Fence-Insertion
for Structured Programs

Arash Pourdamghani

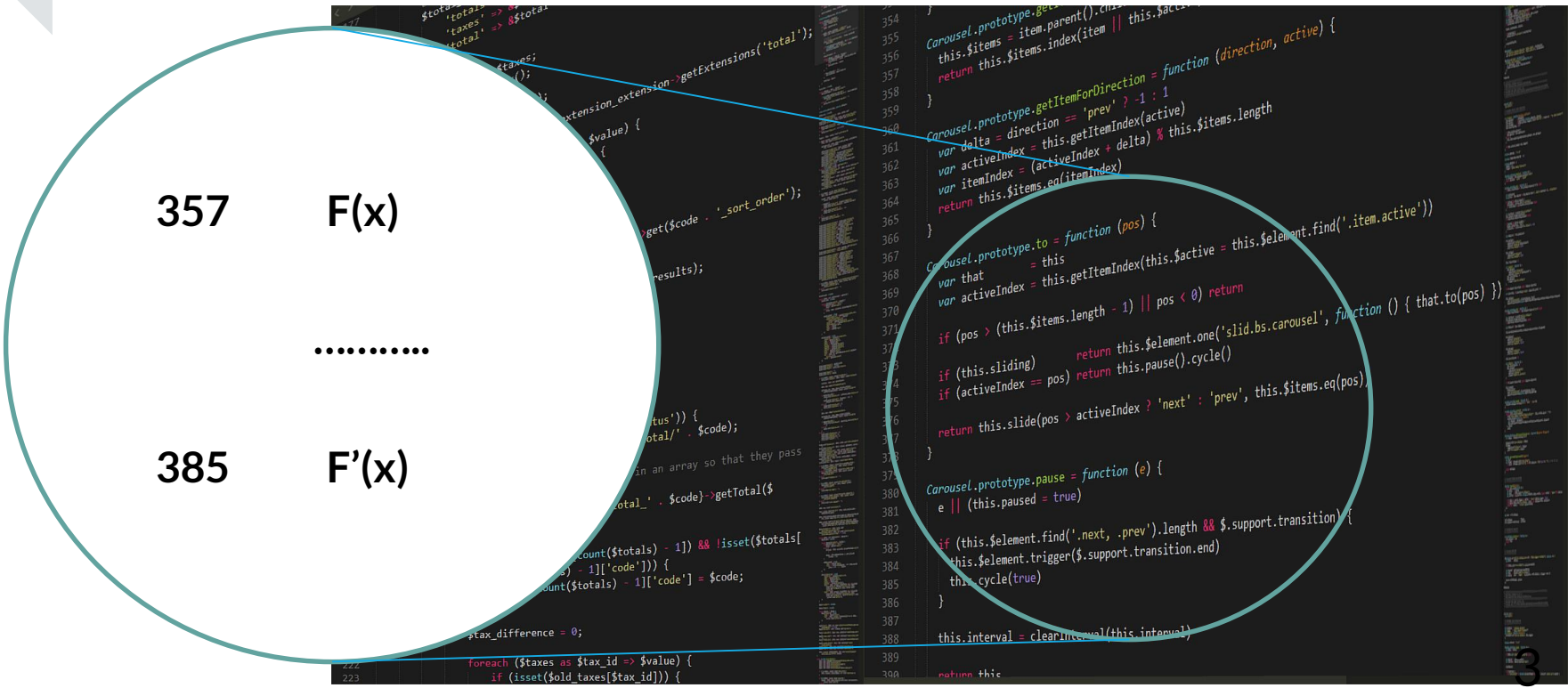
Mohammad Taheri

Mohsen Lesani

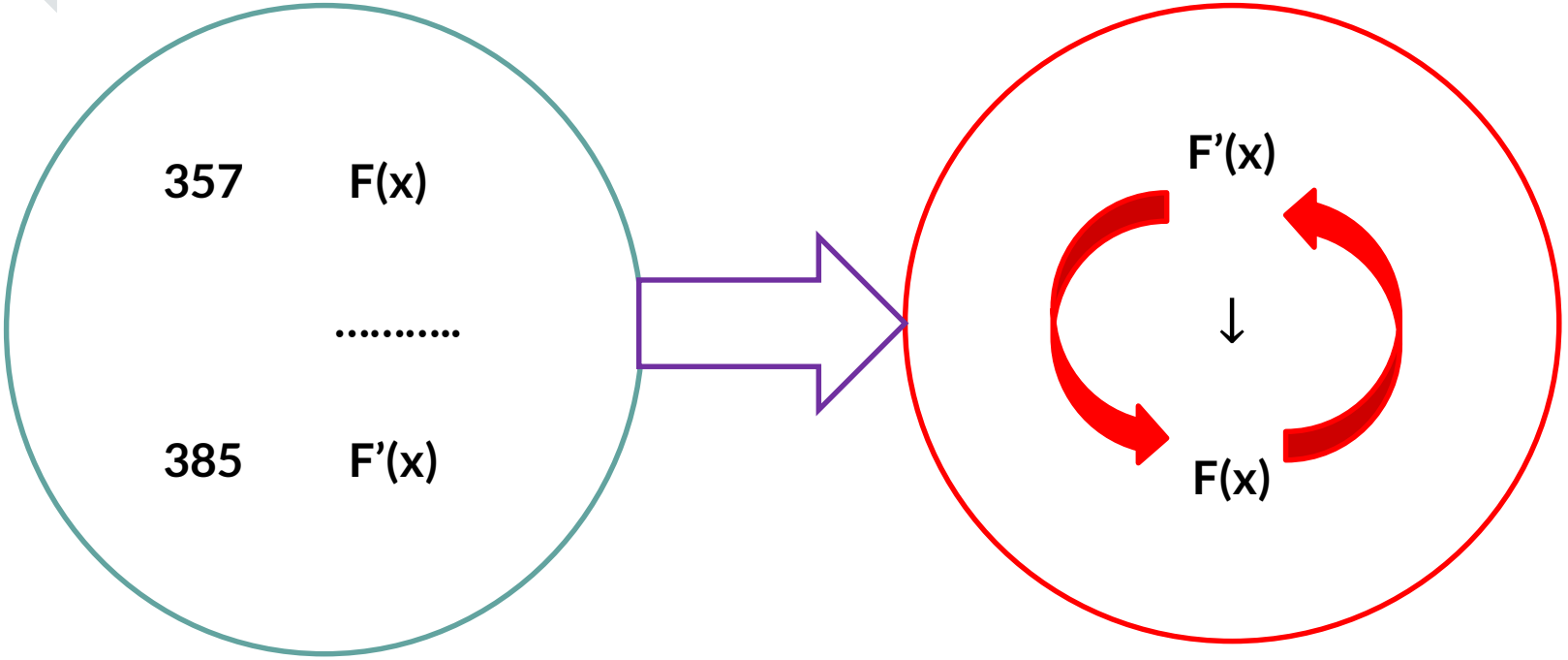
Spring 2020
University of Vienna

Developing a Distributed System!

