Previously

- Resources aren’t free
- Share to reduce costs
- Schedule operations on resources
  - Fixed resources
- Greedy approximation algorithm
- List Scheduling for resource-constrained scheduling

Today

- Time-Constrained Scheduling
  - Force Directed
- Few words on project architecture
- Resource-Constrained
  - Branch-and-Bound

Force-Directed

- **Problem**: how exploit schedule freedom (slack) to minimize instantaneous resources
  - Directly solve time-constrained scheduling
    - (previously only solved indirectly)
  - Minimize resources with timing target

Force-Directed

- Given a node, can schedule anywhere between ASAP and ALAP schedule time
  - Between latest schedule predecessor and ALAP
  - Between ASAP and already scheduled successors
  - Between latest schedule predecessor and earliest schedule successor
- **That is**: Scheduling node will limit freedom of nodes in path
Single Resource Challenge

Force-Directed

- If everything were scheduled, except for the target node, what would we do?:
  - examine resource usage in all timeslots allowed by precedence
  - place in timeslot which has least increase in maximum resources
    - Least energy
    - Where the forces are pulling it

Force-Directed

- **Problem:** don’t know resource utilization during scheduling
- **Strategy:** estimate resource utilization

Force-Directed Estimate

- Assume a node is uniformly distributed within slack region
  - between earliest and latest possible schedule time
- Use this estimate to identify most used timeslots

Slacks on all nodes

Schedule into 12 cycles
Uniform Distribution of Slack

Most Constrained Node, Most Used Timeslot

Single Resource Challenge

Force-Directed

- Scheduling a node will shift distribution
  - all of scheduled node’s cost goes into one timeslot
  - predecessor/successors may have freedom limited so shift their contributions
- Goal: shift distribution to minimize maximum resource utilization (estimate)
Many steps…
Force-Directed Algorithm

1. ASAP/ALAP schedule to determine range of times for each node
2. Compute estimated resource usage
3. Pick most constrained node
   (in largest time slot…)
   - Evaluate effects of placing in feasible time slots (compute forces)
   - Place in minimum cost slot and update estimates
   - Repeat until done

Force-Directed Runtime

- Evaluate force of putting in timeslot $O(N)$
  - Potentially perturbing slack on net prefix/postfix for this node $\rightarrow N$
- Each node potentially in T slots: $\times T$
- $N$ nodes to place: $\times N$
- $O(N^2T)$
  - Loose bound--don’t get both $T$ slots and $N$ perturbations

Project Architecture

FPGA

K-LUT (typical $k=4$)
Compute block w/ optional output Flip-Flop

VLIW

Address
Instruction Memory

ESE171, CIS371
Merge Ideas

- Bit-level, LUT-based mesh
- VLIW, cycle-by-cycle control

Tabula

- March 1, 2010
  - Announced “new” architecture
- Bit-level, LUT-based VLIW with 8 instruction contexts

Branch-and-Bound

(for resource-constrained scheduling)

Brute-Force Scheduling (Exhaustive Search)

- Try all schedules
- Branching/Backtracking Search
- Start w/ nothing scheduled (ready queue)
- At each move (branch) pick:
  - available resource time slot
  - ready task (predecessors completed)
  - schedule task on resource
- Update ready queue

Example

T1 → time 1
T2 → time 1
T3 → time 2
T4 → time 2
T5 → time 2
T6 → time 2

Target: 2 FUs
Branching Search

- Explores entire state space
  - finds optimum schedule
- Exponential work
  - $O(N^{\text{resources}\times\text{time-slots}})$
- Many schedules completely uninteresting

Reducing Work

1. Canonicalize "equivalent" schedule configurations
2. Identify "dominating" schedule configurations
3. Prune partial configurations which will lead to worse (or unacceptable results)

"Equivalent" Schedules

- If multiple resources of same type
  - assignment of task to particular resource at a particular timeslot is not distinguishing

Keep track of resource usage by capacity at time-slot.

"Non-Equivalent" Schedule Prefixes

- Keep track of scheduled set
- Recognize when solving same sub-problem
  - Like dynamic programming finding same sub-problems
  - But no guarantee of small number of subproblems
    - set is power-set so $2^N$
    - ...but not all feasible, so shape of graph may simplify
Dominant Schedules

• A strictly shorter schedule
  – scheduling the same or more tasks
  – will always be superior to the longer schedule

Pruning

• If can establish a particular schedule path will be worse than one we’ve already seen
  – we can discard it w/out further exploration
• In particular:
  – LB=current schedule time + lower_bound_estimate
  – if LB greater than known solution, prune

Pruning Techniques

Establish Lower Bound on schedule time
• Critical Path (ASAP schedule)
• Resource Bound

Alpha-Beta Search

• Generalization
  – keep both upper and lower bound estimates on partial schedule
  • Lower bounds from CP, RB
  • Upper bounds with List Scheduling
  – expand most promising paths
    • (least upper bound, least lower bound)
    – prune based on lower bounds exceeding known upper bound
    – (technique typically used in games/Chess)

Alpha-Beta

• Each scheduling decision will tighten
  – lower/upper bound estimates
• Can choose to expand
  – least current time (breadth first)
  – least lower bound remaining (depth first)
  – least lower bound estimate
  – least upper bound estimate
• Can control greediness
  – weighting lower/upper bound
  – selecting “most promising”

Note

• Aggressive pruning and ordering
  – can sometimes make polynomial time in practice
  – often cannot prove will be polynomial time
  – usually represents problem structure we still need to understand
Multiple Resources

- Works for multiple resource case
- Computing lower-bounds per resource
  - resource constrained
- Sometimes deal with resource coupling
  - e.g. must have 1 A and 1 B simultaneously or in fixed time slot relation
  - e.g. bus and memory port

Summary

- Resource estimates and refinement
- Branch-and-bound search
  - “equivalent” states
  - dominators
  - estimates/pruning

Admin

- Assignment 1 was due at class start
- Reading
  - Wednesday online
- Assignment 2 out
  - Part A due next Monday
  - Part B following

Big Ideas:

- Estimate Resource Usage
- Use dominators to reduce work
- Techniques:
  - Force-Directed
  - Search
    - Branch-and-Bound
    - Alpha-Beta