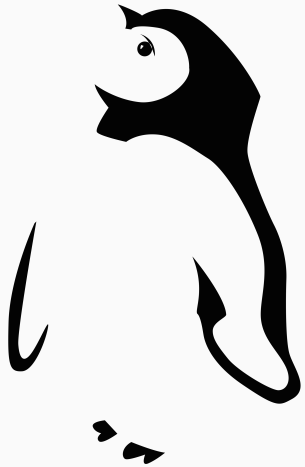
Waddle

Always-Canonical Intermediate Representation

Eric Fritz

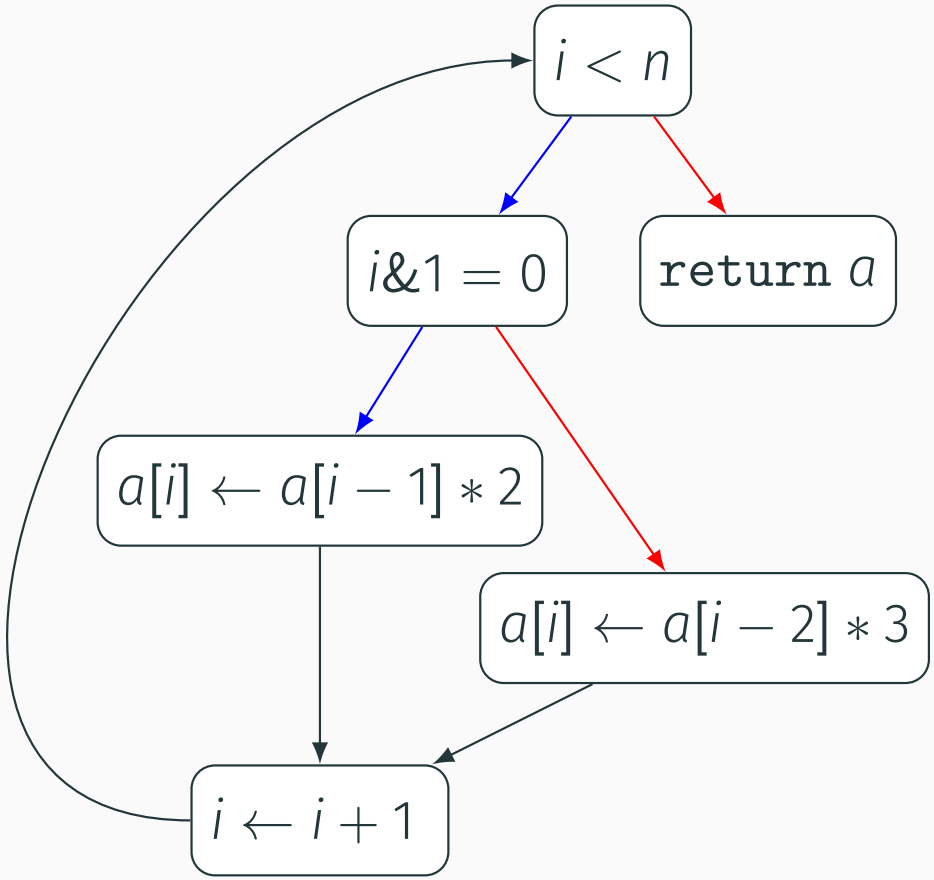
August 4, 2016

University of Wisconsin – Milwaukee



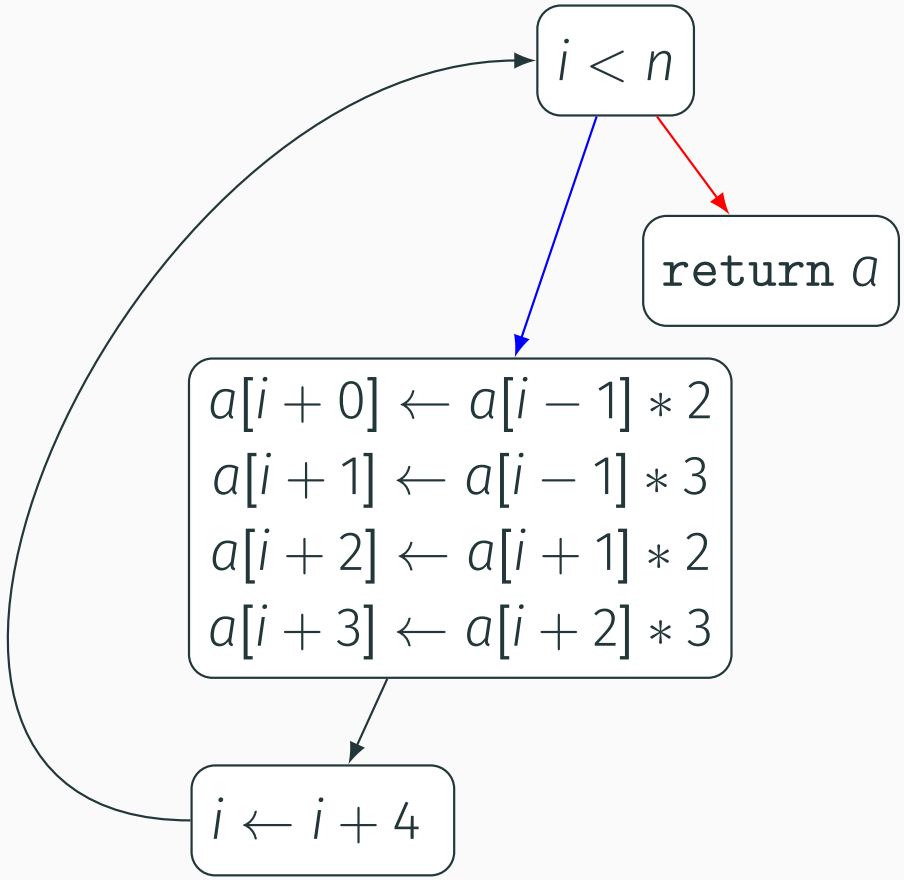
10,000 Foot View

```
1 while i < n do
2 |   if i&1 = 0 then
3 |   |   a[i] ← a[i − 1] ∗ 2
4 |   else
5 |   |   a[i] ← a[i − 2] ∗ 3
6 |   end
7 |   i ← i + 1
8 end
9 return a
```



10,000 Foot View

```
1 while i < n do
2 |   if i&1 = 0 then
3 |   |   a[i] ← a[i − 1] ∗ 2
4 |   else
5 |   |   a[i] ← a[i − 2] ∗ 3
6 |   end
7 |   i ← i + 1
8 end
9 return a
```



10,000 Foot View - LLVM 2.9 -O2 Passes

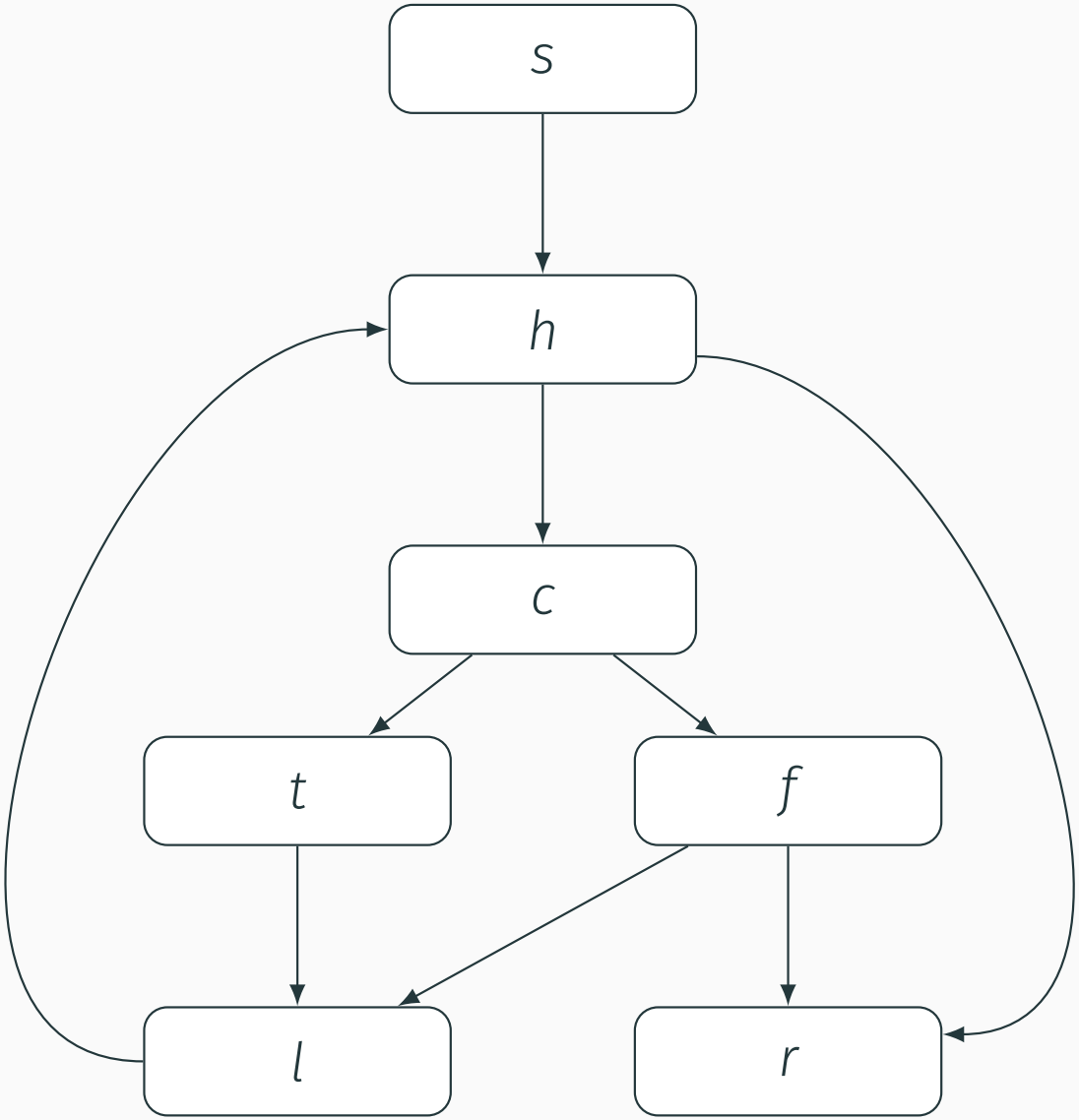
1. no/tb/basic-aa
2. domtree
3. verify
4. lowersetjmp
5. globalopt
6. ipsccp
7. deadargelim
8. basiccg
9. prune-eh
10. inline
11. functionattrs
12. argpromotion
13. scalarrepl-ssa
14. domtree
15. early-cse
16. simplify-libcalls
17. lazy-value-info
18. jump-threading
19. tailcallelim
20. reassociate
21. domtree
22. loops
23. loop-simplify
24. lcssa
25. loop-rotate
26. licm
27. lcssa
28. loop-unswitch
29. scalar-evolution
30. loop-simplify
31. lcssa
32. iv-users
33. indvars
34. loop-idiom
35. loop-deletion
36. loop-unroll
37. memdep
38. gvn
39. memdep
40. memcpyopt
41. sccp
42. lazy-value-info
43. jump-threading
44. domtree
45. memdep
46. dse
47. adce
48. strip-dead-prototypes
49. deadtypeelim
50. globaldce
51. constmerge
52. globalopt
53. ipsccp
54. deadargelim
55. basiccg
56. prune-eh
57. inline
58. functionattrs
59. scalarrepl-ssa
60. domtree
61. early-cse
62. simplify-libcalls
63. lazy-value-info
64. jump-threading
65. tailcallelim
66. reassociate
67. domtree
68. loops
69. loop-simplify
70. lcssa
71. loop-rotate
72. licm
73. lcssa
74. loop-unswitch
75. scalar-evolution
76. loop-simplify
77. lcssa
78. iv-users
79. indvars
80. loop-idiom
81. loop-deletion
82. loop-unroll
83. memdep
84. gvn
85. memdep
86. memcpyopt
87. sccp
88. lazy-value-info
89. jump-threading
90. domtree
91. memdep
92. dse
93. adce
94. strip-dead-prototypes
95. deadtypeelim
96. constmerge
97. domtree
98. verify

Structure of Talk

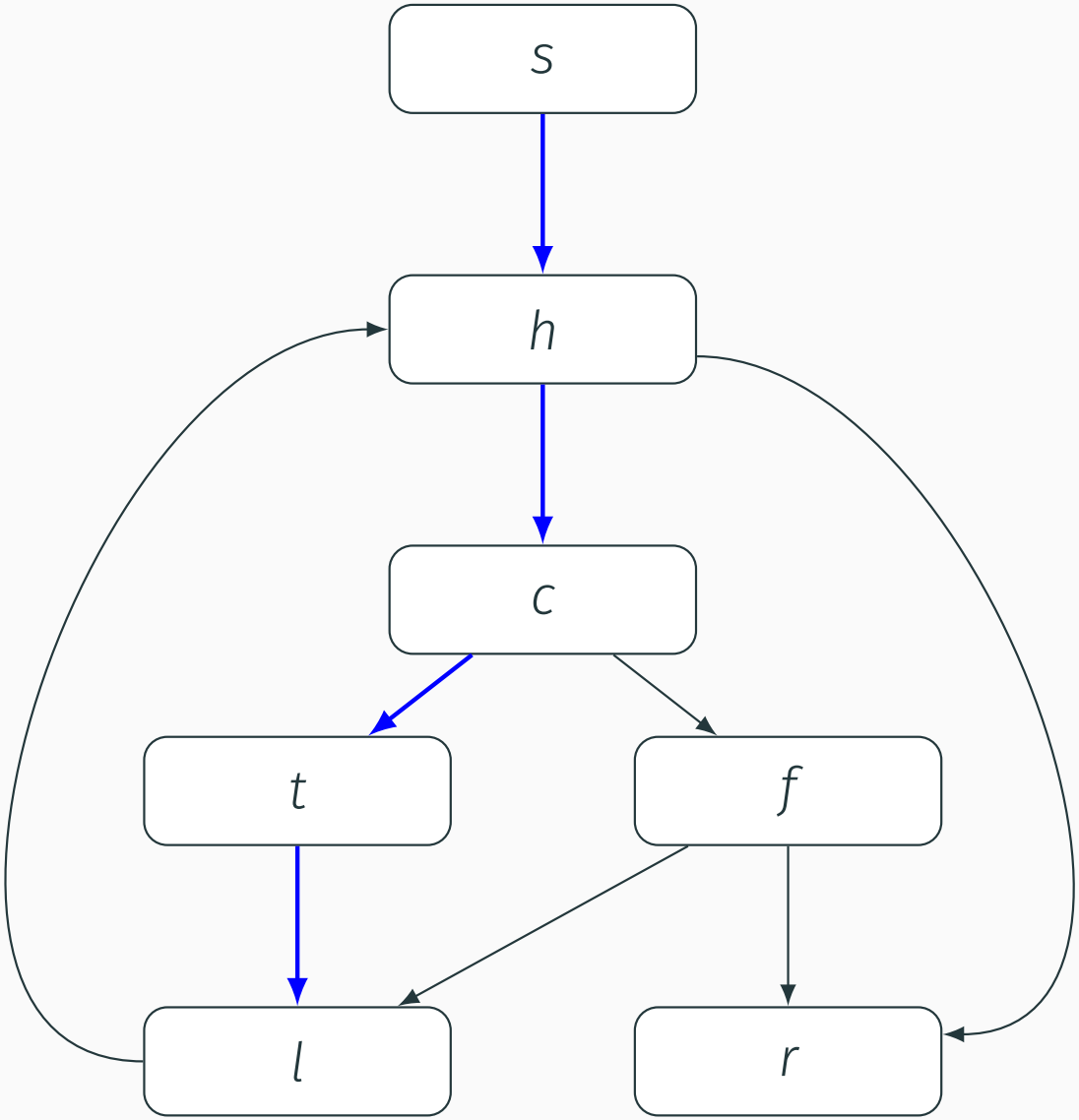
1. Auxiliary Structures, IRs
2. Motivation
3. SSA Reconstruction
4. Dominator Tree Reconstruction
5. Canonical Form Preservation
6. Tasklist

