Advanced Computer Systems Architecture

Chip-Multiprocessors: Applications and Architectures

CSE 526M

Prof. Patrick Crowley

Plan for Today

- Announcements
  - Palmer visit this Friday
  - Commentaries
  - Milestone reports
- Questions
- Today's discussion

Project Logistics

- Dates
  - Today's date: Mar 16
  - End date: (**4.5 weeks later**) Thursday, April 15
- Weekly Milestones

M1	*Mar 4*	*Implementation 1*
M2	Mar 18	Implementation 2
M3	Mar 25	Implementation 3
M4	Apr 1	Implementation Wrap-up
M5	Apr 8	Plan future work, Reports
M6	Apr 15	Presentations

Objectives

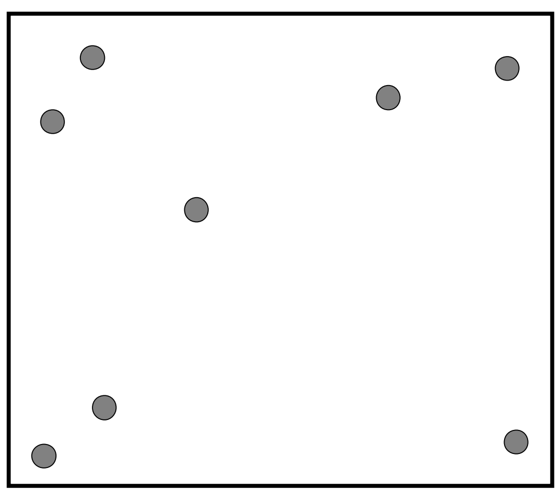
- Consider *performance optimization* in general terms
- Study a particular *performance evaluation* technique

Outline

- Performance optimization
  - Case Study
  - Principles
- Worst-case execution time
  - Motivation
  - Technique

Case Study: The N-body Problem

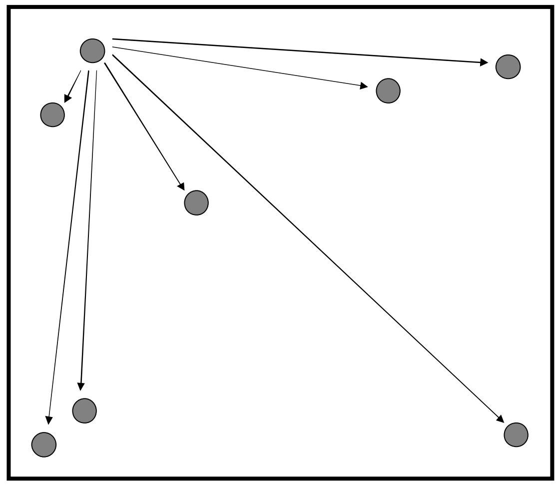
- Given masses, initial positions, and velocities: simulate interactions
- Each object could be a planet, e.g.



Resource: Bentley's Programming Pearls, 2nd edition

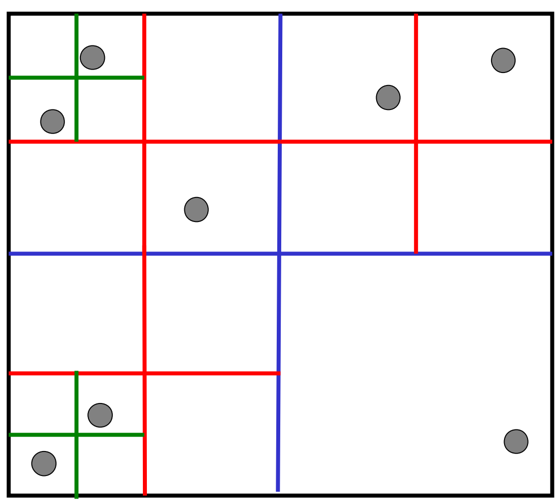
Direct Implementation

- At each (small) time step, compute each objects movement
- Key point: each object influenced by all the others
- Complexity: $O(n^2)$

