I. What is a Loop?

- **Goal:**
  - Define a loop in graph-theoretic terms (control flow graph)
  - Not sensitive to input syntax, a uniform treatment for all loops: DO, while, goto’s
- **Not every cycle is a “loop” from the optimization perspective**

![Diagram of a loop example]

- **Intuitive properties of a loop**
  - single entry point
  - edges must form at least a cycle

---

Formal Definitions

- **Dominator**
  - Node $d$ dominates node $n$ in a graph $(d \text{ dom } n)$ if every path from the start node to $n$ goes through $d$

![Diagram of dominator tree]

- Dominators can be organized as a tree
  - $a \rightarrow b$ in the dominator tree $\iff$ $a$ immediately dominates $b$

---

Natural Loops

- **Definitions**
  - Single entry-point: **header**
    - a header dominates all nodes in the loop
  - A **back edge** is an arc whose head dominates its tail (tail $\rightarrow$ head)
    - a back edge must be a part of at least one loop
  - The **natural loop of a back edge** is the smallest set of nodes that includes the head and tail of the back edge, and has no predecessors outside the set, except for the predecessors of the header.
II. Algorithm to Find Natural Loops

1. Find the dominator relations in a flow graph
2. Identify the back edges
3. Find the natural loop associated with the back edge

1. Finding Dominators

- **Definition**
  - Node \( d \) dominates node \( n \) in a graph (\( d \, dom \, n \)) if every path from the start node to \( n \) goes through \( d \)

- **Formulated as MOP problem**
  - Node \( d \) lies on all possible paths reaching node \( n \) => \( d \, dom \, n \)
    - Direction:
    - Values:
    - Meet operator:
    - Top:
    - Bottom:
    - Boundary condition: start/entry node =
    - Initialization for internal nodes
    - Finite descending chain?
    - Transfer function:

- **Speed:**
  - With reverse postorder, most flow graphs (reducible flow graphs) converge in 1 pass

2. Finding Back Edges

- **Depth-first spanning tree**
  - Edges traversed in a depth-first search of the flow graph form a depth-first spanning tree

- **Categorizing edges in graph**
  - Advancing edges: from ancestor to proper descendant
  - Cross edges: from right to left
  - Retreating edges: from descendant to ancestor (not necessarily proper)

- **Back Edges**

  - **Definition**
    - Back edge: \( t \rightarrow h \), \( h \) dominates \( t \)

  - **Relationships between graph edges and back edges**
    - A back edge must be a retreating edge
      - dominator => visit \( h \) before \( t \), \( t \) must be a descendant of \( h \)
    - A retreating edge is not necessarily a back edge
      - may not satisfy the dominator property

  - **Algorithm**
    - Perform a depth first search
      - For each retreating edge \( t \rightarrow h \), check if \( h \) is in \( t \)'s dominator list

  - **Most programs (all structured code, and most GOTO programs)**
    - reducible flow graphs
      - retreating edges = back edges
3. Constructing Natural Loops

- The natural loop of a back edge is the smallest set of nodes that includes the head and tail of the back edge, and has no predecessors outside the set, except for the predecessors of the header.

- Algorithm
  - delete $h$ from the flow graph
  - find those nodes that can reach $t$ (those nodes plus $h$ form the natural loop of $t \rightarrow h$)

Inner Loops

- If two loops do not have the same header
  - they are either disjoint, or
  - one is entirely contained (nested within) the other -- inner loop, one that contains no other loop.

- If two loops share the same header
  - Hard to tell which is the inner loop
  - Combine as one

Conclusions

- Define loops in graph theoretic terms
- Definitions and algorithms for
  - Dominators
  - Back edges
  - Natural loops

Preheader

- Optimizations often require code to be executed once before the loop
- Create a preheader basic block for every loop