
9. Take out



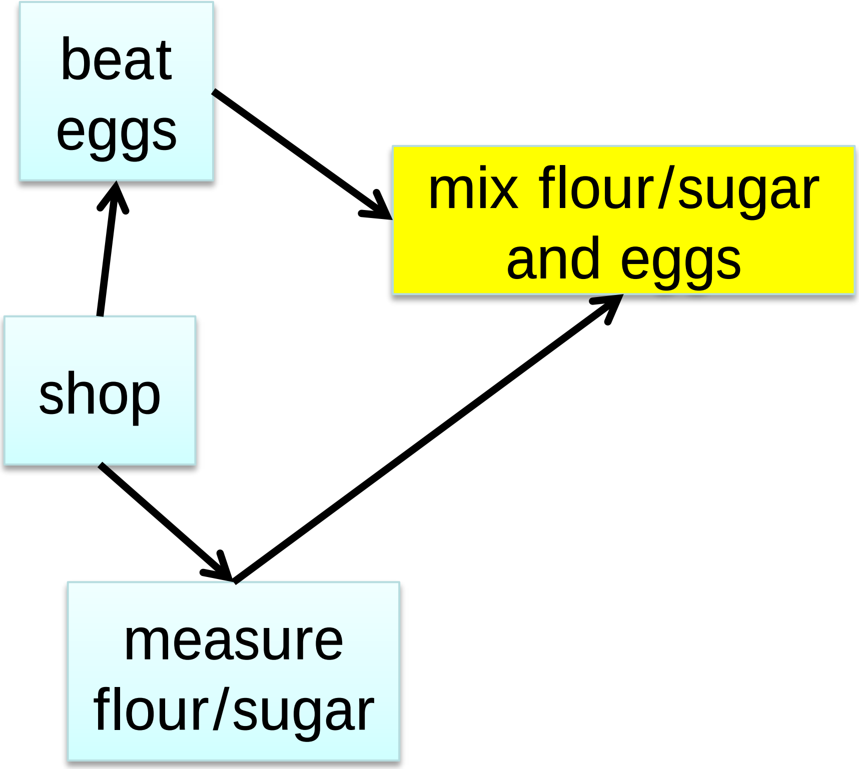
Alternative Algorithm

1.
2.
3.
4.
5.
6.
7. in oven
8. Set Timer
9. Take out



Alternative Algorithm

1.
2.
3.
4.
5. on oven
6. clean
7. in oven
8. Set Timer
9. Take out


beat eggs

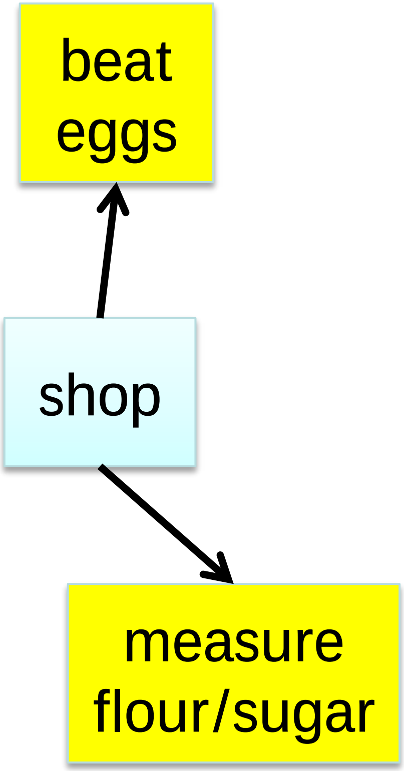
mix flour/sugar and eggs

shop