Auxiliary Structures, IRs

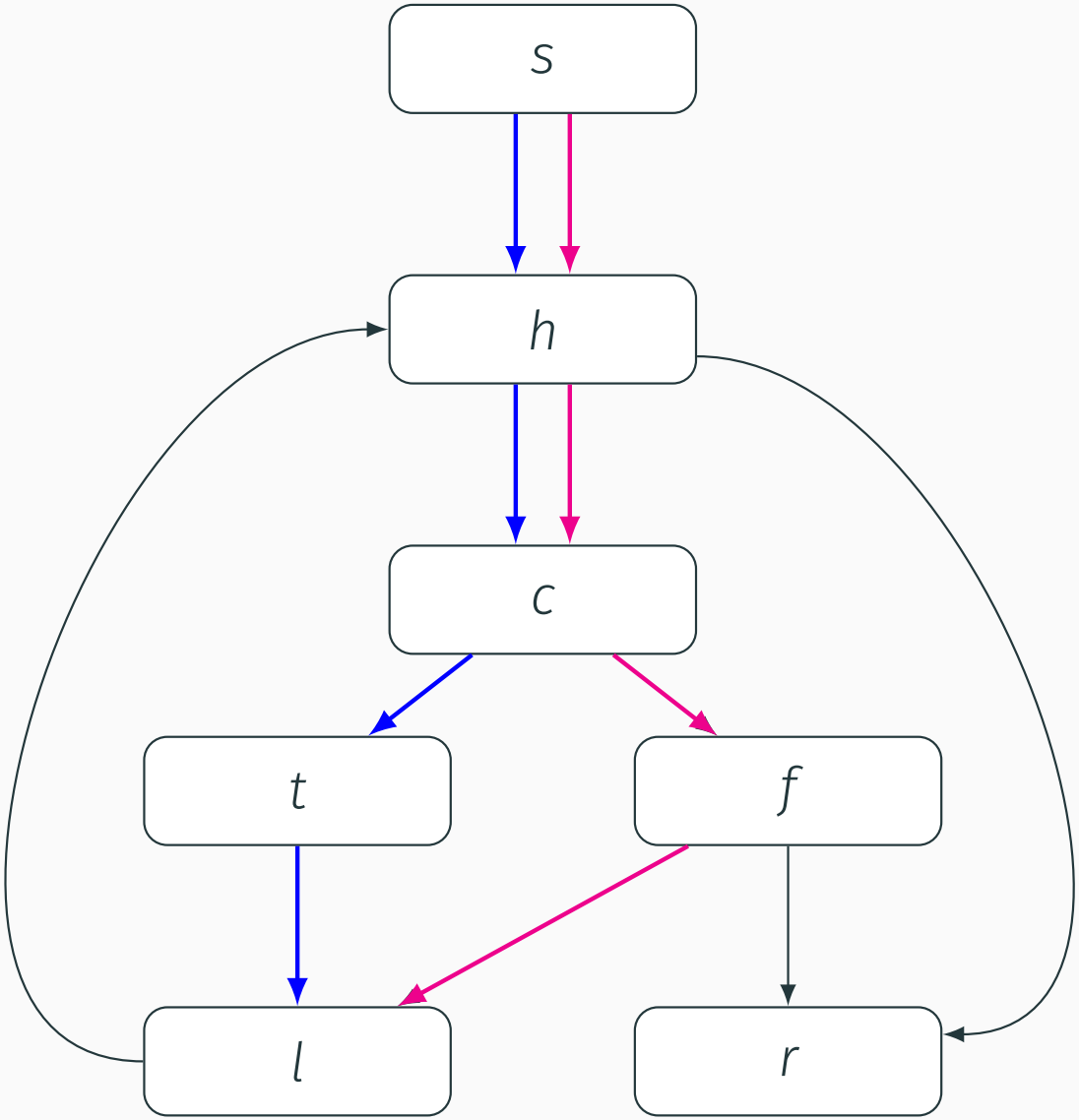
Domination



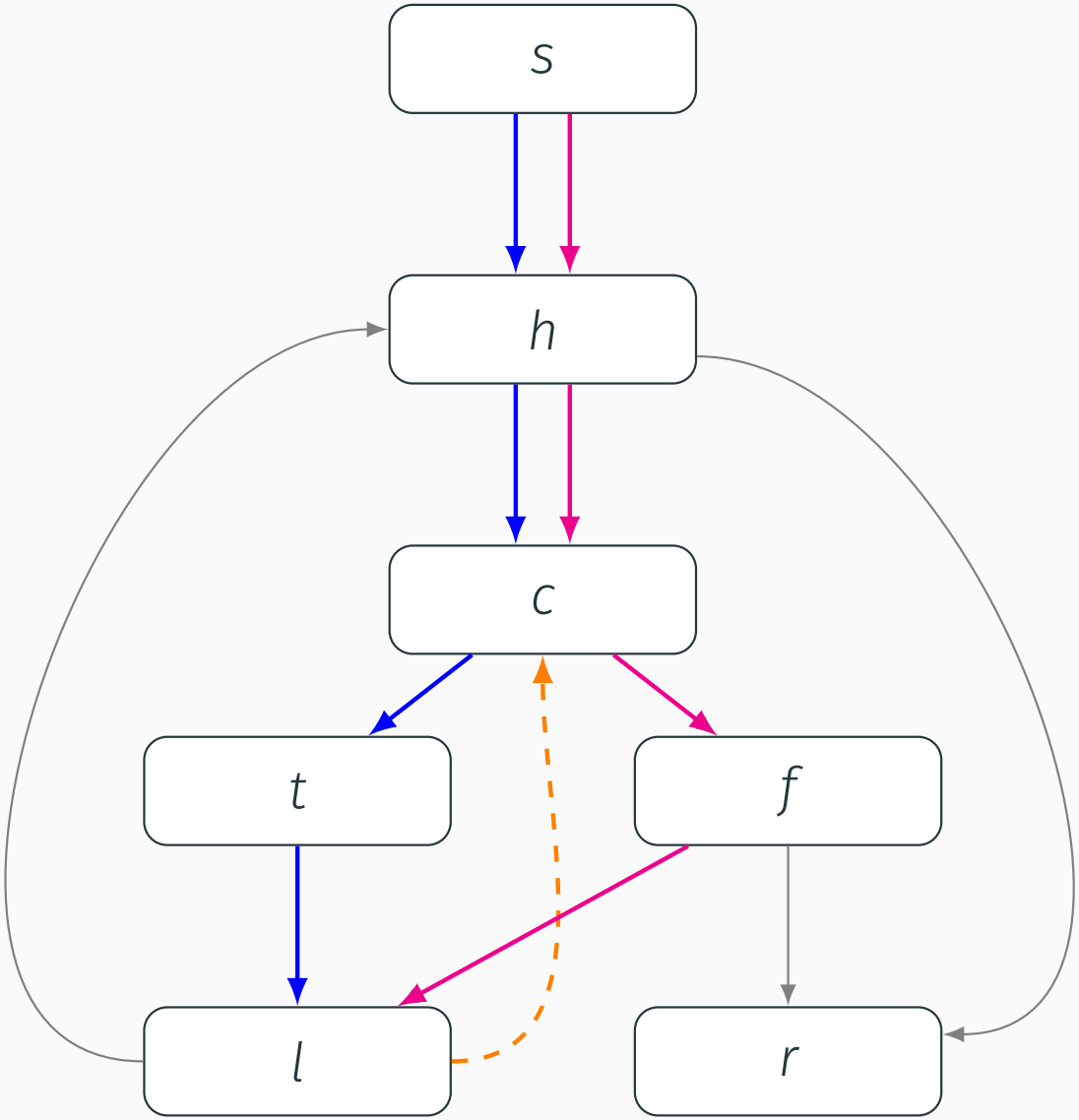
Domination



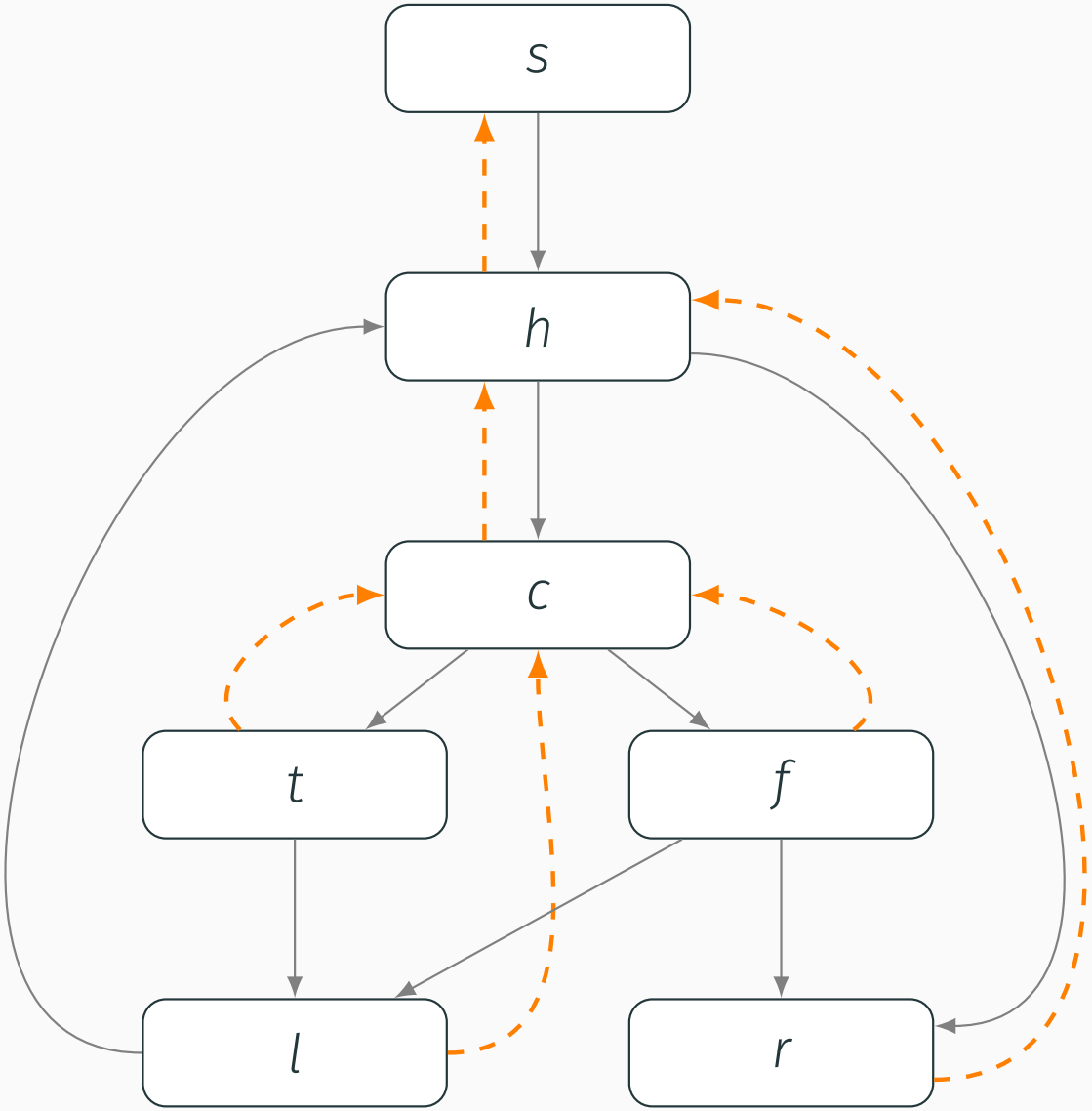
Domination



Domination



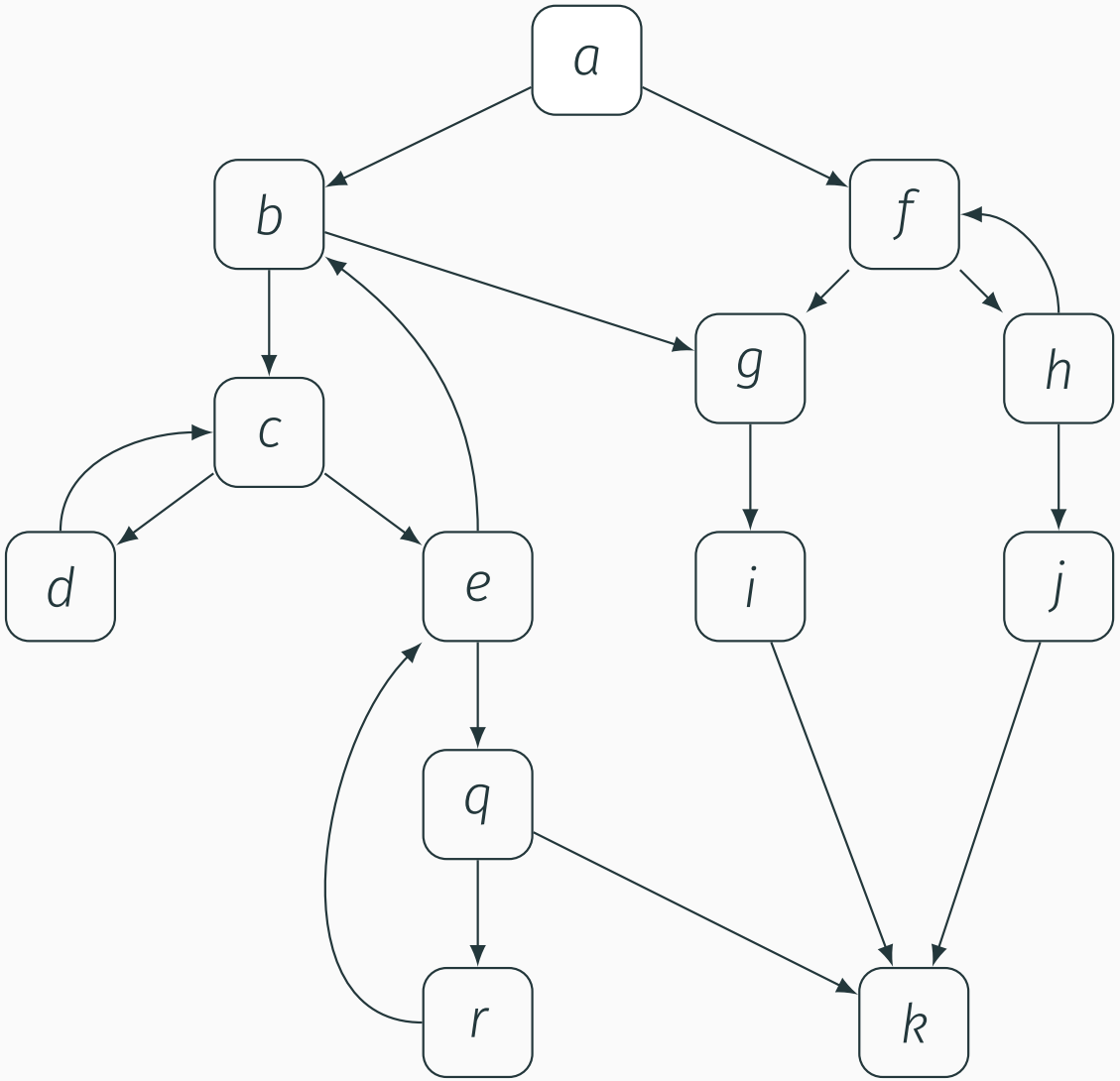
Domination



Domination - An (Extremely) Condensed History

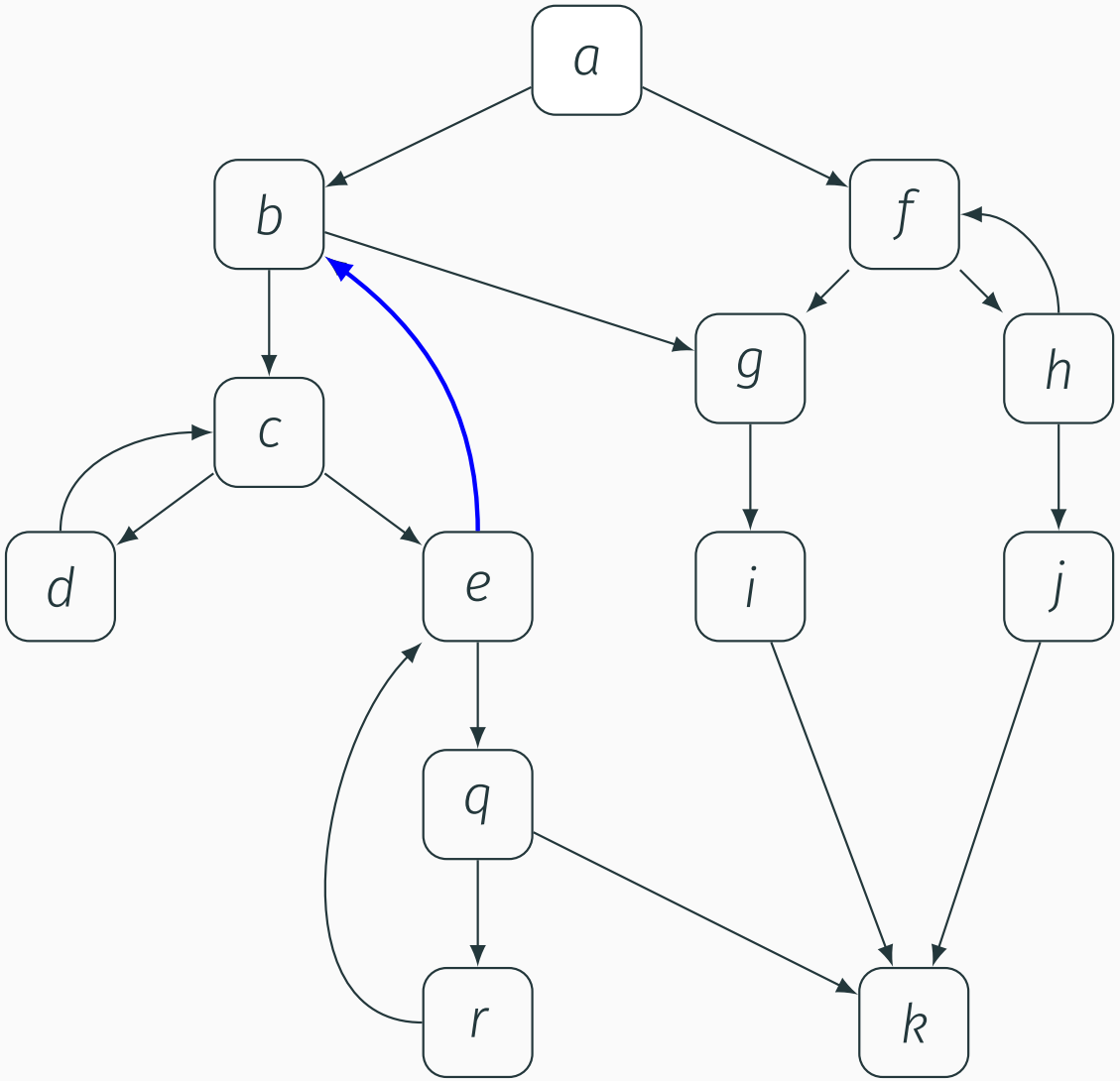
- Purdom-Moore, 1972
- Lengauer-Tarjan, 1979
- Buschbaum, 1998
- Cooper, 2001
- Georgiadis, 2005

