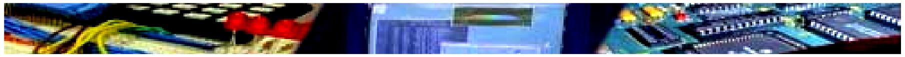
Embedded System Programming

WCET Analysis (1)

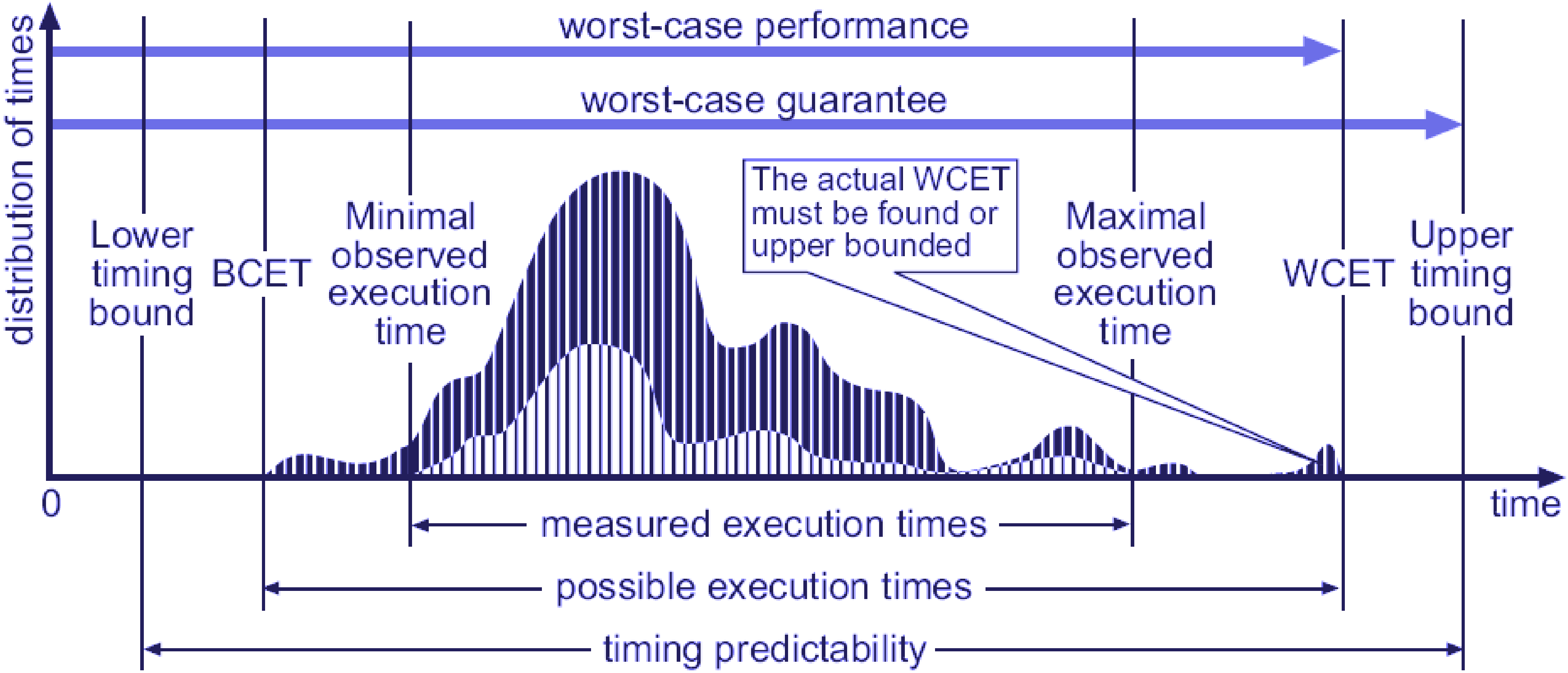
(Module 38)

Yann-Hang Lee
Arizona State University
yhlee@asu.edu
(480) 727-7507

Summer 2014



Execution Time – WCET & BCET



(Figure from R.Wilhelm et al., ACM Trans. Embed. Comput. Sys, 2007.)