Looking Deeper



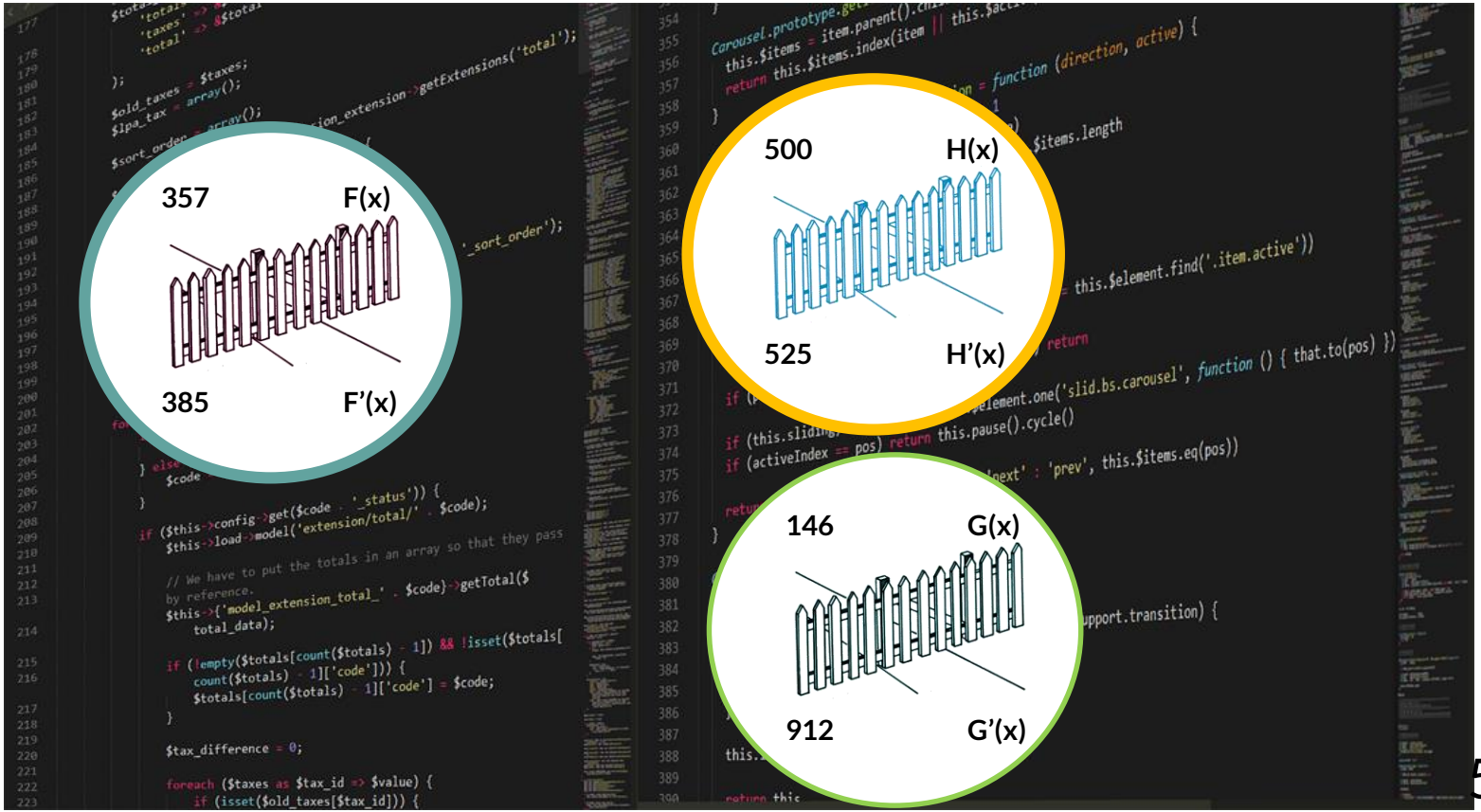
Out of Order Execution



Programmer's Expectation

Runtime Execution

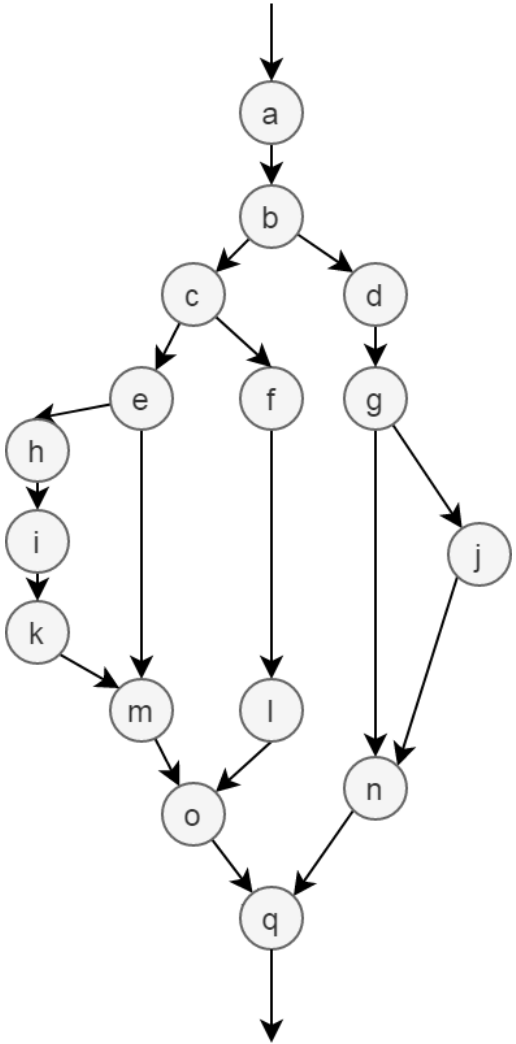
Fence Insertion



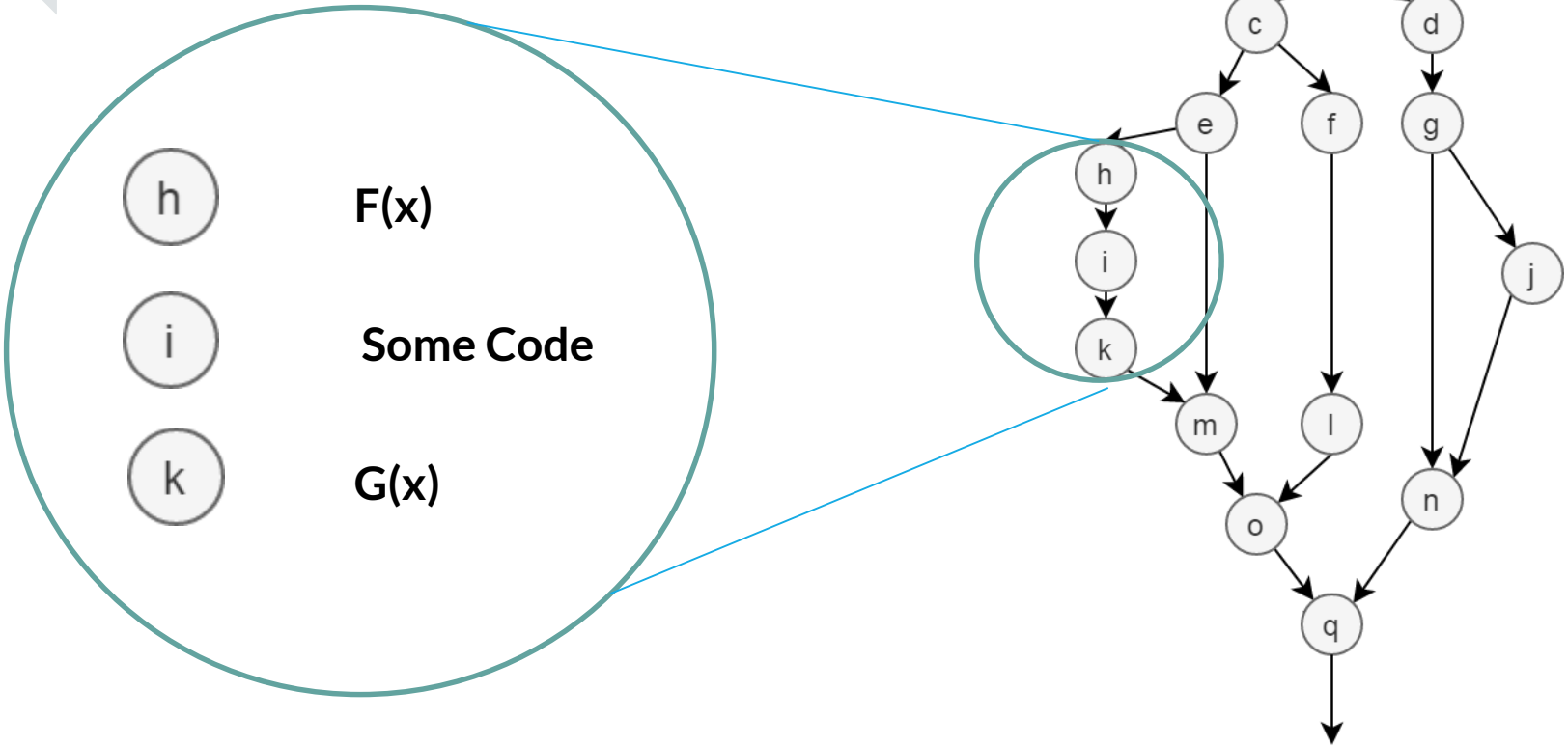
Our Solution

Outline

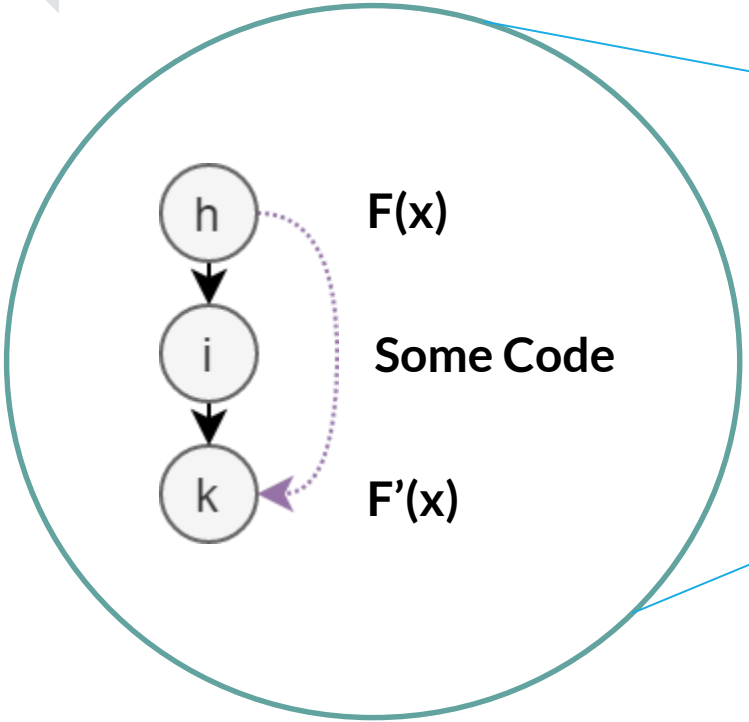
CFG

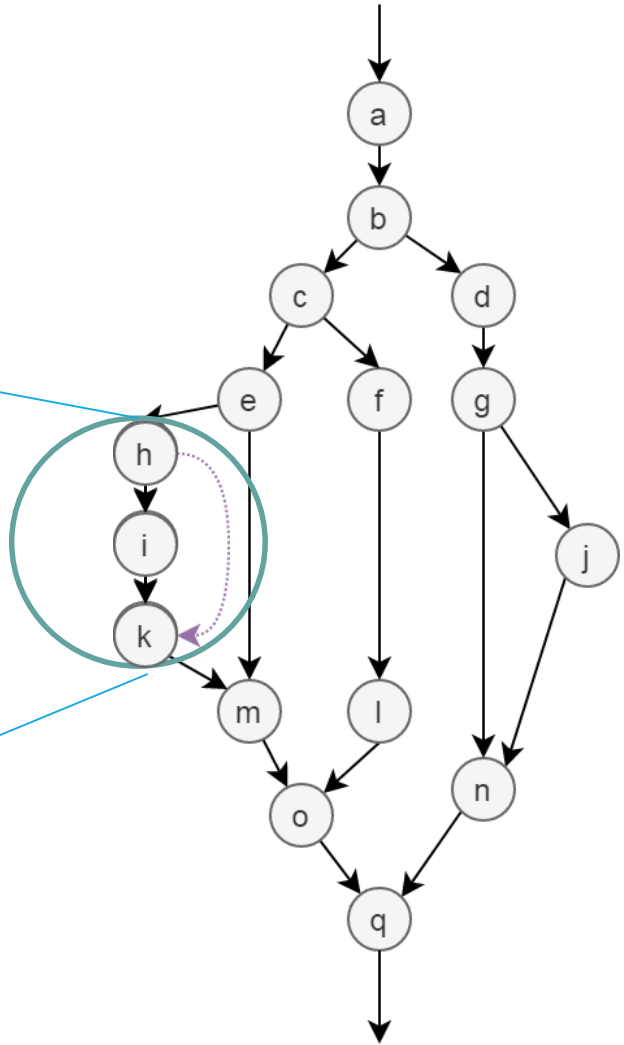


A Closer Look

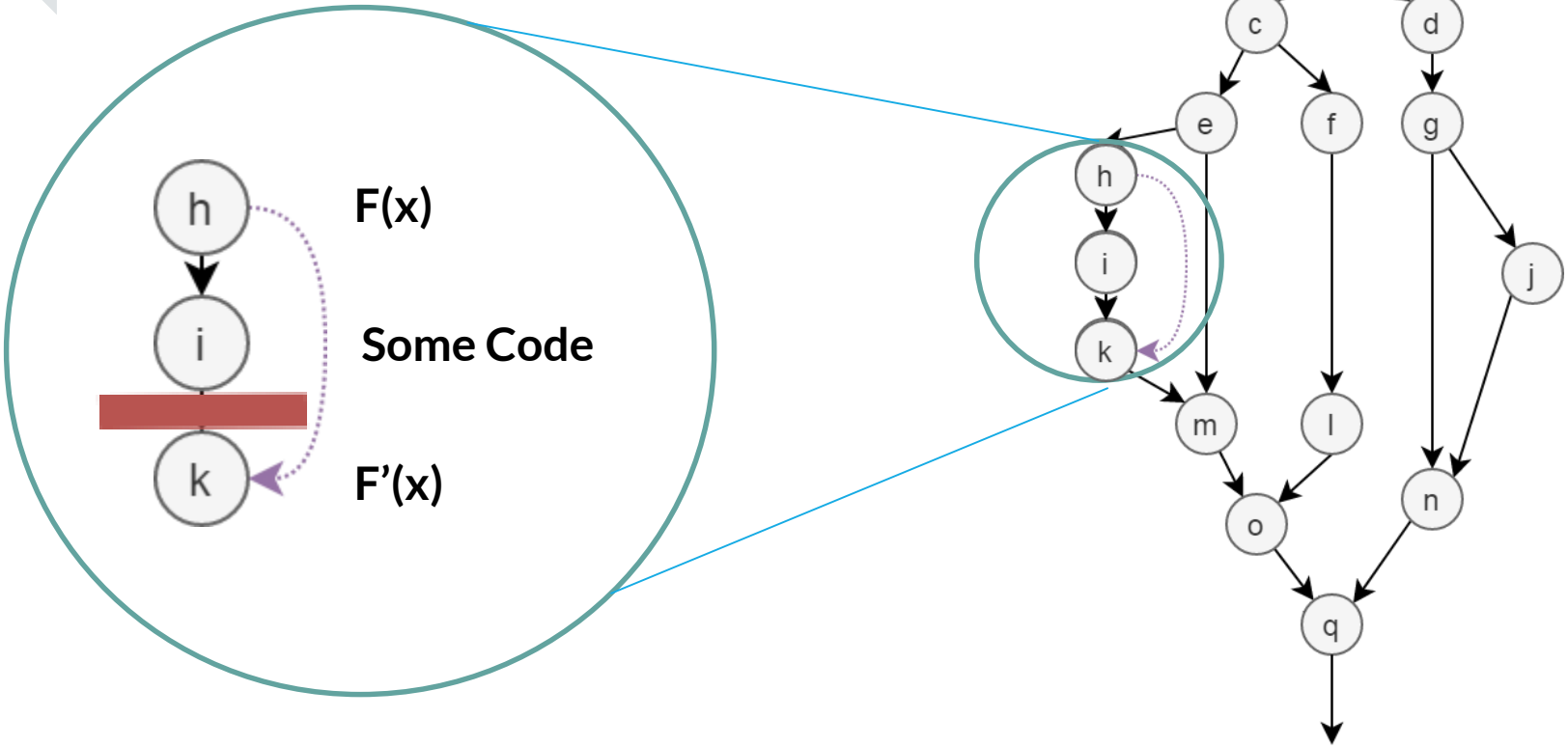


Maintaining Correctness

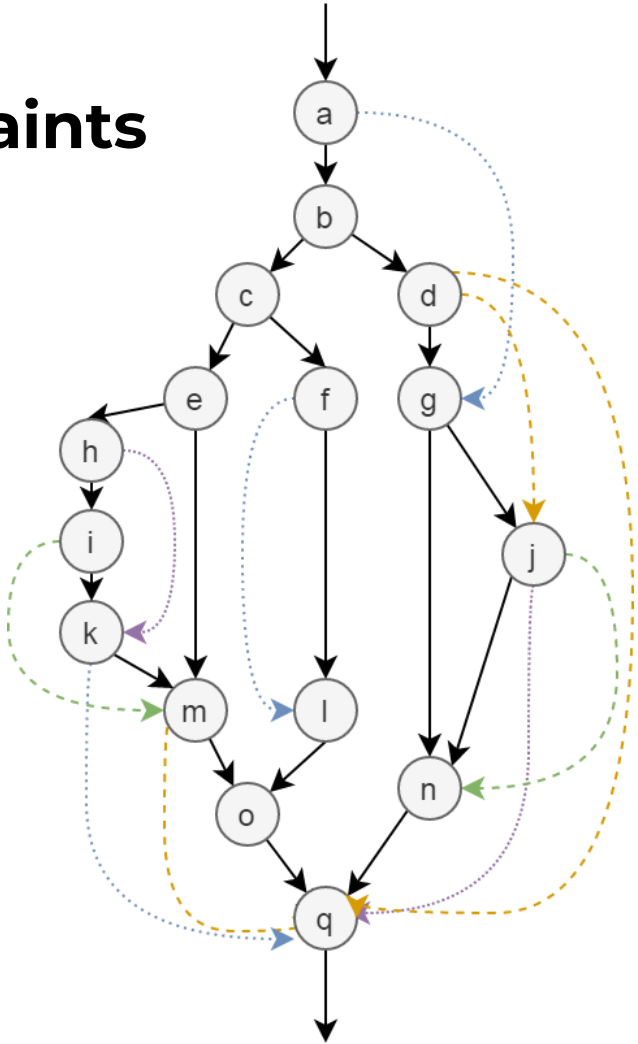




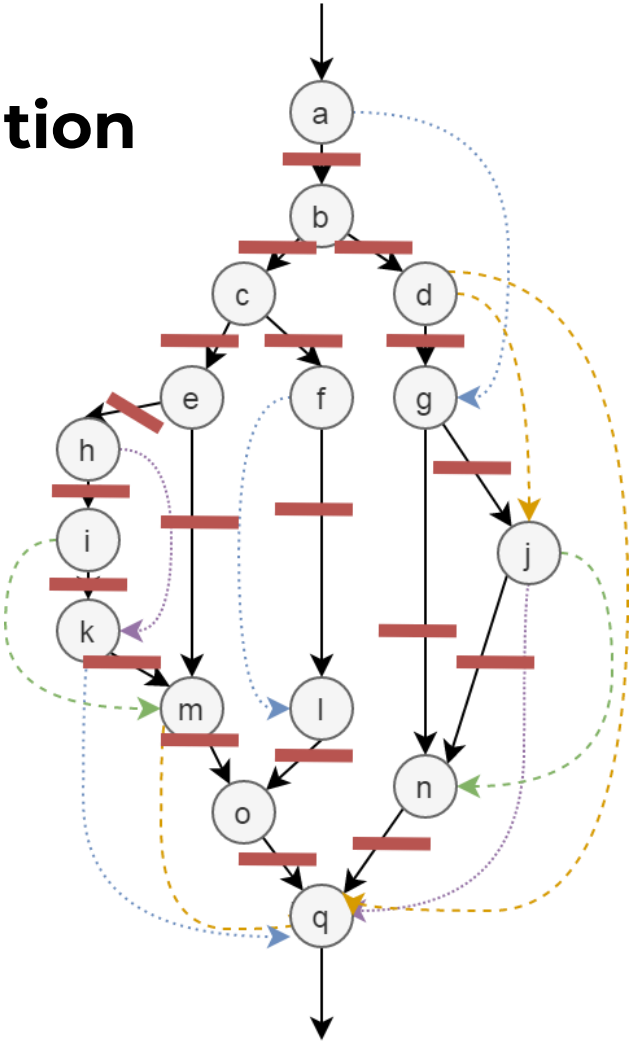
Fence Insertion



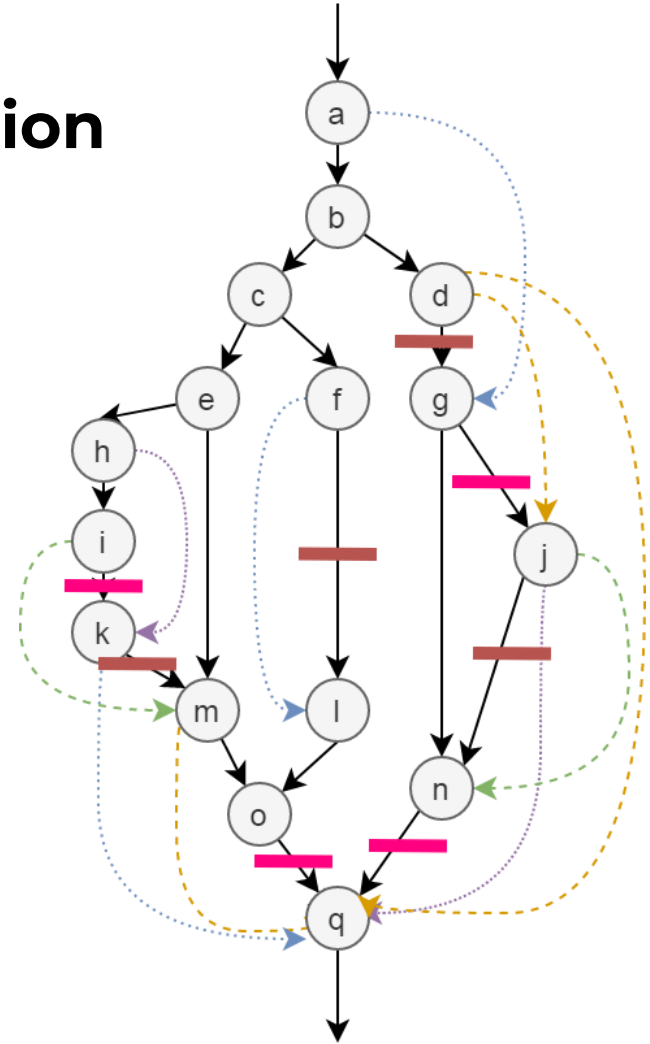
All Constraints



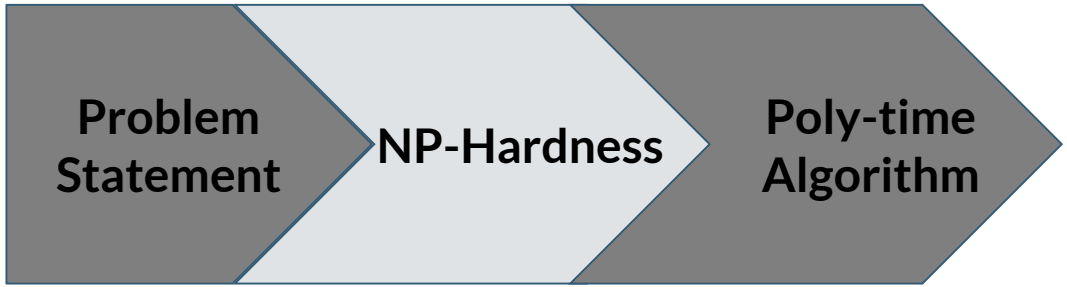
Naive Solution



Optimization



Outline



Reduction from Minimum Set Cover