Loops & Loop Nesting Forest



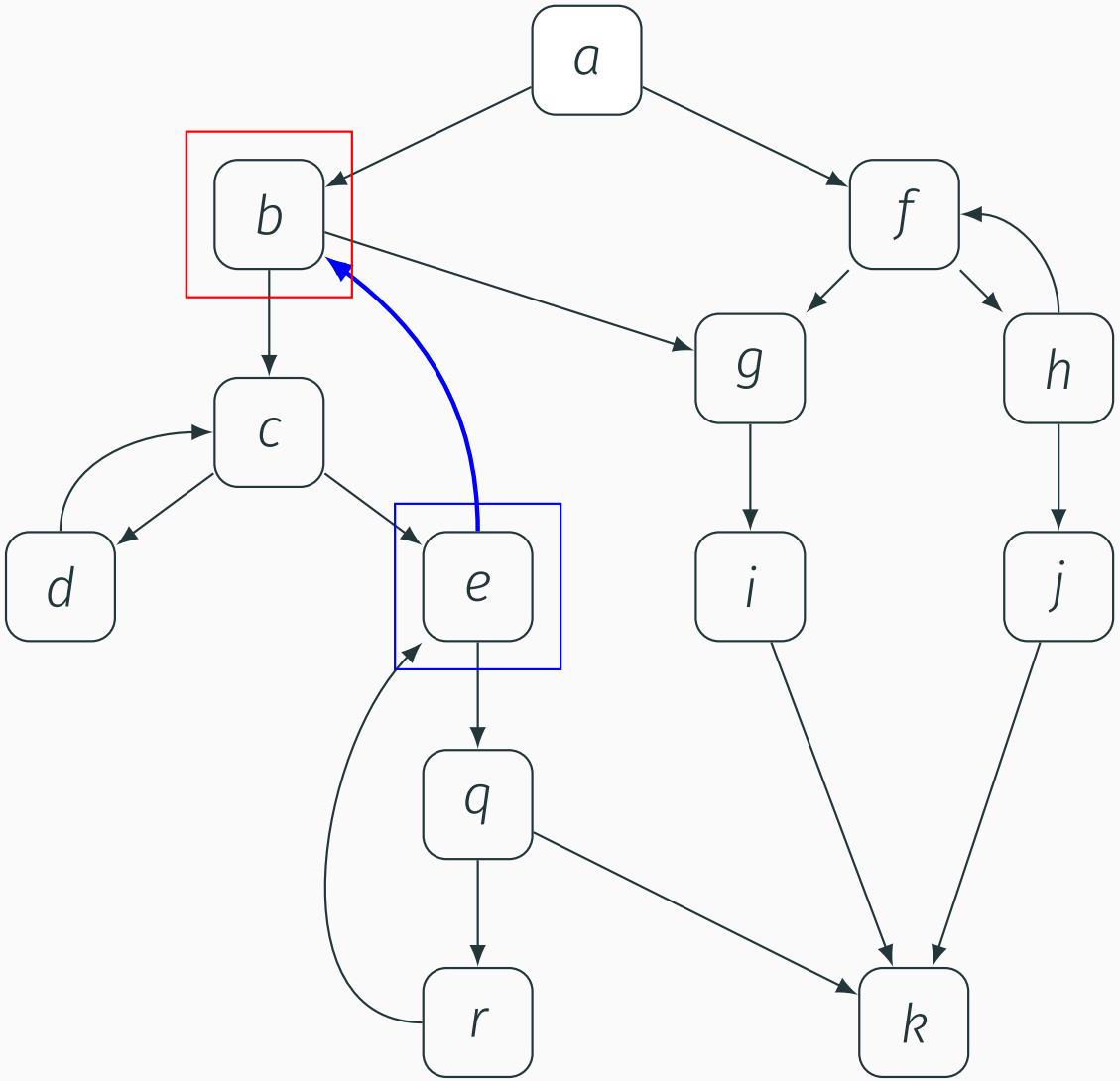
Loops & Loop Nesting Forest

(1) Identify Backedge



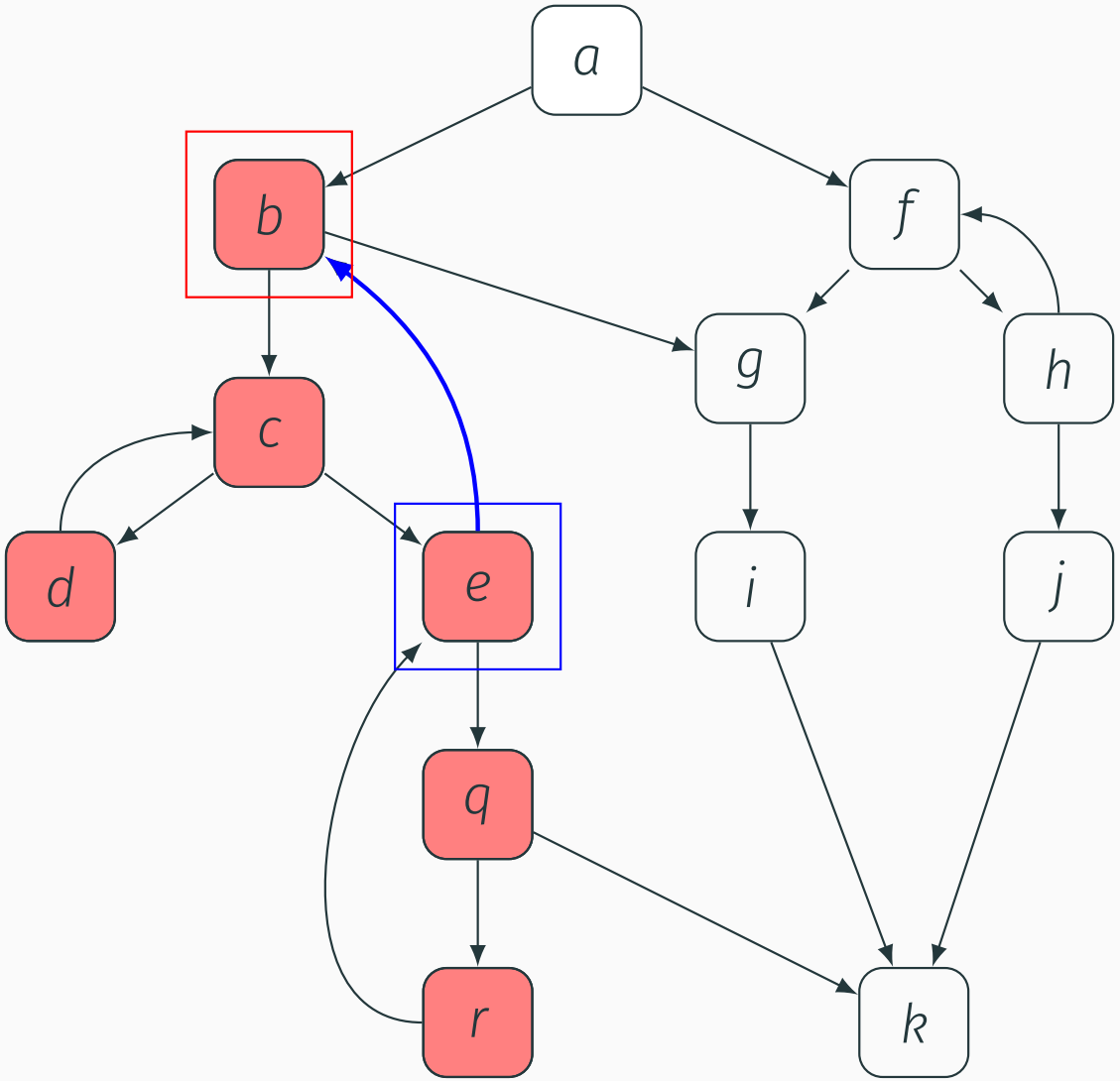
Loops & Loop Nesting Forest

(2) Identify Header



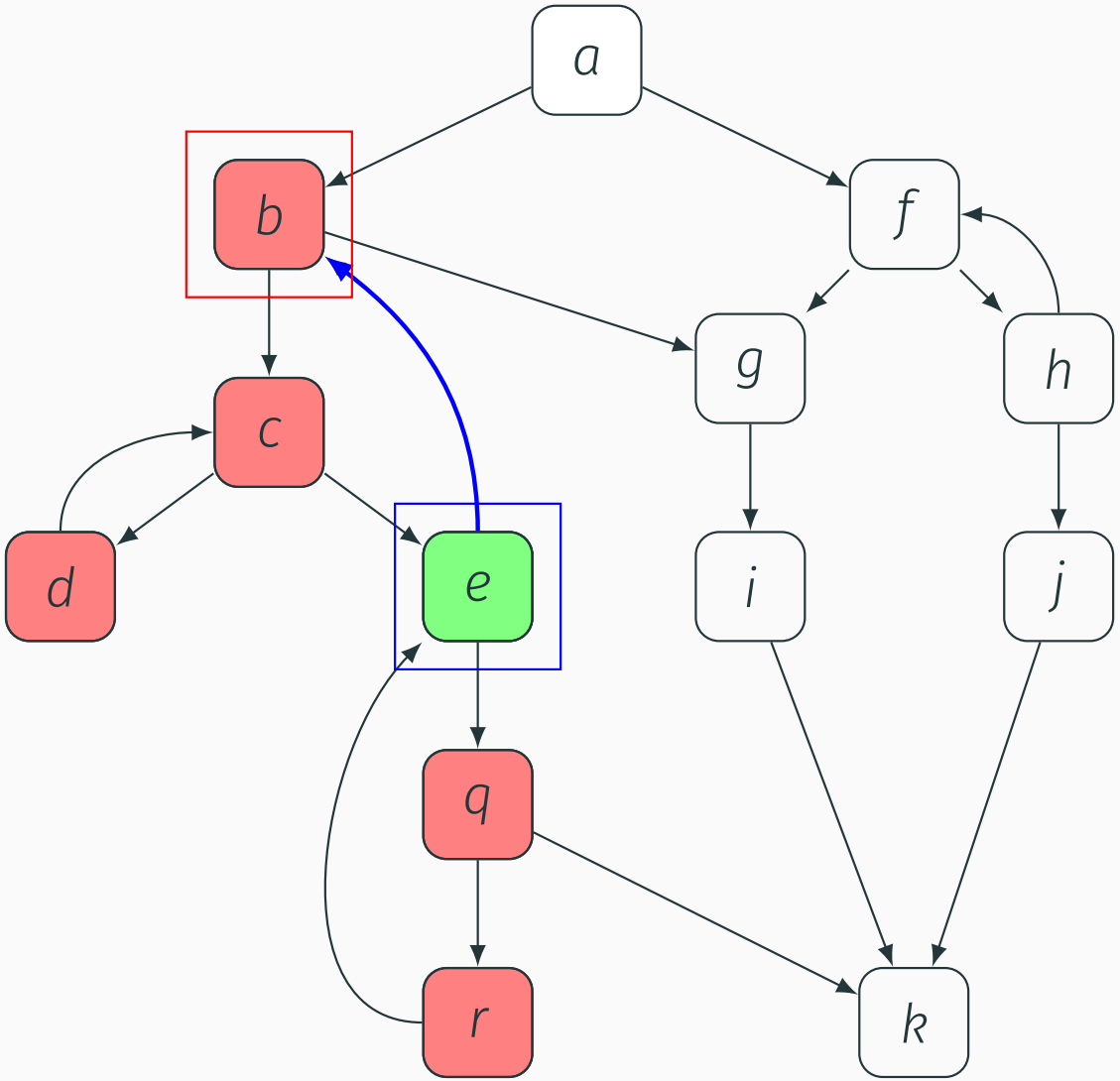
Loops & Loop Nesting Forest

(3) Identify Dominated Blocks



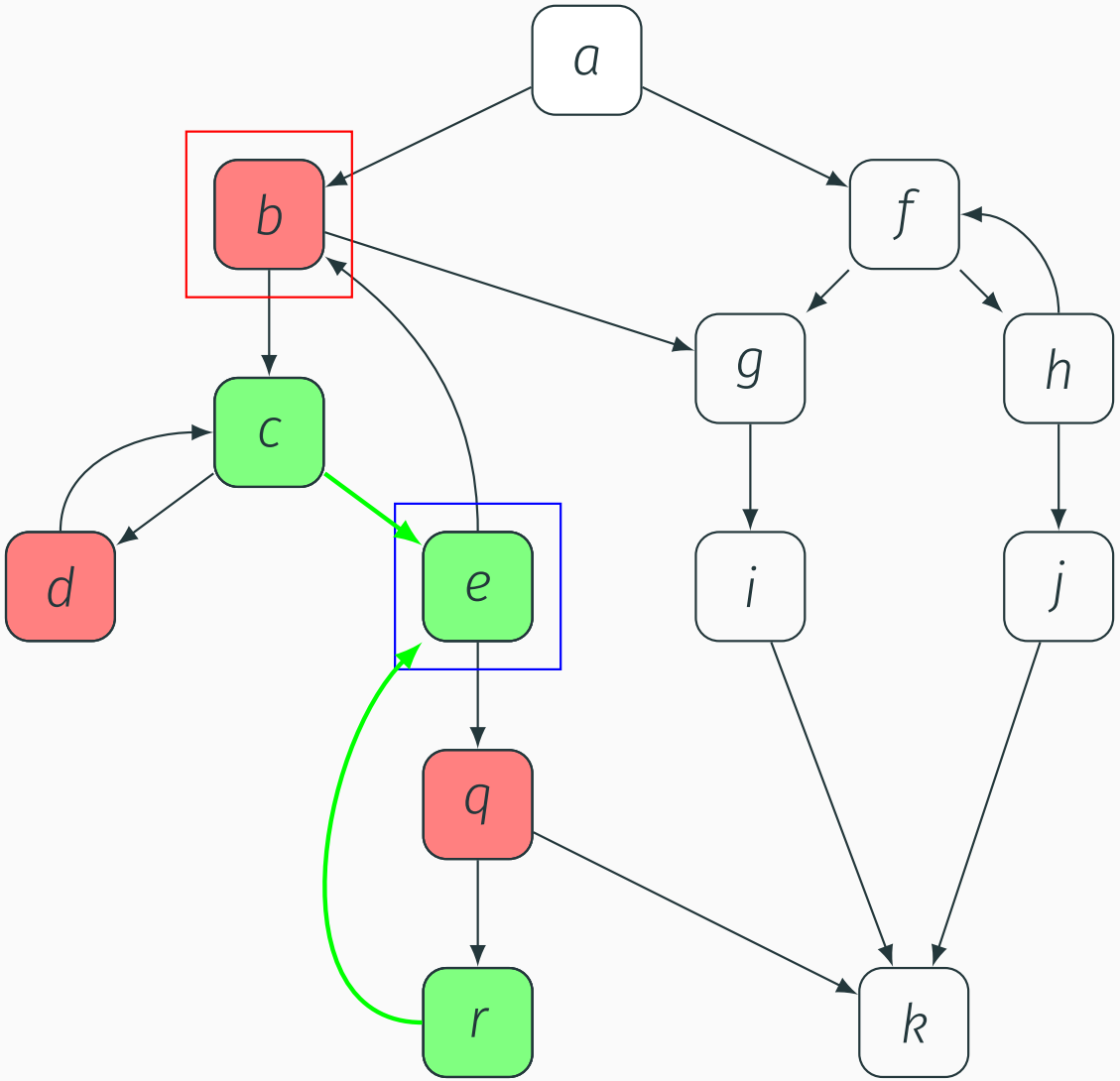
Loops & Loop Nesting Forest

(4) Trace from latch



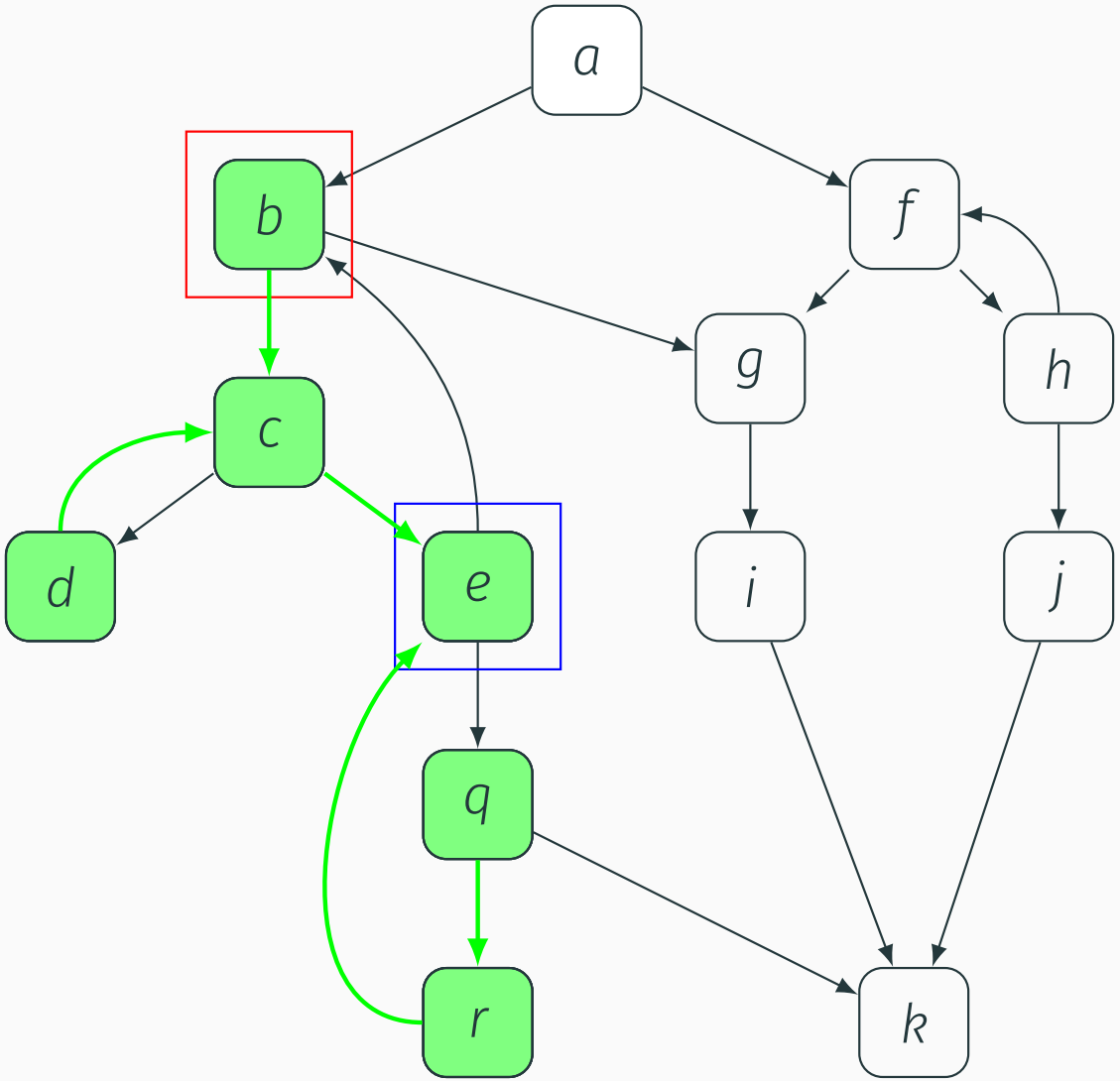
Loops & Loop Nesting Forest

(4) Trace from latch



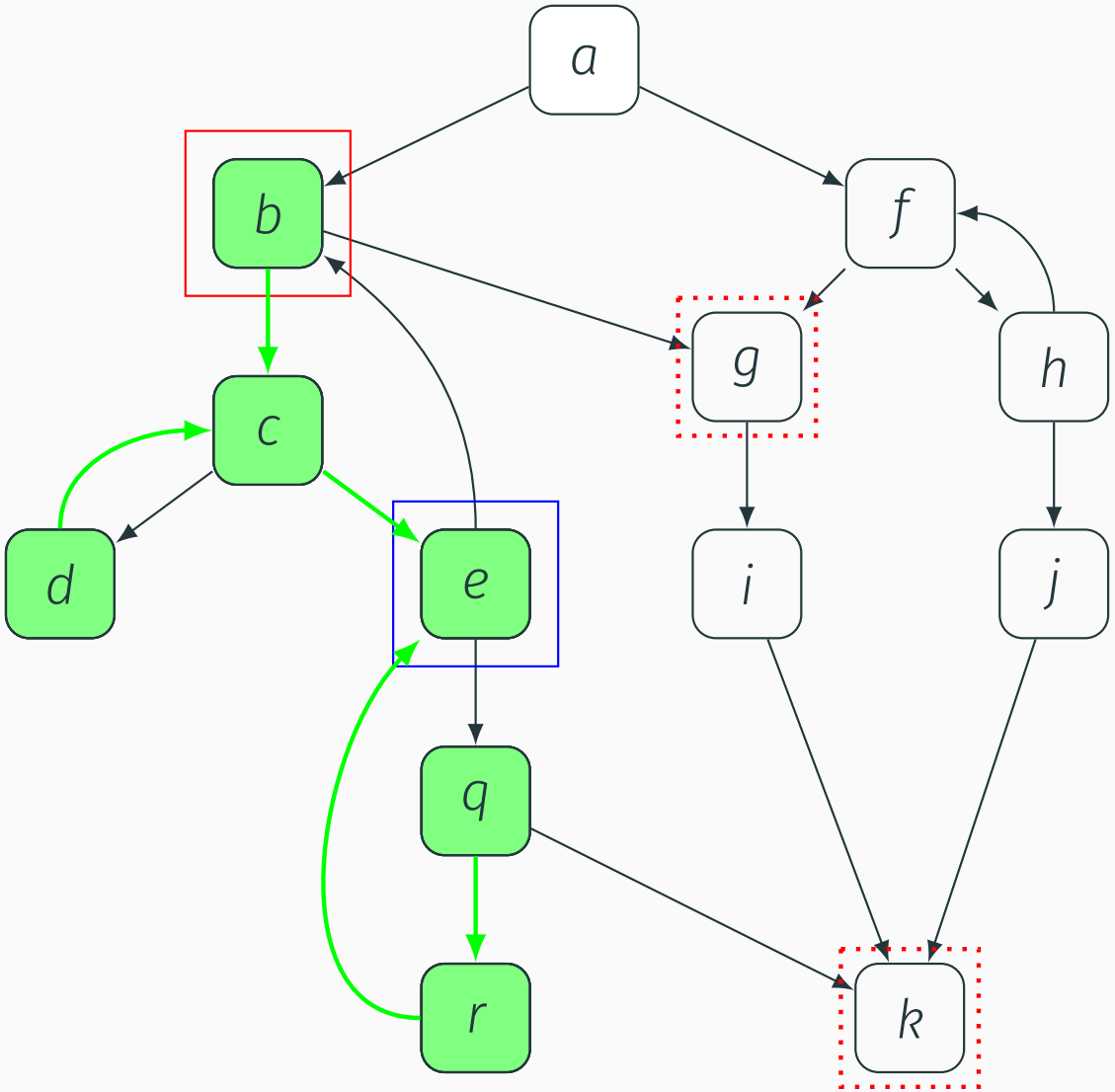
Loops & Loop Nesting Forest

(4) Trace from latch

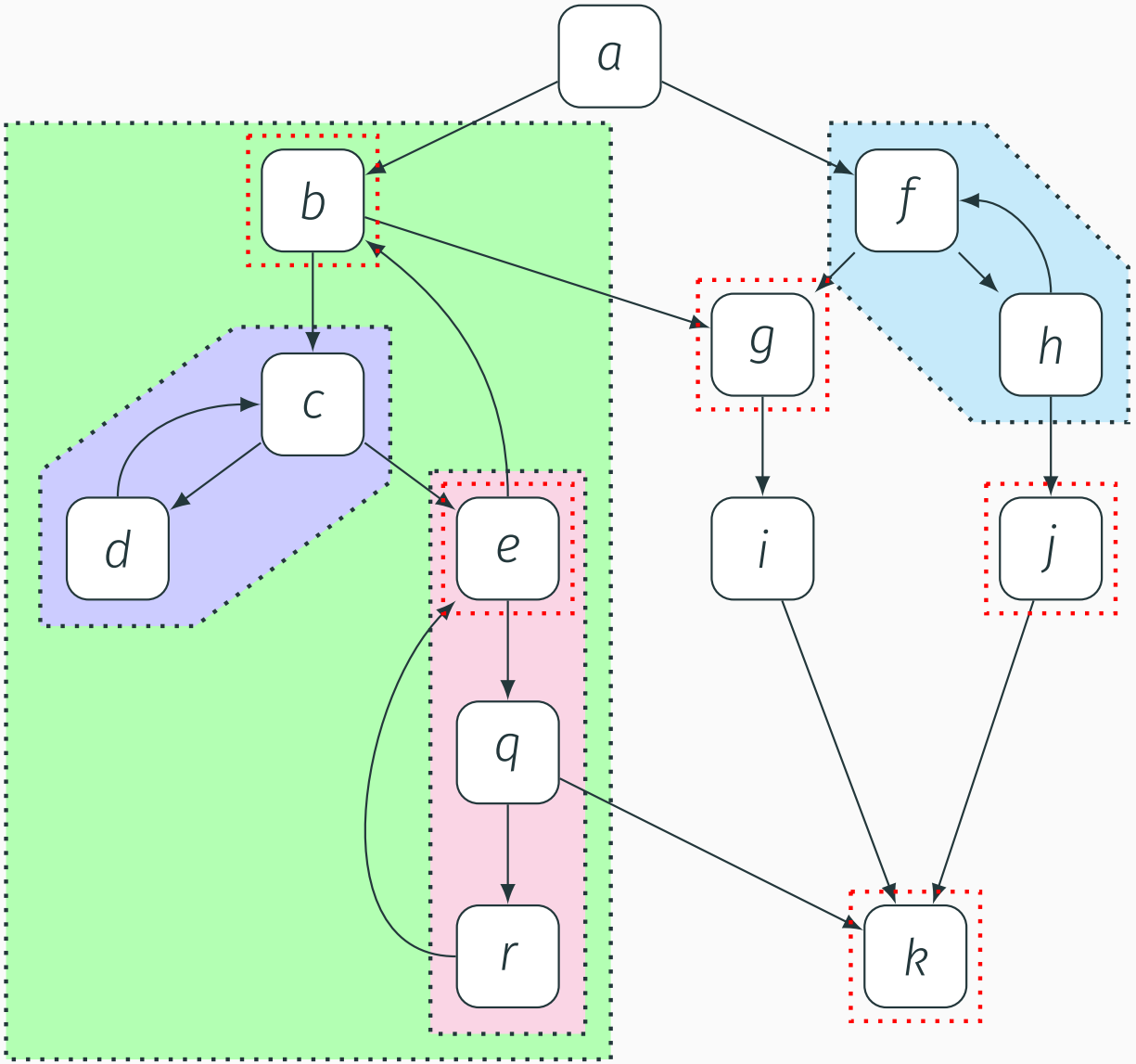


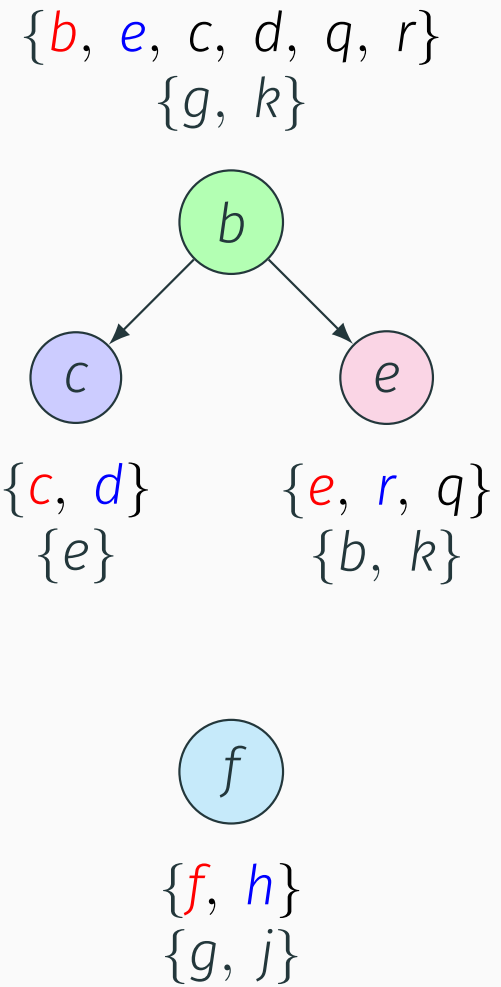
Loops & Loop Nesting Forest

(5) Identify Exits



Loops & Loop Nesting Forest

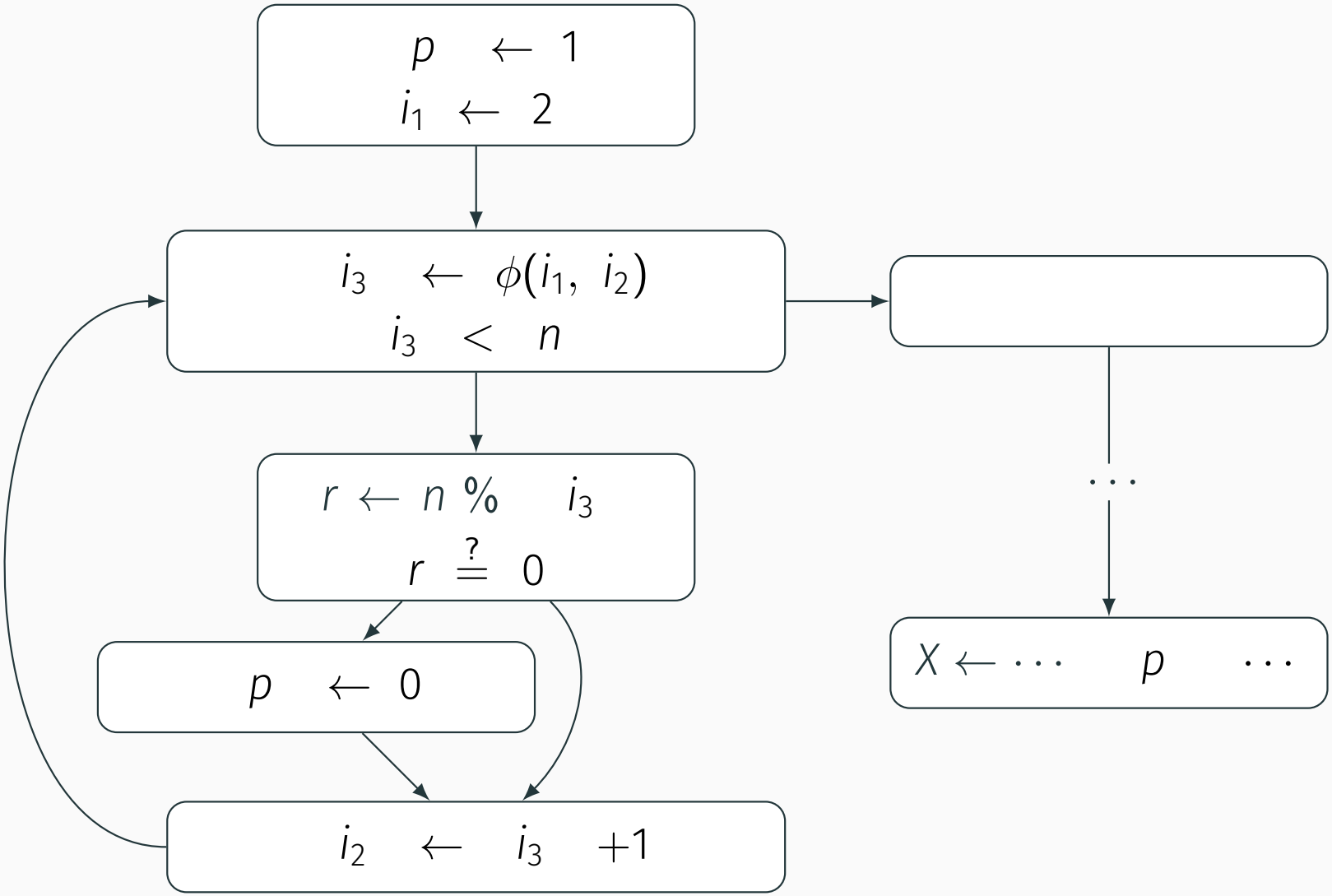




Loop Nesting Forest - An (Extremely) Condensed History

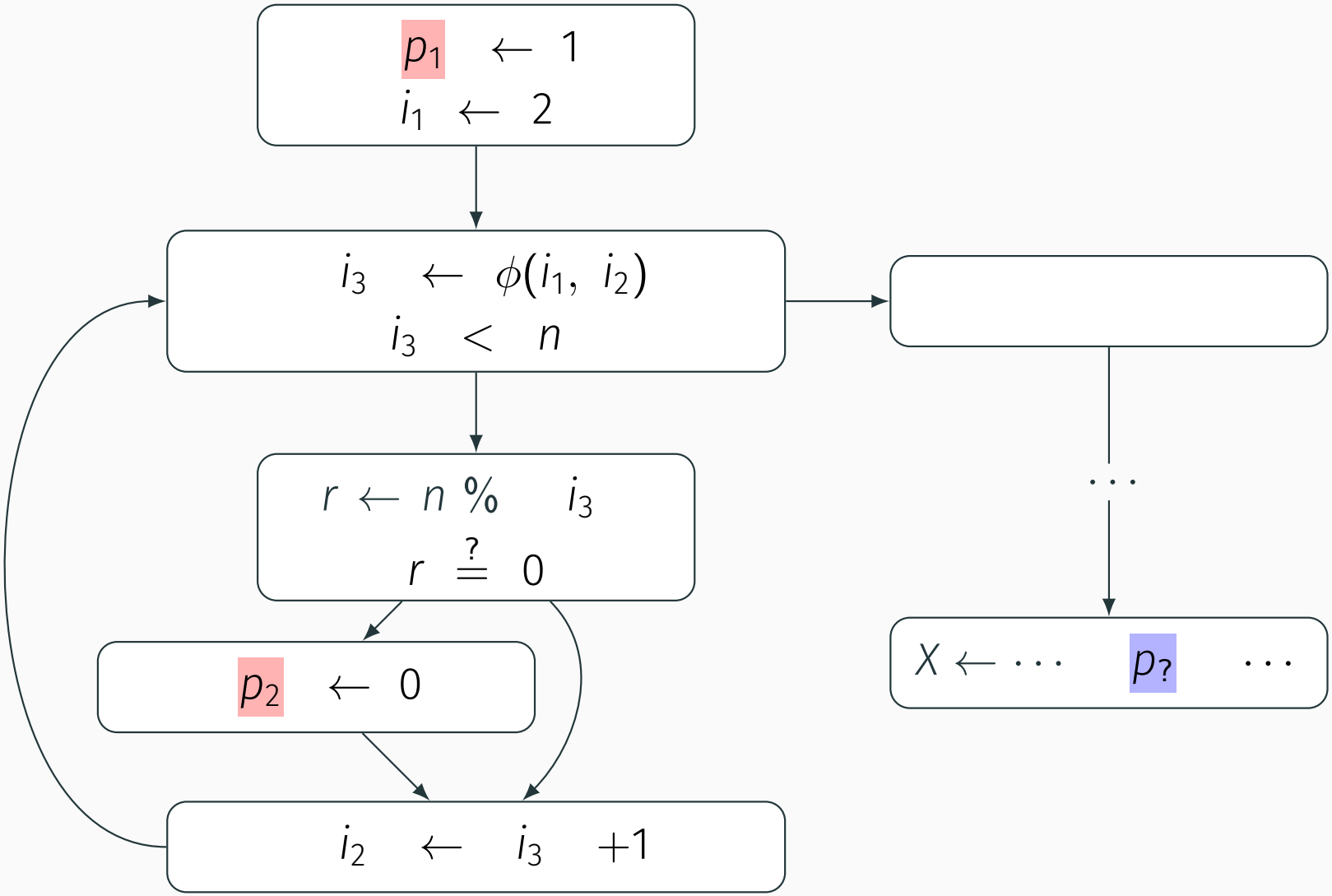
- Tarjan, 1973
- Steensgaard, 1993
- Sreedhar, 1996
- Havlak, 1997

Static Single Assignment Form



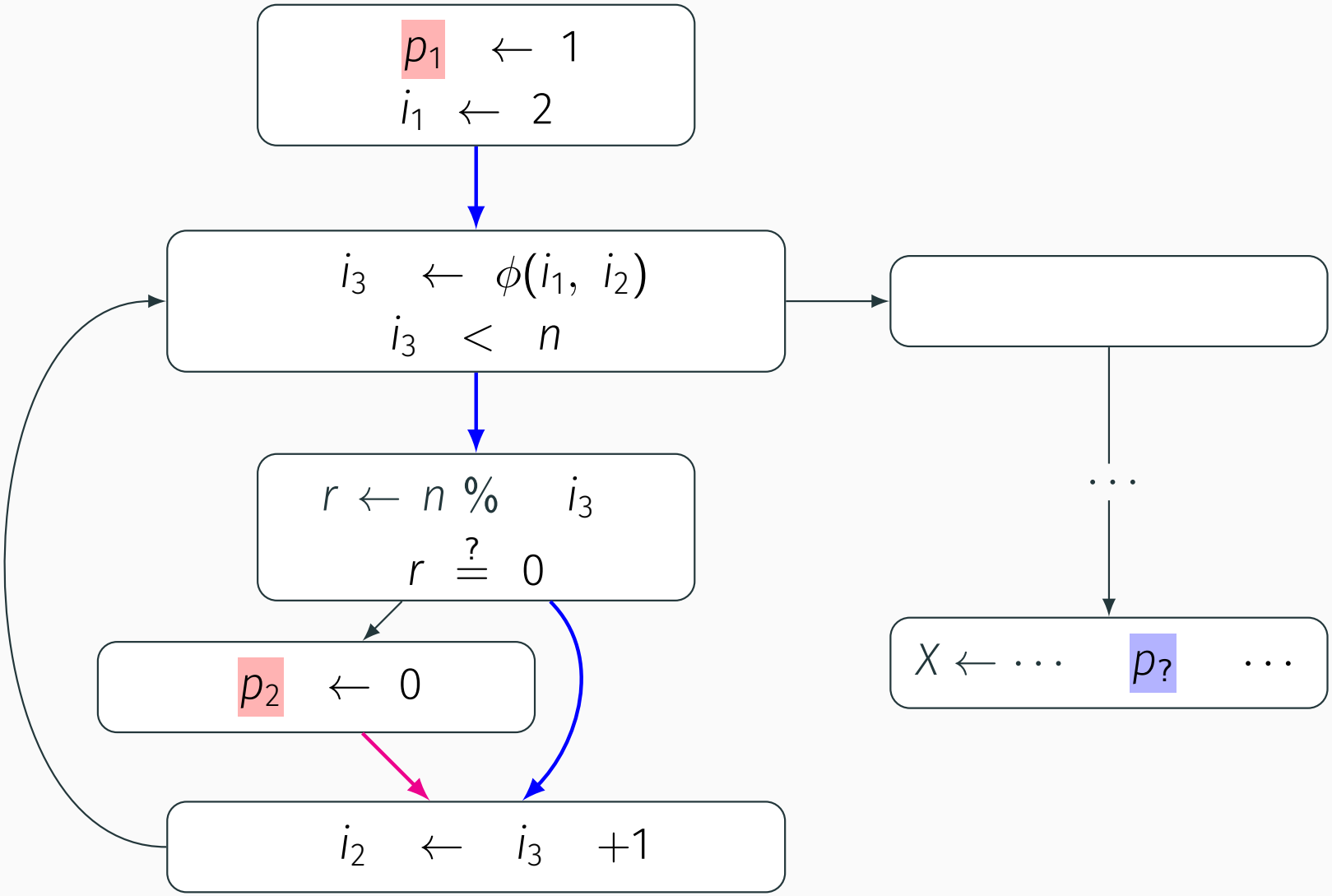
Static Single Assignment Form

(1) Generate unique assignment targets



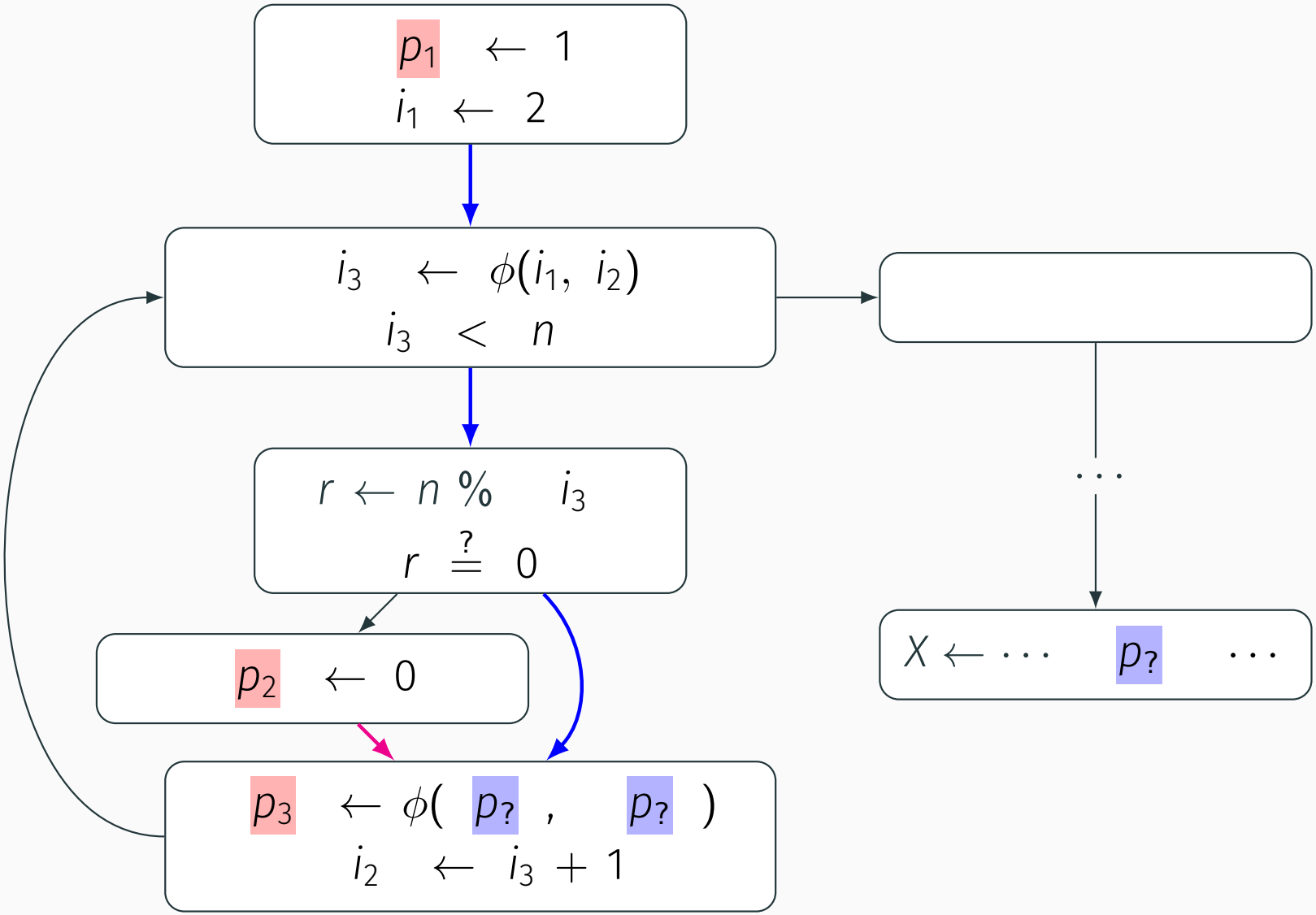
Static Single Assignment Form

(2a) Determine join points with ‘dominance frontier’



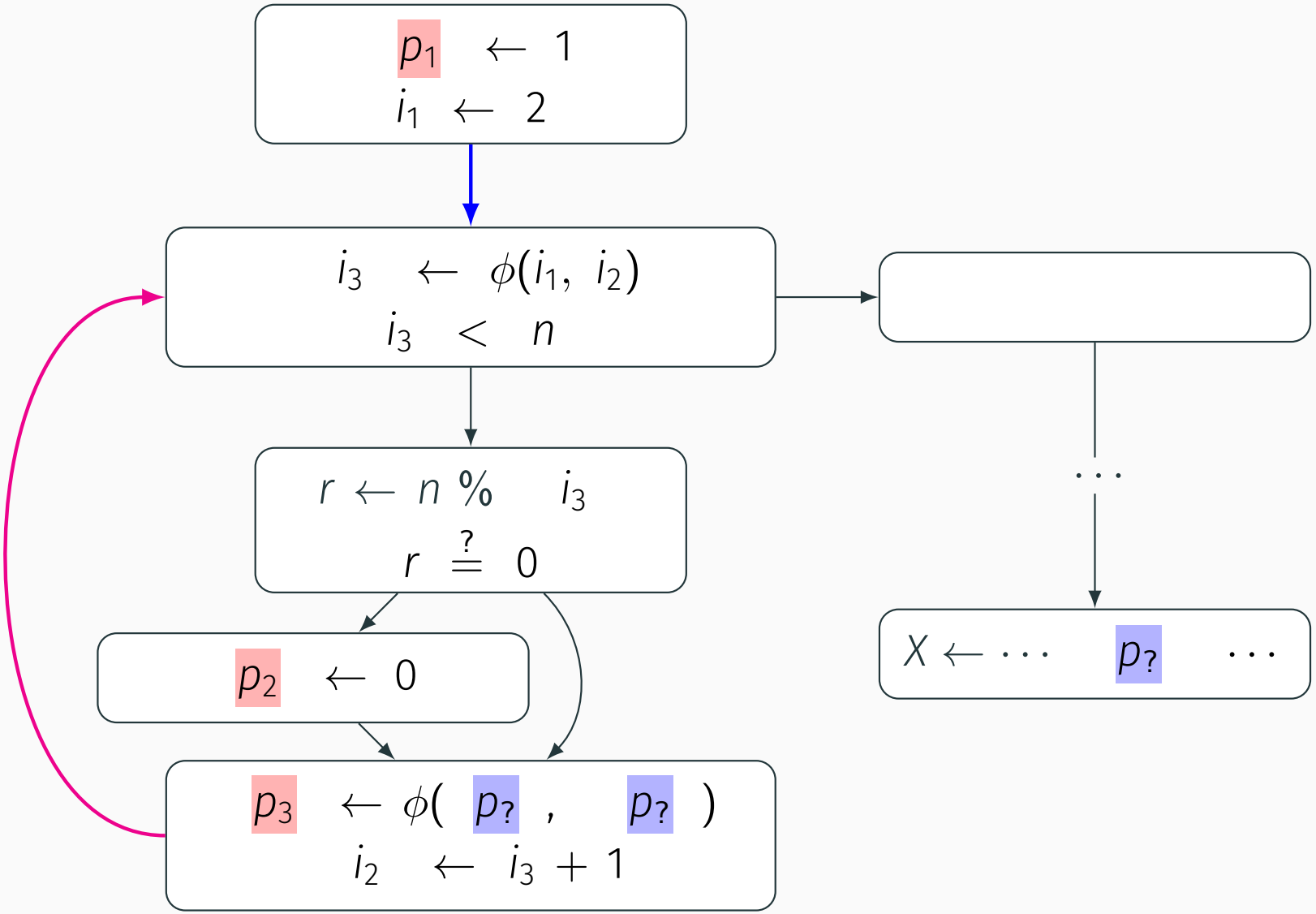
Static Single Assignment Form

(3a) Place ϕ-nodes



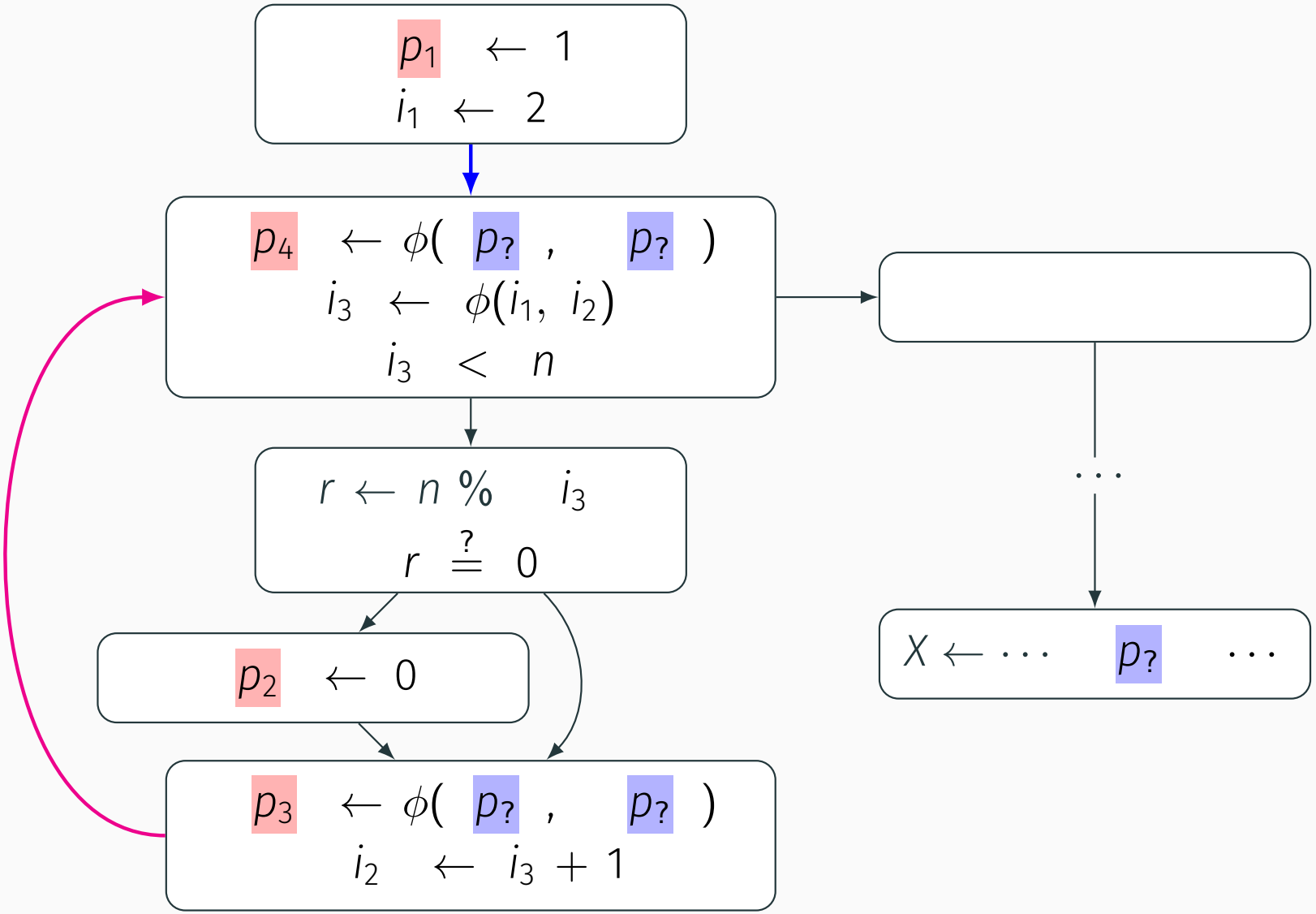
Static Single Assignment Form

(2b) Determine join points with ‘dominance frontier’



