Introduction

Loop Invariants

Let $C$ be a computation in a loop body. $C$ is invariant if it computes the same value during all iterations. $C$ can sometimes be moved out of the loop.

```plaintext
K := 1; I := 2;
REPEAT
    A := K + 1; I := I + A;
UNTIL I <= 10;
K := K + A;
```

How do we know what is a loop???
Loop Terminology

- Head
- Loop Body
- Entry edge
- Exit edge
- Tail
- Preheader

Preheaders

- A preheader is useful, for example if we want to move out loop-invariant computations.
- Not all loops have preheaders — but we can always add one.

Dominators

- To detect what the loops are in a program we first have to perform a *dominator analysis*.
- Definition:
  A node $d$ dominates a node $n$ if every path from the entry node to $n$ must go through $d$. 

- Entry
  - $d$
  - $n$
**Dominators**

- **Notation**: \( d \text{ dom } n \rightarrow d \) strictly dominates \( n \).
- **Intuition**: Given a node \( n \), which blocks are guaranteed to have executed prior to executing \( n \).

- Every node dominates itself: \( d \text{ dom } d \).

**Strict Dominator**

- **Definition**: A node \( d \) strictly dominates a node \( n \) if \( d \) dominates \( n \) and \( d \neq n \).
- **Notation**: \( d \text{ sdom } n \rightarrow d \) strictly dominates \( n \).

**Immediate Dominator**

- **Definition**: The immediate dominator \( d \) of a node \( n \) is the unique node that strictly dominates \( n \) but does not strictly dominate any other node that strictly dominates \( n \).
- **Entry nodes don’t have an immediate dominator.**
- **Notation**: \( d \text{ idom } n \rightarrow d \) immediately dominates \( n \).

**Post dominator**

- **Definition**: A node \( d \) post dominates a node \( n \) if every path from \( n \) to the exit node must go through \( d \).
- **Notation**: \( d \text{ pdom } n \rightarrow d \) post dominates \( n \).
- **Intuition**: Given a node \( n \), which blocks are guaranteed to execute after executing \( n \).
Definition:
A back edge \( b \rightarrow h \), where \( h \text{dom} b \), induces a natural loop consisting of all nodes \( x \), where \( h \text{dom} x \) and there is a path from \( x \) to \( b \) not containing \( b \).

Example — Not a Natural Loop

Back edge \( b \rightarrow h \) since there is a path to \( b \) from the Entry node that does not go through \( h \).

\( h \) does not dominate \( H \)ence, \( b \rightarrow h \) does not induce a natural loop.

Computing Dominators

The dominators of a node \( n \) are given by

\[
\text{dom}(\text{entry node}) = \{\text{entry node}\}
\]

\[
\text{dom}(n) = \{n\} \cup \left( \bigcap_{\text{preds } p \text{ of } n} \text{dom}(p) \right)
\]

- The dominator of the entry node is the entry node itself.
- The set of dominators for a node \( n \) is the intersection of the set of dominators for all predecessors of \( n \).
- \( n \) is also in the set of dominators for \( n \).
Dataflow Equations — Intuition

\[ \text{dom}(n) = \{n\} \cup \bigcap_{\text{preds } p \text{ of } n} \text{dom}(p) \]

- If \( d \) dominates all predecessors of \( n \), then it also dominates \( n \)

Algorithm

- \( N \) is the set of all nodes.
- \( n_0 \) is the entry node.

\[
\begin{align*}
\text{dom}(n_0) &:= \{n_0\}; \\
\text{dom}(n) &:= N; \\
\text{dom}(n) &:= \{n\} \cup \left( \bigcap_{\text{preds } p \text{ of } n} \text{dom}(p) \right)
\end{align*}
\]

Example 1

```
K:=1;
I:=2;
REPEAT
A:=K+1;
I:=I+A;
UNTIL I<=10;
K:=K+A;
```

Example 1 — Initialization
Example 1 — First Iteration

Example 1 — Final Result

- A back edge \( b \rightarrow h \), where \( h \text{dom} b \), induces a natural loop consisting of all nodes \( x \), where \( h \text{dom} x \) and there is a path from \( x \) to \( b \) not containing \( b \).

Example 2
Example 2 — Initialization

Example 2 — First iteration

Example 2 — Identifying loops

Back edge $b \rightarrow h$, $h \text{dom } b$, induces a loop with all nodes $x$, where $h \text{dom } x$ and there there is a path $x \sim b$ not containing $b$.

Summary
Each node dominates itself.
If \( x \) dominates \( y \), and \( y \) dominates \( z \), then \( x \) dominates \( z \).
If \( x \) dominates \( z \) and \( y \) dominates \( z \), then either \( x \) dominates \( y \) or \( y \) dominates \( x \).