WCET Analysis: High-Level Overview

CPEN 432 Real-Time System Design

Arpan Gujarati
University of British Columbia

Worst-Case Execution Time Analysis (WCET)

How to determine C_i?

WCET analysis

Schedulability analysis

Execution Time Histogram[Wil08]

Terminology

WCET = maximum ever observed (on target platform)

BCET = minimum ever observed

ACET = average, dependent on input, BCET ≤ ACET ≤ WCET

It's important to distinguish between *bounds* (or *estimates*) and the *actual* WCET/BCET.
→ **Safety**: bound ≥ actual.
→ **Tightness**: bound close to actual value.

WCET Analysis Challenges

Typical Software Restrictions

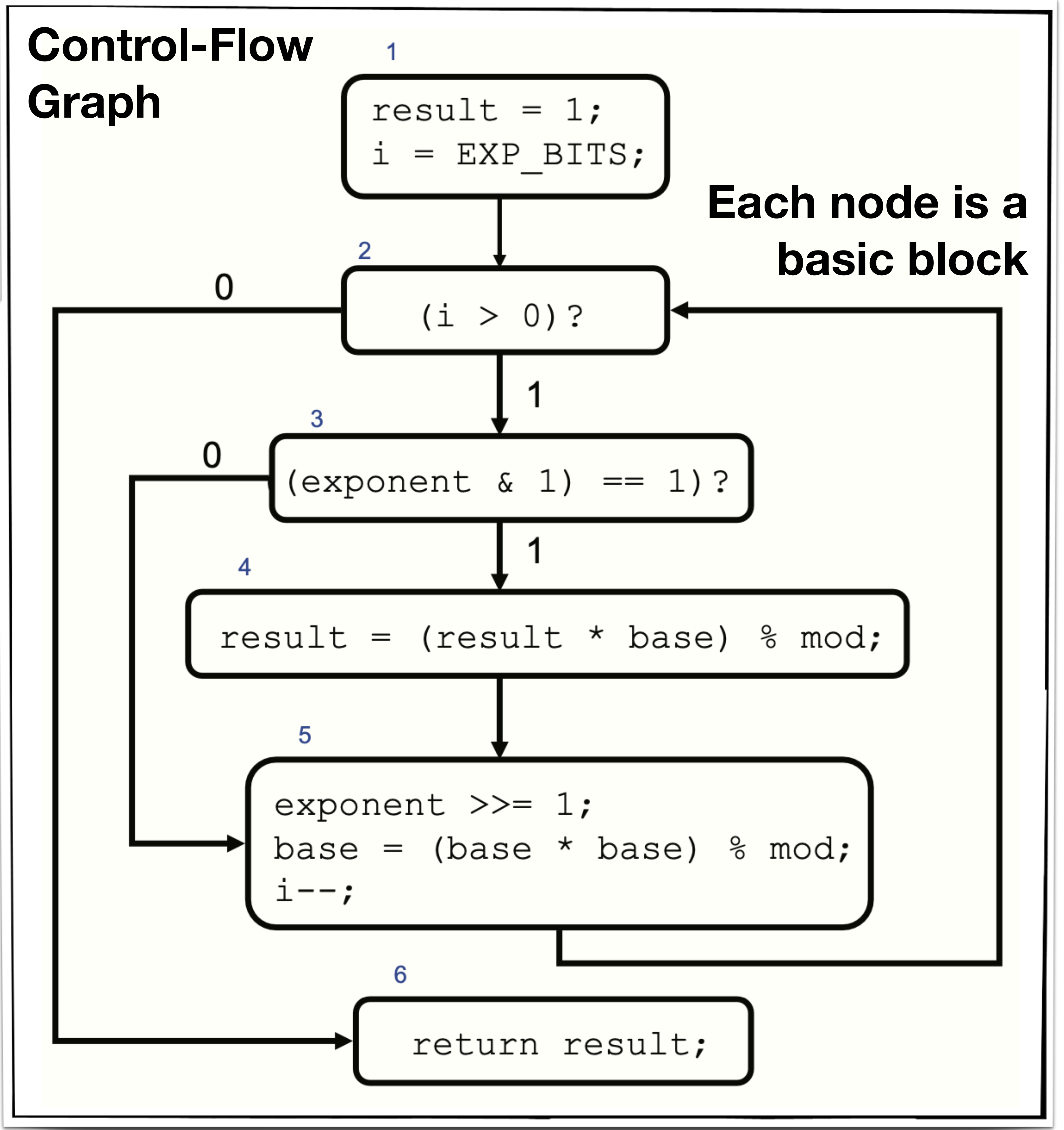
- no recursion
- no unbounded loops
- no function pointers / virtual method dispatch
- no/restricted pointer aliasing
- no dynamic linking
- no dynamic memory management