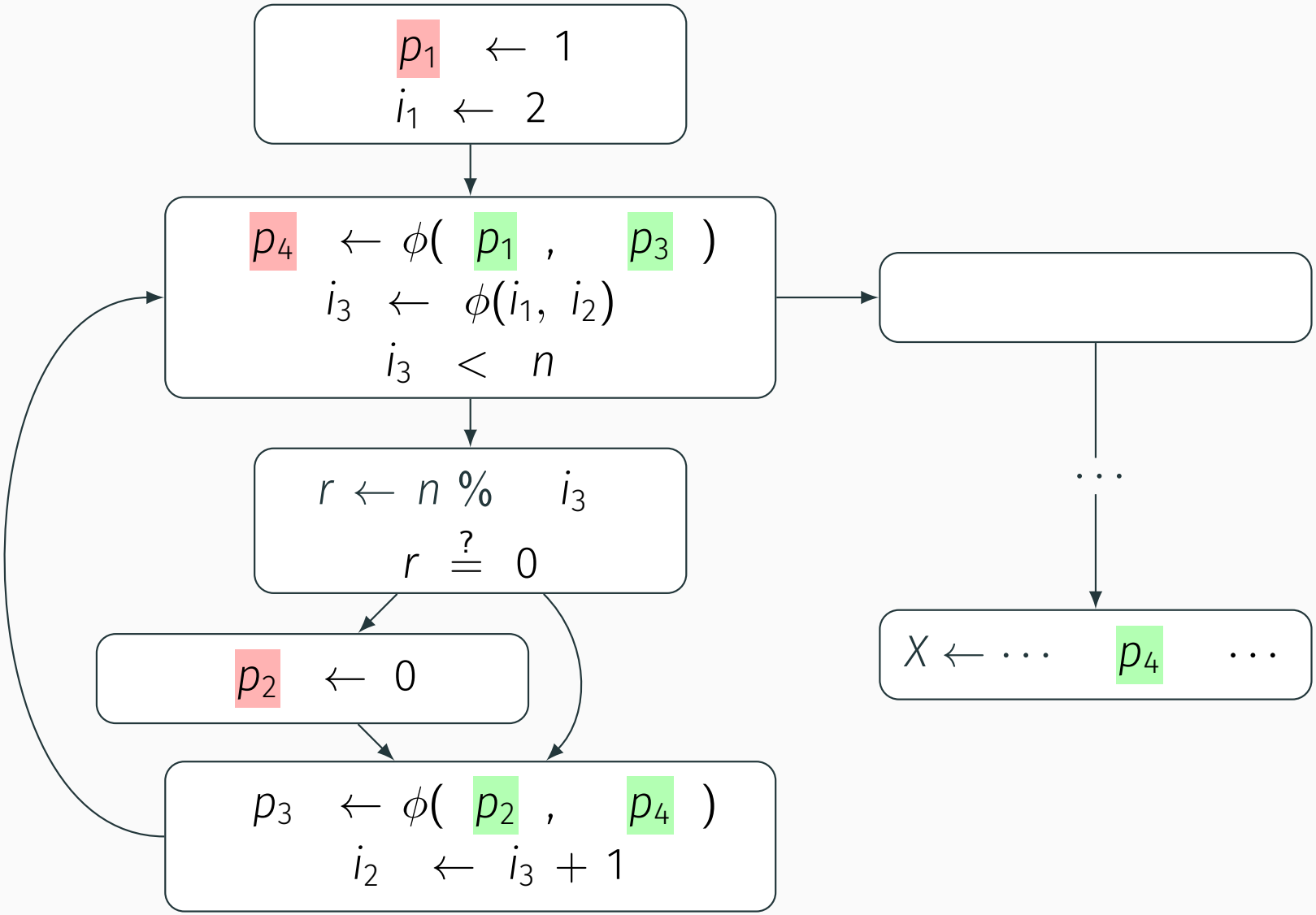
Static Single Assignment Form

(3b) Place ϕ-nodes



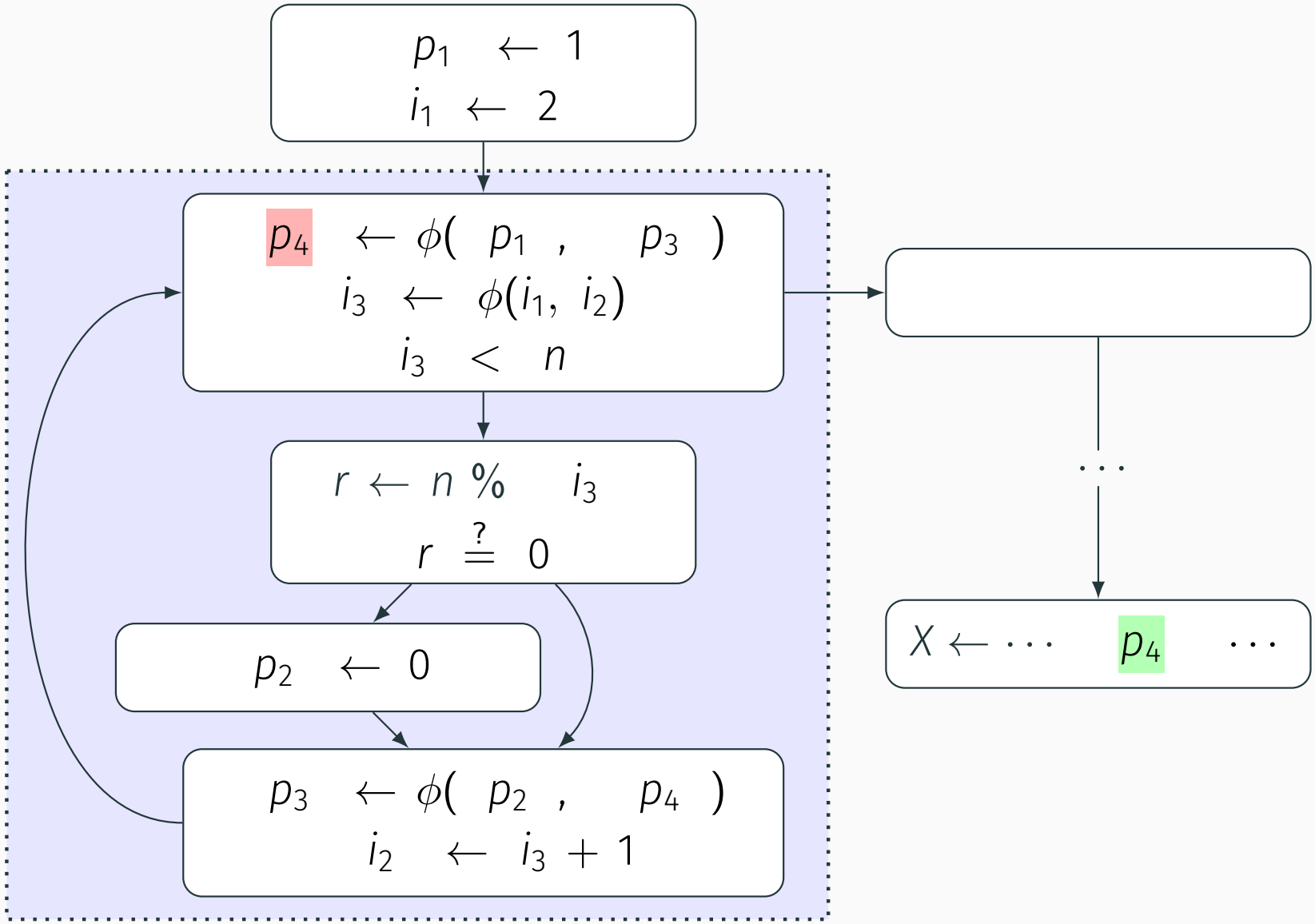
Static Single Assignment Form

(4) Rewrite uses with *RD*



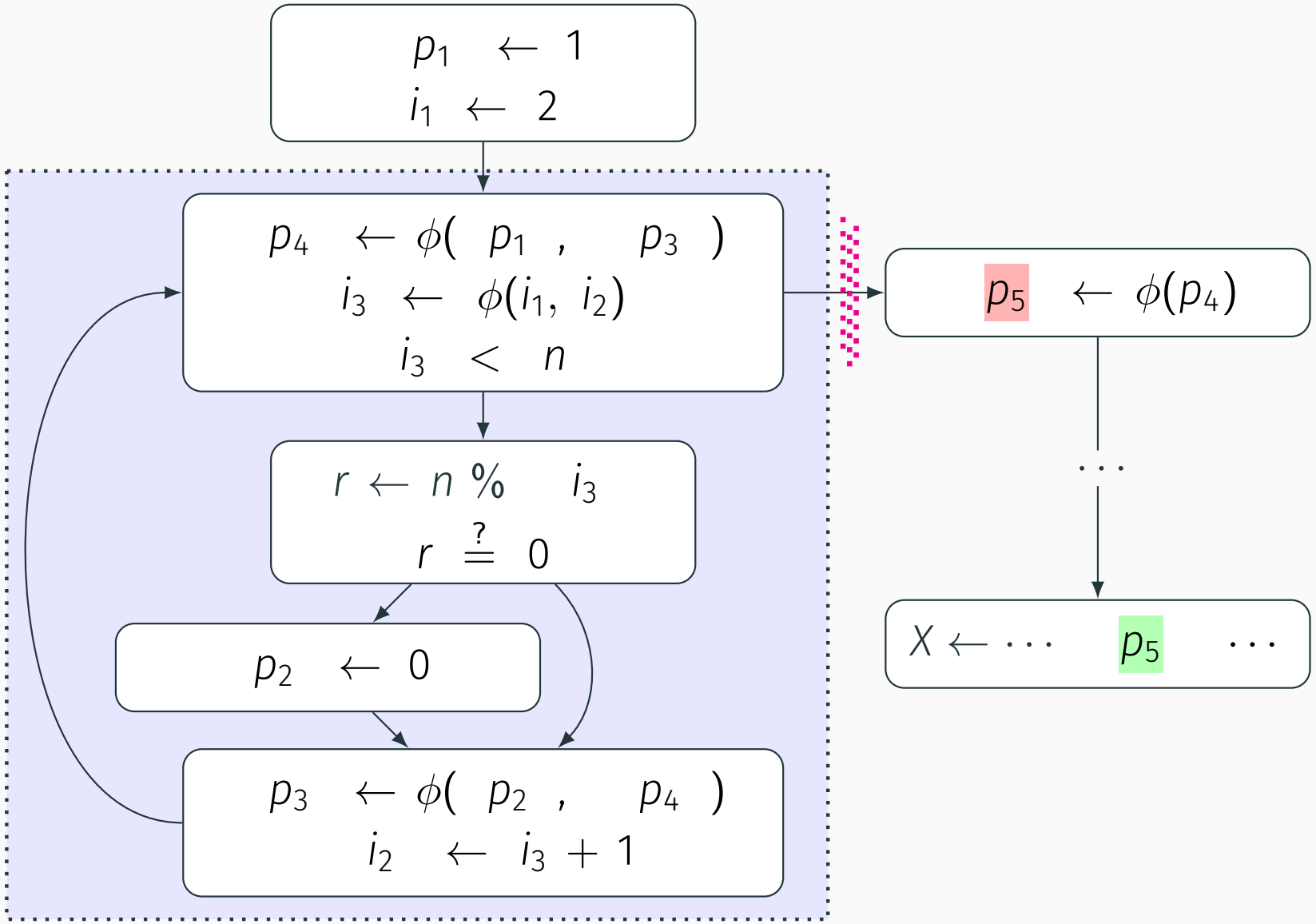
Loop-Closed Static Single Assignment Form

Loop Closed: No uses outside loop of register defined within loop



Loop-Closed Static Single Assignment Form

Loop Closed: No uses outside loop of register defined within loop

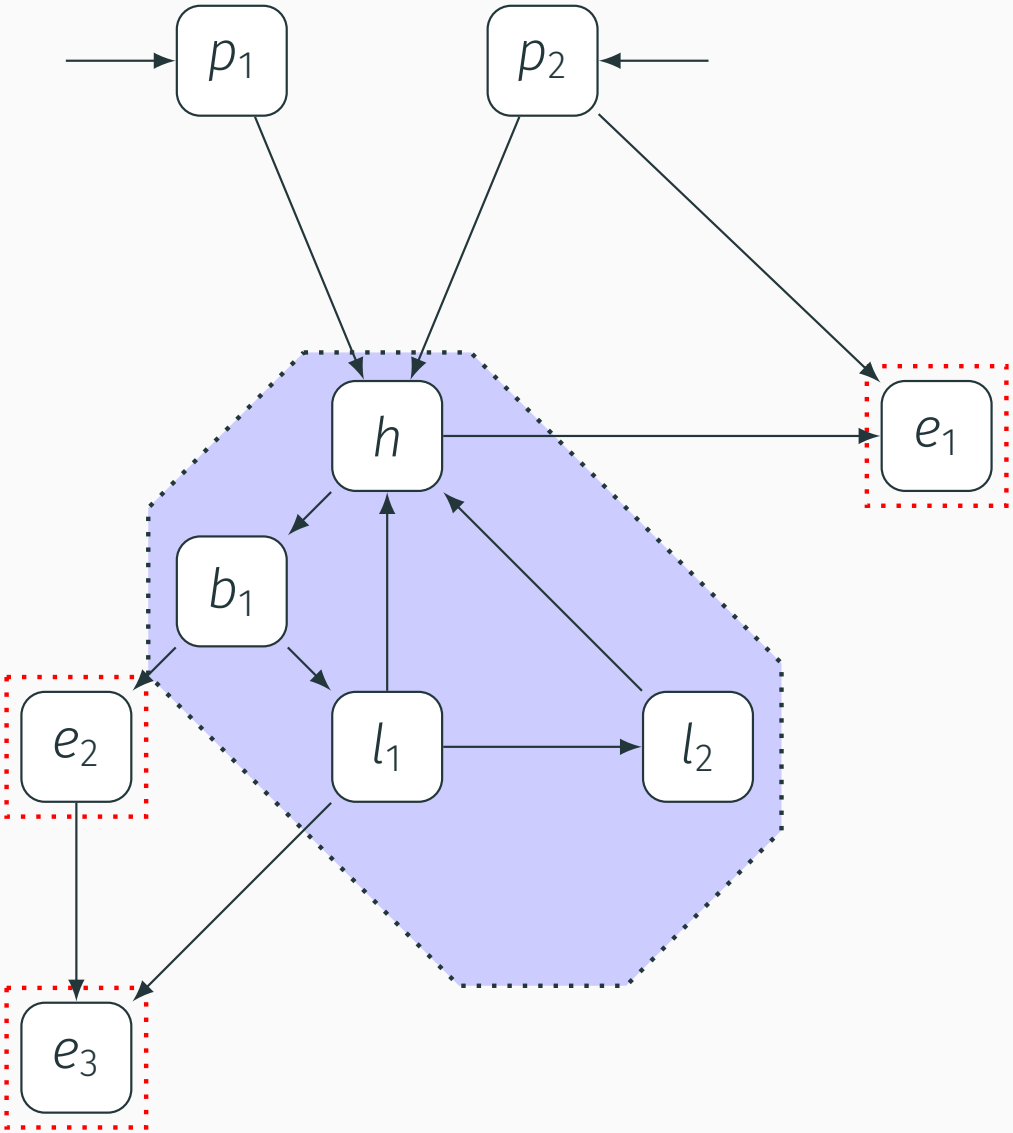


SSA - An (Extremely) Condensed History

- Cytron & Rosen & Zadeck, 1991
- Choi, 1991
- Briggs, 1998
- Braun & Hack, 2013

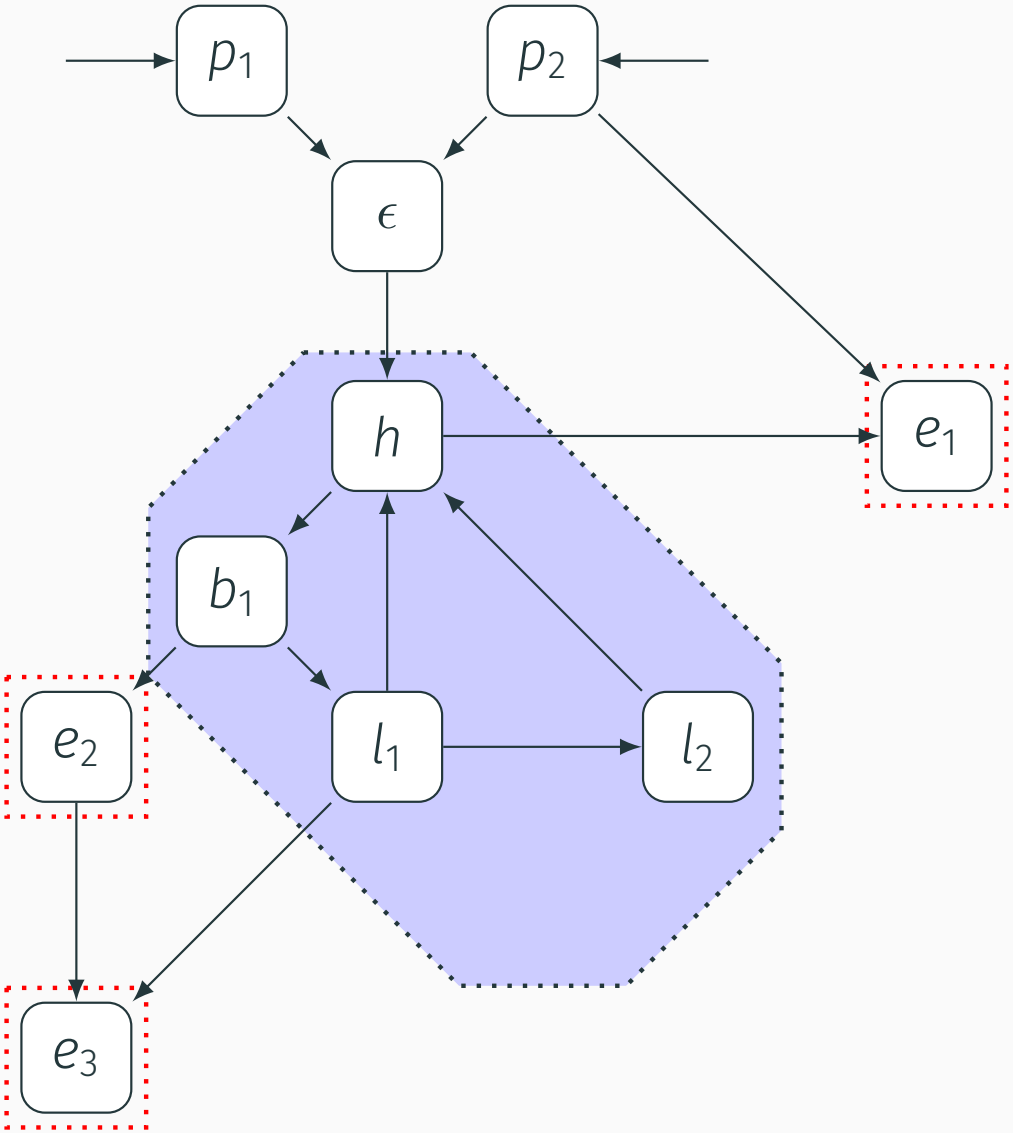
- (LCSSA) Zadeck (?), After 1991 (Probably)