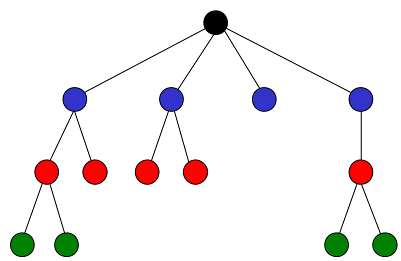


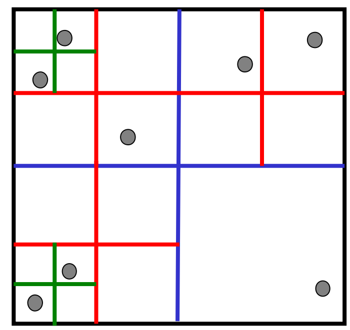
Appel's Algorithmic Approach

- Construct a quadtree (for 2d)
- Treat *distant* clusters as a single point



Quadtree Representation





- Leaves are objects
- Each non-leaf is a cluster
- Given a few constraints, each object is influenced by the clusters above it, O(n log n)

System-wide Contributions

Design Level	Speedup	Change
Alg and Data Structure	12	Binary tree reduces $O(n^2)$ to O(n log n)
Alg Tuning	2	Larger time steps
Data Structure	2	Tree-specific D.S.
System-ind Code Tune	2	Use single-precision integers
System-dep Code Tune	2.5	Hand tuned critical function
Hardware	2	F.P. accelerator
Total	**400**	

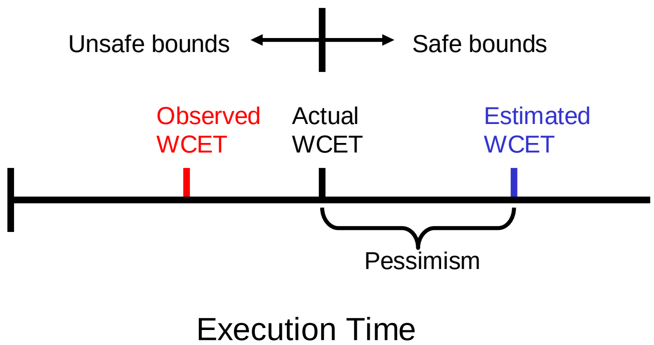
Principles

- Dramatic performance available across all design levels in the system
- Improvements in algorithms have high impact, and can lead to additional improvements
- If you need a little speedup: choose the right level
- If you need a large speedup: work at multiple levels

Worst-Case Execution Time (WCET)

- Networks are real time systems
  - Implementations must be both *functionally* and *temporally* correct
  - If service time exceeds inter-arrival time, the system will be unstable
- Minimum reliable performance *can* be stated in terms of worst-case conditions
  - Best effort routers
  - Beyond best effort: differentiated services

WCET: Safety & Tightness



WCET on Network Processors

- Challenges:
  - Implementations are software
  - Programs run on multithreaded multiprocessors
- Related real-time systems work
  - Studied recently
  - No work on *multithreading*

Question: How do you find a safe & tight worst-case bound?

A Method for Multithreaded WCET Estimation

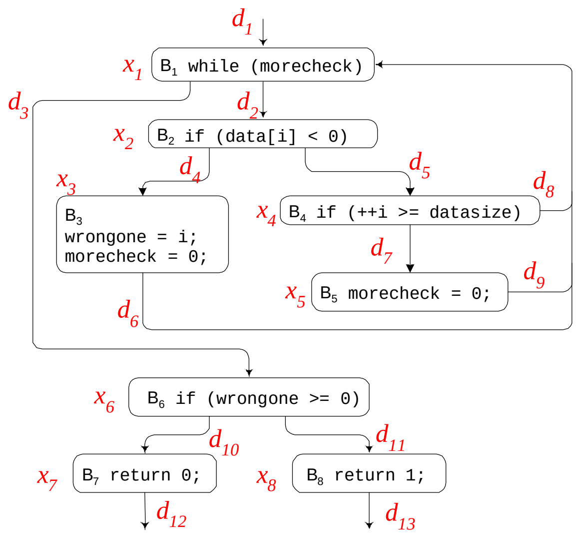
- Basic idea:
  - Use integer linear programming techniques to find the most expensive of all paths through a program's control-flow graph (CFG).
- Decidability restrictions on programs:
  - No unbounded loops
  - No dynamic data structures
  - No recursion
- Based on Implicit Path Enumeration (IPET)
  - Work by Steven Li, *et al.* from Princeton