Programs as Graphs

```
#define EXP_BITS 32

typedef unsigned int UI;

UI modexp(UI base, UI exponent, UI mod) {
  int i;
  UI result = 1;

  i = EXP_BITS;
  while(i > 0) {
    if ((exponent & 1) == 1) {
      result = (result * base) % mod;
    }
    exponent >>= 1;
    base = (base * base) % mod;
    i--;
  }
  return result;
}
```

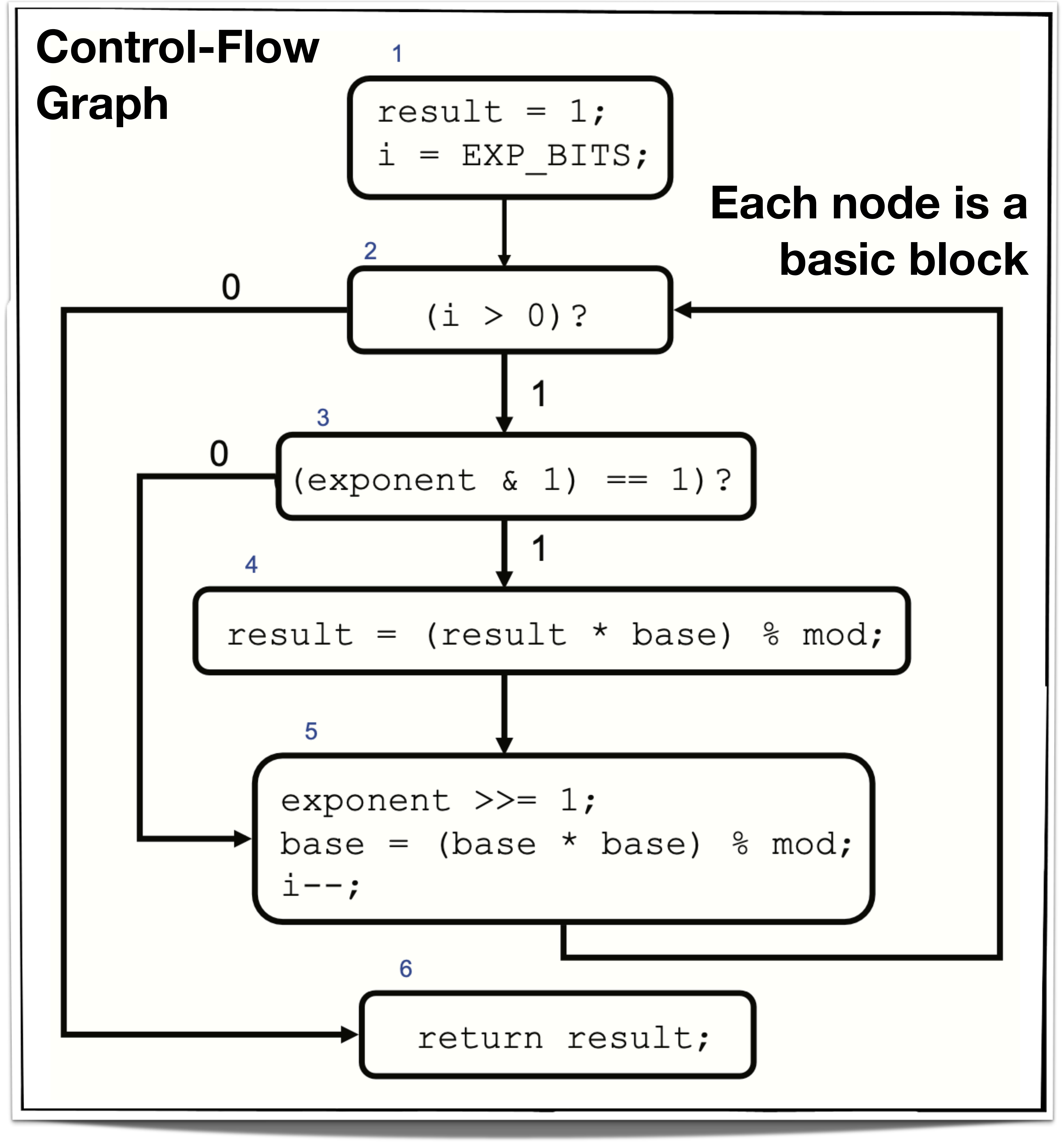
If e is even: $b^e = (b^2)^{\frac{e}{2}}$