Canonical Form



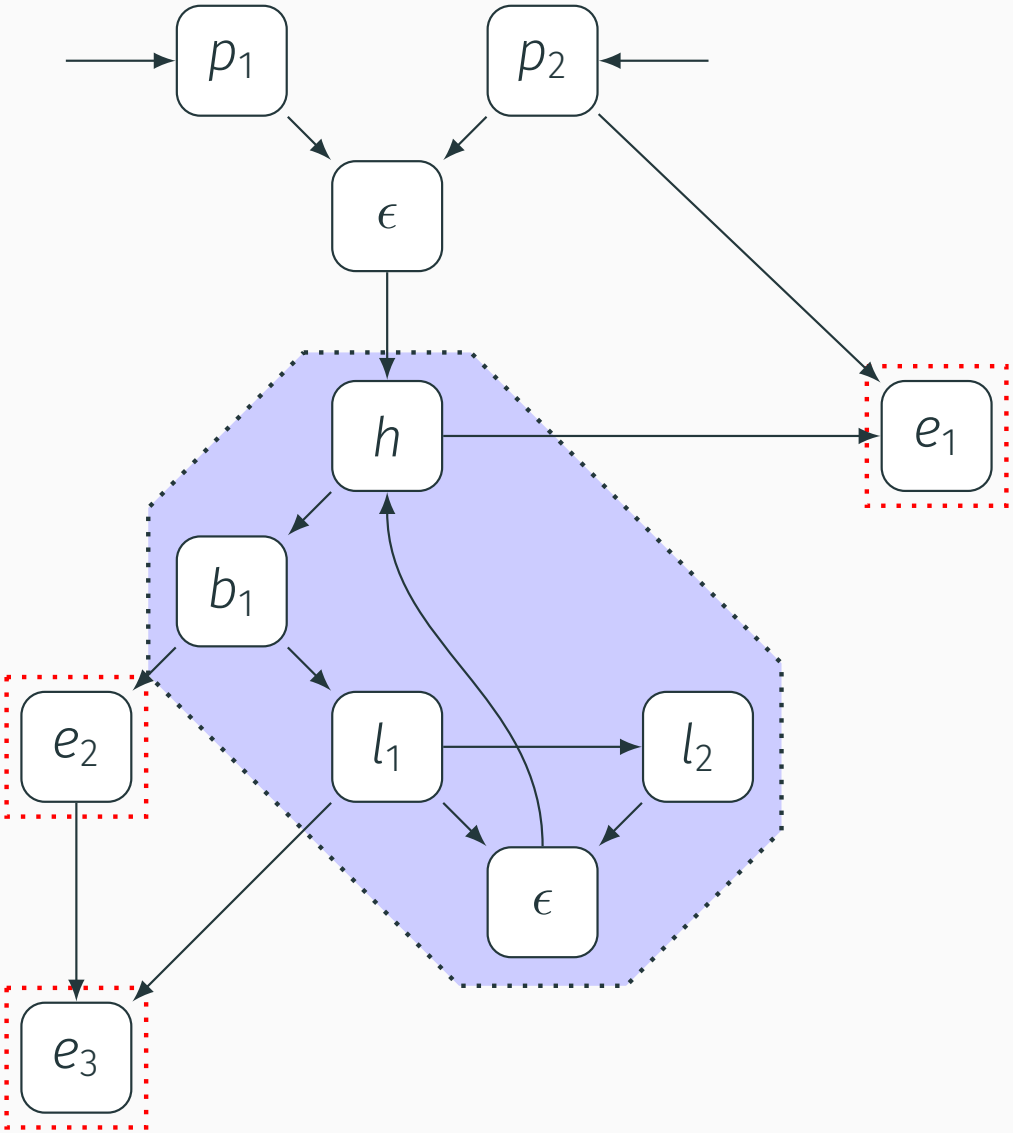
Canonical Form

(Property 1) Dedicated preheader



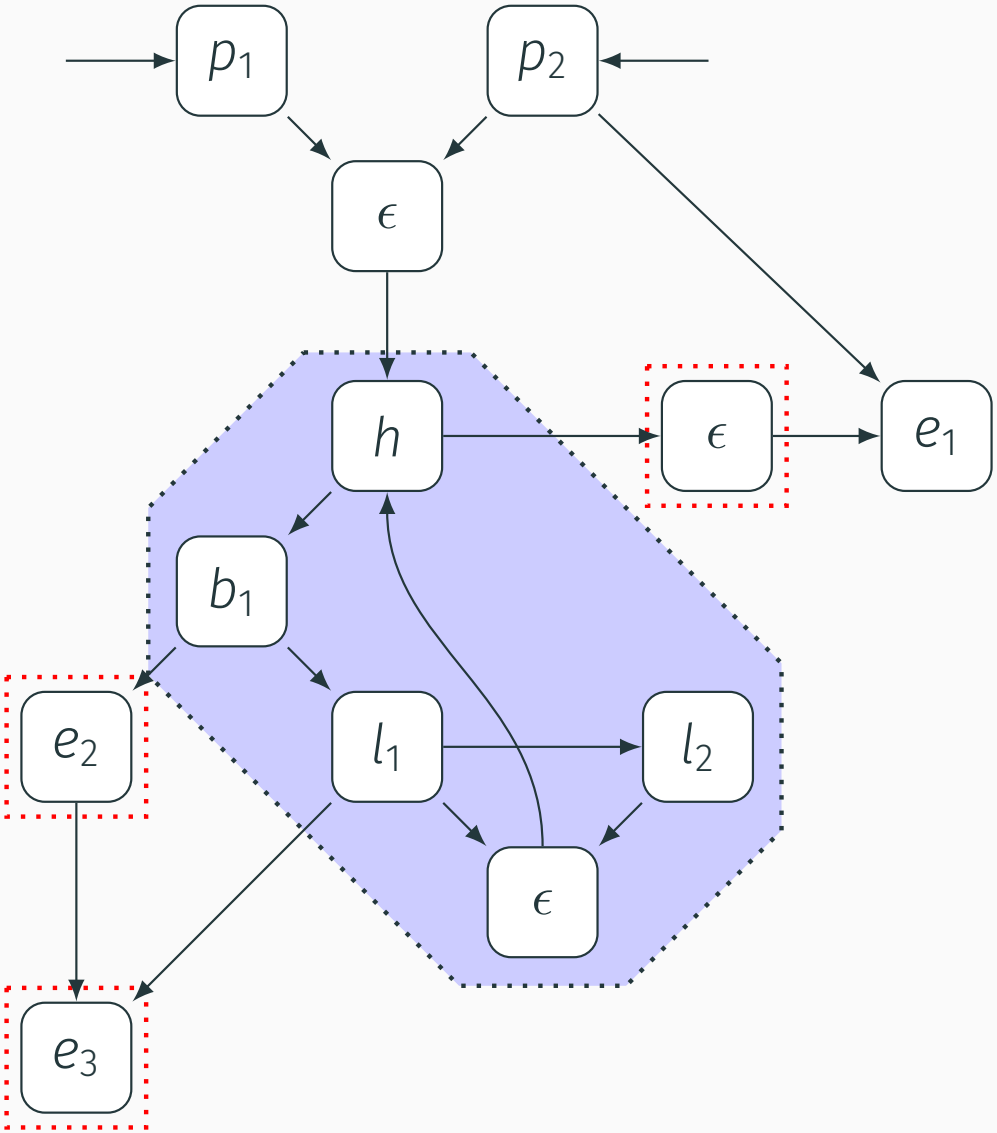
Canonical Form

(Property 2) Single Latch



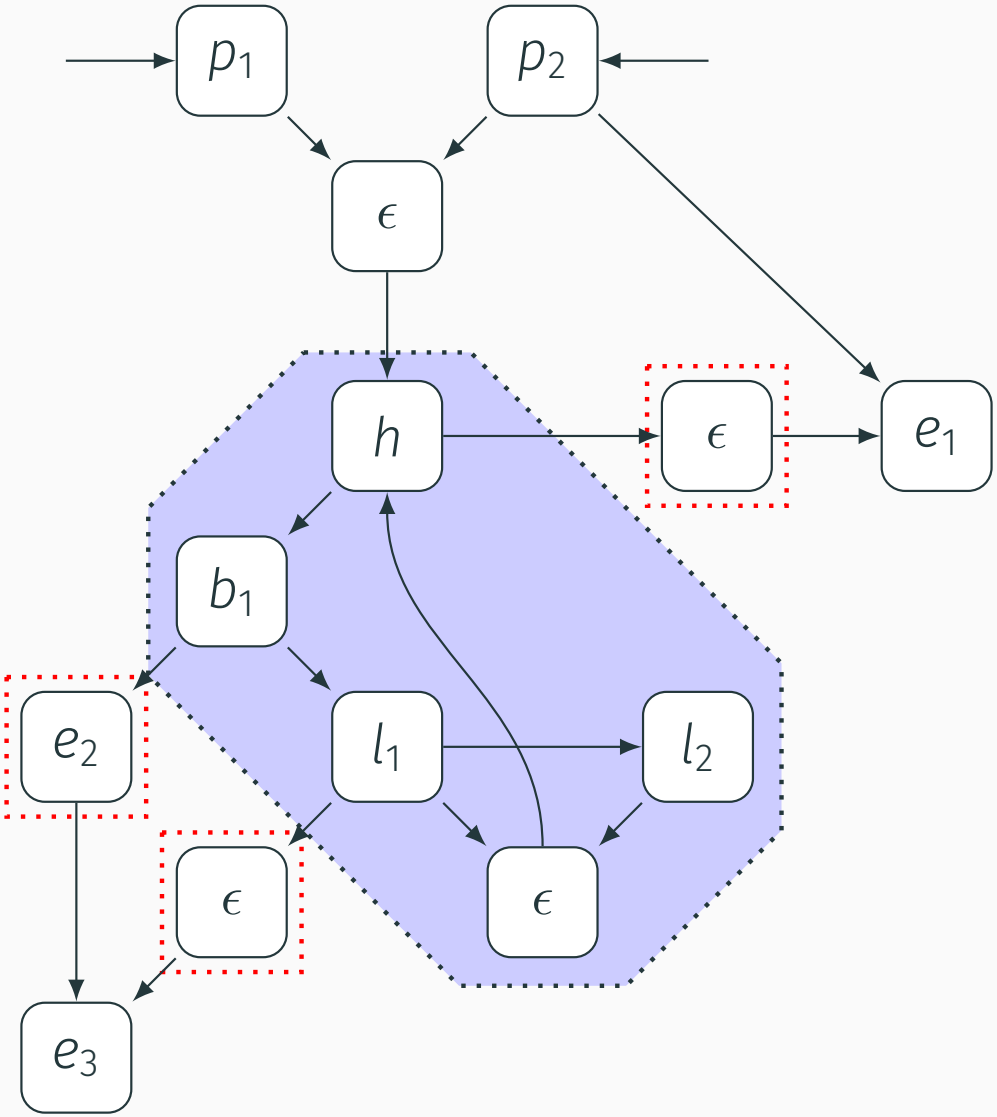
Canonical Form

(Property 3) $\forall b \in exit(L),\ pred(b) \subseteq L$



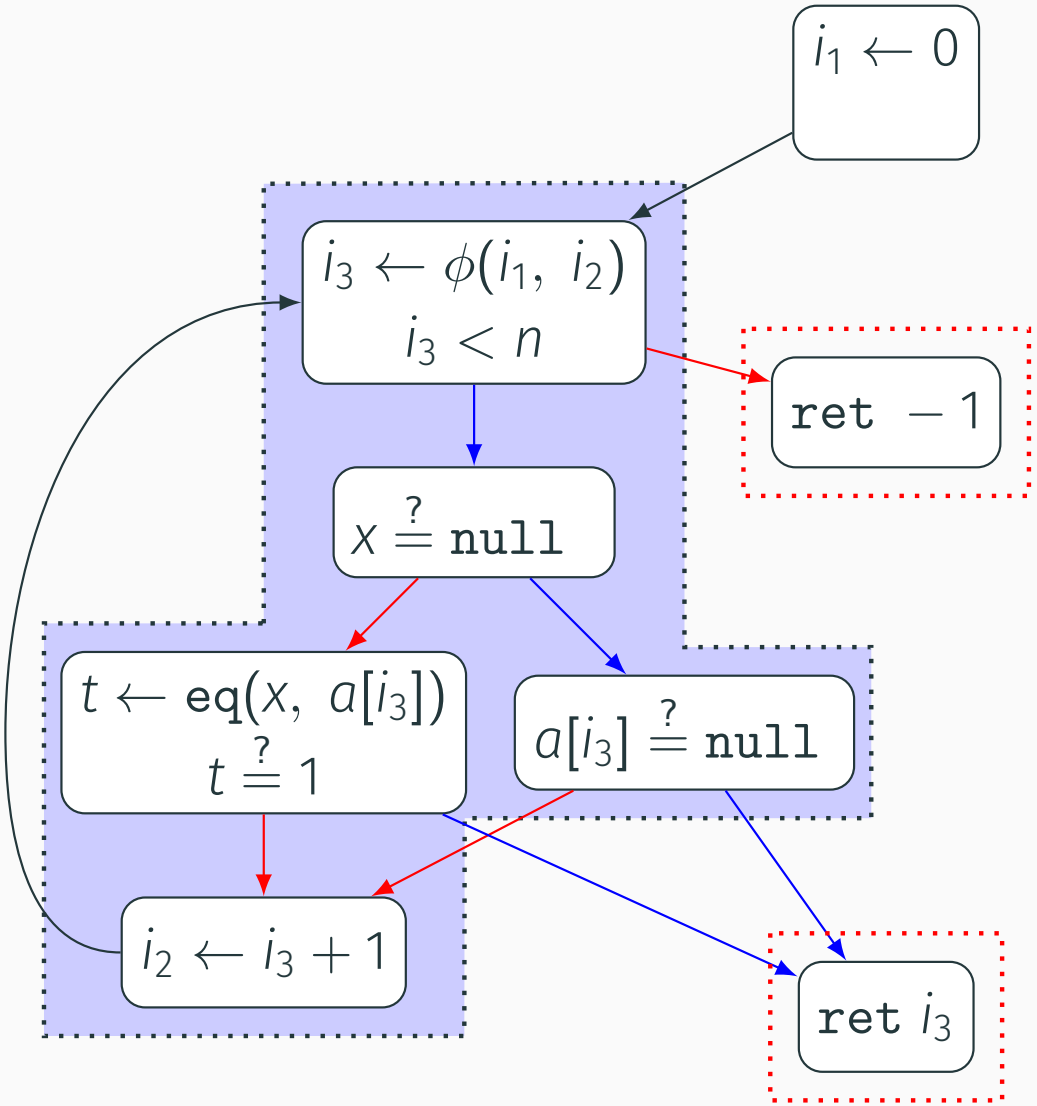
Canonical Form

(Property 3) $\forall b \in exit(L),\ pred(b) \subseteq L$

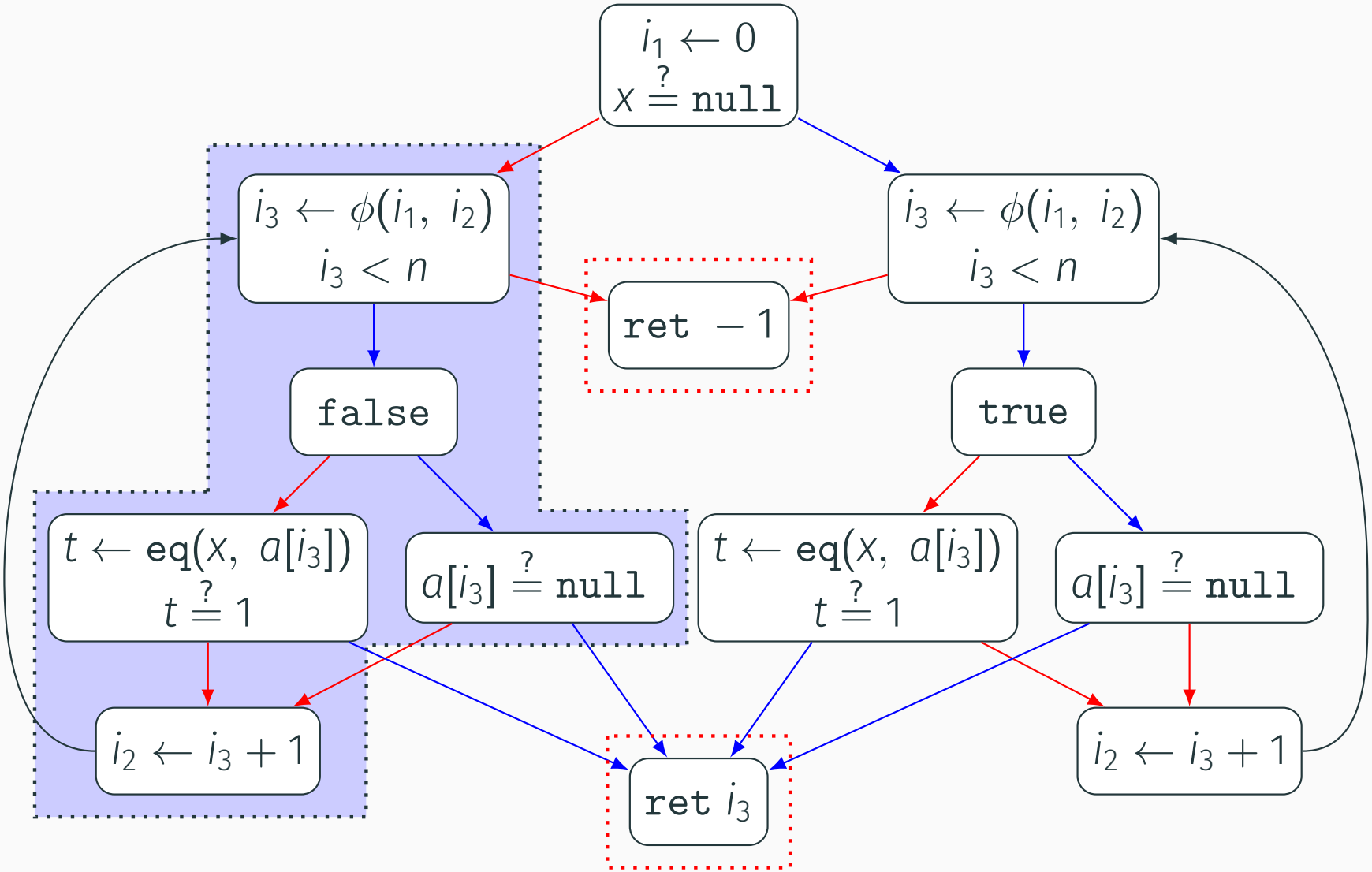


Motivation

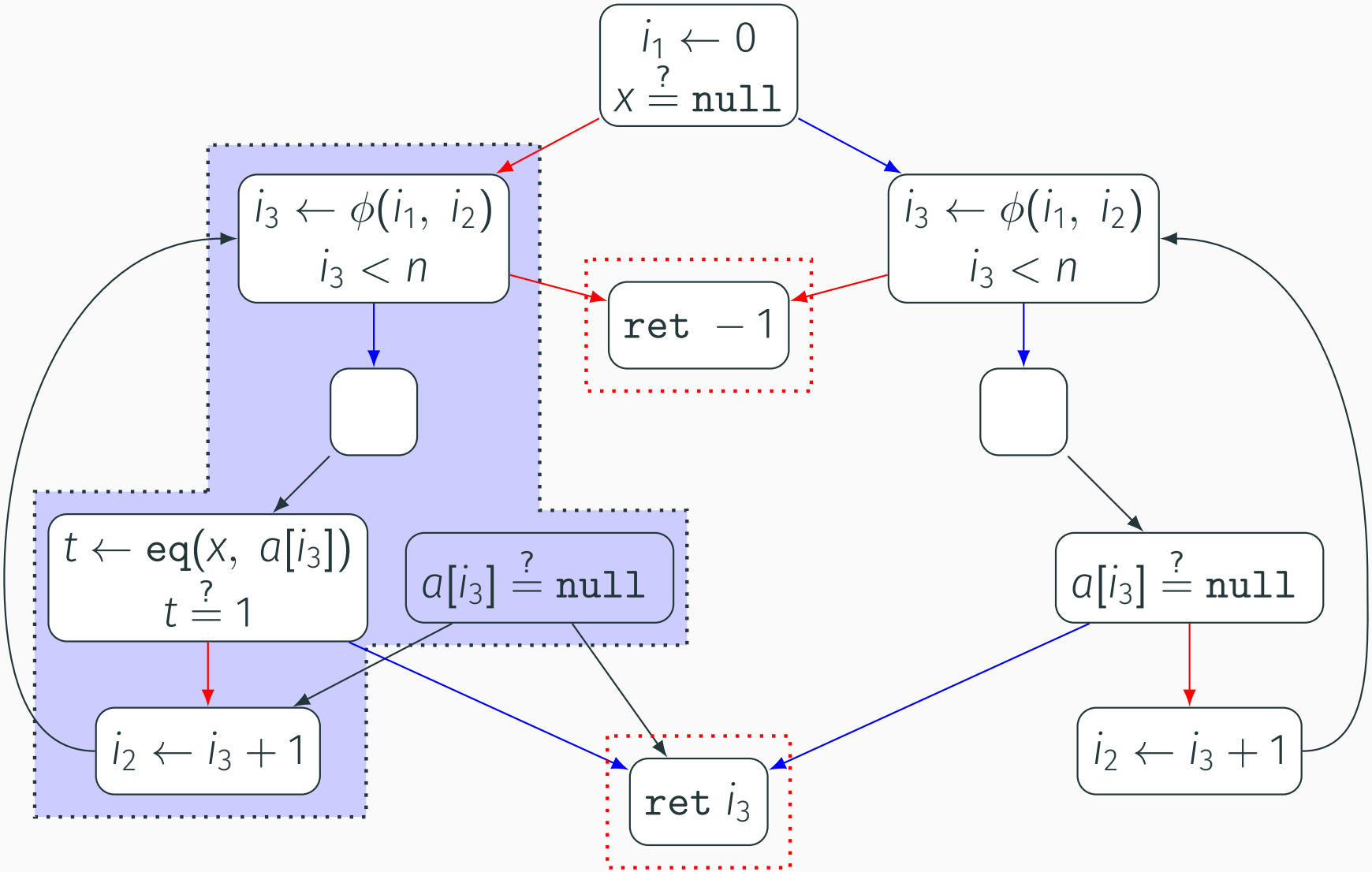
Loop Unswitching + If Simplification



Loop Unswitching + If Simplification



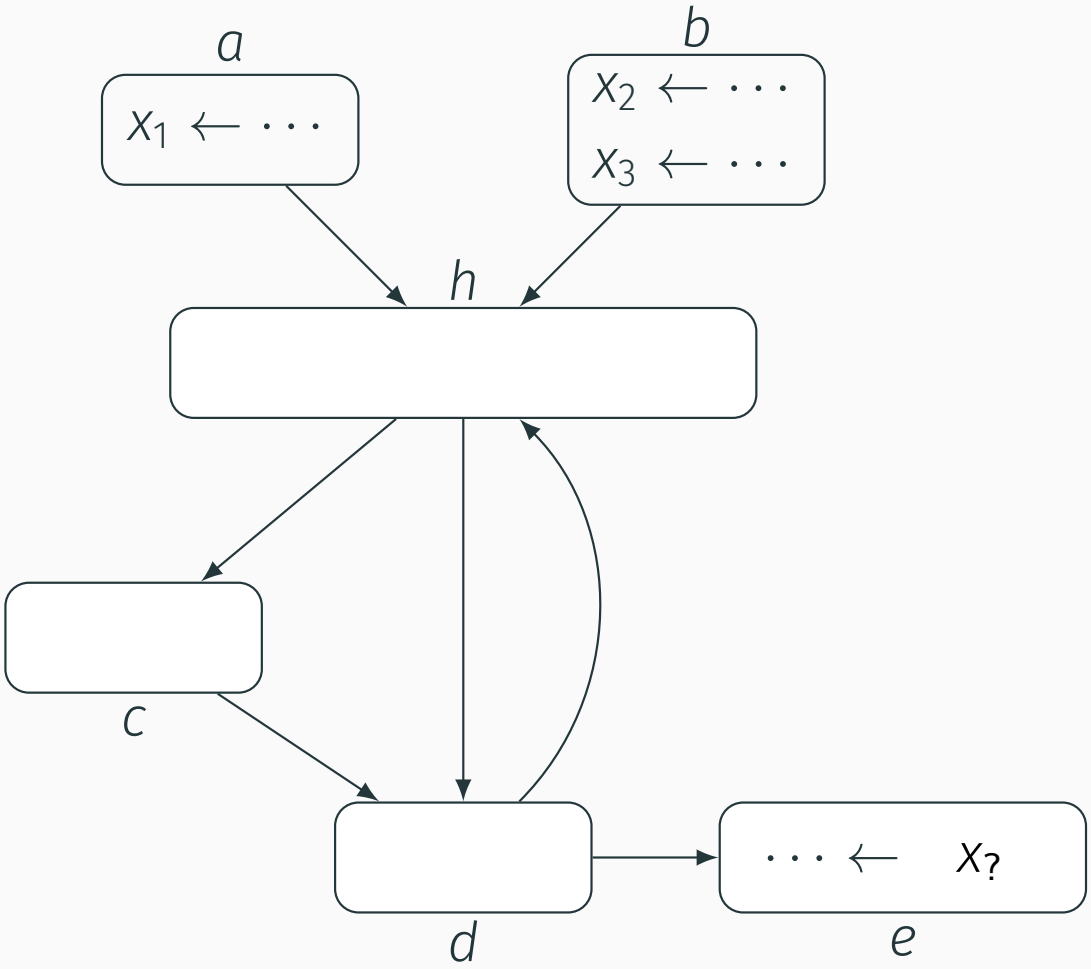
Loop Unswitching + If Simplification



SSA Reconstruction

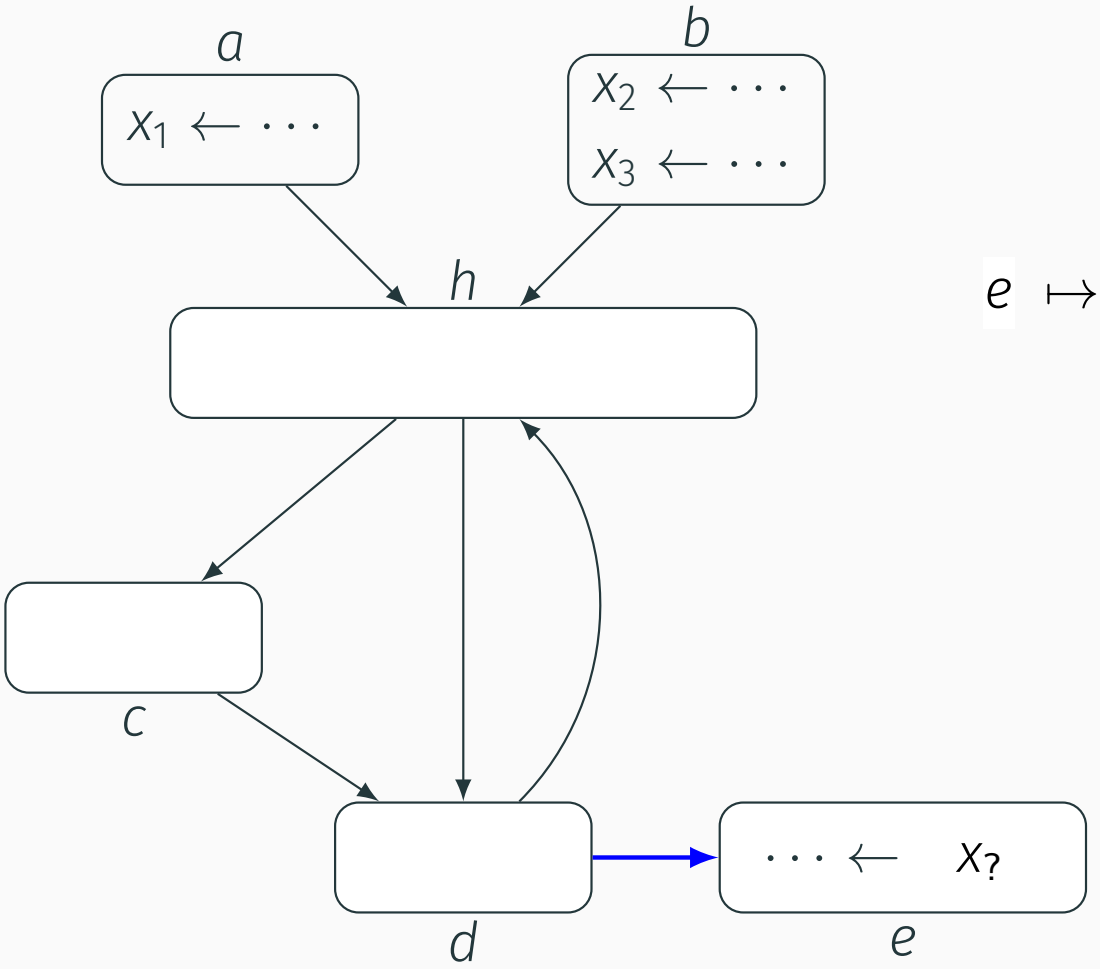
Search-Based Algorithm (Braun 2013)

Starting from a use, search $\overleftarrow{G}$ for *RD*s



Search-Based Algorithm (Braun 2013)

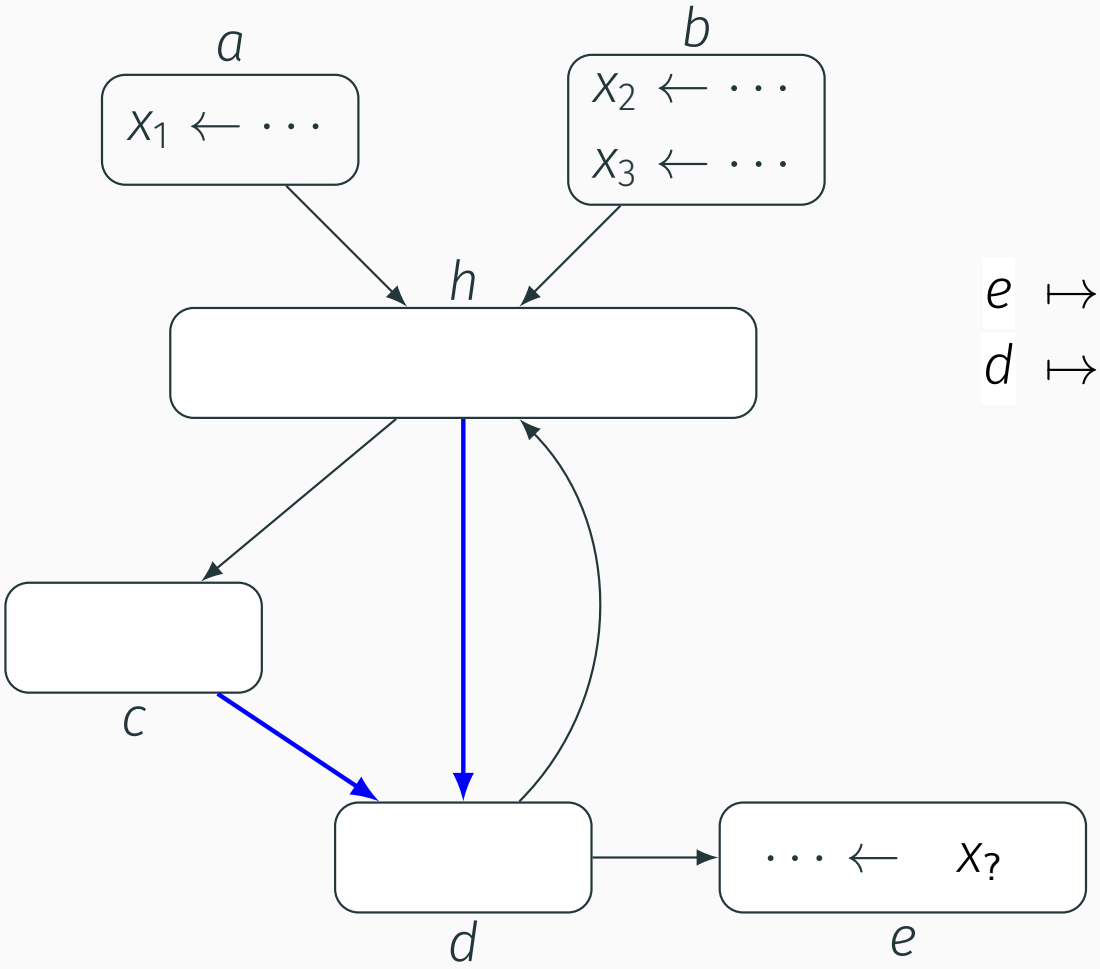
If a block does not define x, search its predecessors



$e \mapsto RD(d)$

Search-Based Algorithm (Braun 2013)

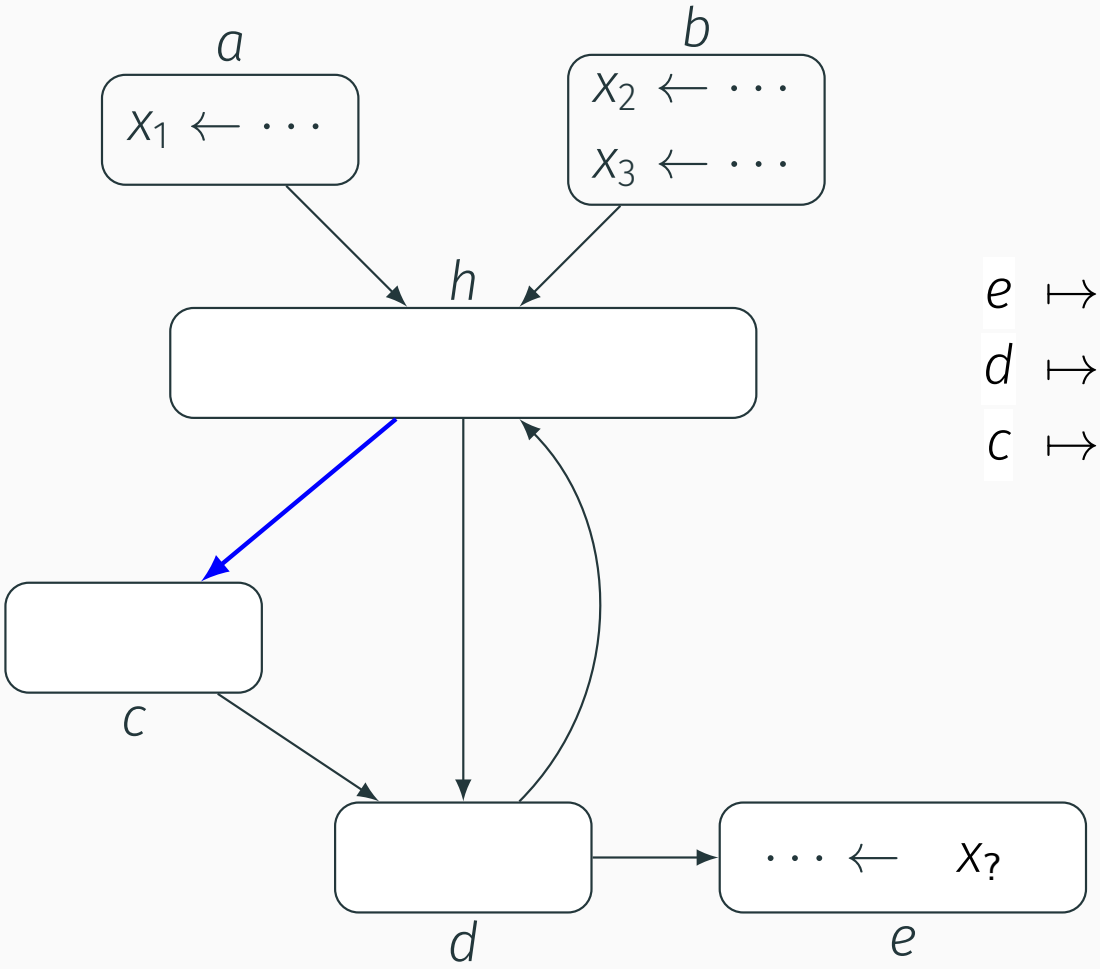
If a block has multiple predecessors, *join* result of predecessors



$e \mapsto RD(d)$

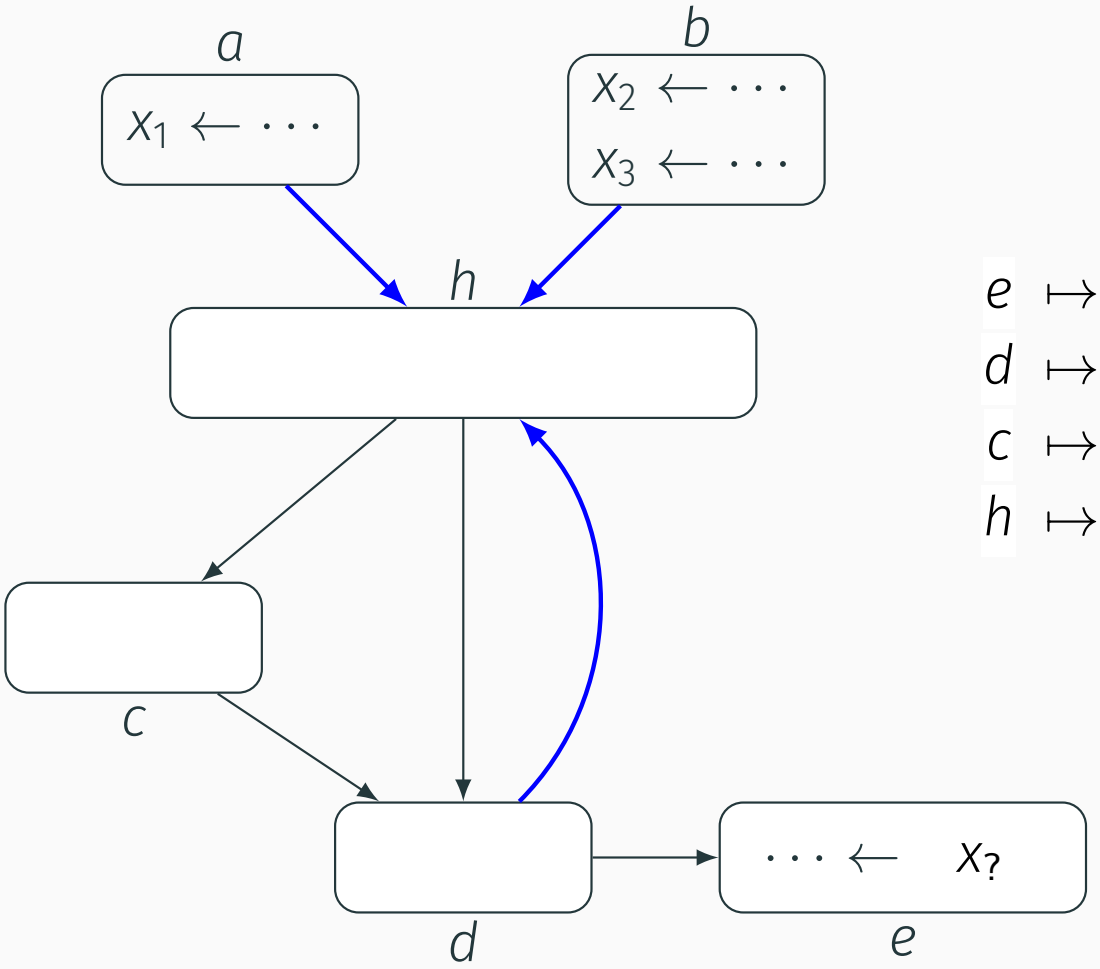
$d \mapsto RD(c) \oplus RD(h) = t_1$

Search-Based Algorithm (Braun 2013)



$$
\begin{aligned}
e &\mapsto RD(d) \\
d &\mapsto RD(c) \oplus RD(h) = t_1 \\
c &\mapsto RD(h)
\end{aligned}
$$

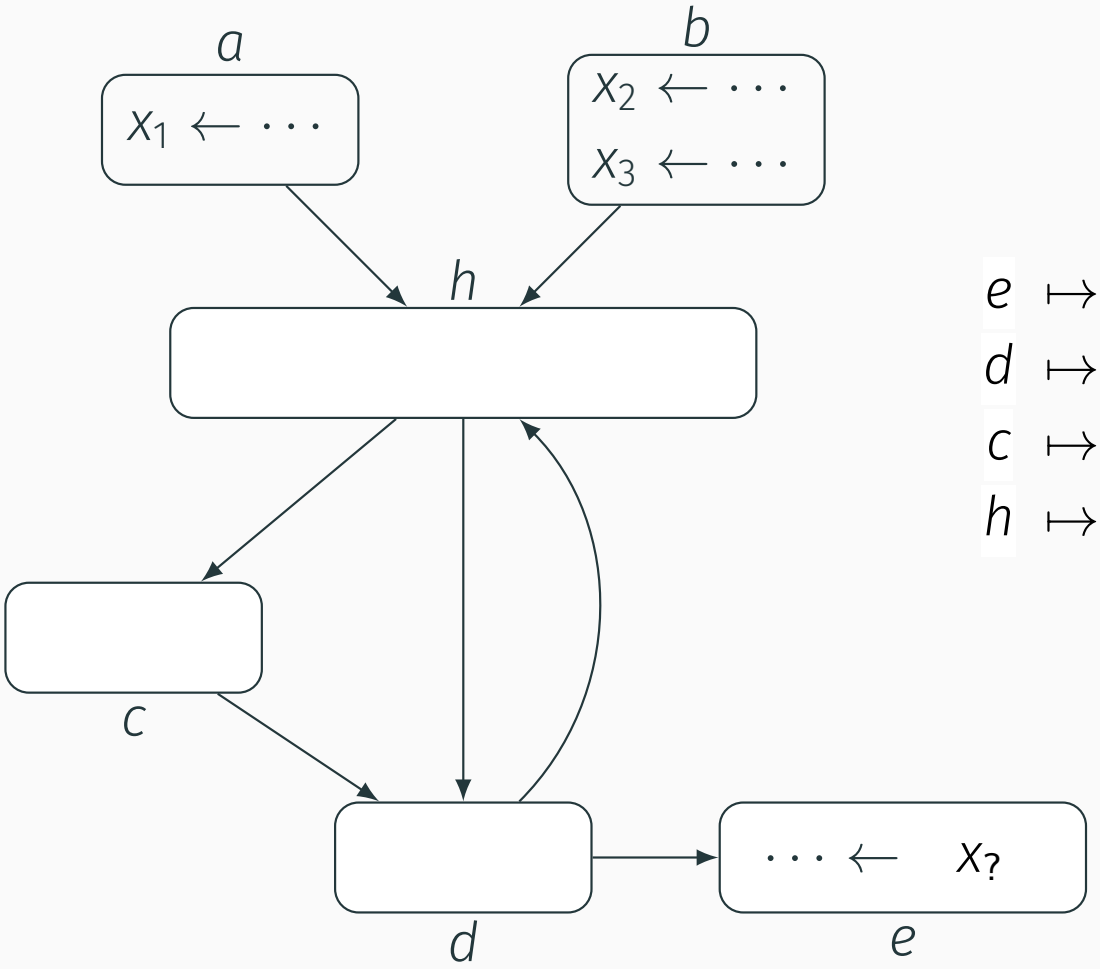
Search-Based Algorithm (Braun 2013)



$$
\begin{aligned}
e &\mapsto RD(d) \\
d &\mapsto RD(c) \oplus RD(h) = t_1 \\
c &\mapsto RD(h) \\
h &\mapsto RD(a) \oplus RD(b) \oplus RD(d) = t_2
\end{aligned}
$$

Search-Based Algorithm (Braun 2013)

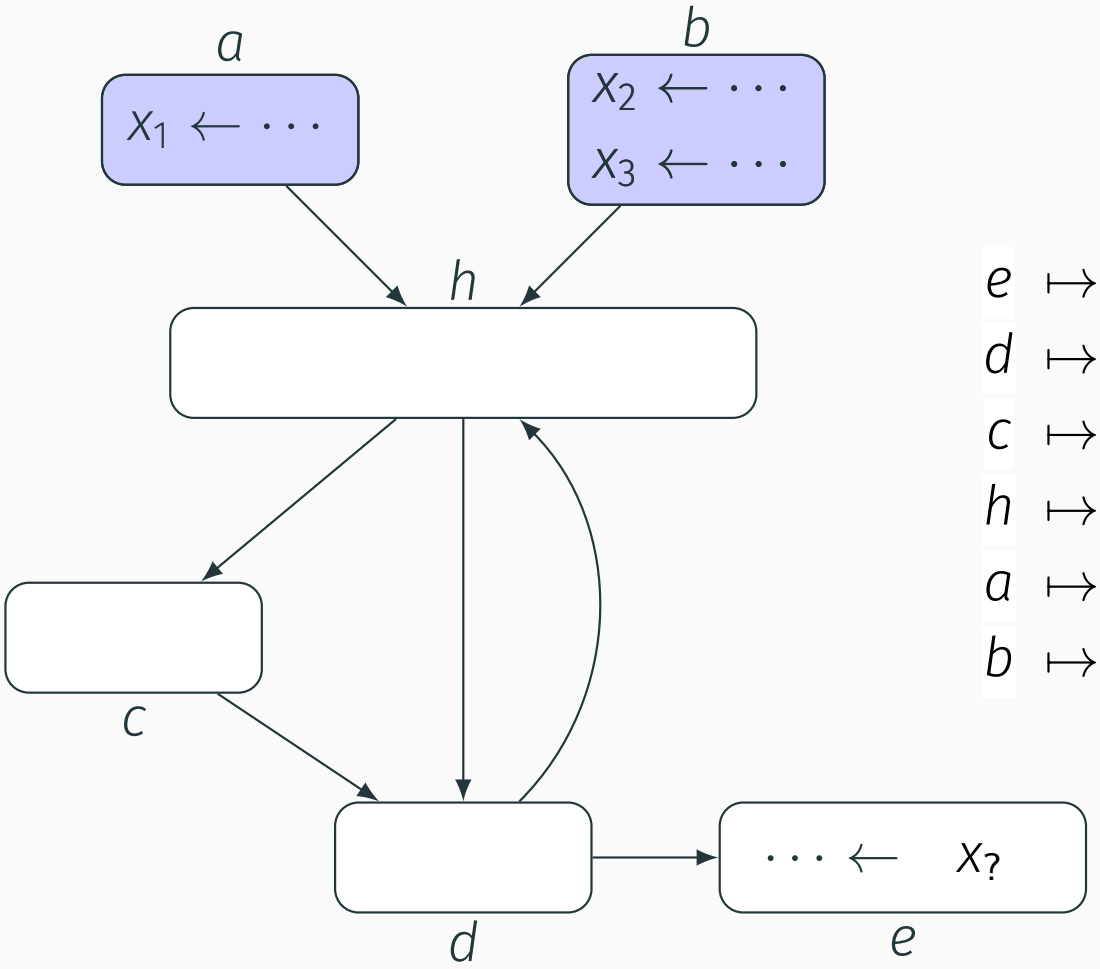
$RD(d)$ term collapses



$$
\begin{aligned}
e &\mapsto RD(d) \\
d &\mapsto RD(c) \oplus RD(h) = t_1 \\
c &\mapsto RD(h) \\
h &\mapsto RD(a) \oplus RD(b) \oplus t_1 = t_2
\end{aligned}
$$