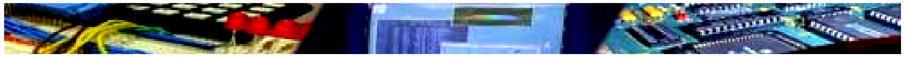
The WCET Problem

- **Given**
  - the code for a software task
  - the platform (OS + hardware) that it will run on
- **Determine the WCET of the task.**

- **Why is this problem important?**
  - The WCET is central in the design of real-time computing
- **Can the WCET always be found?**
  - In general, not a decidability problem, but a complexity problem
- **Compute bounds for the execution times of instructions and basic blocks and determine a longest path in the basic-block graph of the program.**

Components of Execution Time Analysis

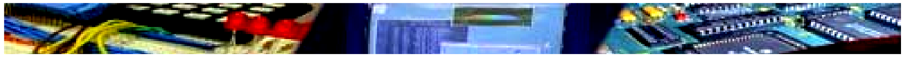
- **Program path (Control flow) analysis**
  - Want to find longest path through the program
  - Identify feasible paths through the program
  - Find loop bounds
  - Identify dependencies amongst different code fragments
- **Processor behavior analysis**
  - For small code fragments (basic blocks), generate bounds on run-times on the platform
  - Model details of architecture, including cache behavior, pipeline stalls, branch prediction, etc.
- **Outputs of both analyses feed into each other**



Program Path Analysis: Overall Approach (1)

- **Construct Control-Flow Graph (CFG) for the task**
  - Nodes represent Basic Blocks of the task
    - Basic block: a sequence of consecutive program statements where there is no possibility of branching
    - We have a single entry and a single exit node
  - Edges represent flow of control (jumps, branches, calls, …)

- **The problem is to identify the longest path in the CFG**
  - Note: CFG can have loops, so need to infer loop bounds and unroll them
  - This gives us a directed acyclic graph (DAG). How do we find the longest path in this DAG?

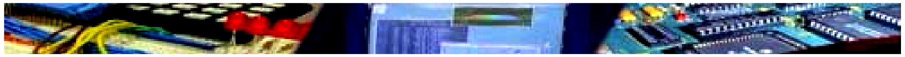


Program Path Analysis: Overall Approach (2)

- **In a CFG**
  - *B_i* = basic block *i*
  - *x_i* = number of times the block B_i is executed
  - *d_j* = number of times edge is executed
  - *c_i* = worst case running time of block *B_i*
- **Objective: find**

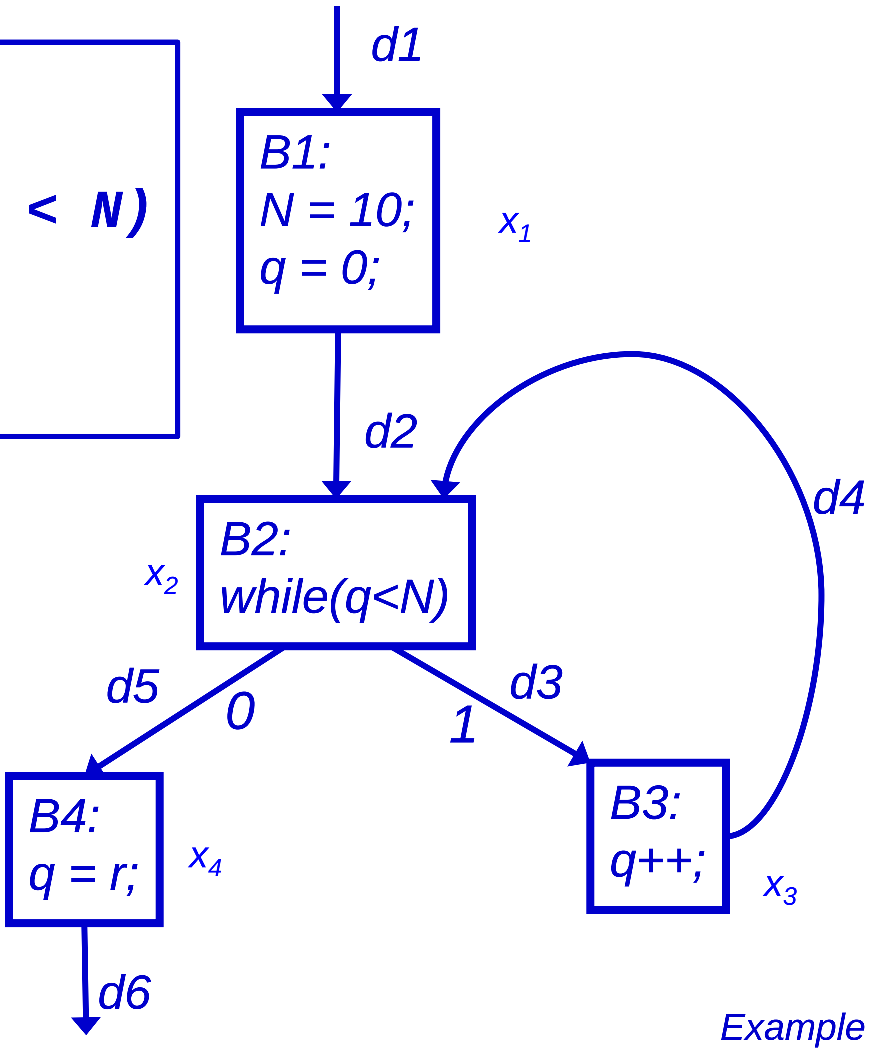
$$\mathrm{WCET} = \max_{x_i} \sum_{i=1}^{N} c_i x_i$$