$U = \{u_1, u_2, u_3, u_4, \dots, \mathrm{u_n}\}$

$S = \{S_1, S_2, \dots, S_m\}$

$Example\colon\ S1 = \{u_1, u_2, u_3\}, S2 = \{u_2, u_4\}$

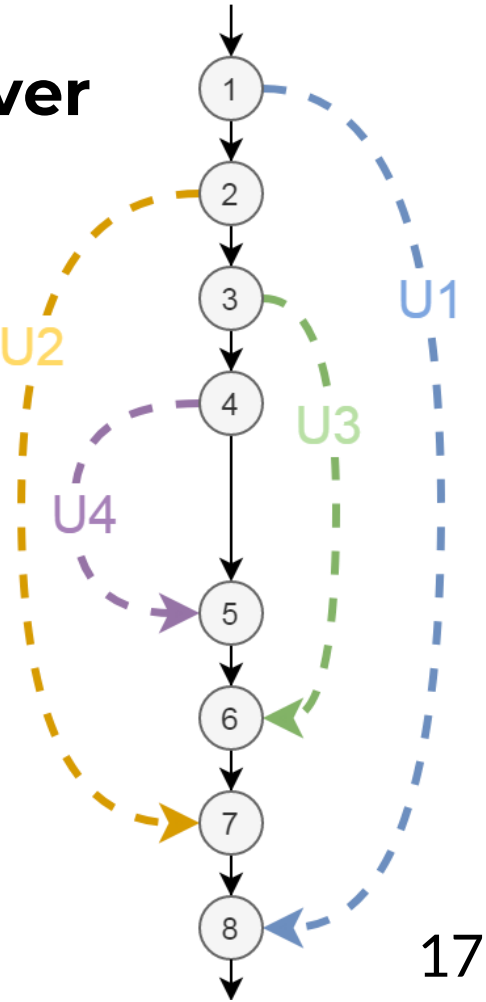
Reduction from Minimum Set Cover

$U = \{u_1, u_2, u_3, u_4, \ldots, u_n\}$: Constraint Types,

Straight line program with 2n instructions,

Constraints with type u_i on (v_i, v_{2n-i+1}),

$(v_{\frac{n}{2}}, v_{\frac{n}{2}} + 1)$:The middle edge



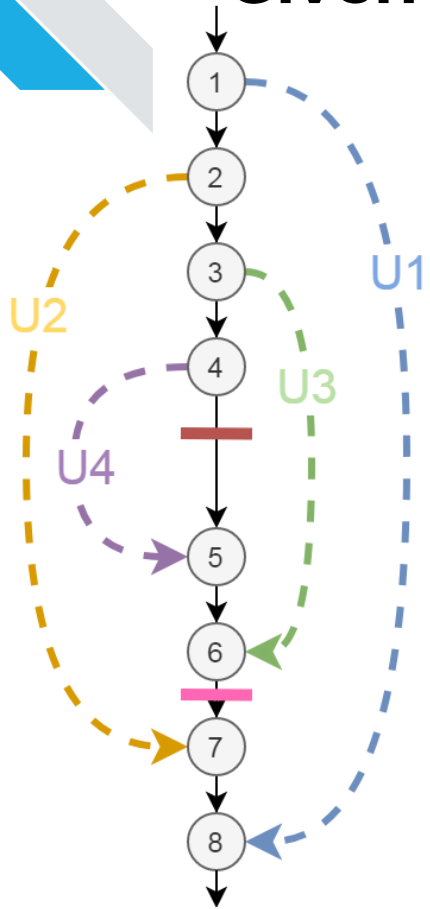
Reduction from Minimum Set Cover

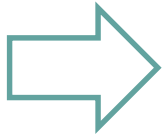
$S = \{S_1, S_2, \dots\}$: Fence Types

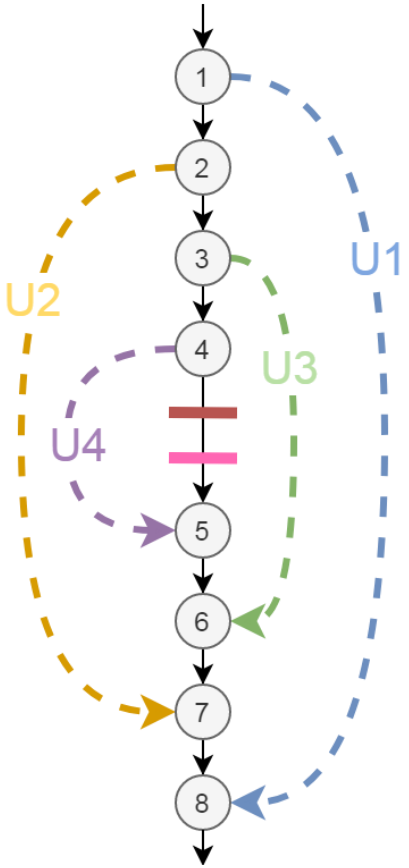
We could many fences of S_i

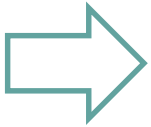
$Example$: $S1 = \{u_1, u_2, u_3\}, S2 = \{u_2, u_4\}$