measure flour/sugar

Alternative Algorithm

1.
2.
3.
4. mix
5. on oven
6. clean
7. in oven
8. Set Timer
9. Take out



beat eggs

shop

measure flour/sugar

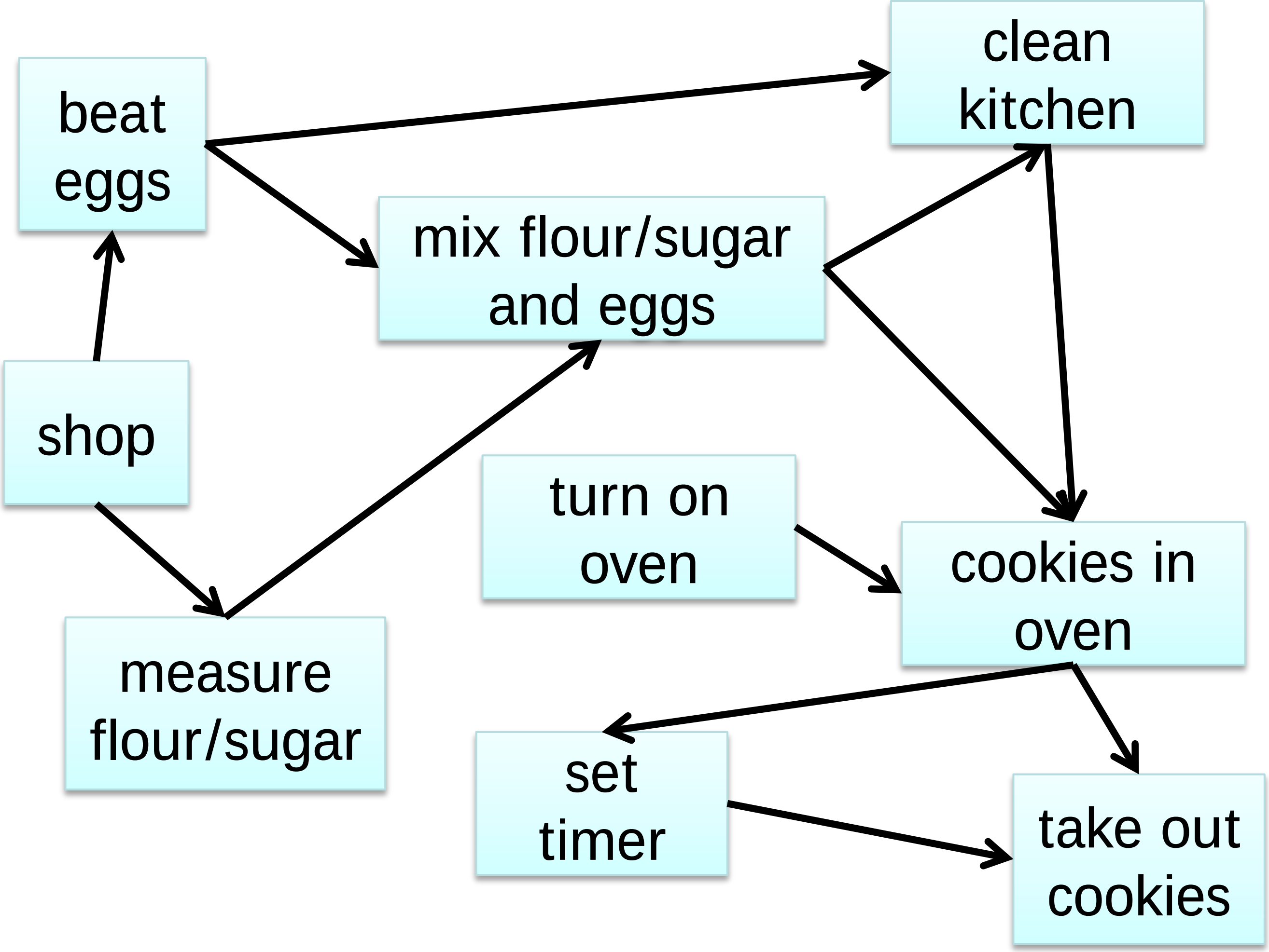
Alternative Algorithm

1.
2. beat eggs
3. measure
4. mix
5. on oven
6. clean
7. in oven
8. Set Timer
9. Take out