If e is odd: $b^e = b \times (b^2)^{\frac{e-1}{2}}$



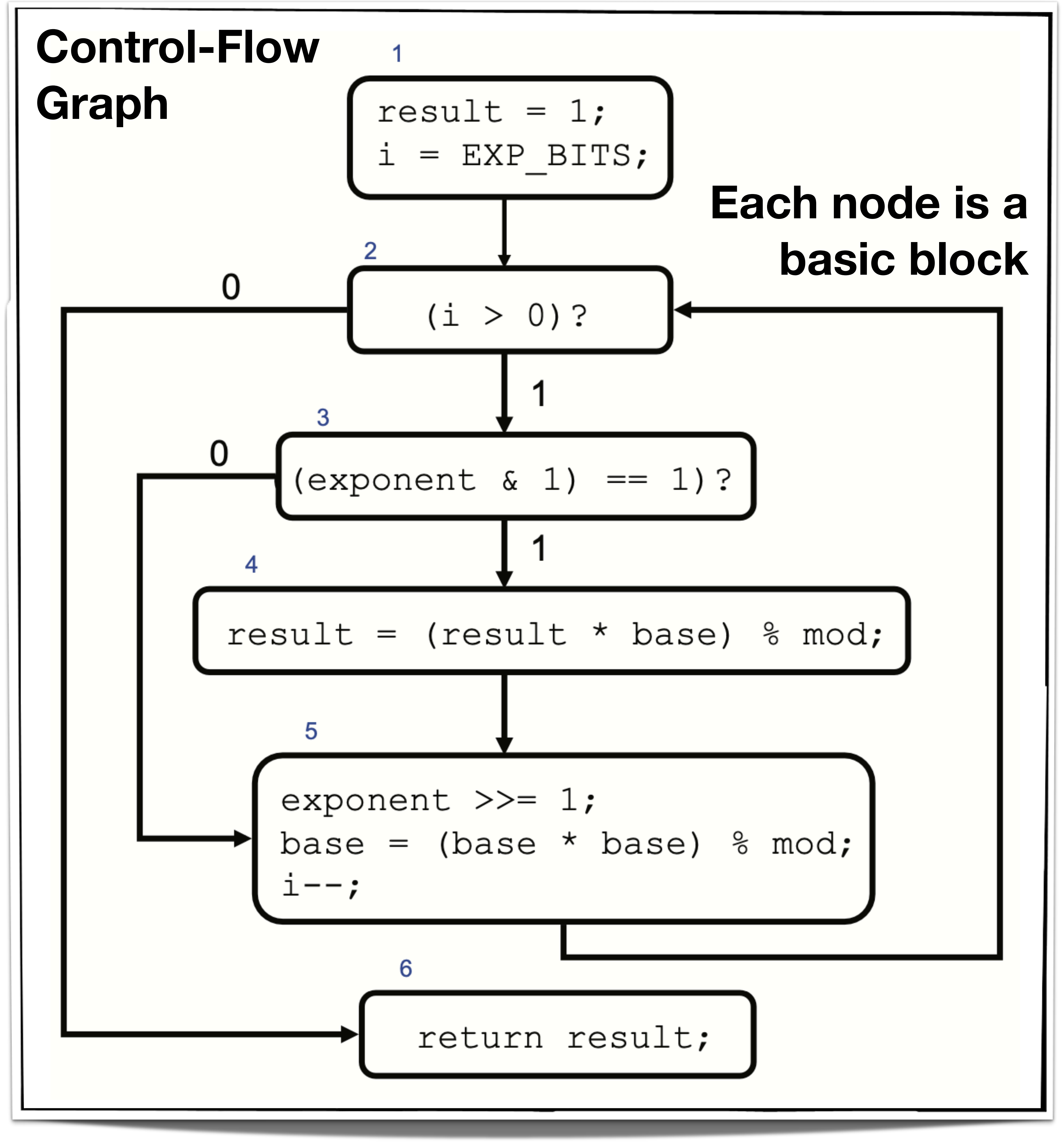
Optimization Formulation [1/3]

