BB: BLACK BOX

(Building Blocks-See motivation below)

Algo1 : BFS(G)$_u$: (M,P) / O(V+E)
Algo2 : BFS(G)$_X$: BLOCK / O(V+E)
Algo3 : DFS(G)$_u$: RPO / O(V+E)
Algo4 : DFS(G)$_X$: RPO / O(V+E)
Algo5 : META($G^{\rightarrow}$) : $G^M = (V^M, E^M)$ / O(V+E)
Algo6 : SP_BFS(G)$_s$: ($Dist_s$[.], P) / O(V+E)
Algo7 : DIJK(G)$_s$: ($Dist_s$[.], P) / O((V+E)*logV)
Algo8 : BFM($G^{\rightarrow}$)$_s$: ($Dist_s$[.], P) / O(V*E)
Algo9 : SP_DAG($G^{\rightarrow}$)$_s$: ($Dist_s$[.], P) / O(V+E)
Algo10 : CYCLE($G^{\rightarrow}$)$_s$: YES/NO / O(V*E)
Algo11 : MST(G): T/O((V+E)*logV) / O(E*logV)
Algo12 : COL(G) : M = array of colors / O(V+E)

Algo13: TO($G^{\rightarrow}$) :topological order+cycle / O(V+E)
Algo14 : SCC($G^{\rightarrow}$) : M / O(V+E)
Algo15: TC($G^{\rightarrow}$) : M transitive closure, /O(V(V+E))
Algo16: SINK(G^M) = the set of strongly components without outgoing edges / O(V+E)
Algo17: SOURCE(G^M) = the set of strongly components without ingoing edges / O(V+E)
Algo18: JOHNSON(G):(Dist[.][.])/O(V*(V+E)*logV)

A1: PATH(P, s, t): path from s to t / O(V+E)
A2:REVERSE($G^{\rightarrow}$) : reverse of $G^{\rightarrow}$/ O(V+E)
A3:DEGREE(G) : degrees of V/ O(V+E)

Black Box: Motivation

Developing new graph algorithms is hard! Often, it is easier to solve a problem on graphs by *reusing* existing graph algorithms (BB above).

Key idea: *Use* an existing graph algorithm as a "*black box*" with known properties and a known runtime.

a) **Makes algorithm easier to write: can just use an *off-the-shelf* implementation.**
b) **Makes correctness proof easier: can "*piggyback*" on top of the existing correctness proof.**
c) **Makes algorithm easier to analyze: runtime of key subroutine is known.**