Given Solution of Fence Insertion









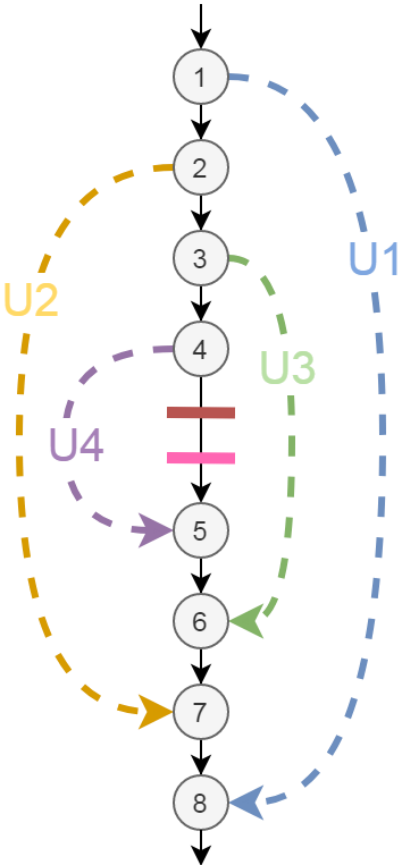
$$U = \{u_1, u_2, u_3, u_4, \dots\}$$

$$S = \{S_1, S_2, \dots\}$$

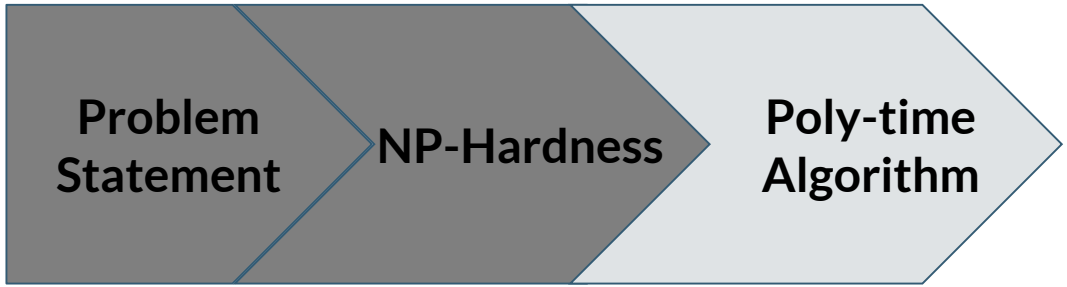
Given Solution of Minimum Set Cover

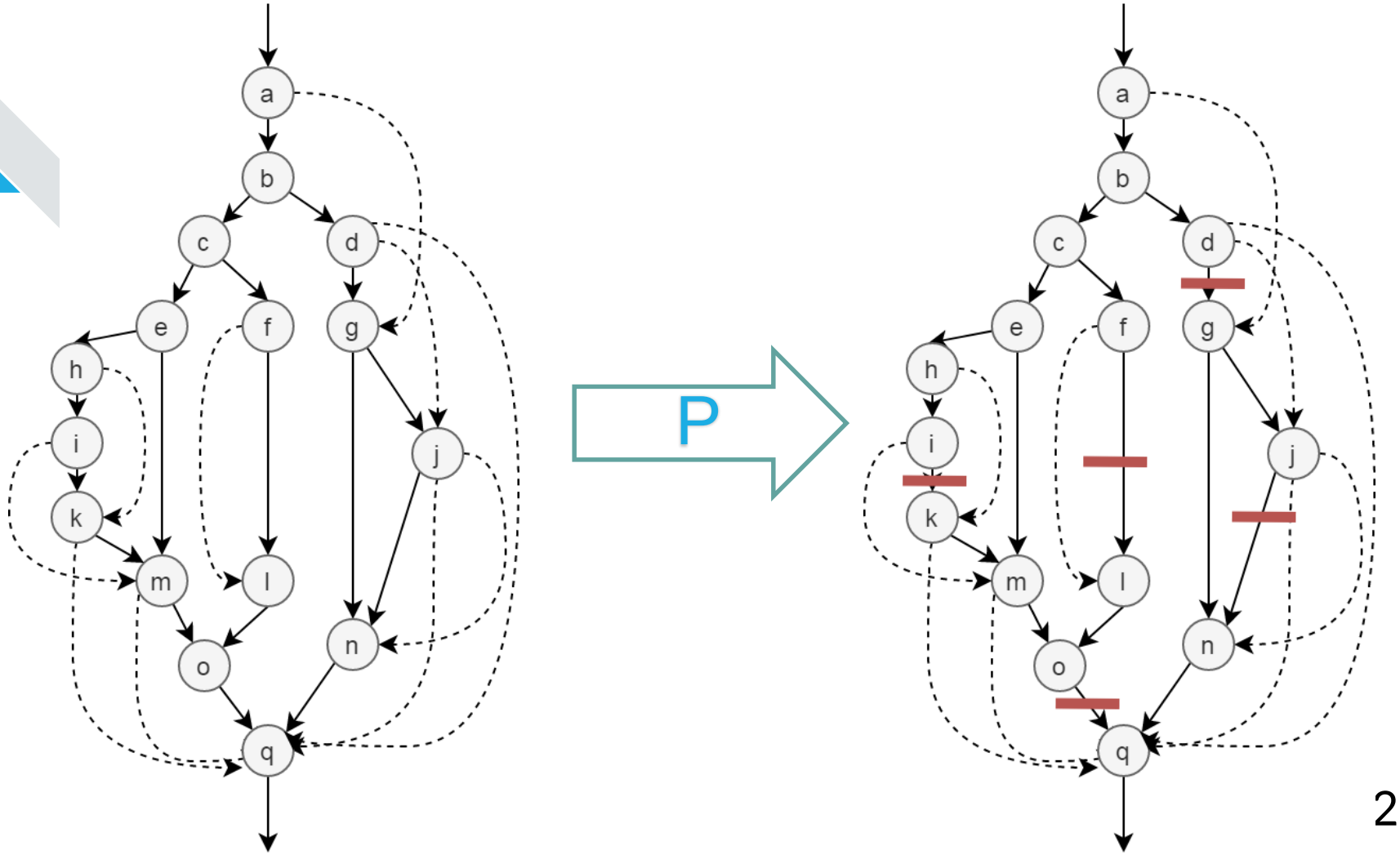
$U = \{U_1, U_2, U_3, U_4, \dots\}$

$S = \{S_1, S_2, \dots\}$



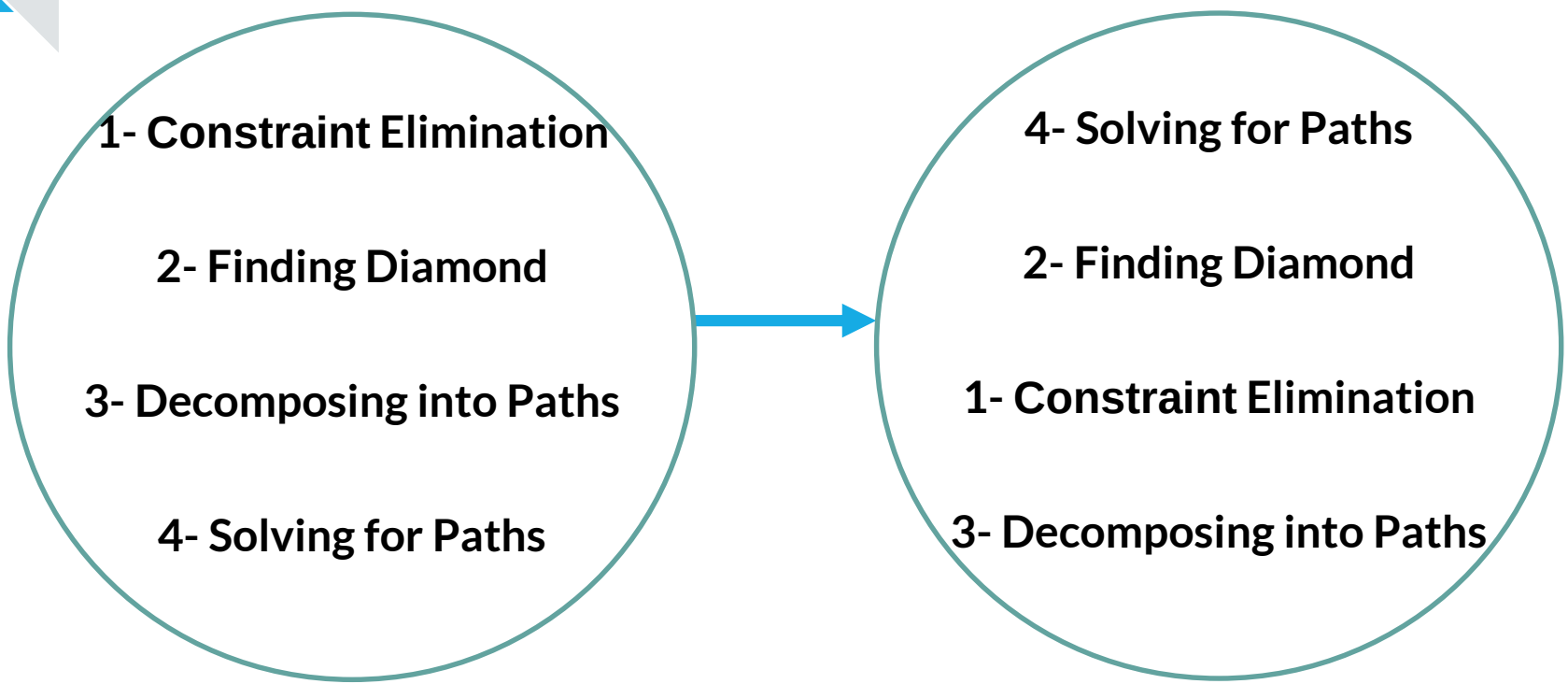
Outline



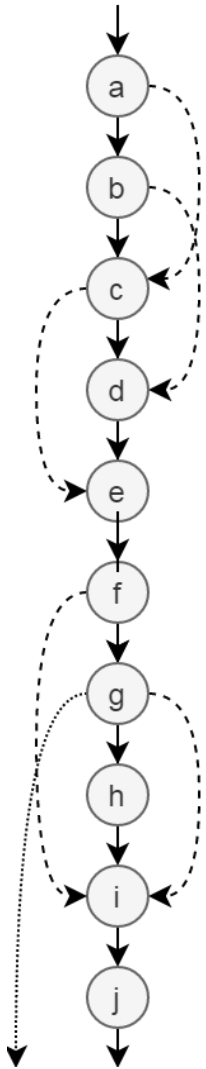

a
b
c
d
e
f
g
h
i
j
k
m
l
n
o
q
P
a
b
c
d
e
f
g
h
i
j
k
m
l
n
o
q

Algorithm Overview

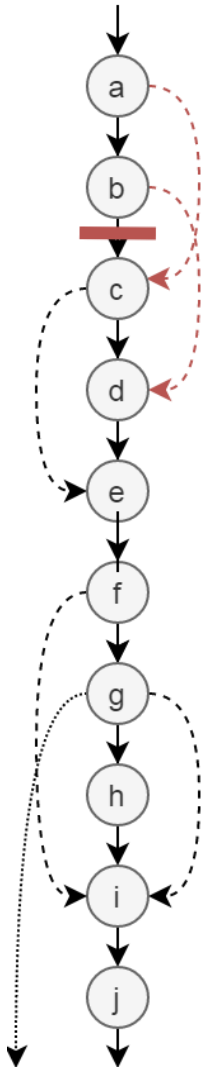
Out of Order Execution!