- Let $G = (V, E)$ denote the CFG
  - $n = |V|$ and $m = |E|$
- Let **X** $= (x_1, x_2, \ldots, x_n)$ be a vector of variables recording execution counts
  - x_i = no. of times basic block i is executed
- **X** is **valid** if its elements correspond to a feasible execution of the program
  - E.g., in the CFG on the right, for a valid **X**
    - $x_1 = x_6 = 1, x_2 = x_3 + 1, x_3 = x_5$



Optimization Formulation [2/3]

- Flow constraints
  - Unit flow at source: $x_1 = 1$ and $x_n = 1$
  - Conservation of flow: $x_i = \sum_{j \in P_i} d_{ji} = \sum_{k \in S_i} d_{ik}$
    - $d_{i,j}$ = no. of times the edge from node i to j is executed
    - $P_1 = \varnothing$ and $S_n = \varnothing$
- E.g., in the CFG on the right
  - $x_1 = 1$ and $x_6 = 1$
  - $x_1 = d_{12}$ and $x_2 = d_{12} + d_{52} = d_{23} + d_{26}$
  - $x_3 = d_{23} = d_{34} + d_{35}$ and $x_4 = d_{34} = d_{45}$
  - $x_5 = d_{35} + d_{45} = d_{52}$ and $x_6 = d_{26}$
- One valid solution: **X** $= (1, 2, 3, 0, 1, 1)$