Search-Based Algorithm (Braun 2013)

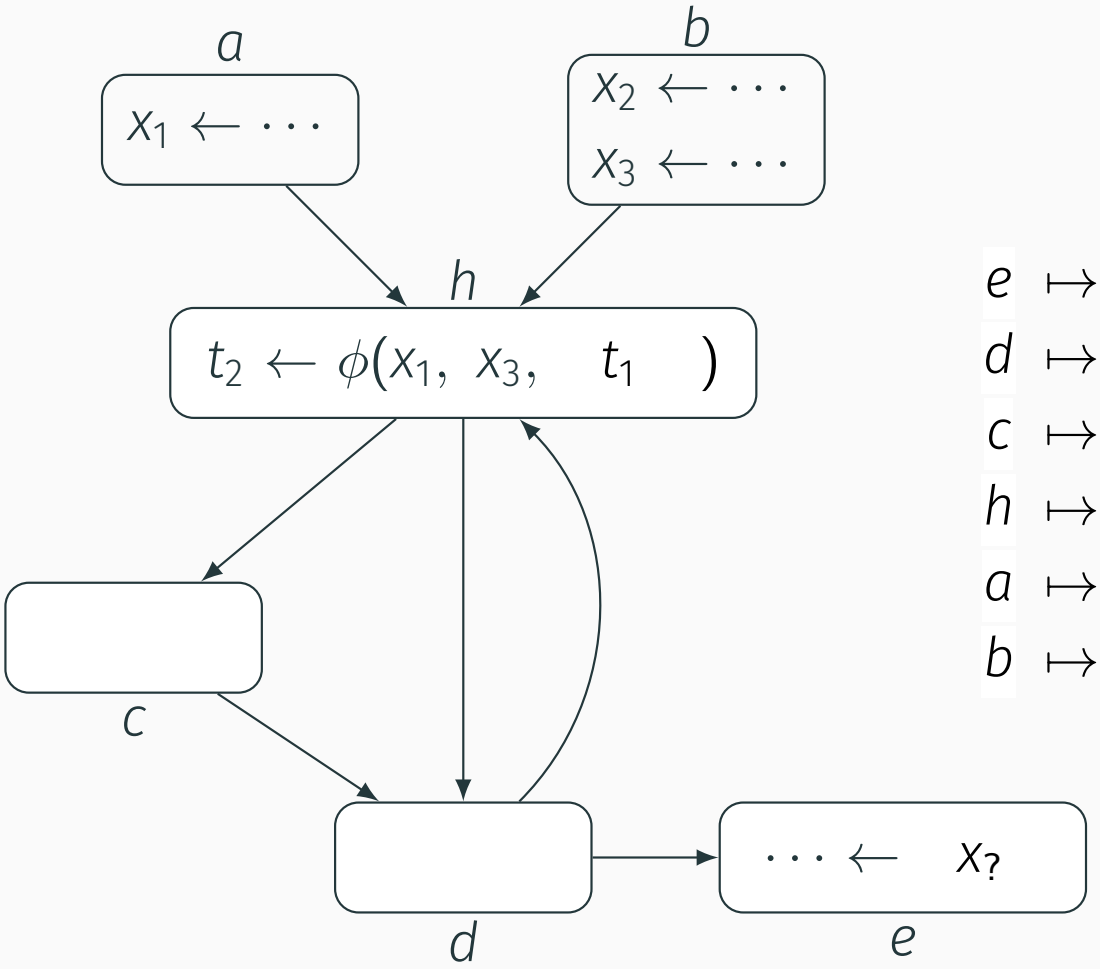
Use **last** definition of x in block



$$
\begin{aligned}
e &\mapsto RD(d) \\
d &\mapsto RD(c) \oplus RD(h) = t_1 \\
c &\mapsto RD(h) \\
h &\mapsto RD(a) \oplus RD(b) \oplus t_1 = t_2 \\
a &\mapsto x_1 \\
b &\mapsto x_3
\end{aligned}
$$

Search-Based Algorithm (Braun 2013)

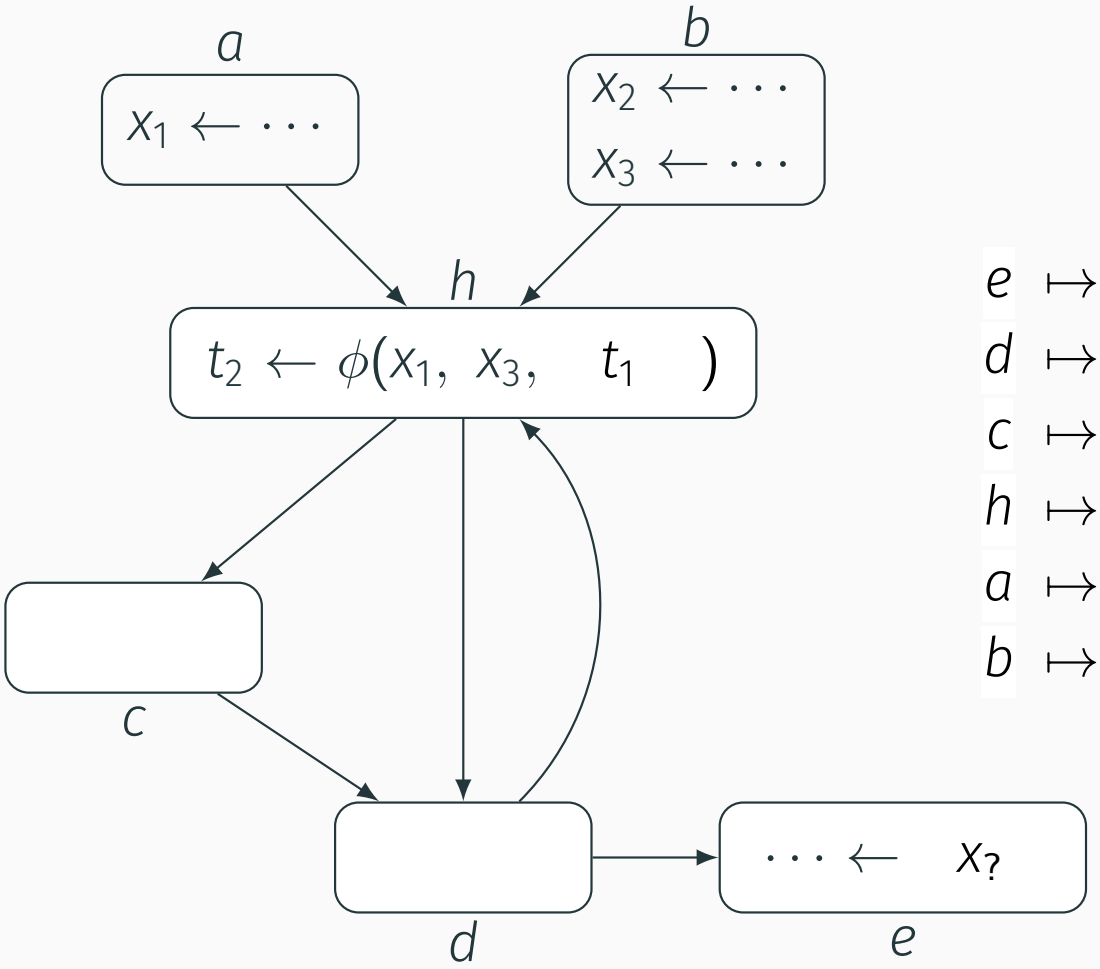
$RD(a)$ and $RD(b)$ terms collapse, $RD(h)$ has concrete ϕ-arguments



$$
\begin{aligned}
e &\mapsto RD(d) \\
d &\mapsto RD(c) \oplus RD(h) = t_1 \\
c &\mapsto RD(h) \\
h &\mapsto x_1 \oplus x_3 \oplus t_1 = t_2 \\
a &\mapsto x_1 \\
b &\mapsto x_3
\end{aligned}
$$

Search-Based Algorithm (Braun 2013)

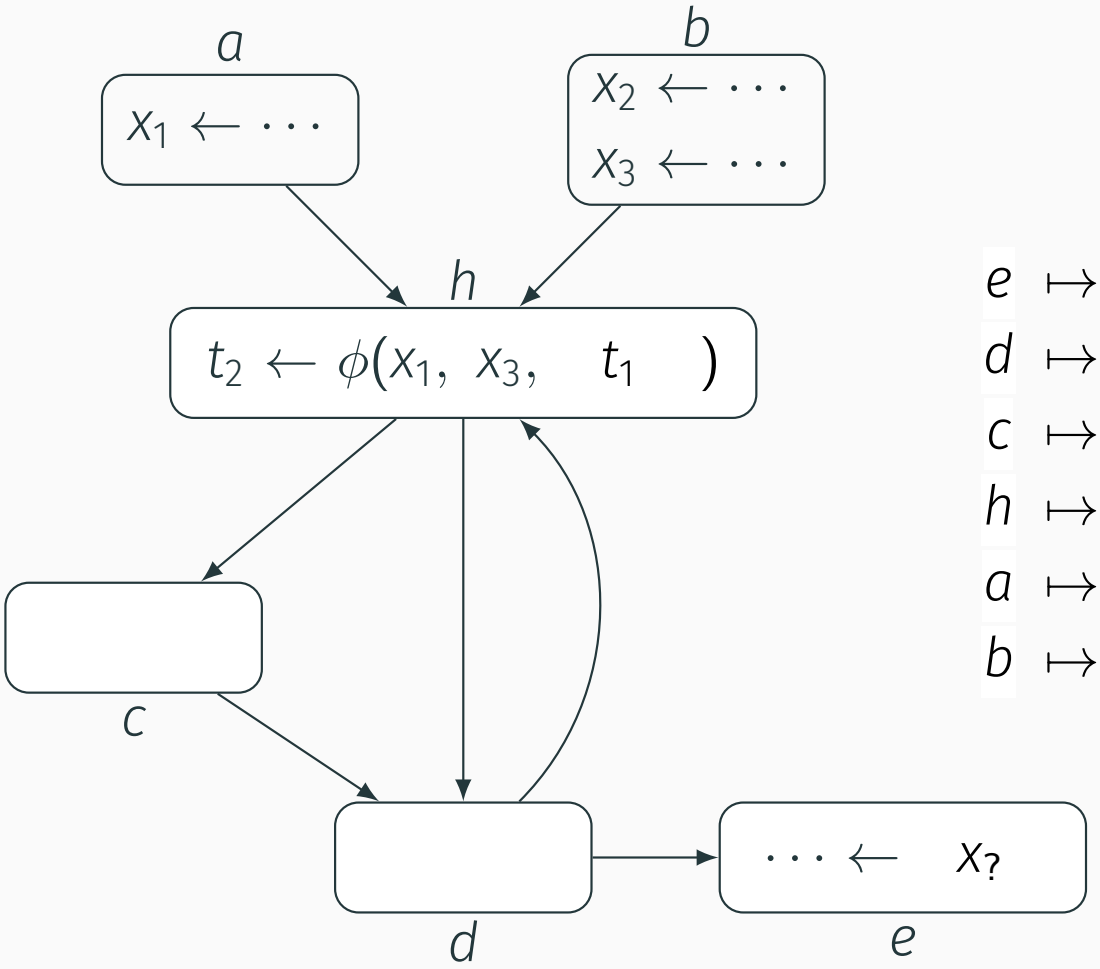
$RD(h)$ term collapses



$$
\begin{aligned}
e &\mapsto RD(d) \\
d &\mapsto RD(c) \oplus t_2 = t_1 \\
c &\mapsto t_2 \\
h &\mapsto x_1 \oplus x_3 \oplus t_1 = t_2 \\
a &\mapsto x_1 \\
b &\mapsto x_3
\end{aligned}
$$

Search-Based Algorithm (Braun 2013)

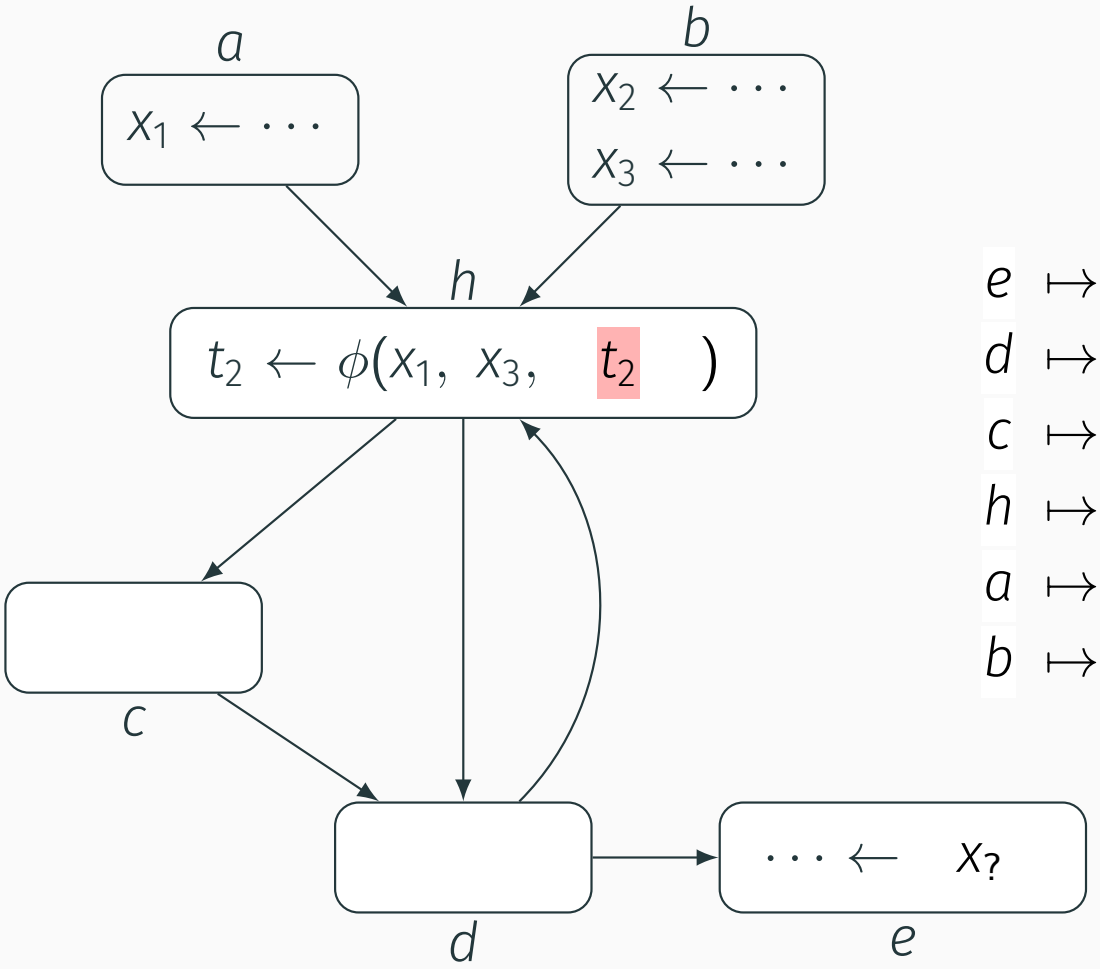
$RD(c)$ term collapses



$$
\begin{aligned}
e &\mapsto RD(d) \\
d &\mapsto t_2 \oplus t_2 = t_1 \\
c &\mapsto t_2 \\
h &\mapsto x_1 \oplus x_3 \oplus t_1 = t_2 \\
a &\mapsto x_1 \\
b &\mapsto x_3
\end{aligned}
$$

Search-Based Algorithm (Braun 2013)

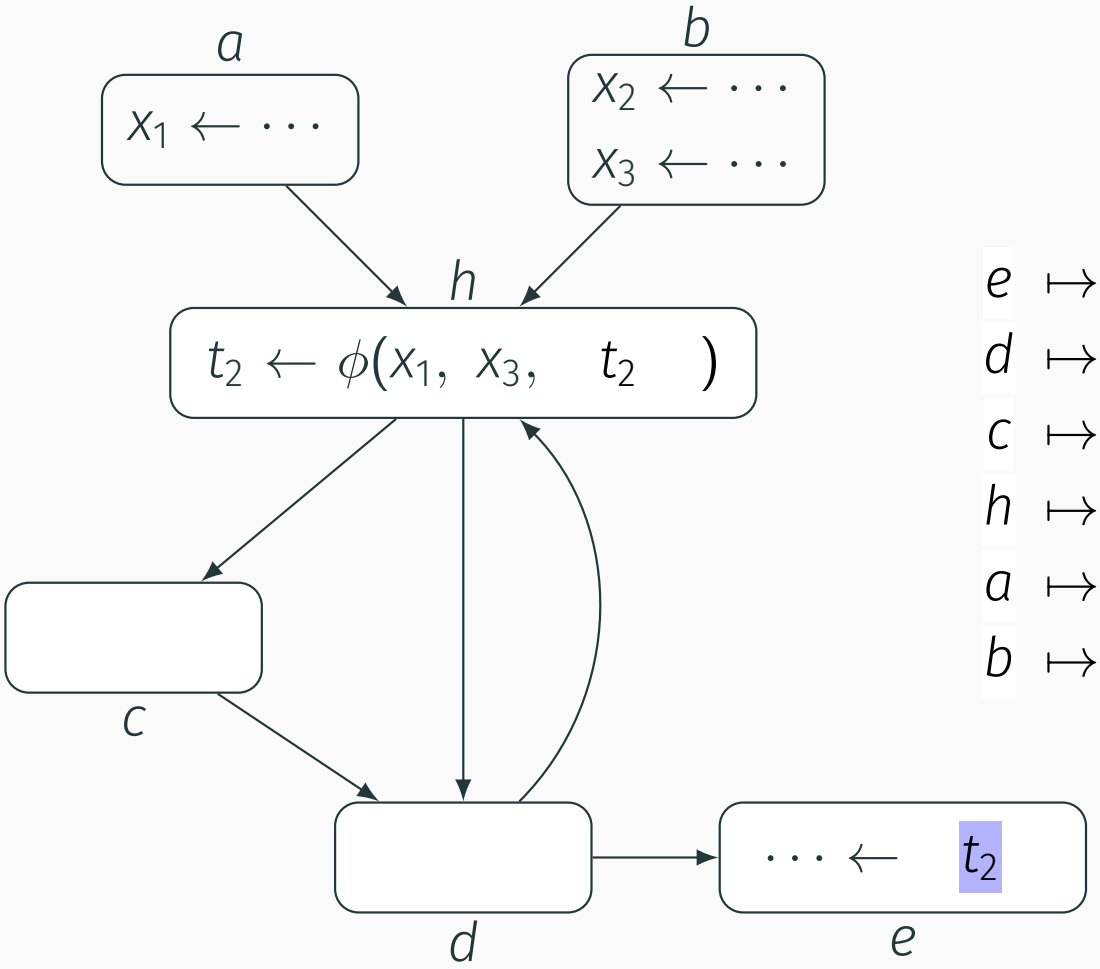
$t_1 = \phi(t_2,\ t_2)$ is trivially unnecessary



$$
\begin{aligned}
e &\mapsto RD(d) \\
d &\mapsto t_2 \oplus t_2 = t_2 \\
c &\mapsto t_2 \\
h &\mapsto x_1 \oplus x_3 \oplus t_2 = t_2 \\
a &\mapsto x_1 \\
b &\mapsto x_3
\end{aligned}
$$

Search-Based Algorithm (Braun 2013)

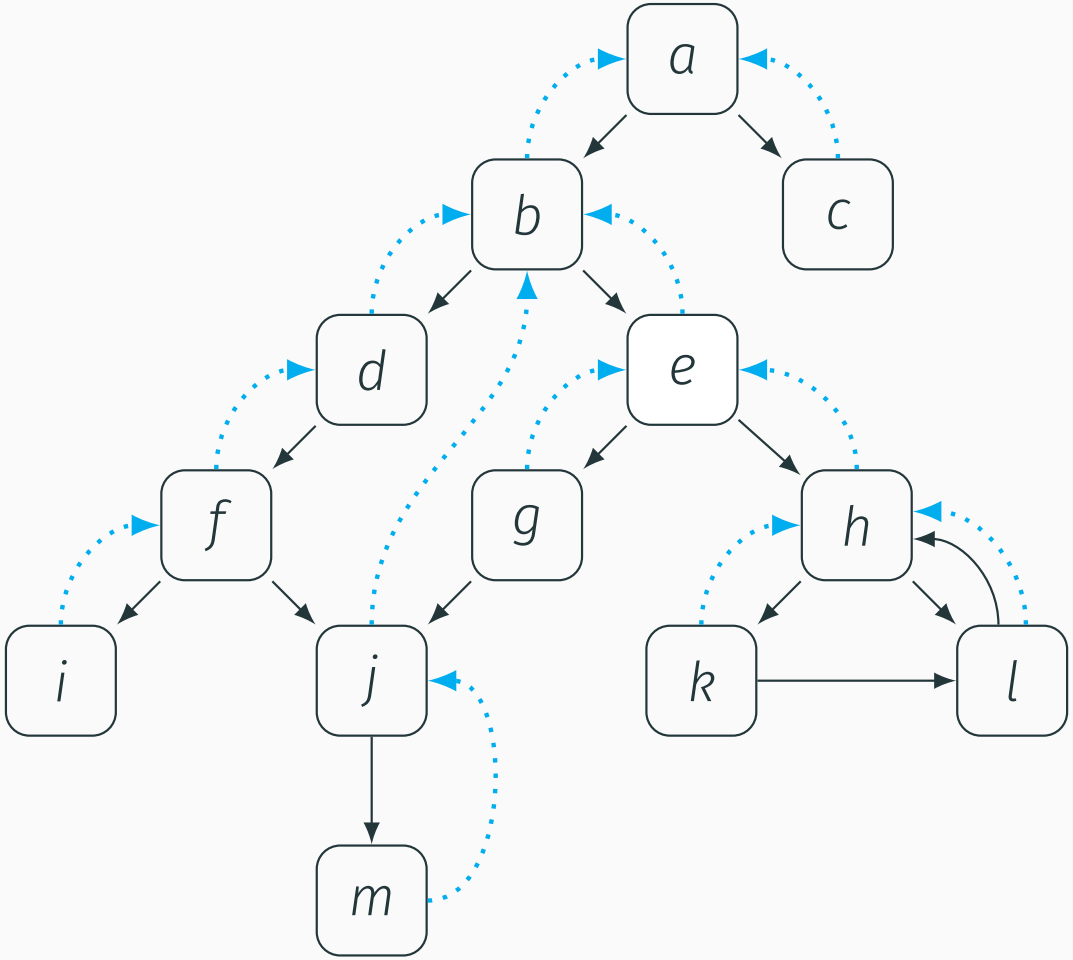
$RD(d)$ term collapses, $RD(e)$ has a solution



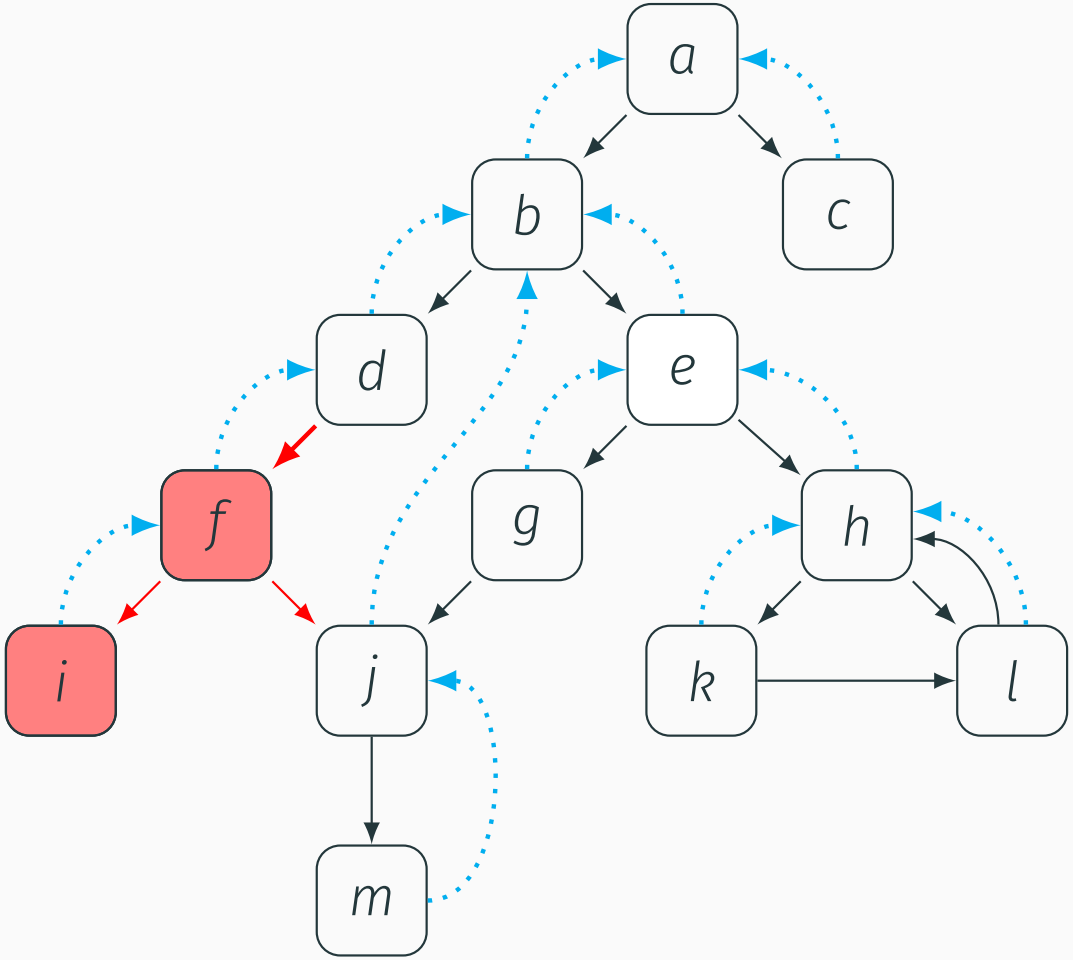
$$
\begin{aligned}
e &\mapsto t_2 \\
d &\mapsto t_2 \oplus t_2 = t_2 \\
c &\mapsto t_2 \\
h &\mapsto x_1 \oplus x_3 \oplus t_2 = t_2 \\
a &\mapsto x_1 \\
b &\mapsto x_3
\end{aligned}
$$

Dominator Tree Reconstruction

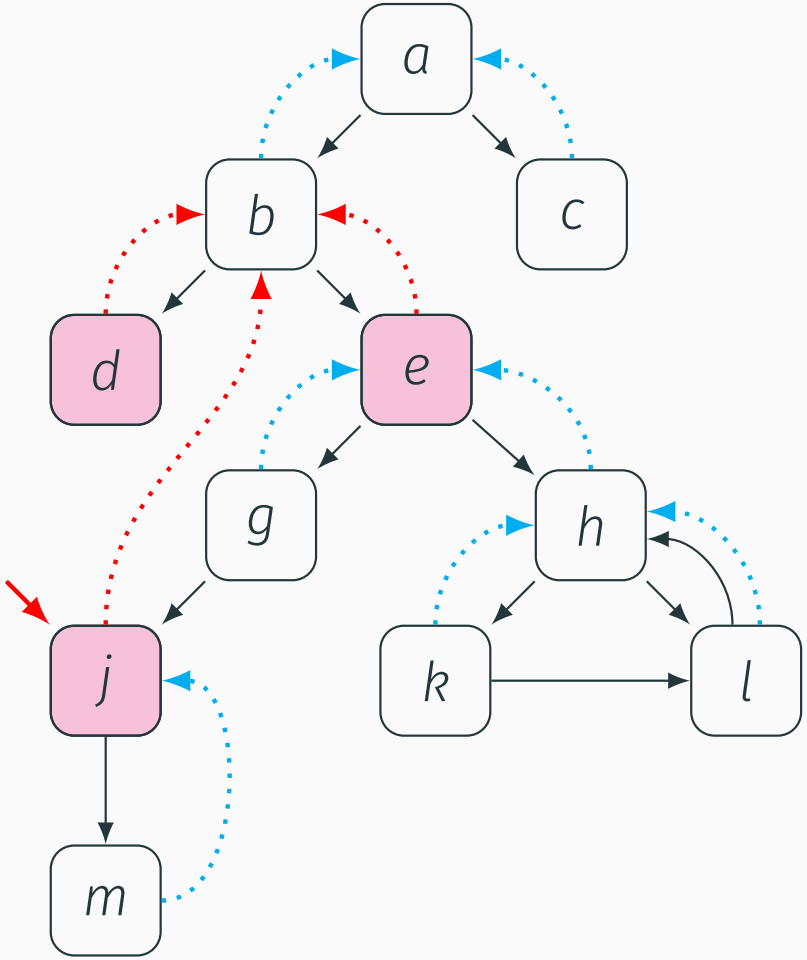
Edge Deletion (Ramalingam & Reps 1994)