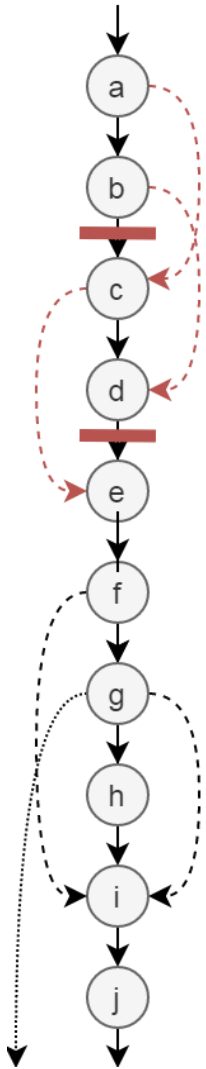
A Simple Path



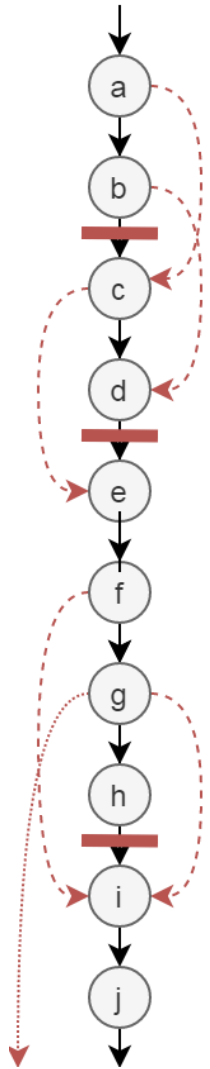
A Simple Path



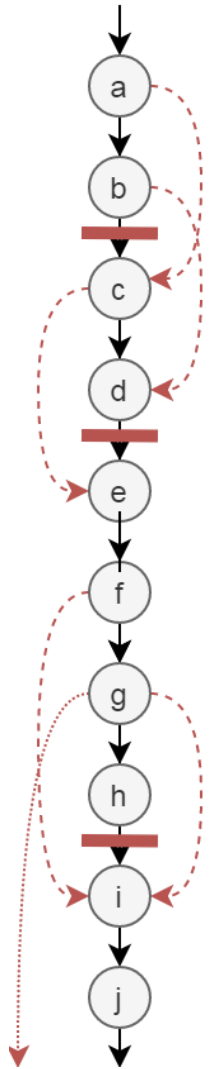
A Simple Path



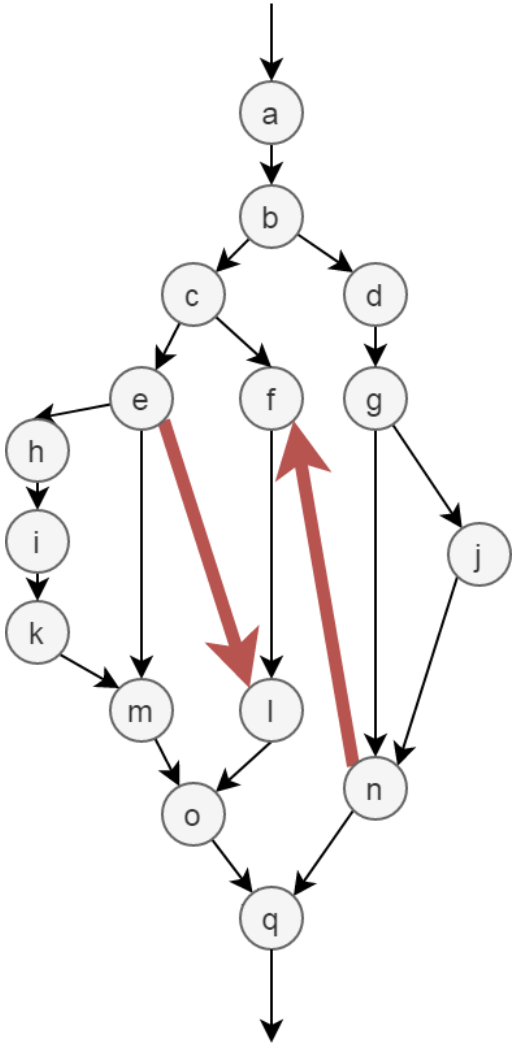
A Simple Path



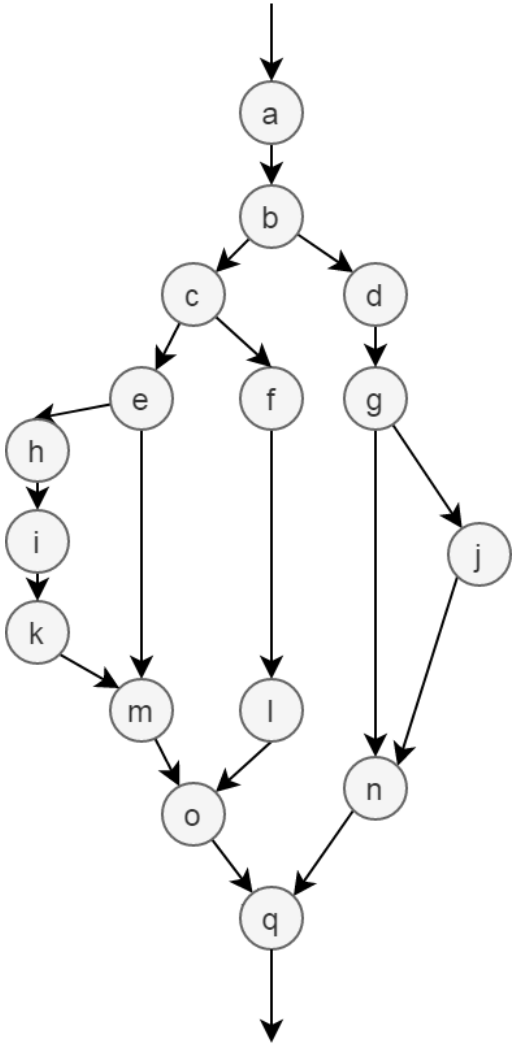
A Simple Path



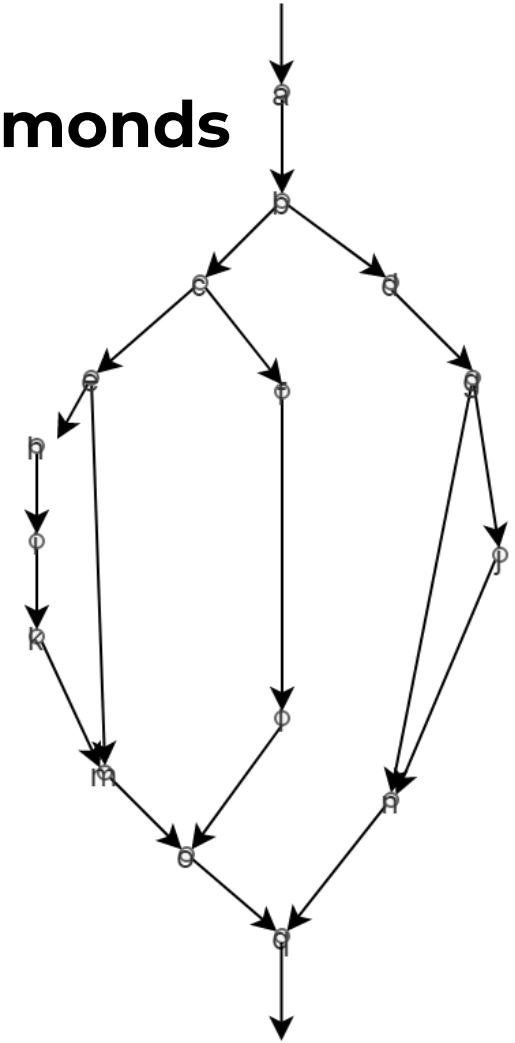
Structured



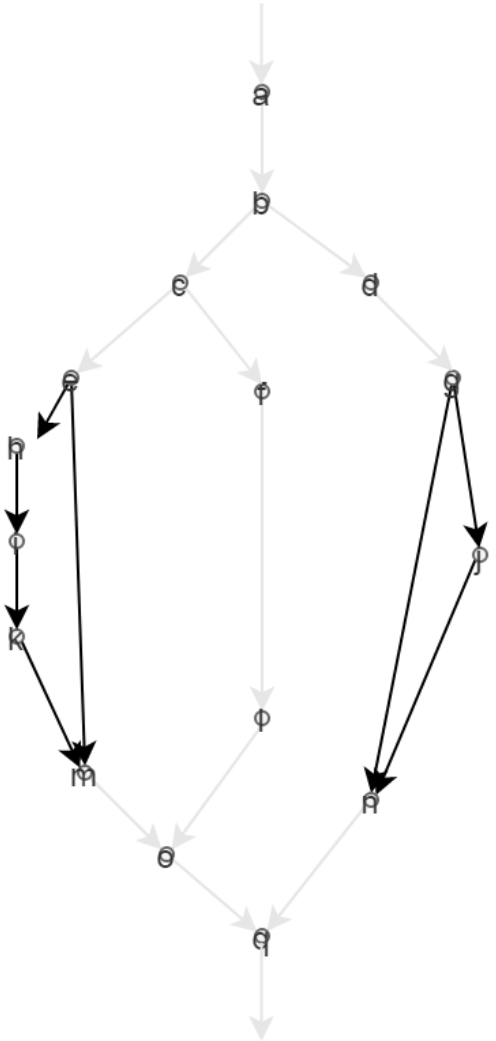
A(C)FG



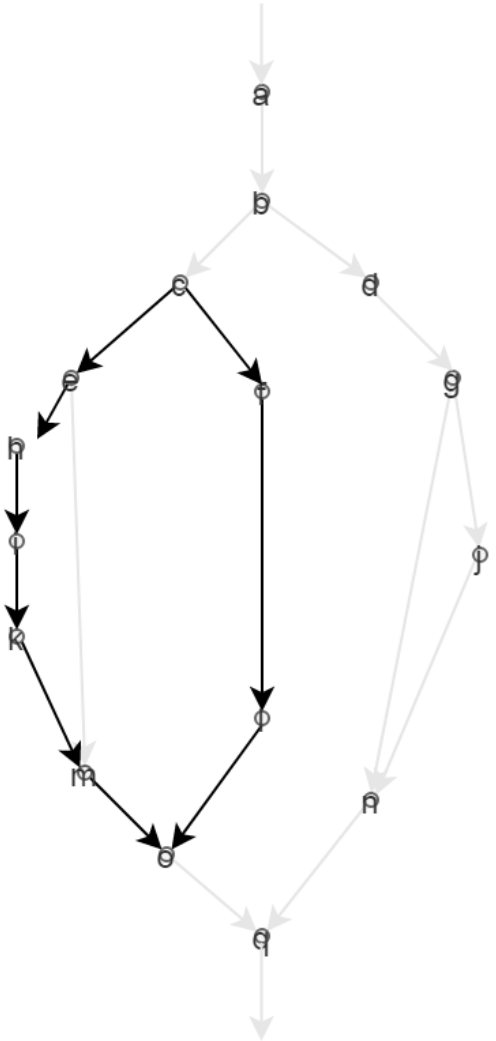
Finding Diamonds



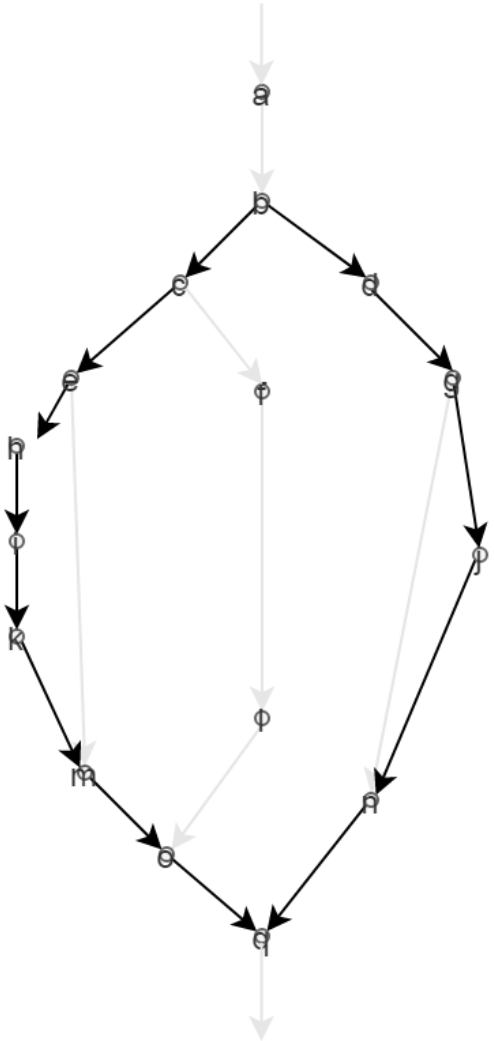
Level 0



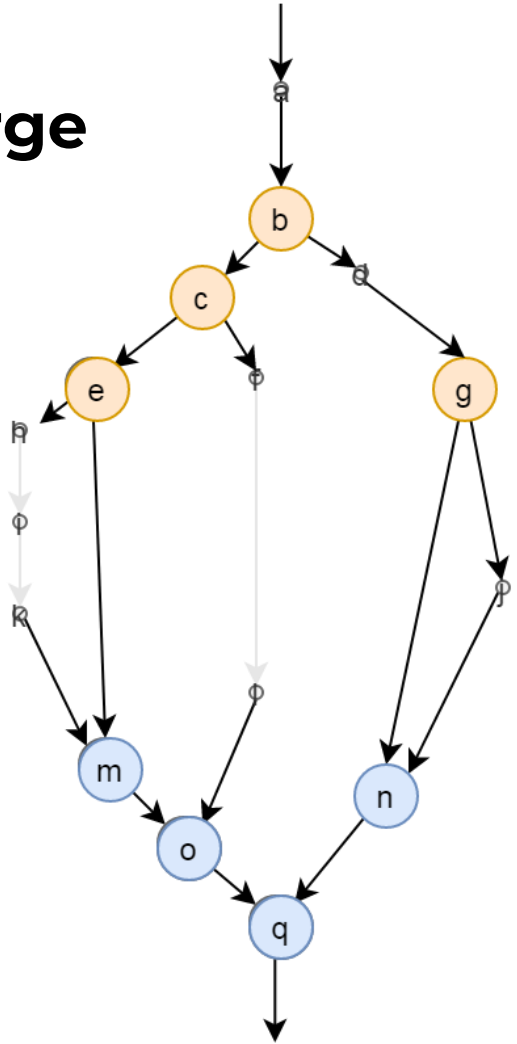