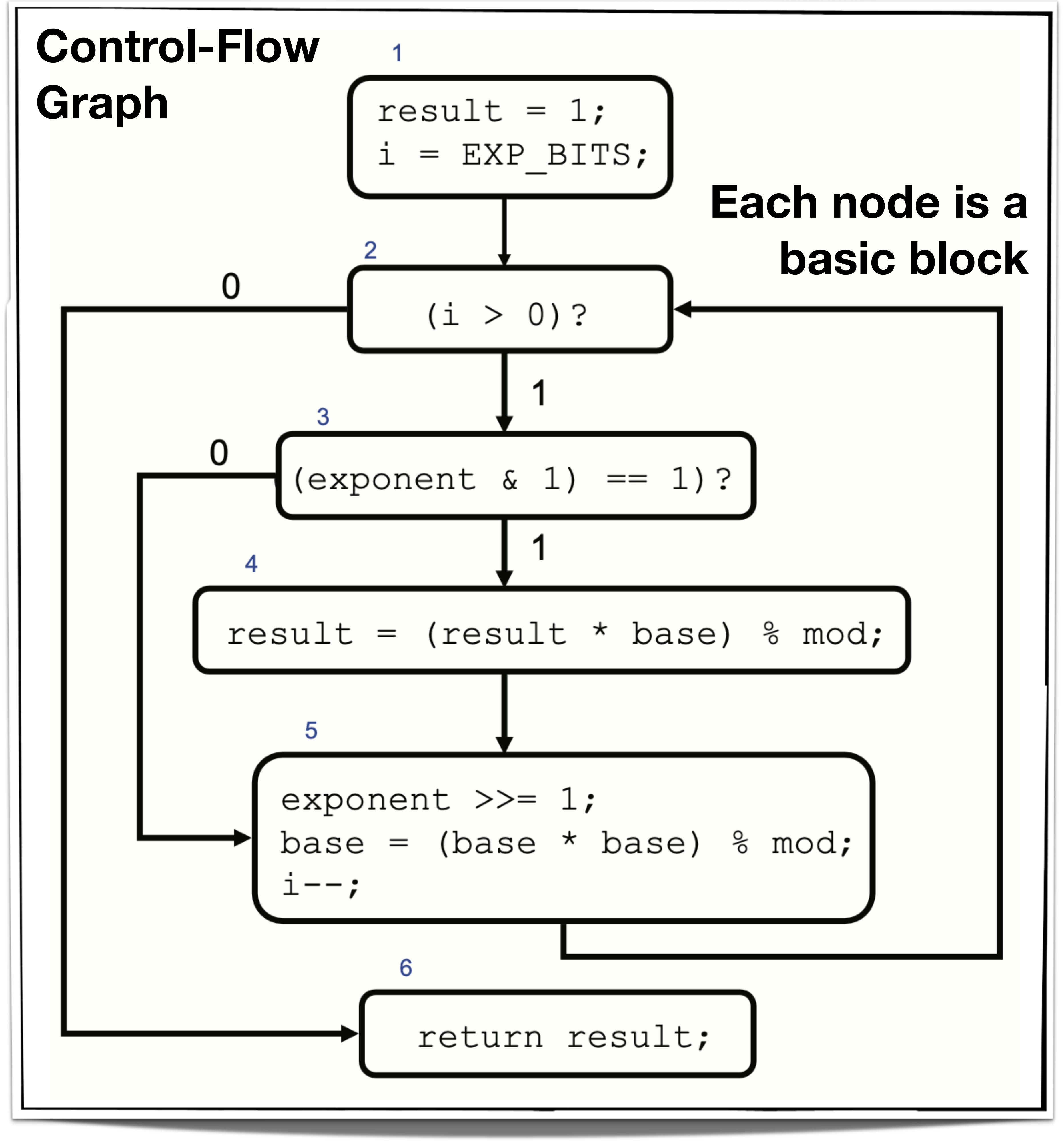


Optimization Formulation [3/3]

- Let w_i be an upper bound on the execution time of basic block i
  - WCET = maximum possible $\sum_{i=1}^{n} w_i x_i$ over all valid **X**

- Linear programming (LP) formulation
  - Find **X** that gives $\max_{x_i,\ 1 \le i \le n} \sum_{i=1}^{n} w_i x_i$
  - Subject to $x_1 = x_n = 1$ and $x_i = \sum_{j \in P_i} d_{ji} = \sum_{k \in S_i} d_{ik}$

- Drawbacks?



Logical Flow Constraints [1/2]