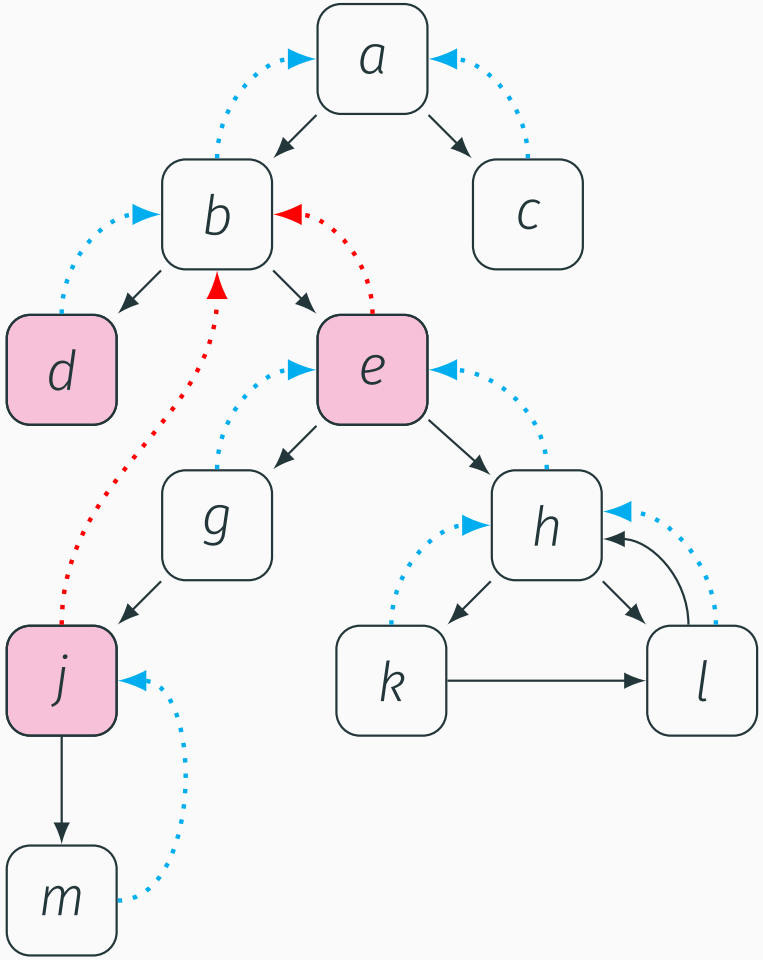
Edge Deletion (Ramalingam & Reps 1994)



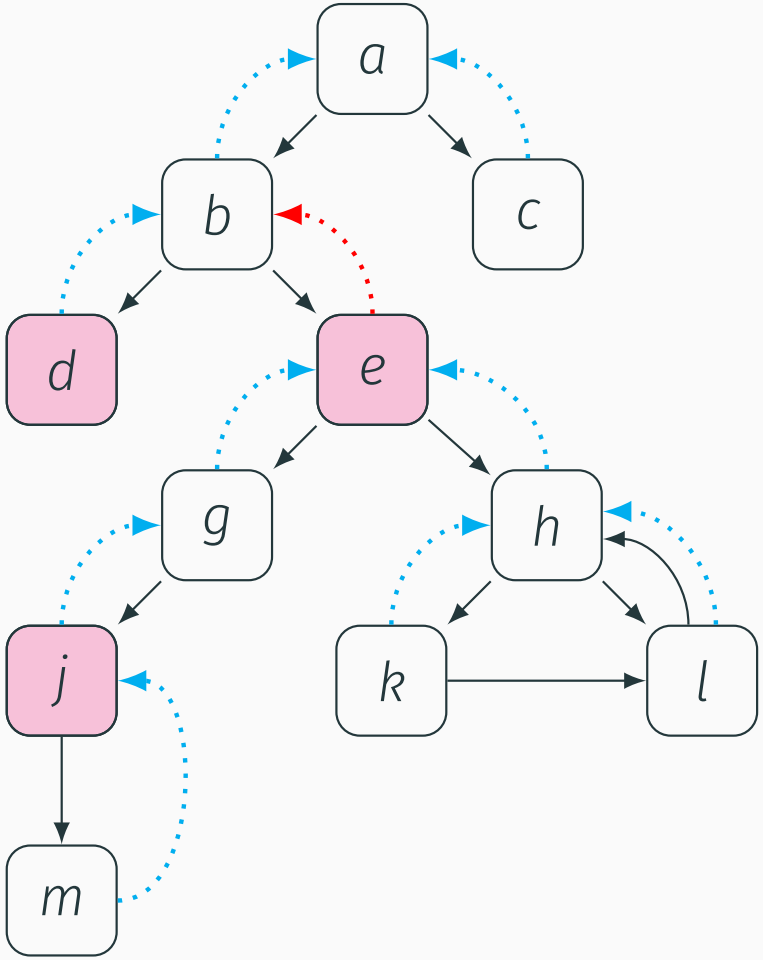
Edge Deletion (Ramalingam & Reps 1994)



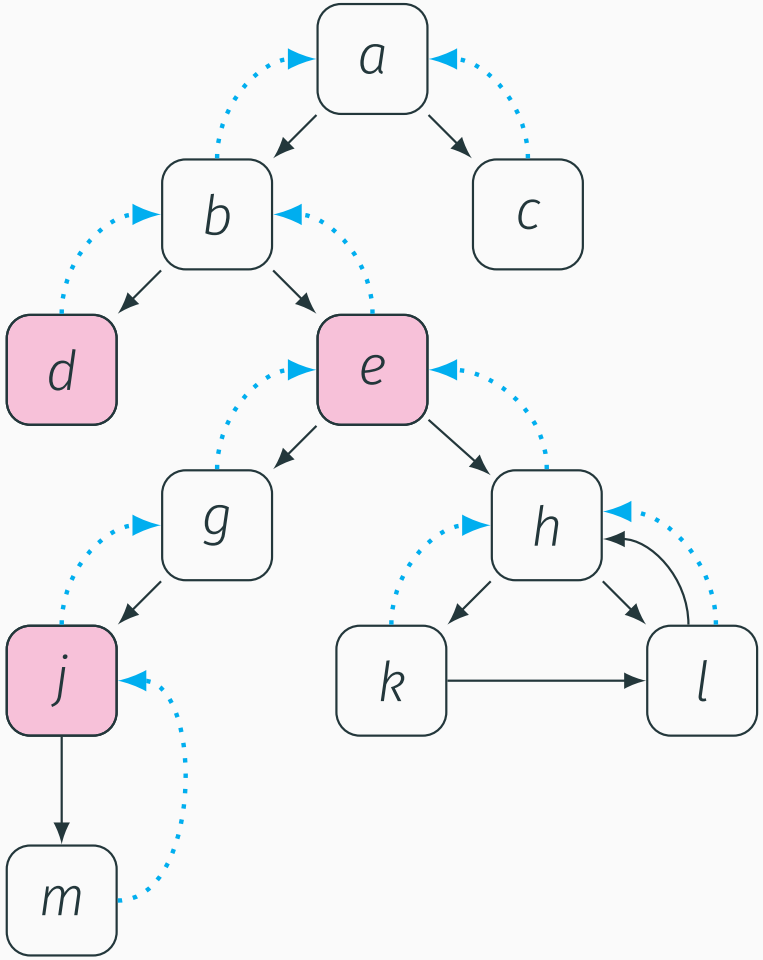
Edge Deletion (Ramalingam & Reps 1994)



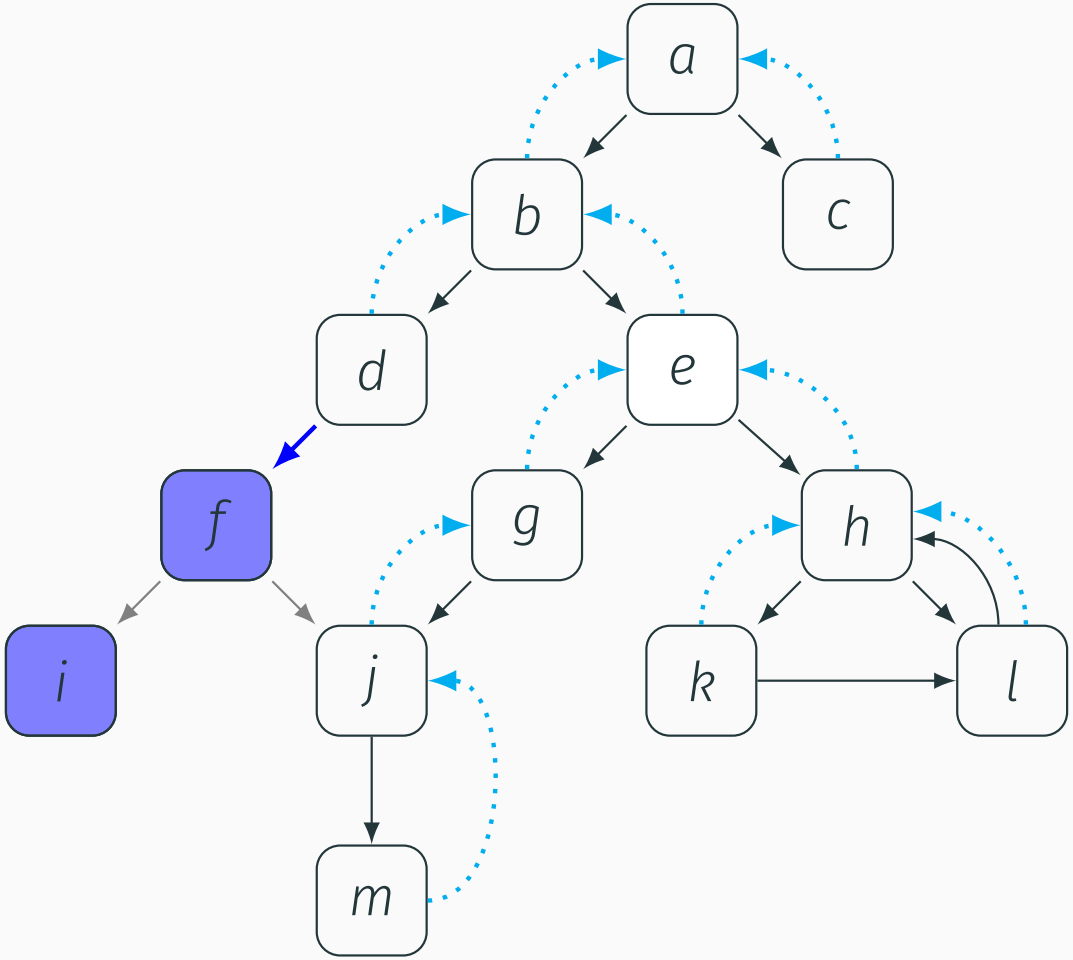
Edge Deletion (Ramalingam & Reps 1994)



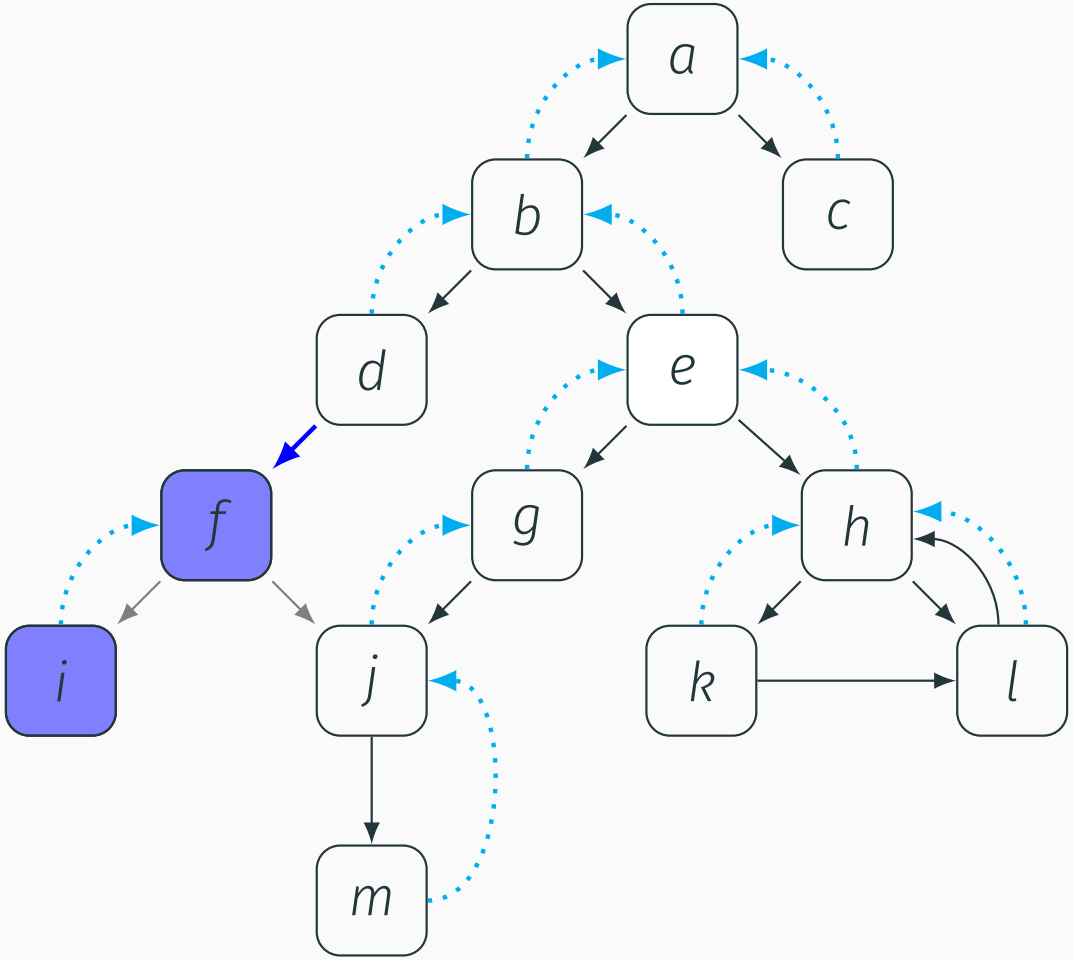
Edge Deletion (Ramalingam & Reps 1994)



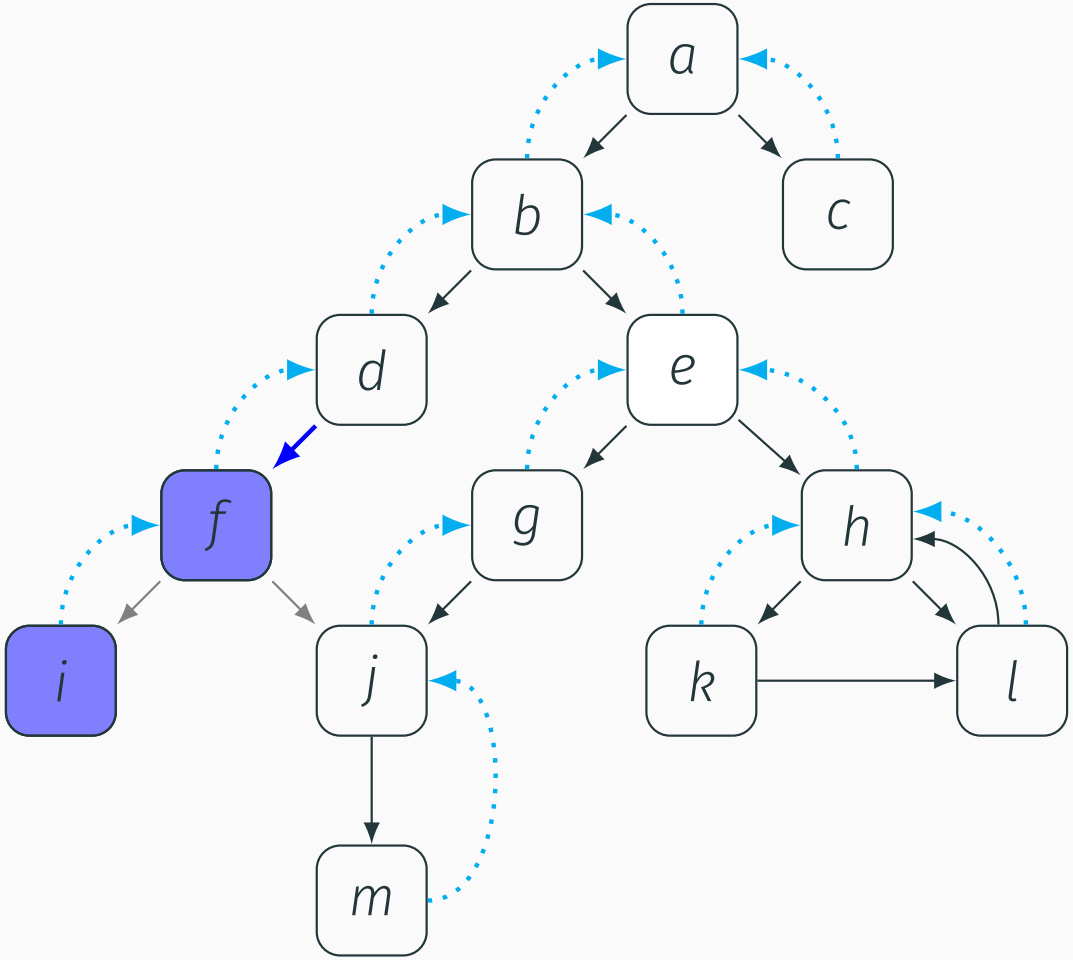
Edge Insertion (Ramalingam & Reps 1994)



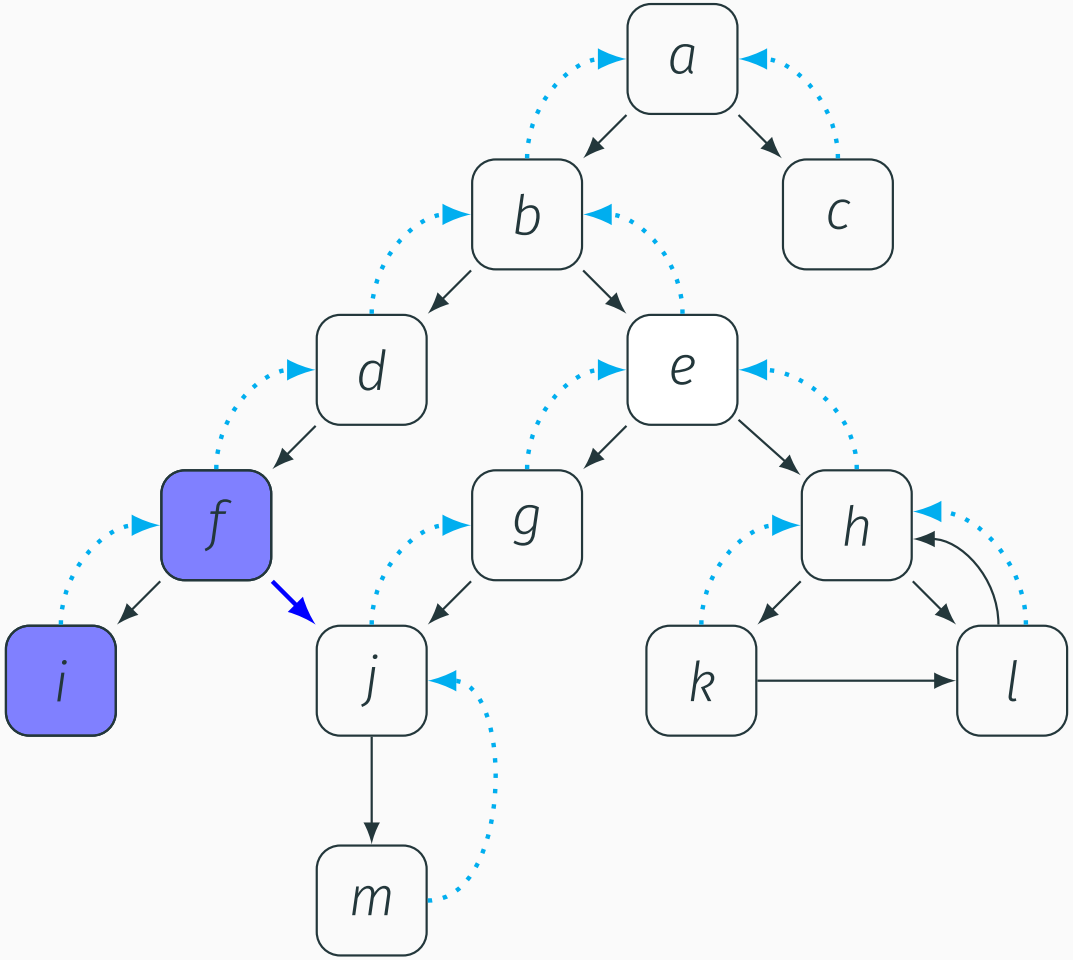
Edge Insertion (Ramalingam & Reps 1994)



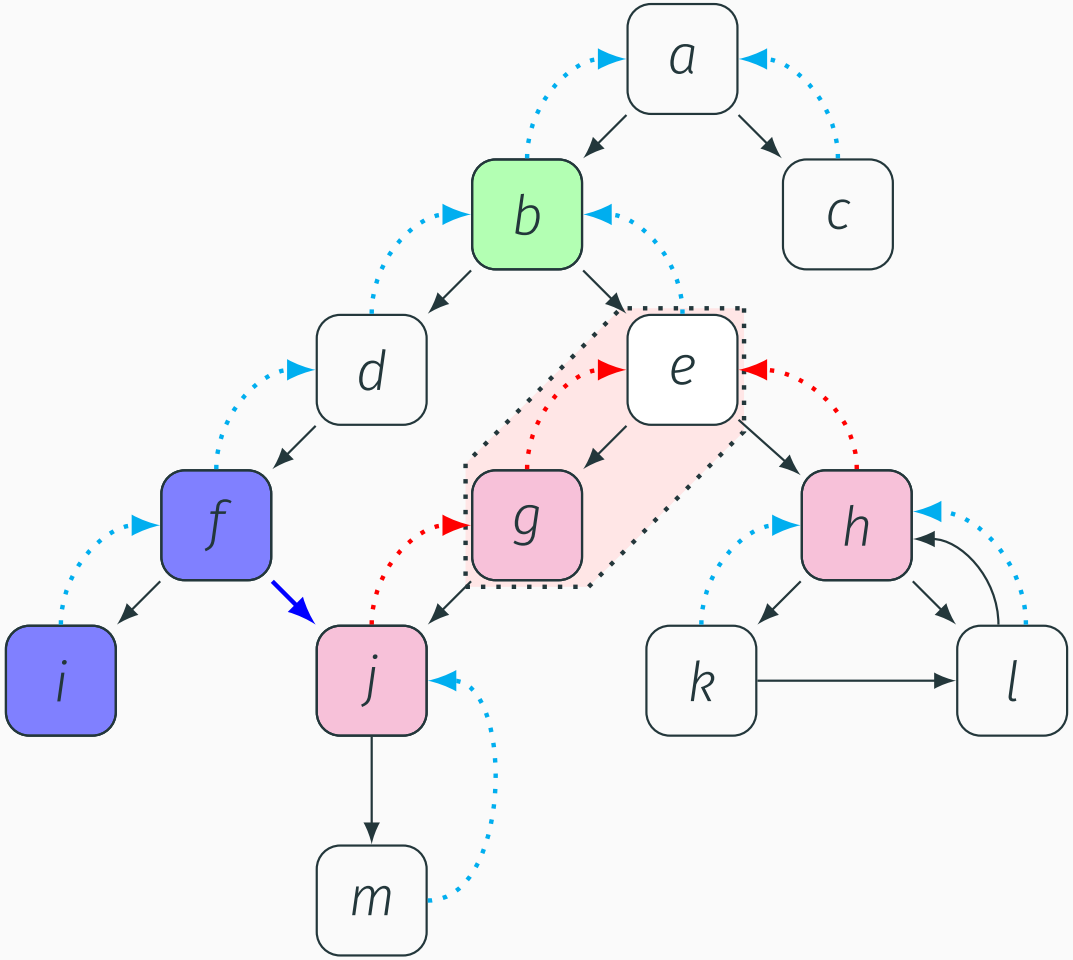
Edge Insertion (Ramalingam & Reps 1994)



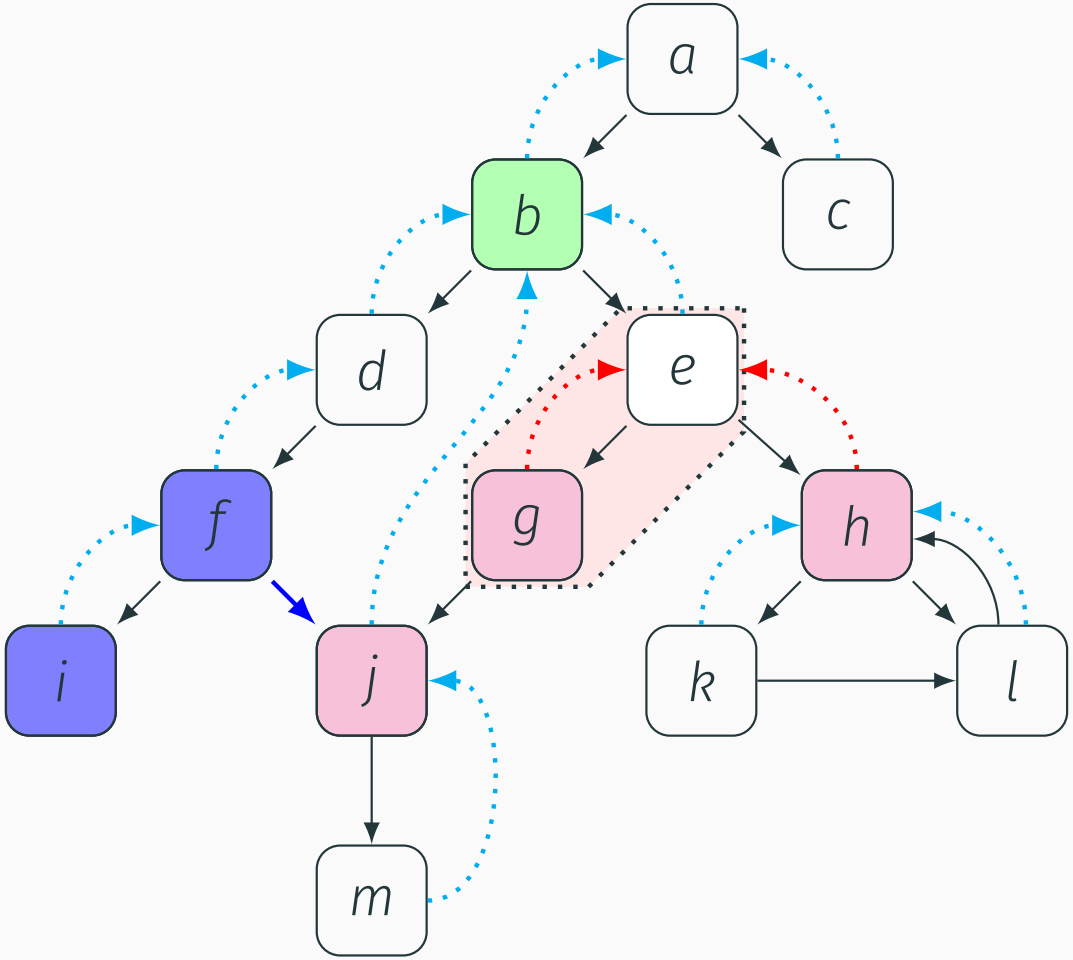
Edge Insertion (Ramalingam & Reps 1994)



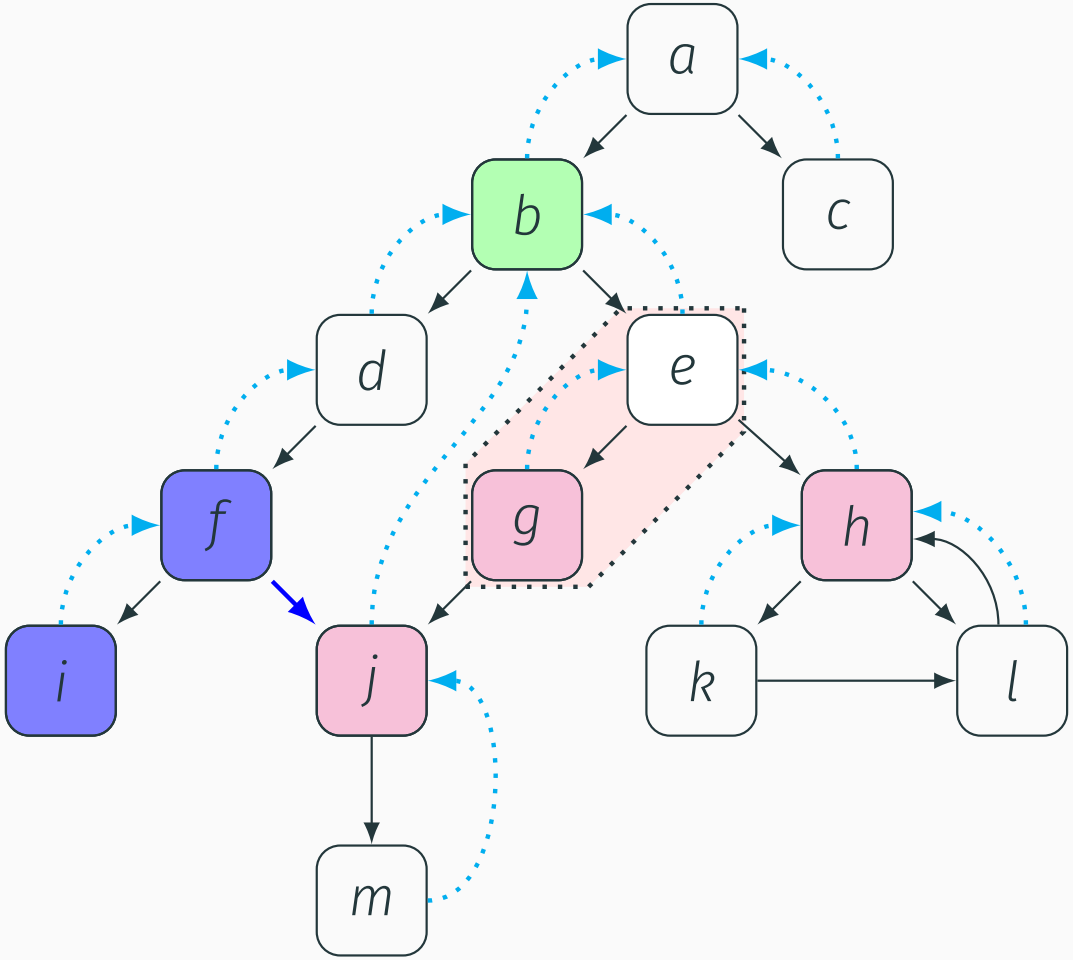
Edge Insertion (Ramalingam & Reps 1994)