Level 1



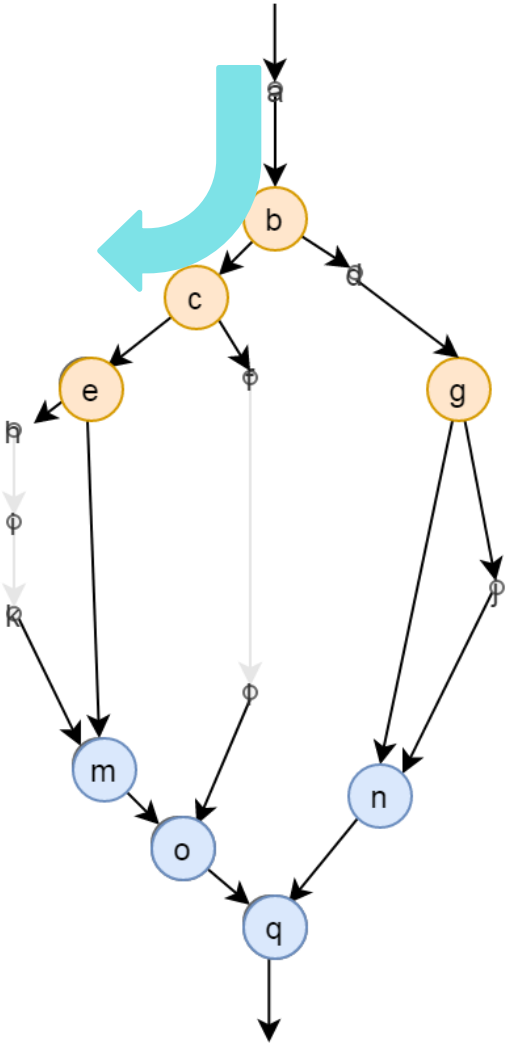
Level 2



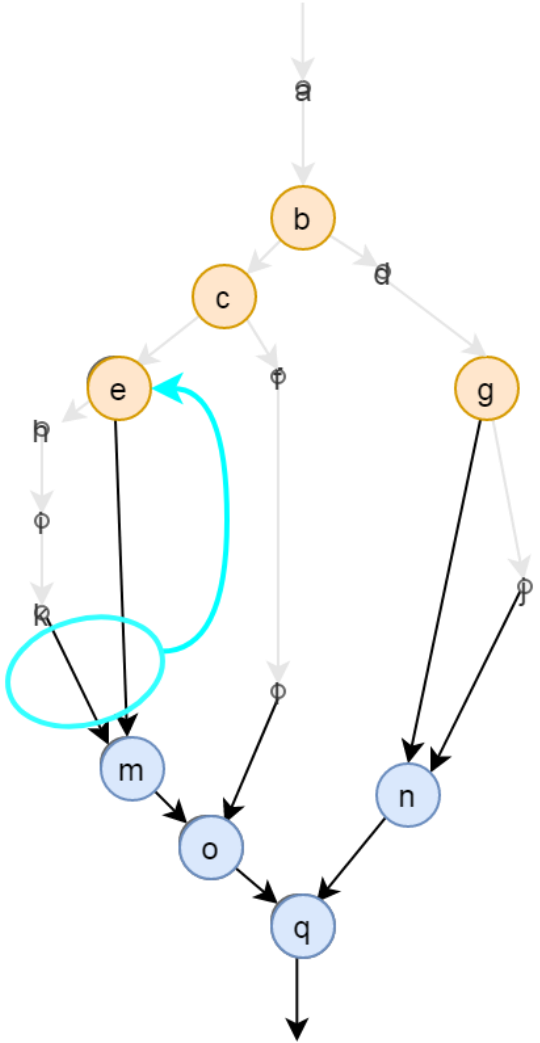
Branch/Merge



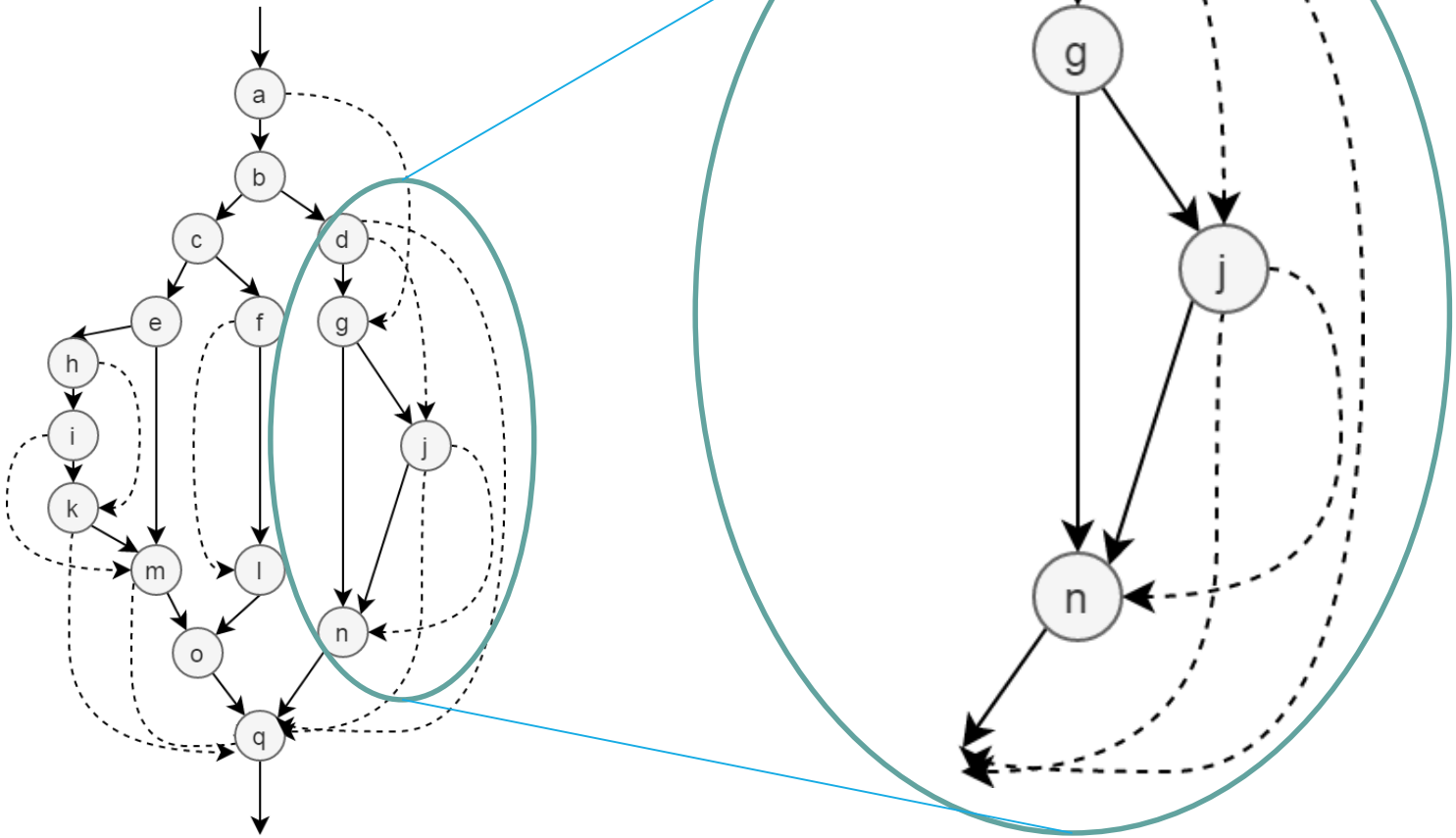
Algorithm



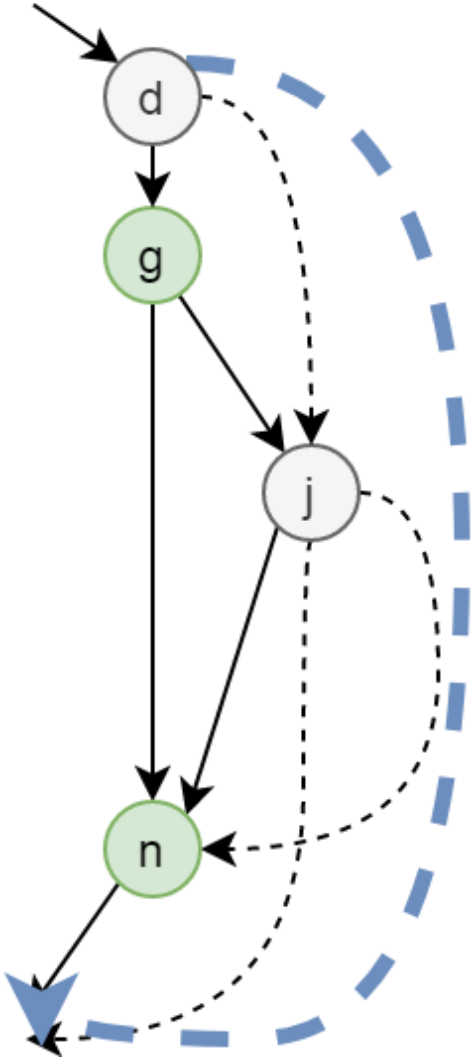
Algorithm



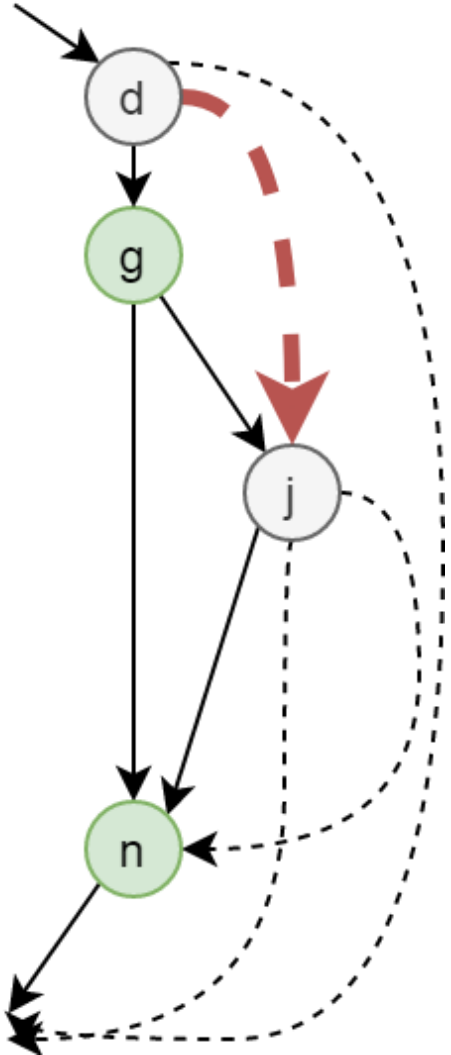
Fence Elimination



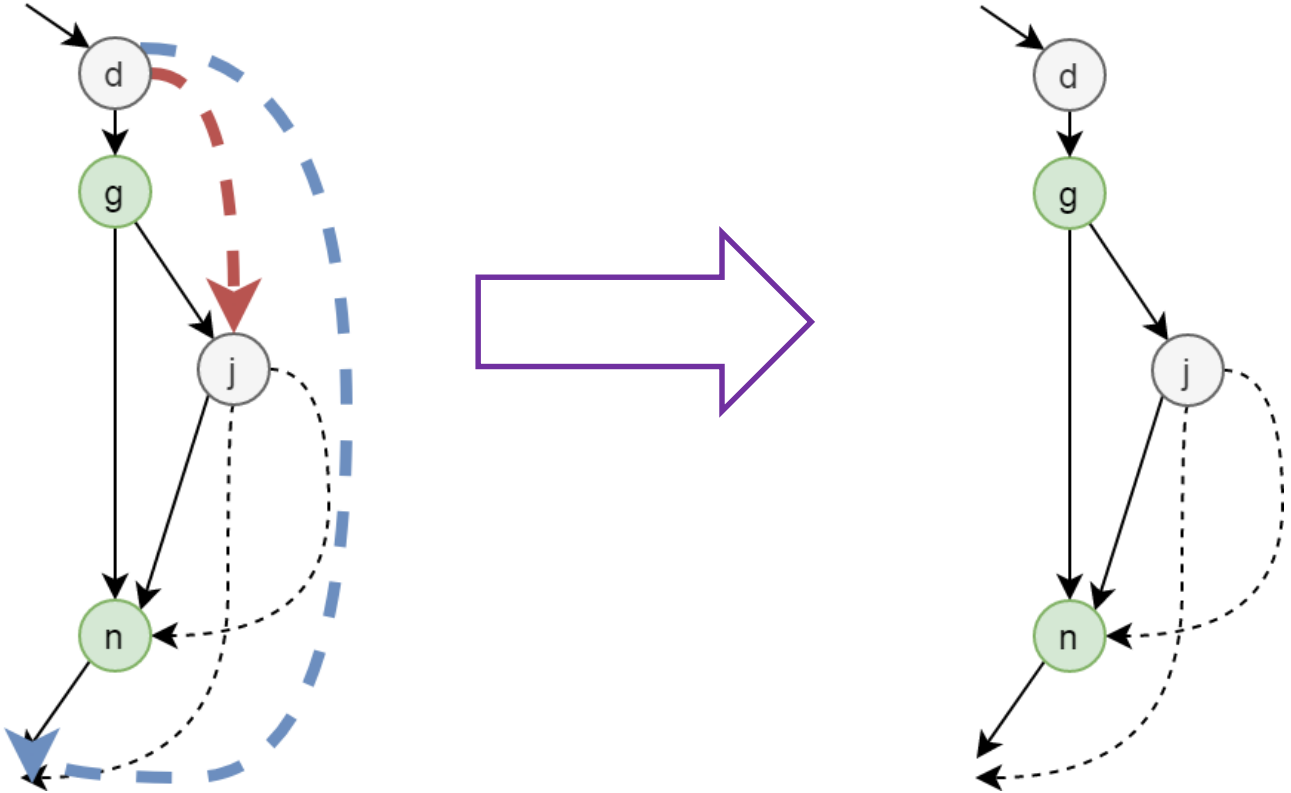
Passing Constraints



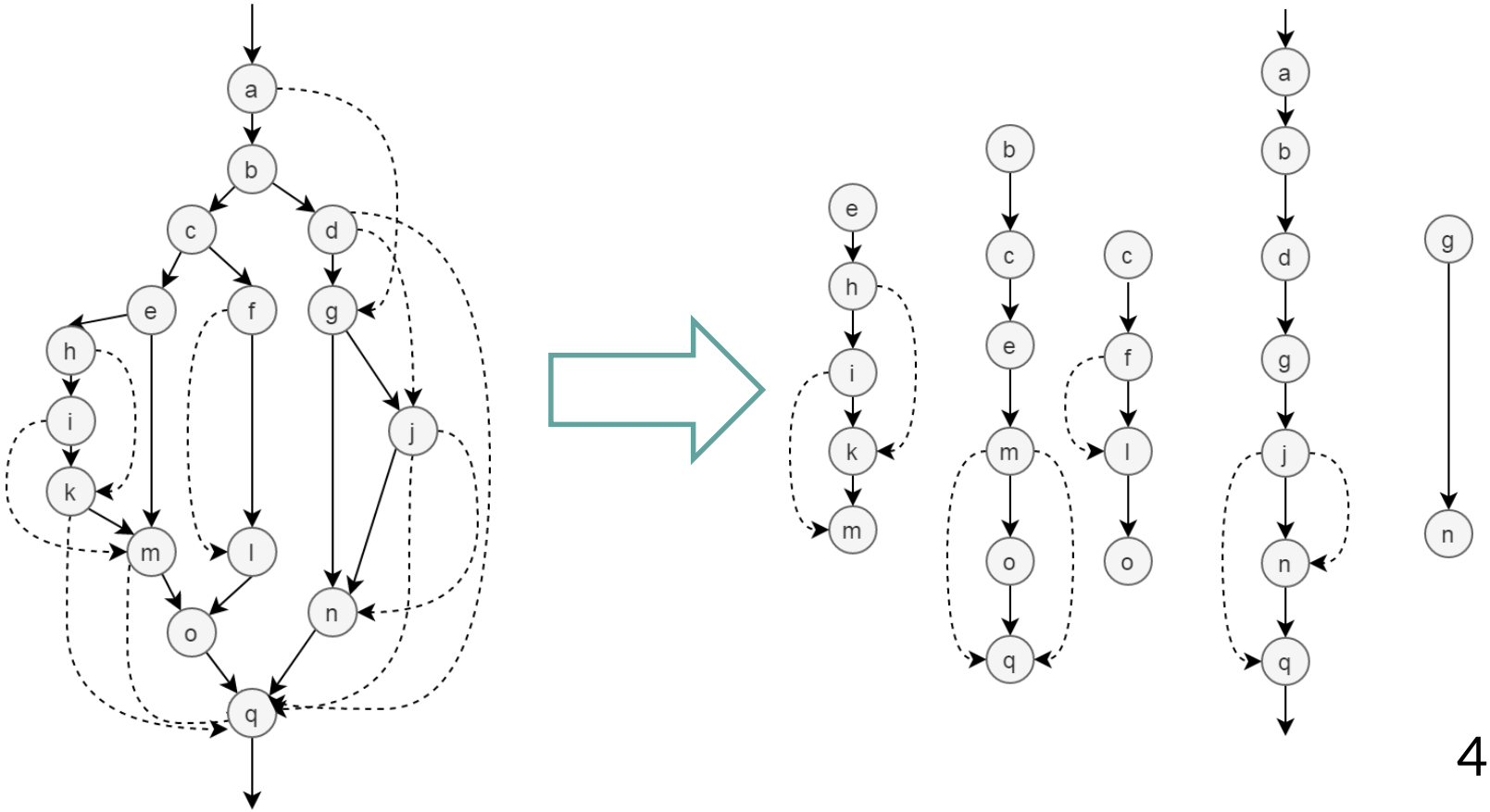
Spanning Constraints



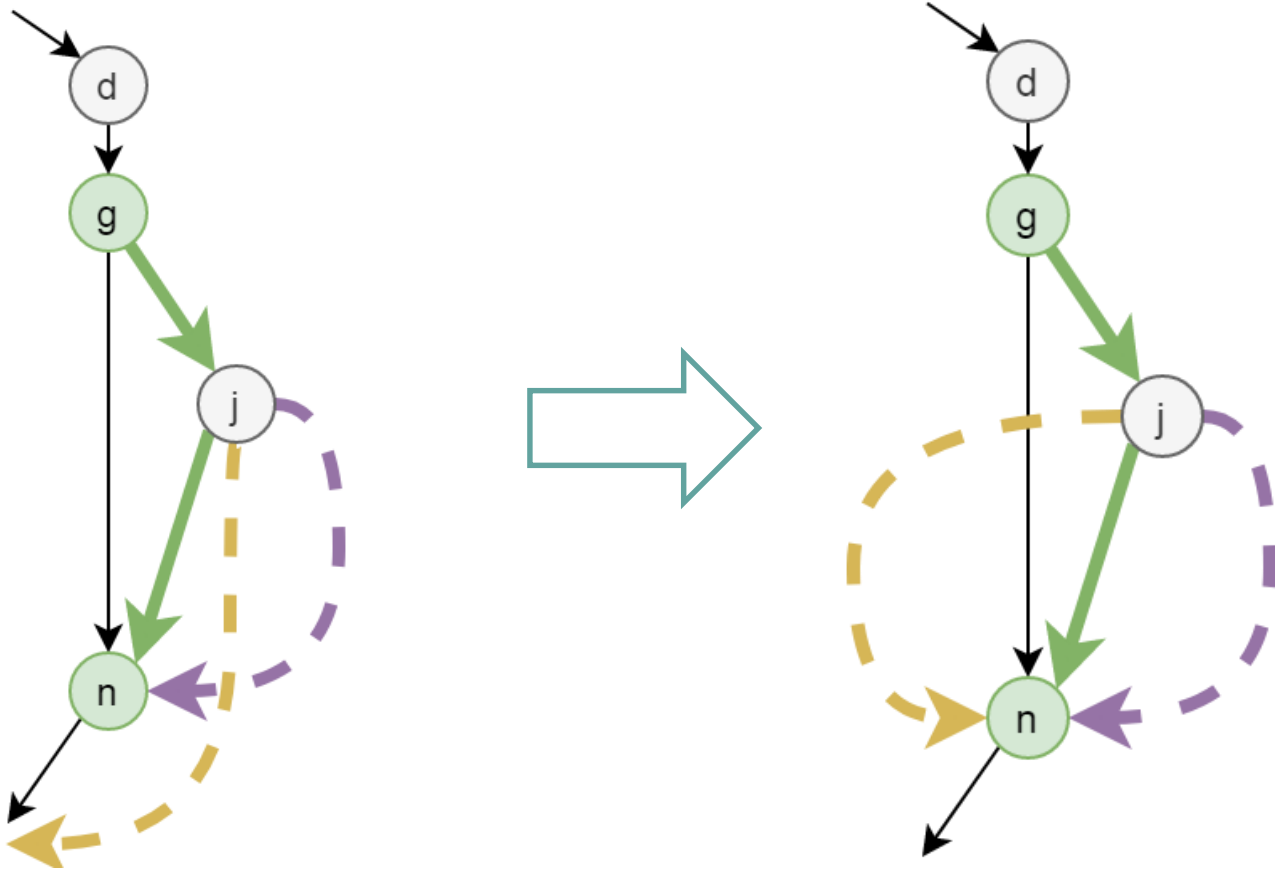
Control Dependency Preservation



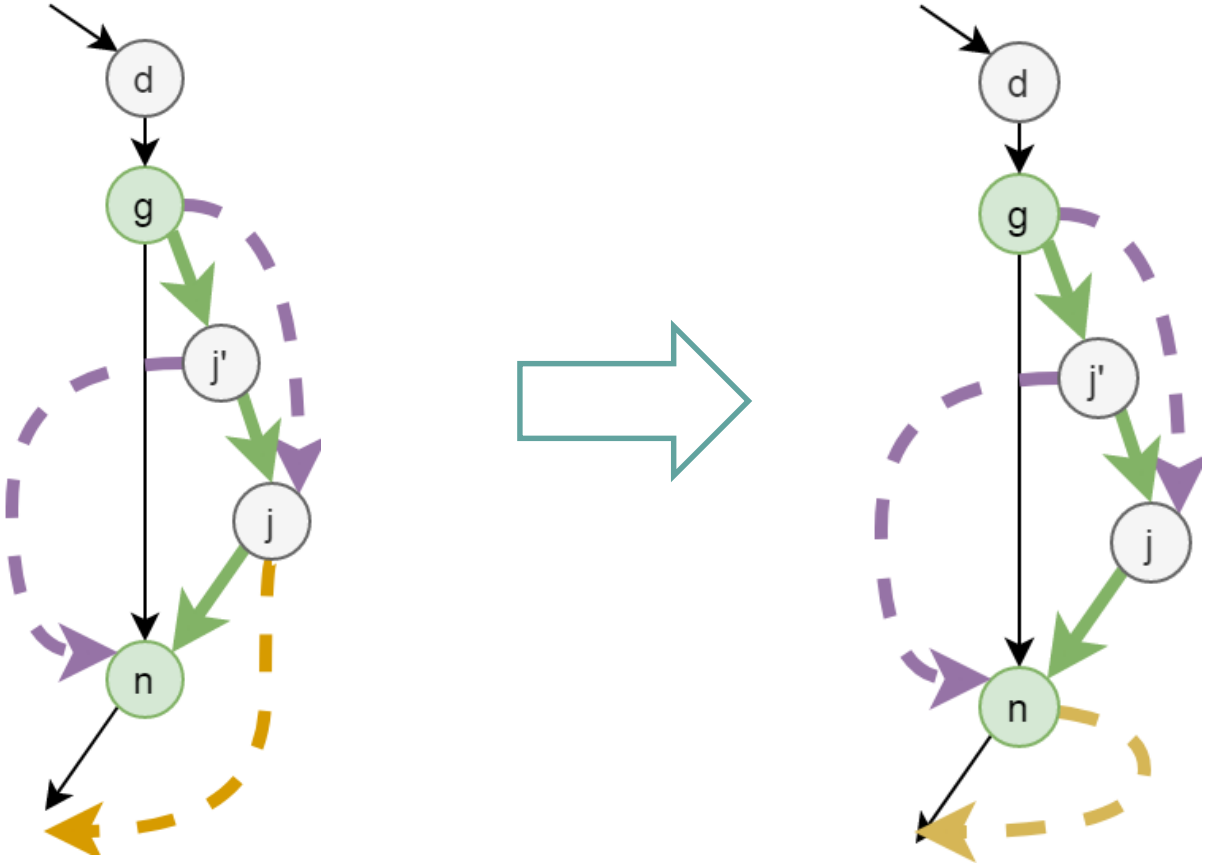
Decomposing into simple paths