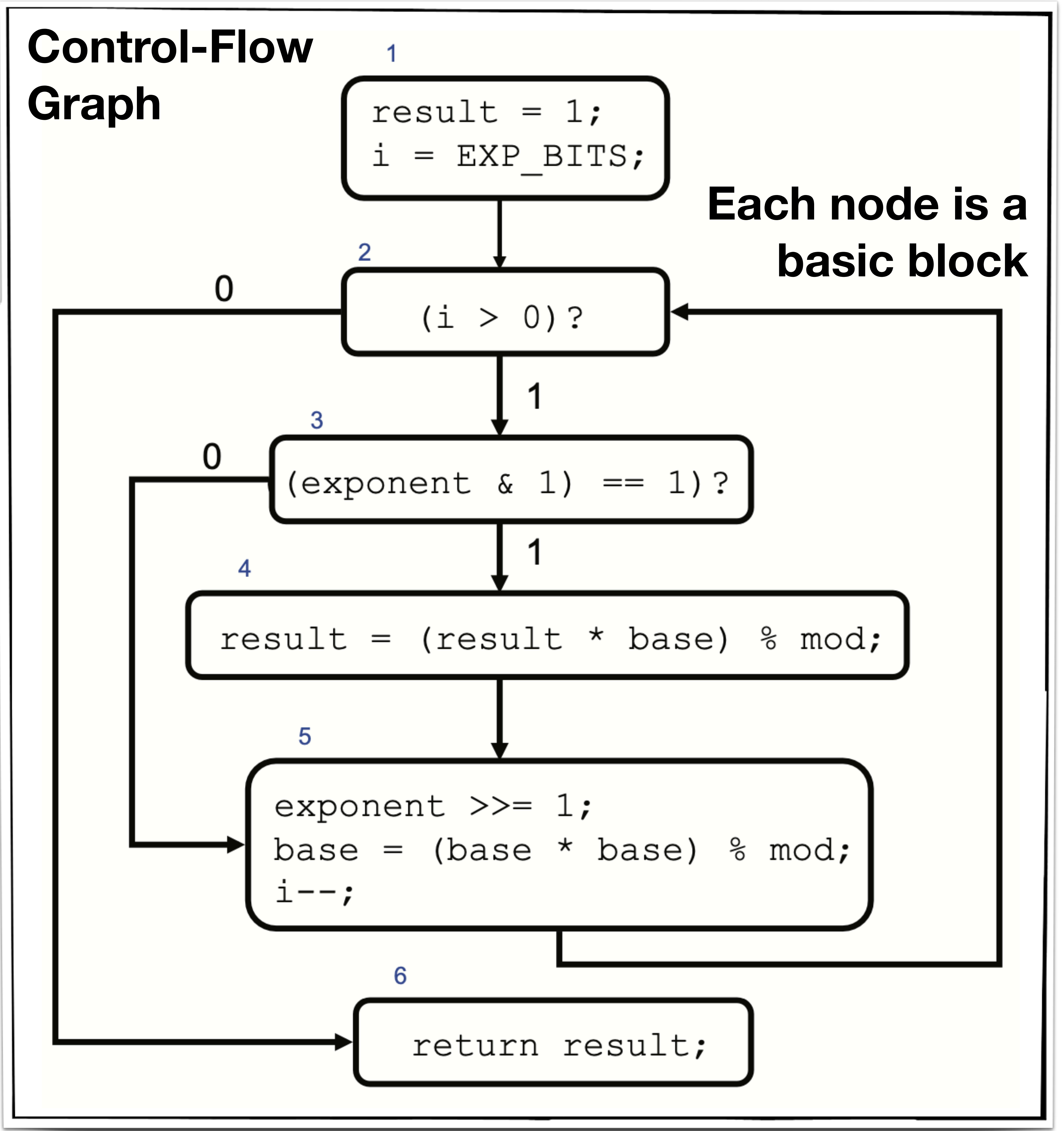
```c
#define EXP_BITS 32

typedef unsigned int UI;

UI modexp(UI base, UI exponent, UI mod) {
  int i;
  UI result = 1;

  i = EXP_BITS;
  while(i > 0) {
    if ((exponent & 1) == 1) {
      result = (result * base) % mod;
    }
    exponent >>= 1;
    base = (base * base) % mod;
    i--;
  }
  return result;
}
```

If e is even: $b^e = (b^2)^{\frac{e}{2}}$

If e is odd: $b^e = b \times (b^2)^{\frac{e-1}{2}}$

How many times around the while loop?

$x_3 \leq 32$



Logical Flow Constraints [2/2]

```c
#define CLIMB_MAX 1.0
```

```c
void altitude_control_task(void) {
    ...
    err = estimator_z - desired_altitude;
    desired_climb
          = pre_climb + altitude_pgain * err;
    if (desired_climb < -CLIMB_MAX) {
      desired_climb = -CLIMB_MAX;
    }
    if (desired_climb > CLIMB_MAX) {
      desired_climb = CLIMB_MAX;
    }
    return;
}
```