Edge Insertion (Ramalingam & Reps 1994)

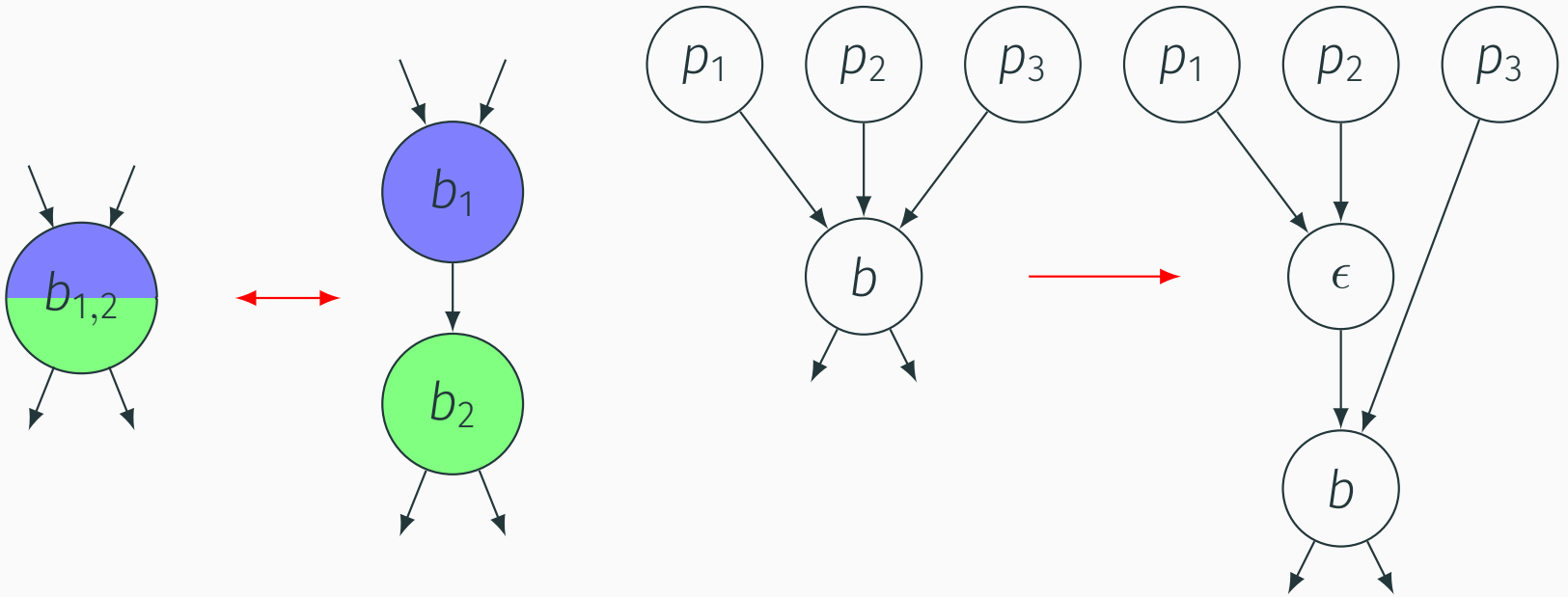


Edge Insertion (Ramalingam & Reps 1994)

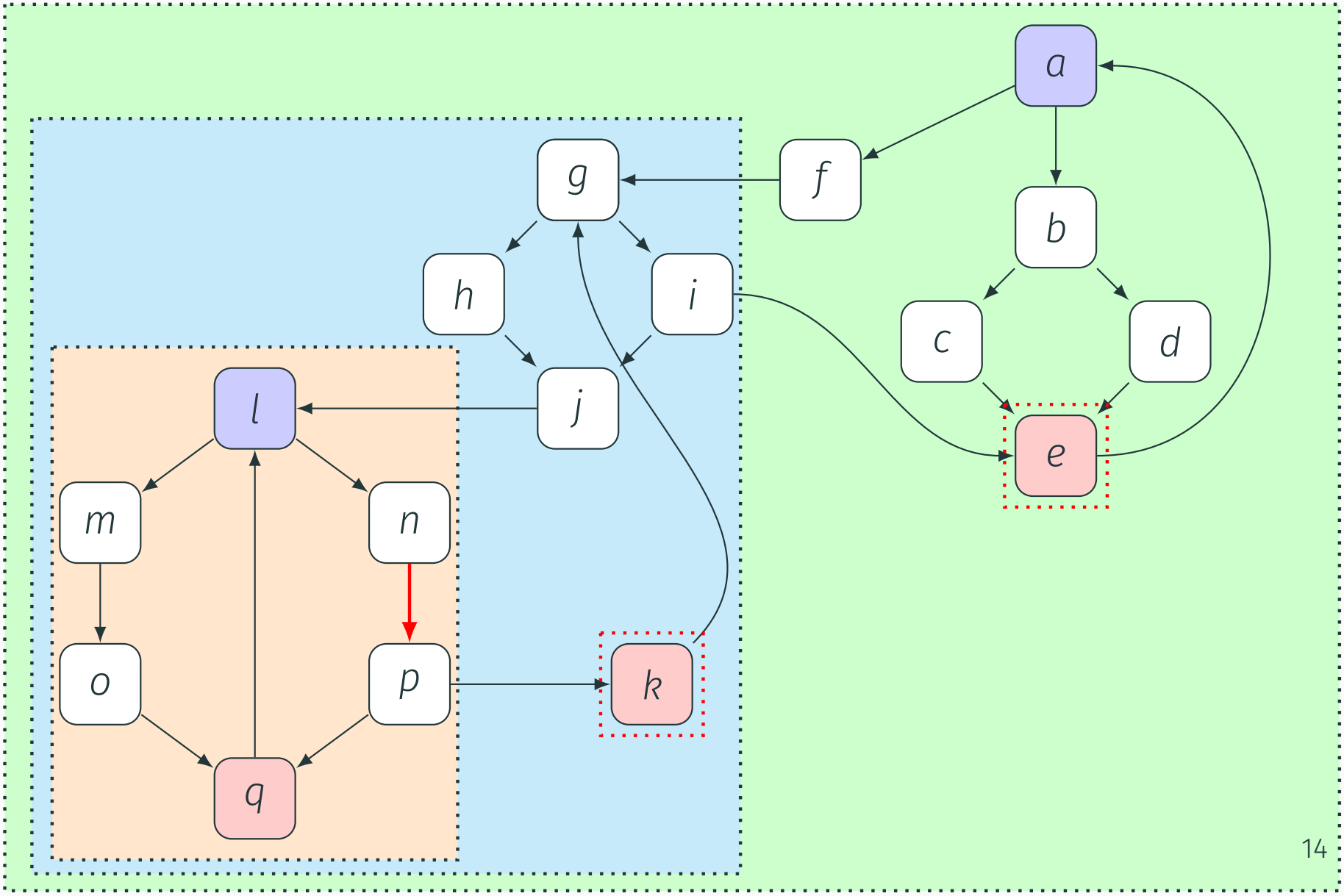


Canonical Form Preservation

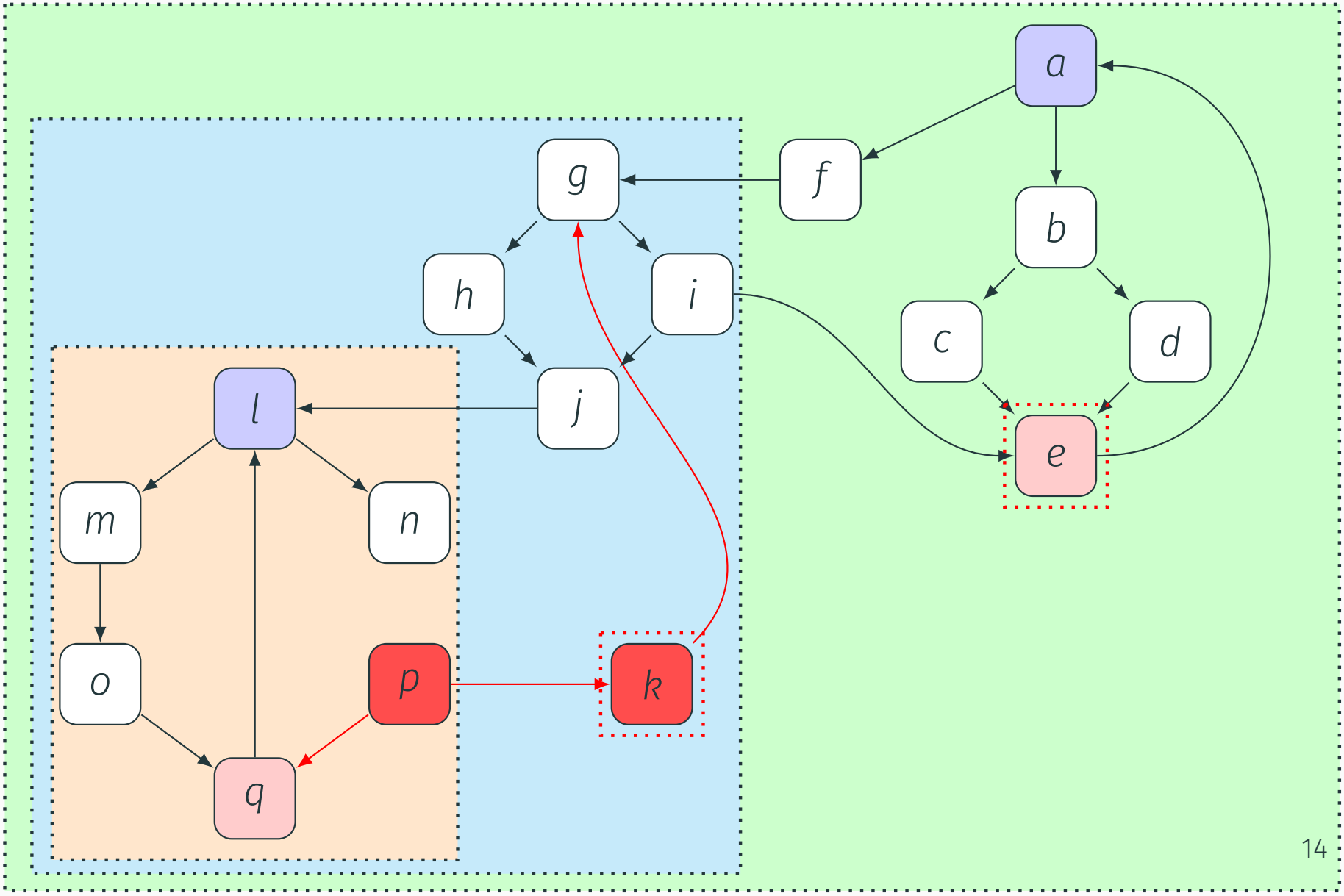
Block Splitting / Collapsing & Edge-Set Splitting



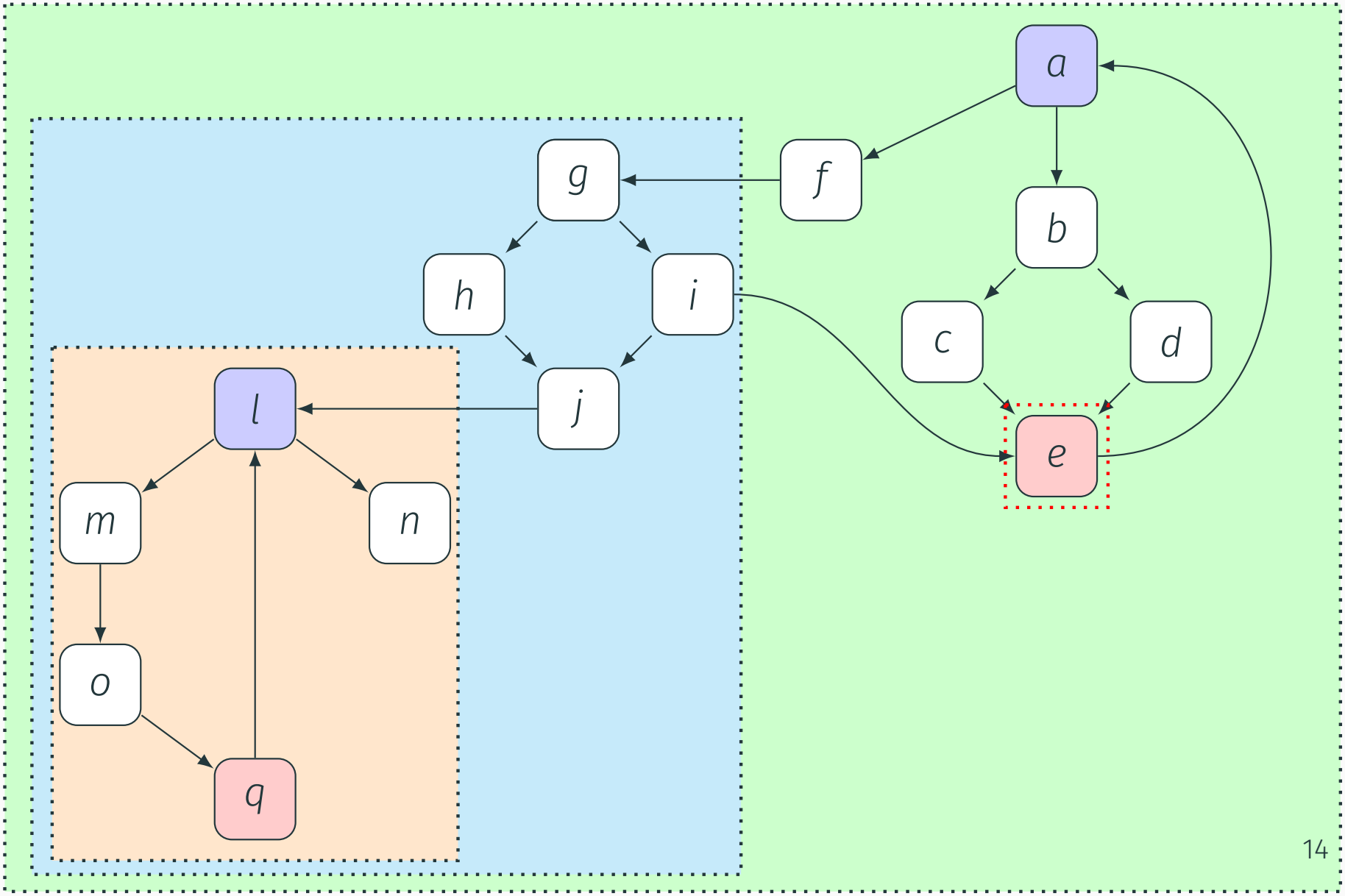
Edge Deletion



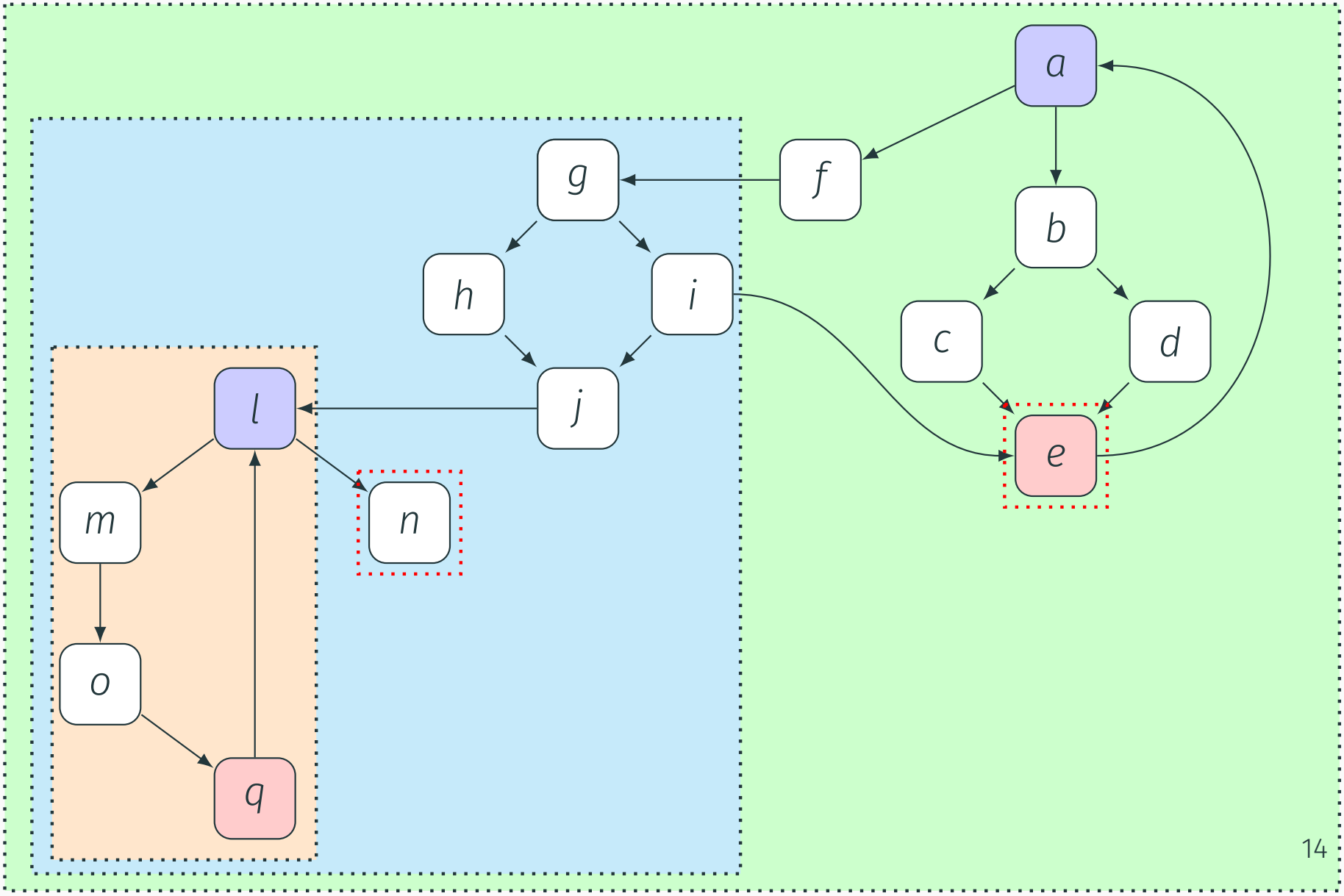
Edge Deletion



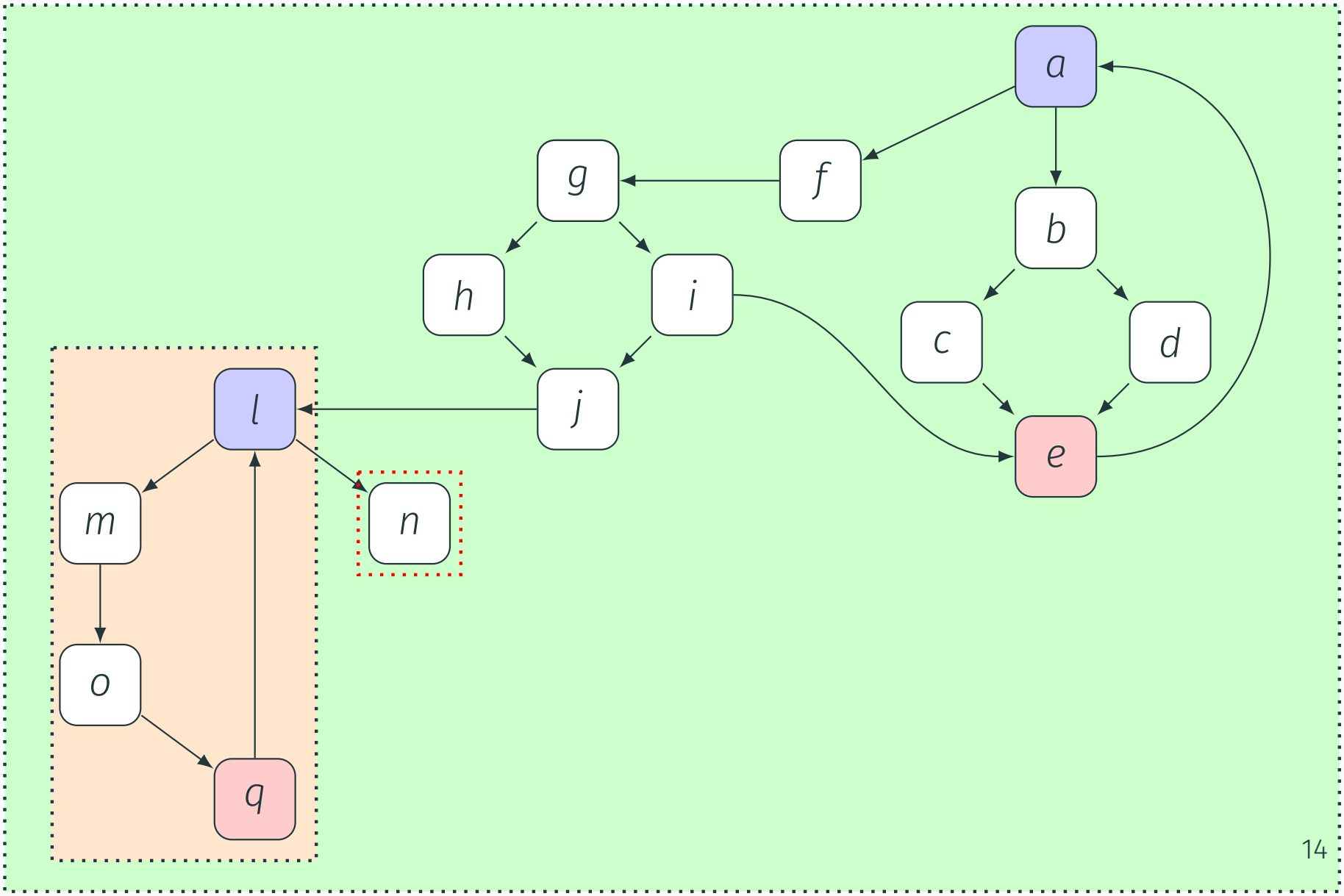
Edge Deletion



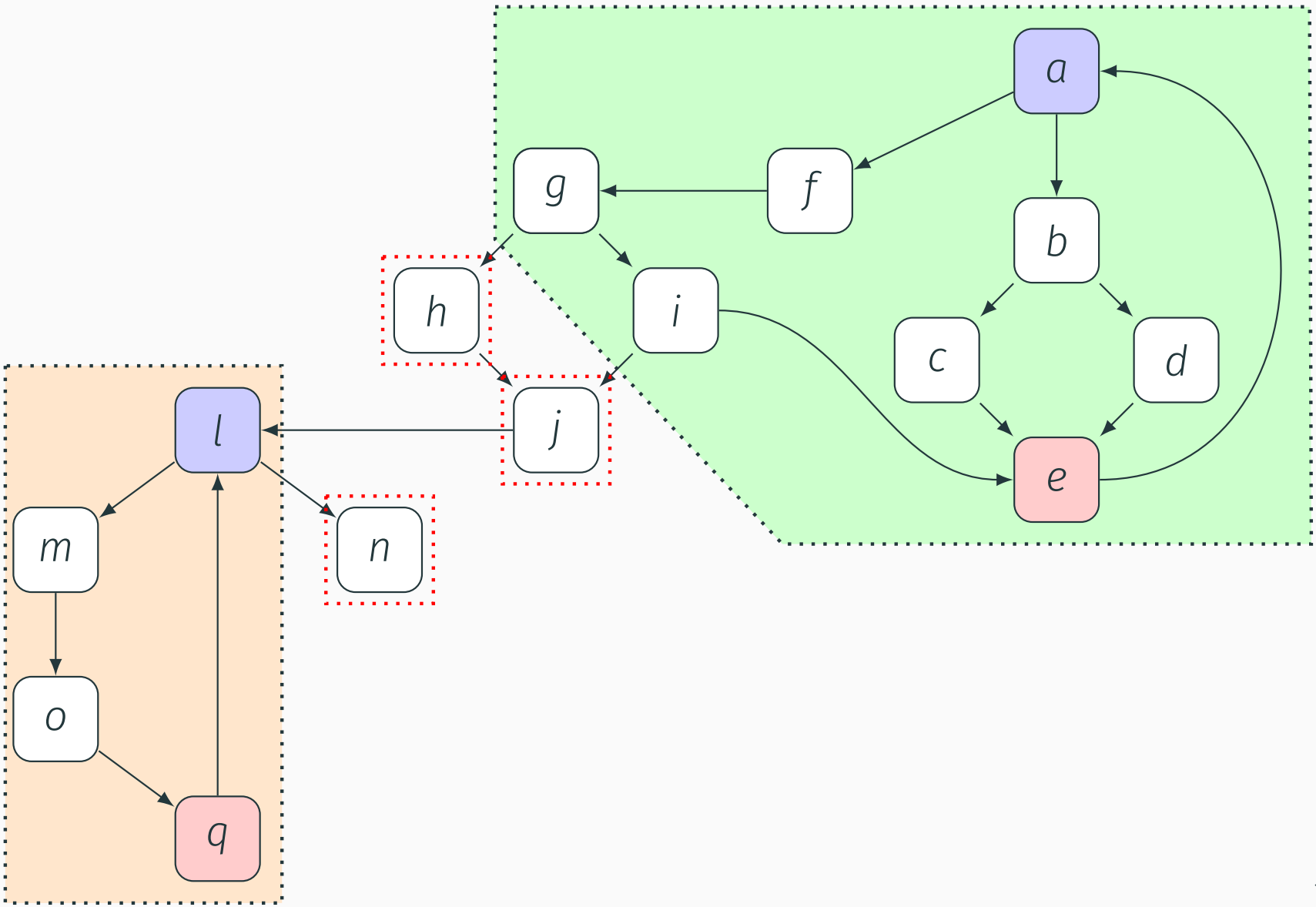
Edge Deletion



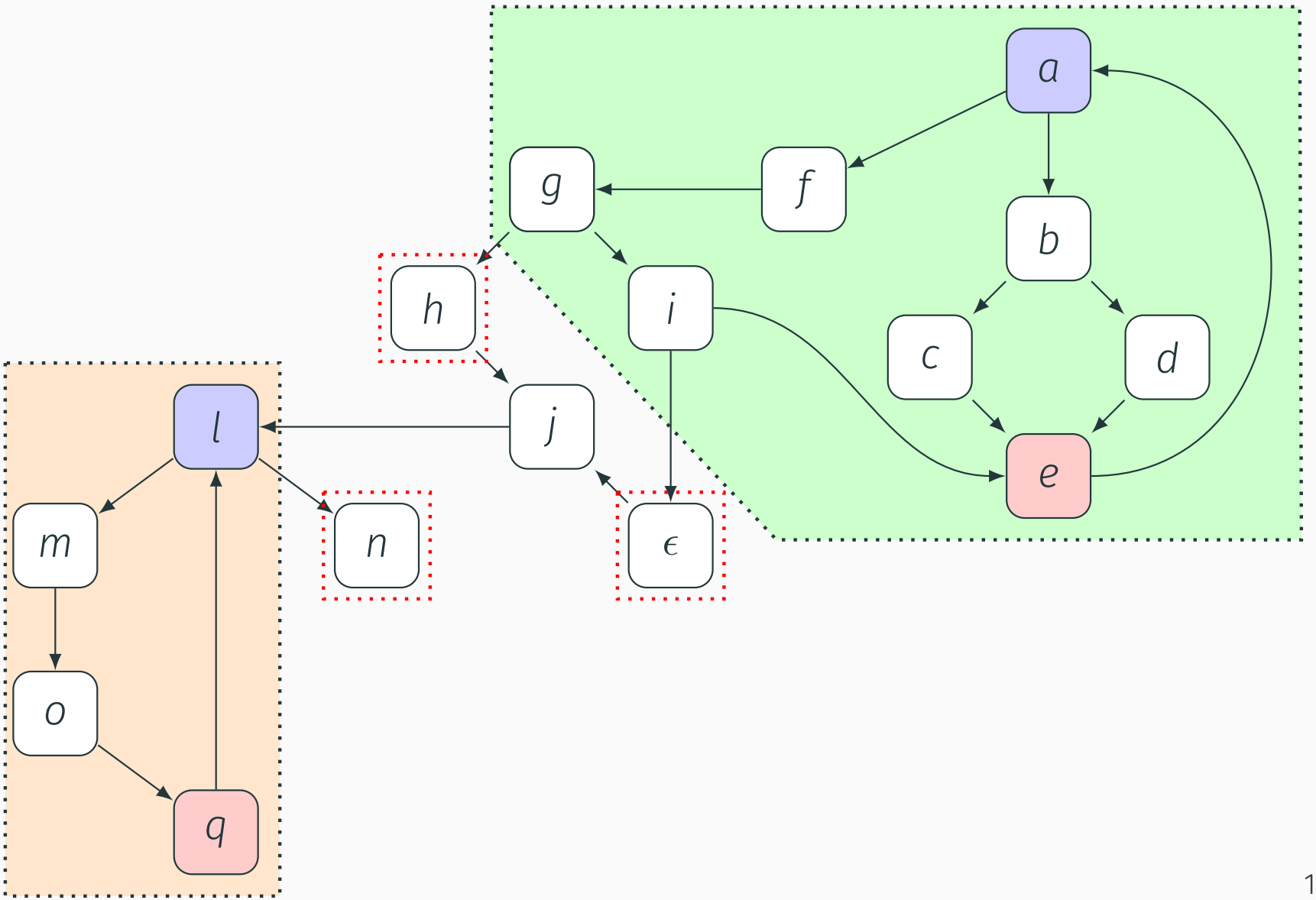
Edge Deletion



Edge Deletion



Edge Deletion



Tasklist

Remaining Work

1. Solidify Current Work
2. CF Preservation on Edge Insertion (e.g. Jump Threading)
3. Implement Bag of Optimizations
4. Evaluation

Questions?