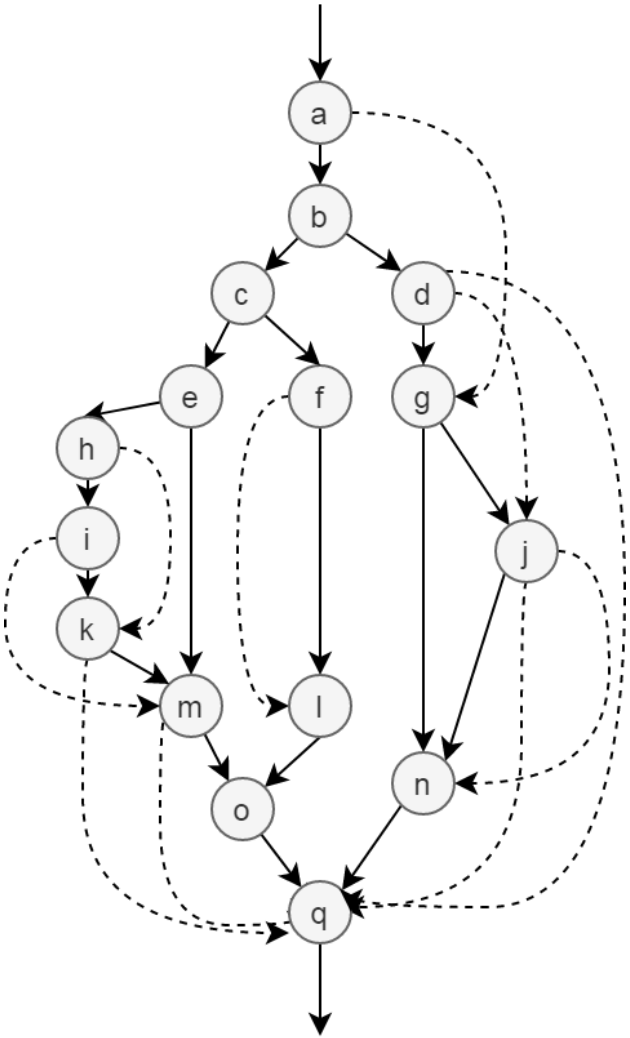
Absorption in a Path



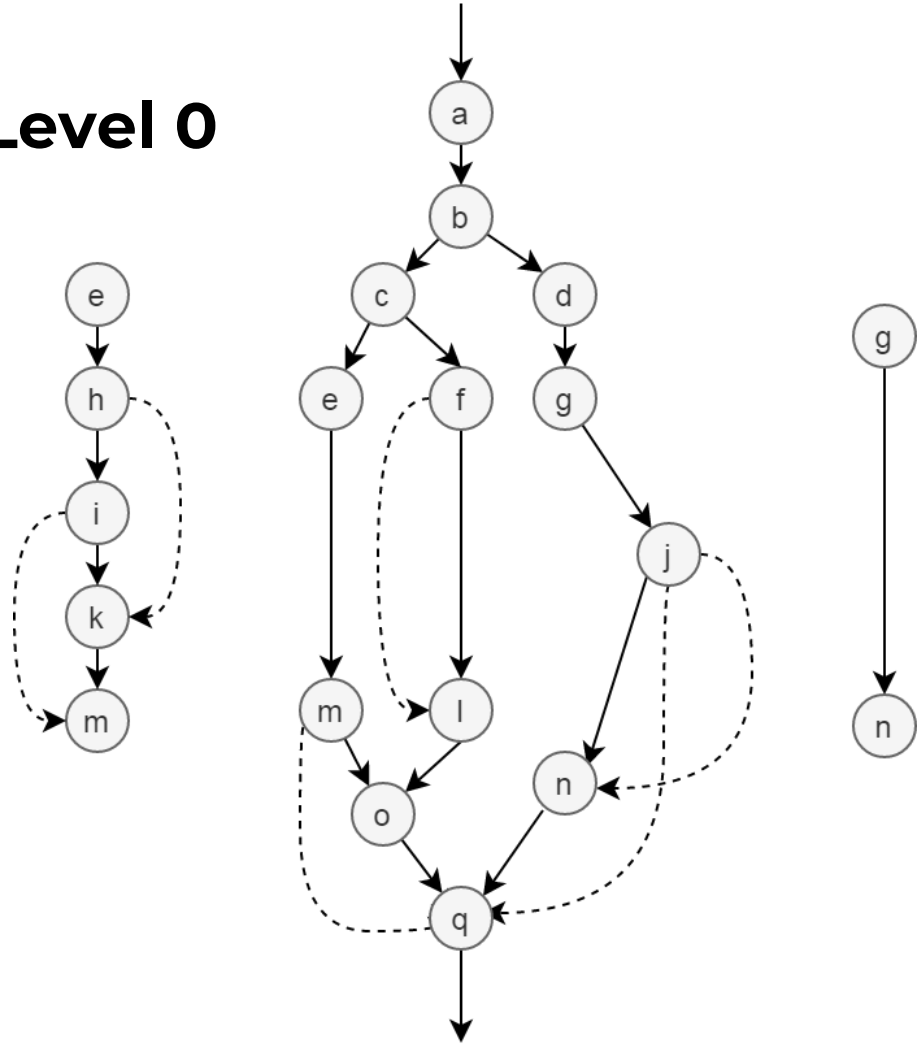
Emission in a Path



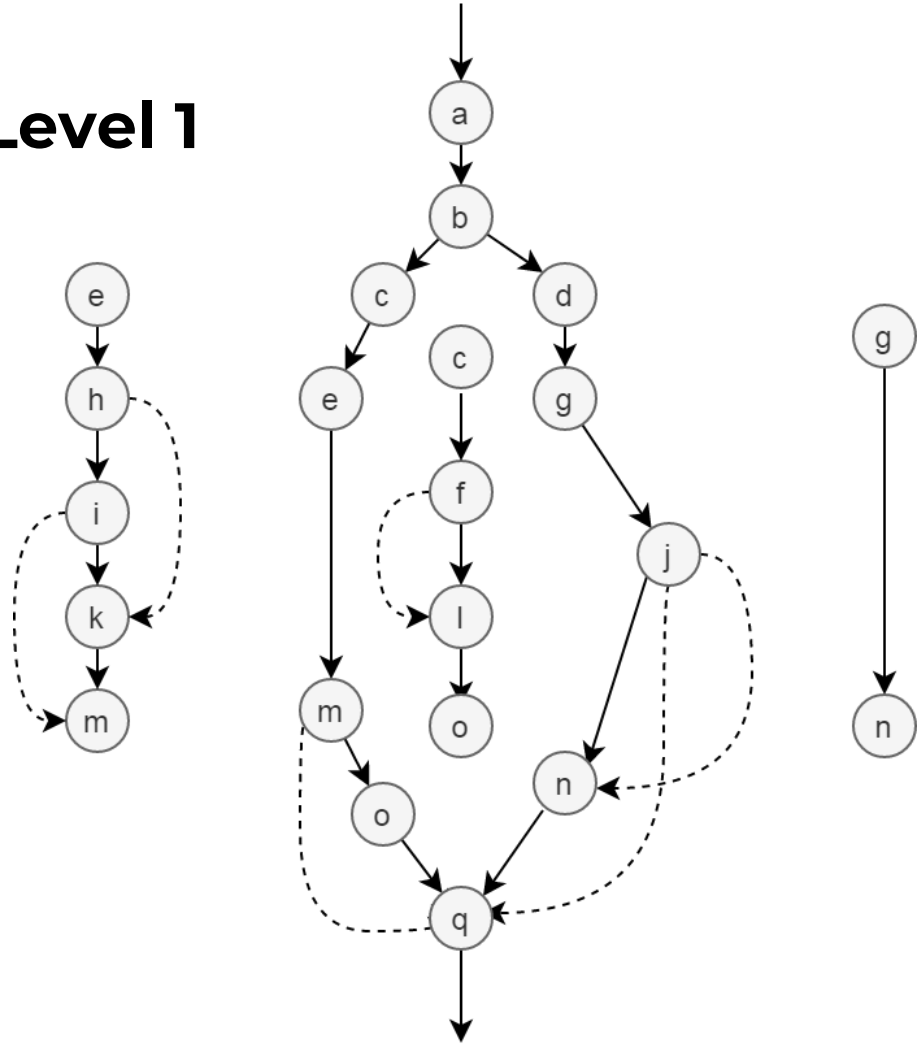
Before



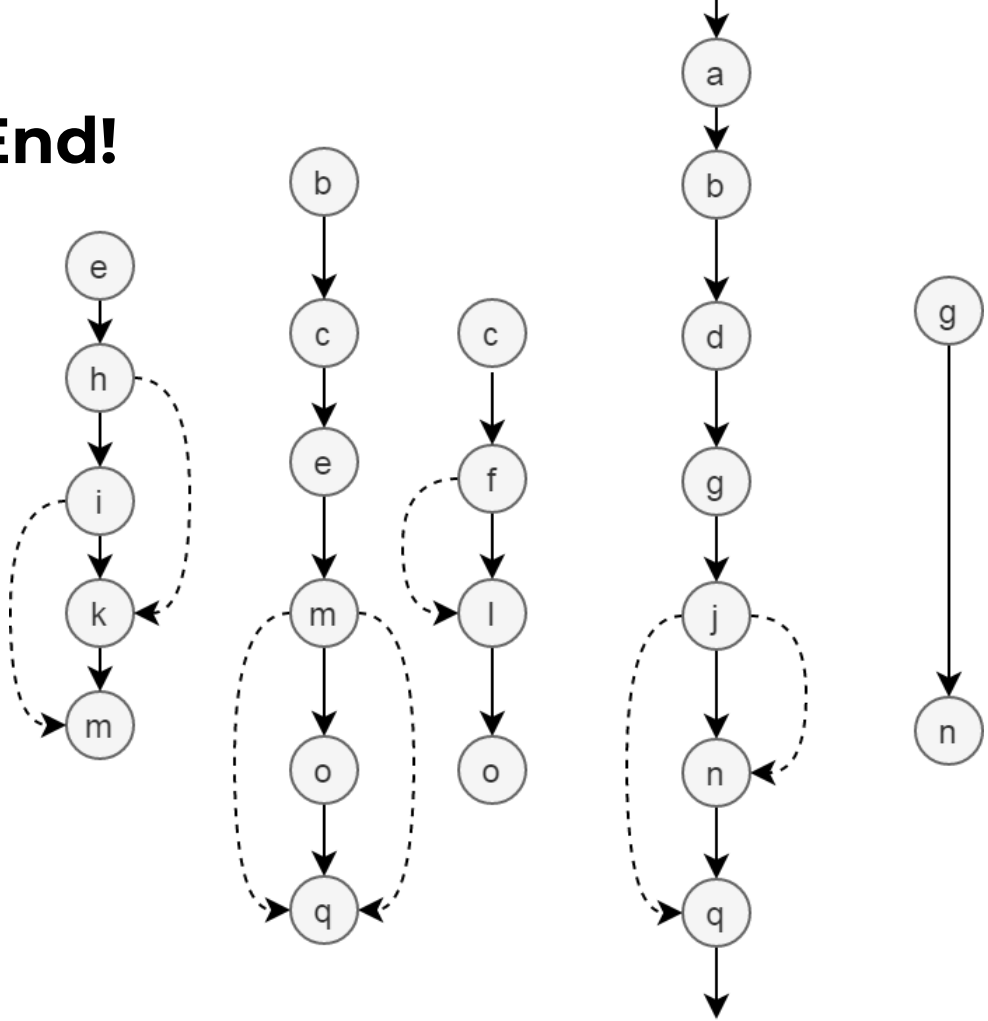
After Level 0



After Level 1



The End!



Complexity

1-$|E| \in O(V)$

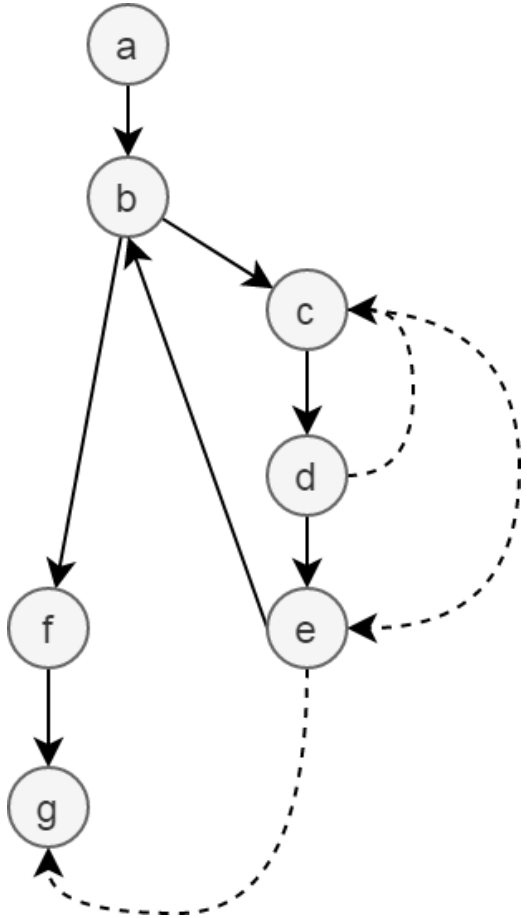
2-Elimination: $O(|C| + |V|)$

3-Finding Diamonds: $O(VlogV)$

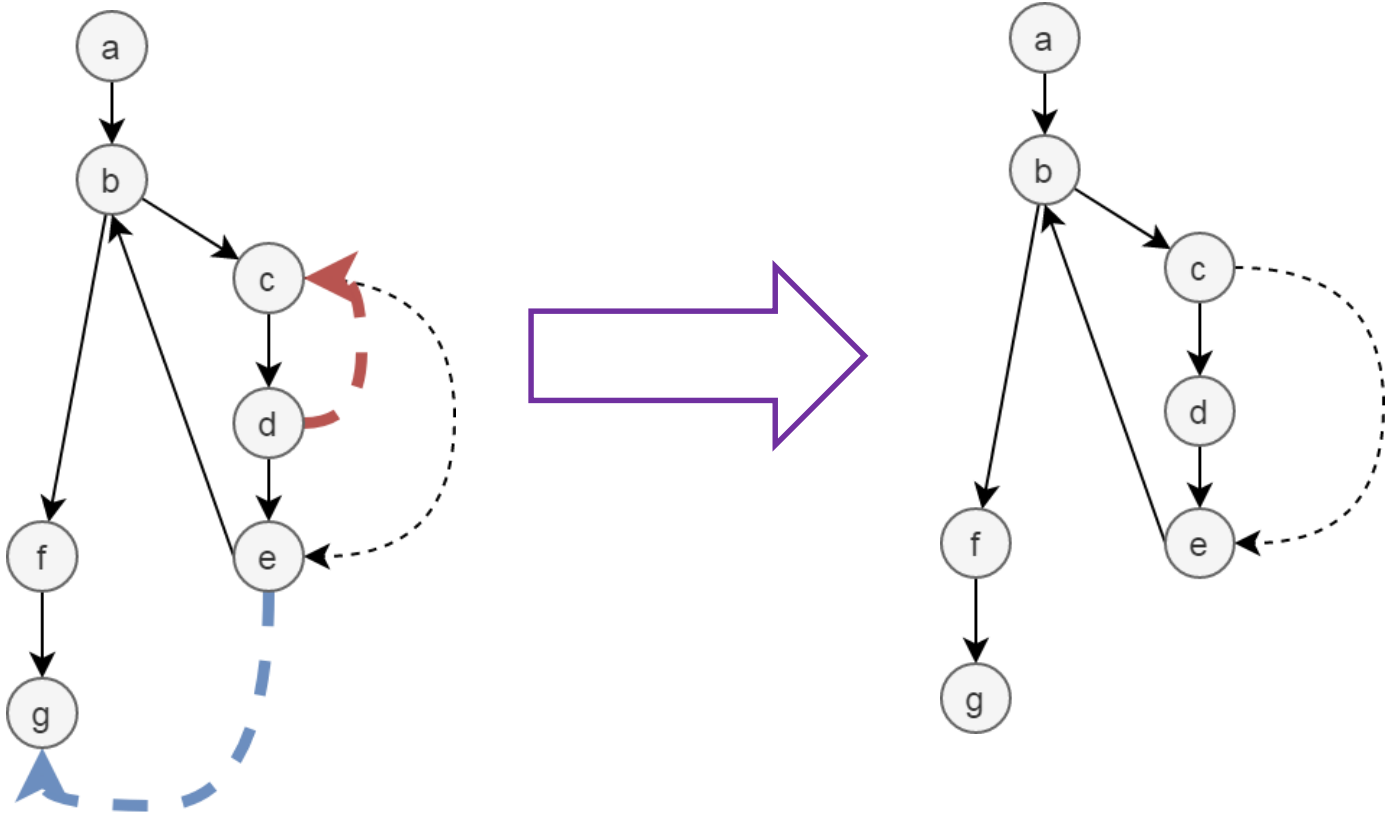
4-Decompsiton and Insertion: $\sum_{|p|} |C_{p_i}| log C_{p_i} + |V_{p_i}|$

$\Rightarrow$Polynomial Time!