The Problem

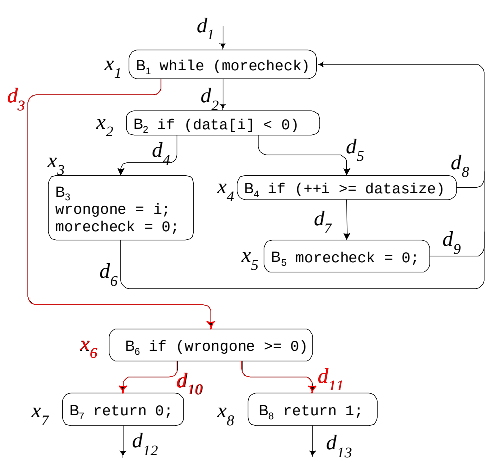
- Question 1
  - How long will one thread take to complete?
  - Assuming a simple, single issue processor
- *Question 2*
  - *How long will group of threads take to complete?*
  - *Assuming a single issue processor with 0-cycle context switches*
- Only considering worst-case throughput here

Example

```
/*
morecheck = 1;
i = 0; datasize = 10;
wrongone = -1;
*/
while (morecheck) {
  if (data[i] < 0) {
    wrongone = i;
    morecheck = 0;
  }
  else
    if (++i >= datasize)
      morecheck = 0;
}
if (wrongone >= 0)
  return 0;
else
  return 1;
```



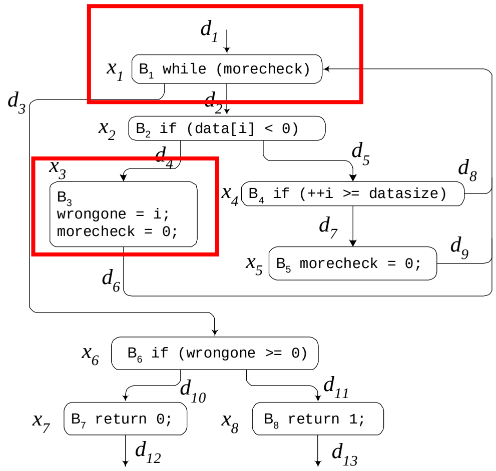
Structural constraints



$$d_1 = 1$$
$$x_1 = d_1 + d_6 + d_8 + d_9 = d_2 + d_3$$
$$x_2 = d_2 = d_4 + d_5$$
$$x_3 = d_4 = d_6$$
$$x_4 = d_5 = d_7 + d_8$$
$$x_5 = d_7 = d_9$$
$$x_6 = d_3 = d_{10} + d_{11}$$
$$x_7 = d_{10} = d_{12}$$
$$x_8 = d_{11} = d_{13}$$

- Flow in = flow out

Functional constraints



- Loop bound:

  $1d_1 \leq x_1 \leq 10d_1$

- Additionally:

  $x_3 \leq 1d_1$

Problem Statement

- Assume each basic block has a constant cost c_i
- The constraints bound the feasible x_i values.
- Solve for WCET by maximizing the sum:

$$\text{Program Execution Time} = \sum_{i=1}^{N} c_i x_i$$

Subject to the structural and functional constraints

Assignment

- Thursday (3/18)
  - Milestone 2
- Tuesday (3/23):
  - ***Commentary:*** *Hints for Computer System Design, by Butler Lampson*