shop

Alternative Algorithm

1. Shop
2. beat eggs
3. measure
4. mix
5. on oven
6. clean
7. in oven
8. Set Timer
9. Take out



Kruskal’s Algorithm

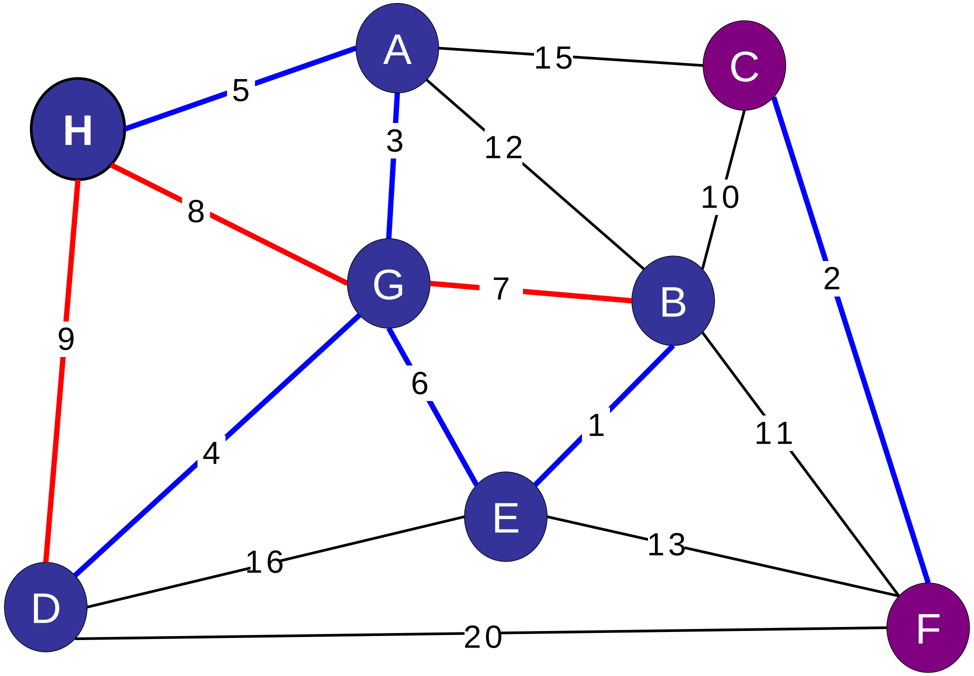
Kruskal’s Algorithm. (Kruskal 1956)

Basic idea:

- Sort edges by weight.
- Consider edges in ascending order:
  - If both endpoints are in the **same** blue tree, then color the edge red.
  - Otherwise, color the edge blue.

Data structure:

- Union-Find
- Connect two nodes if they are in the same blue tree.



Prim’s Algorithm

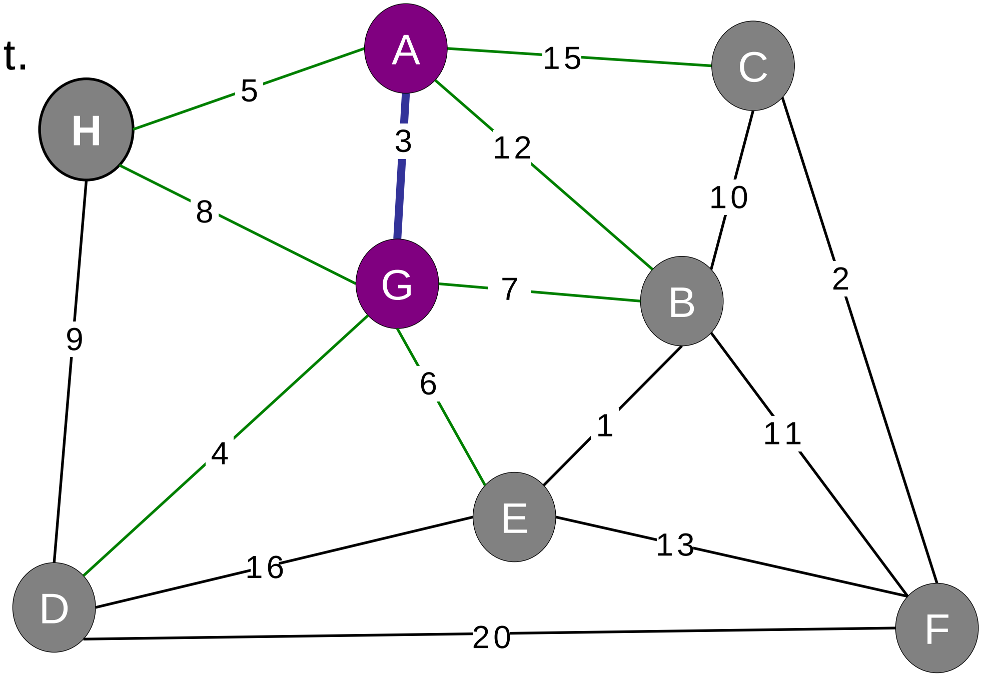
Prim’s Algorithm. (Jarnik 1930, Dijkstra 1957, Prim 1959)

Basic idea:

- S : set of nodes connected by blue edges.
- Initially: S = {A}
- Repeat:
  - Identify cut: {S, V–S}
  - Find minimum weight edge on cut.
  - Add new node to S.

Proof:

- Each added edge is the lightest on some cut.
- Hence each edge is in the MST.


A
C
H
G
B
E
D
F
15
5
3
12
10
8
7
2
9
6
1
11
4
13
16
20