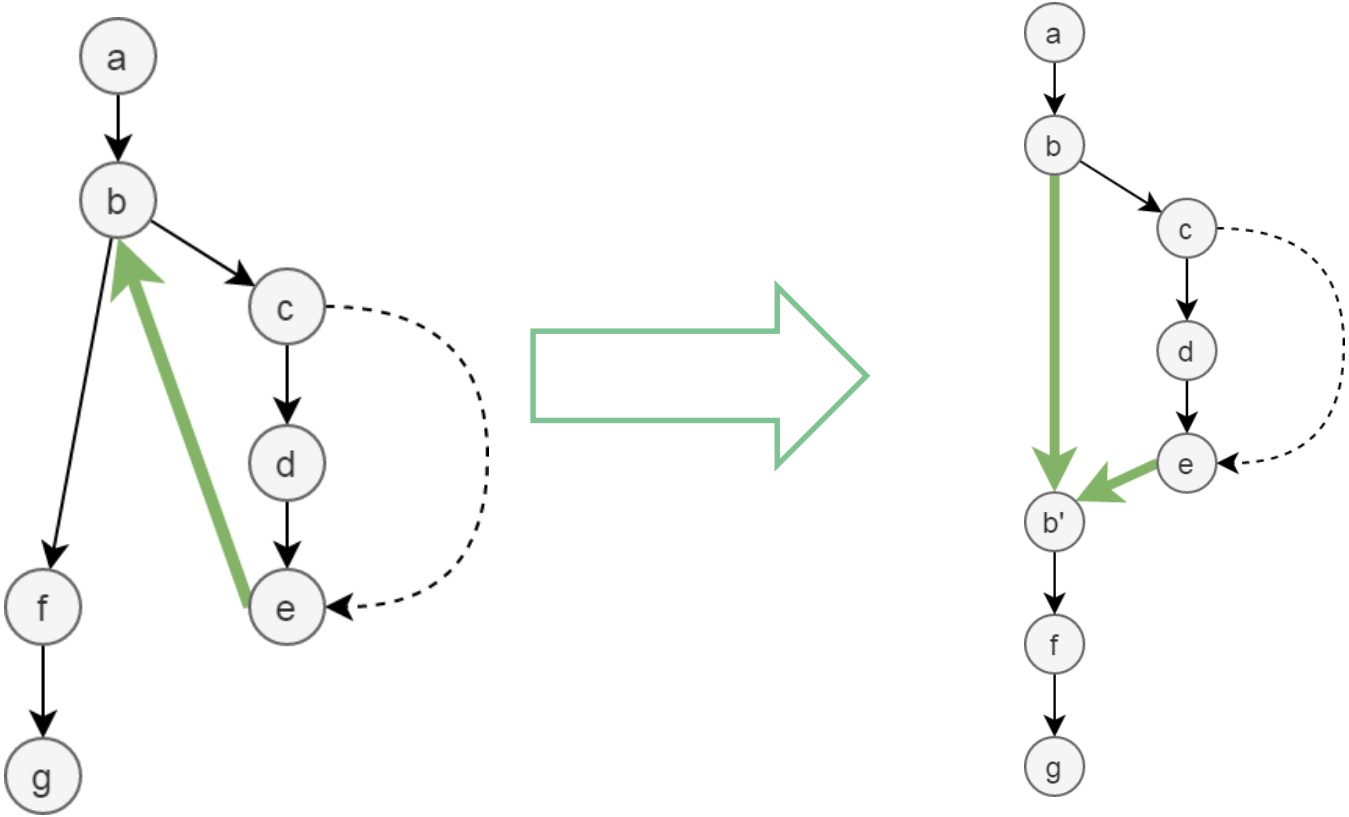
What about loops?



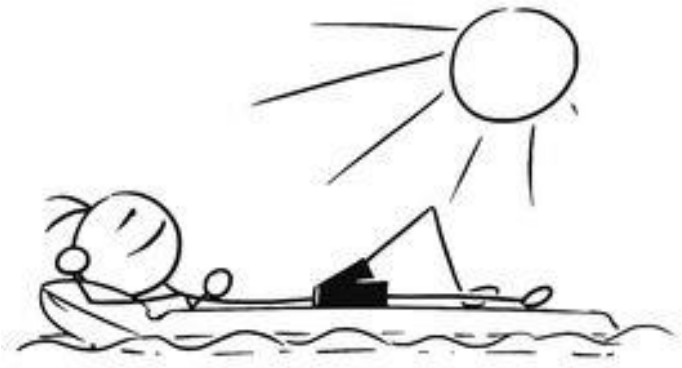
Elimination



Transformation



Thank you



Probabilistic Smart Contracts: Secure Randomness on the Blockchain [ICBC 2019]

Propose a novel game-theoretic approach for generating provably unmanipulatable pseudorandom numbers on the blockchain.

Hybrid Mining: Exploiting Blockchain's Computational Power for Distributed Problem Solving [SAC 2019]

A new mining protocol that combines solving real-world useful problems with Hashcash.