- **How to get *x_i*?**
  - Structural constraints
  - Functionality constraints
  - Loop bounds -- need to be known



CFG Example

```
N = 10;
q = 0;
while(q < N)
   q++;
q = r;
```



Want to

maximize $\sum_i c_i x_i$

subject to constraints

$x_1 = d_1 = d_2$

$d_1 = 1$

$x_2 = d_2+d_4 = d_3+d_5$

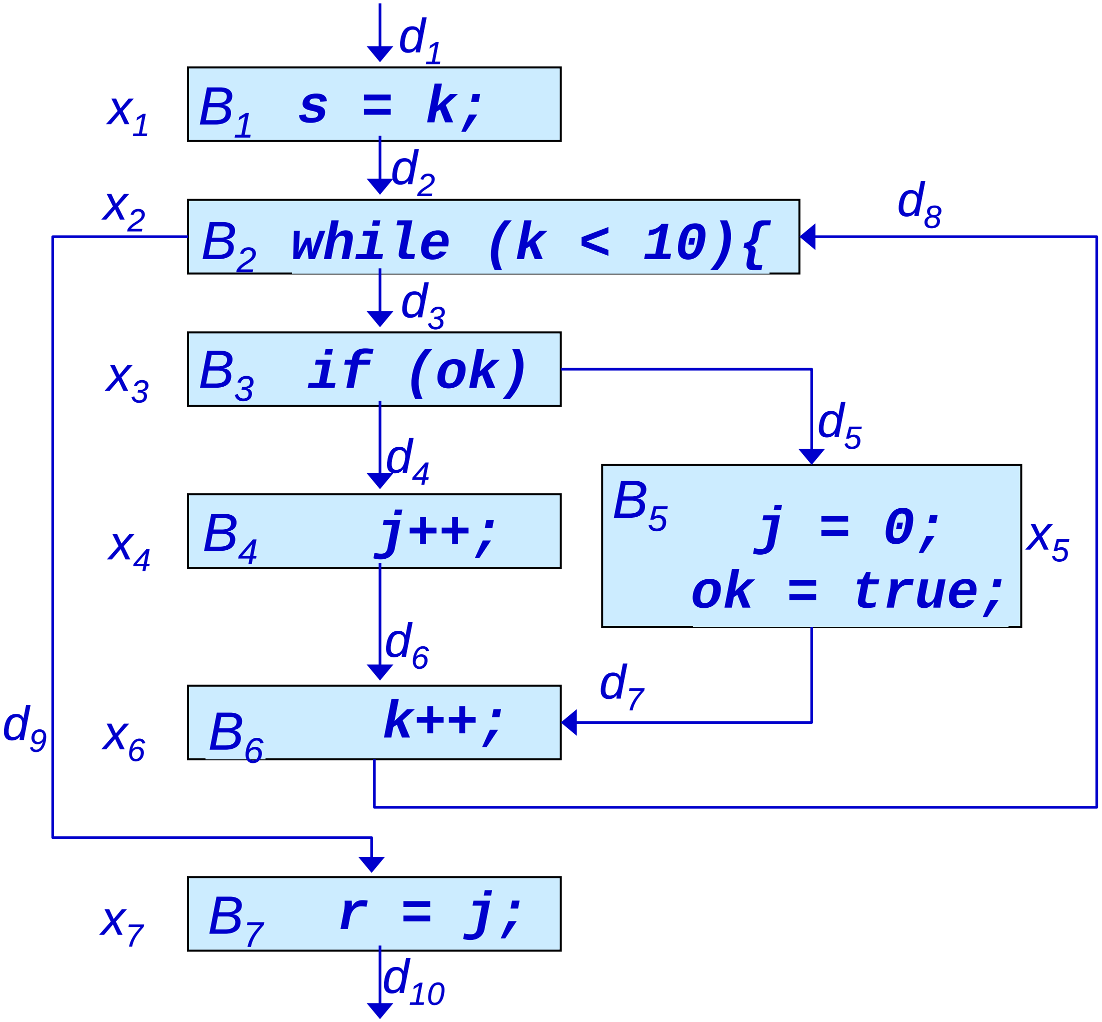
$x_3 = d_3 = d_4 = 10$

$x_4 = d_5 = d_6$

Example due to Y.T. Li and S. Malik

CFG – Another example

```
/* k >=0 */
s = k;
while (k < 10){
   if (ok)
      j++;
   else {
      j = 0;
      ok = true;
   }
   k++;
}
r = j;
```



Supplementary Slides