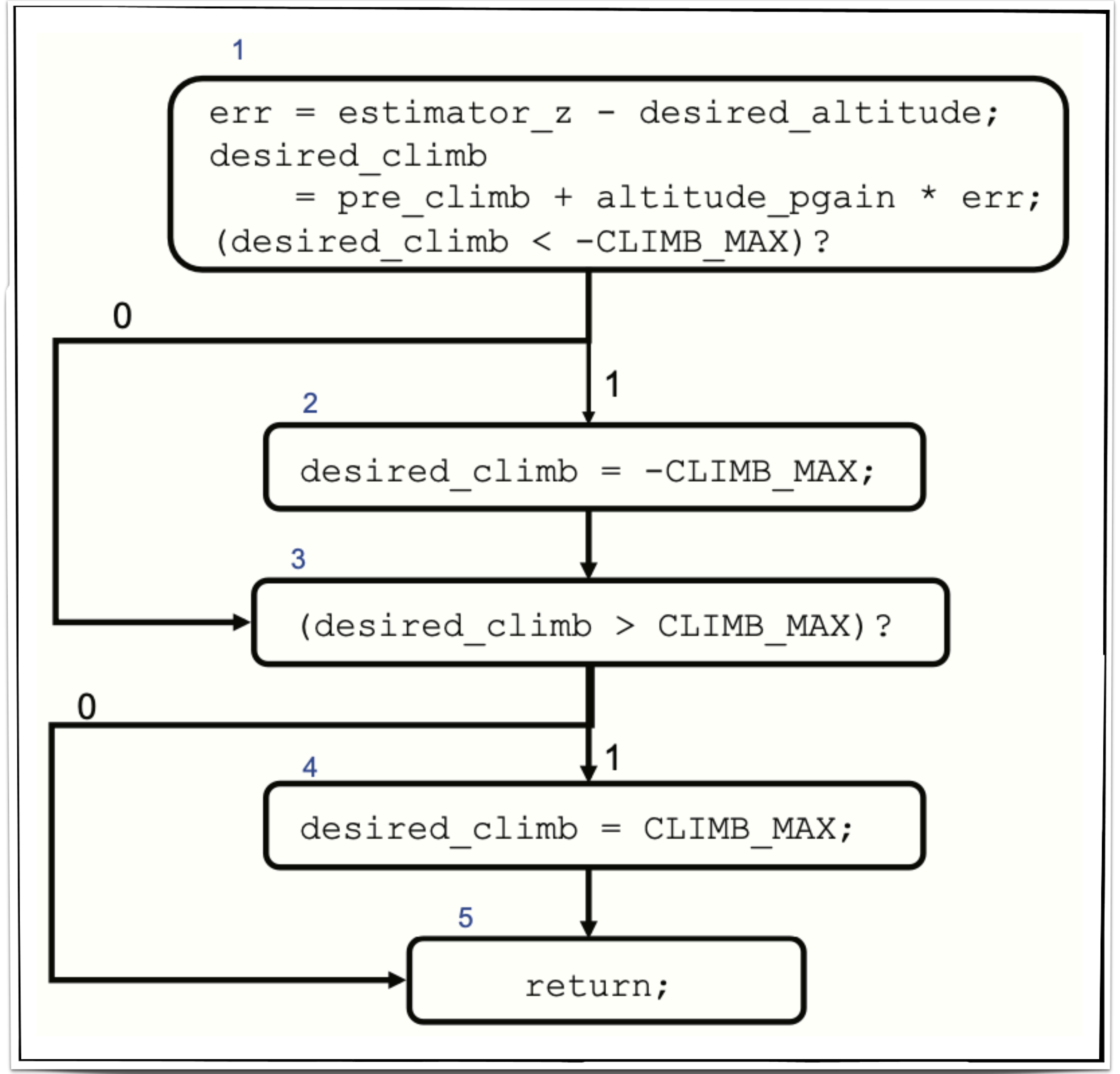
Are there any infeasible paths?

$$d_{12} + d_{34} \leq 1$$

$$
\begin{aligned}
x_1 &= 1 \\
x_5 &= 1 \\
x_1 &= d_{12} + d_{13} \\
x_2 &= d_{12} = d_{23} \\
x_3 &= d_{13} + d_{23} = d_{34} + d_{35} \\
x_4 &= d_{34} = d_{45} \\
x_5 &= d_{35} + d_{45}
\end{aligned}
$$

A solution for the above system of equations is

$$x_1 = x_2 = x_3 = x_4 = x_5 = 1,$$



Bounds for Basic Blocks

- How to estimate upper bound w_i on the execution time of basic block i?
- Challenges
  - Requires detailed micro-architectural modelling
  - Cache miss versus a hit can change latency by a factor of 100
    - If the analysis does not differentiate between cache hits and misses, the computed bound may be a hundred times larger than the actual execution time

Examples

```
void testFn(int *x, int flag) {
  while (flag != 1) {
    flag = 1;
    *x = flag;
  }
  if (*x > 0)
    *x += 2;
}
```

- Is there a bound on the number of iterations of the while loop? Justify your answer?
- How many total paths does this program have? How many of them are feasible, and why?
- Write down the system of flow constraints, including any logical flow constraints, for the control-flow graph of this program?