

CS137: Electronic Design Automation

Day 13: March 1, 2003
Scheduling Introduction



CALTECH CS137 Winter2004 -- DeHon

Today

- Scheduling
 - Basic problem
 - variants
 - branching
 - bounds
 - and pruning

CALTECH CS137 Winter2004 -- DeHon

General Problem

- Resources are not free
 - wires, io ports
 - functional units
 - LUTs, ALUs, Multipliers,
 - memory locations
 - memory access ports

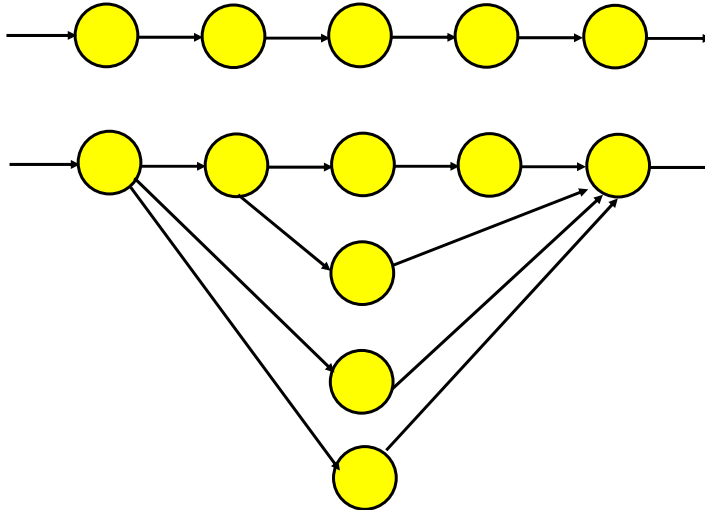
CALTECH CS137 Winter2004 -- DeHon

Trick/Technique

- Resources can be shared (reused) in time
- Sharing resources can reduce
 - instantaneous resource requirements
 - total costs (area)
- **Pattern:** scheduled operator sharing

CALTECH CS137 Winter2004 -- DeHon

Example



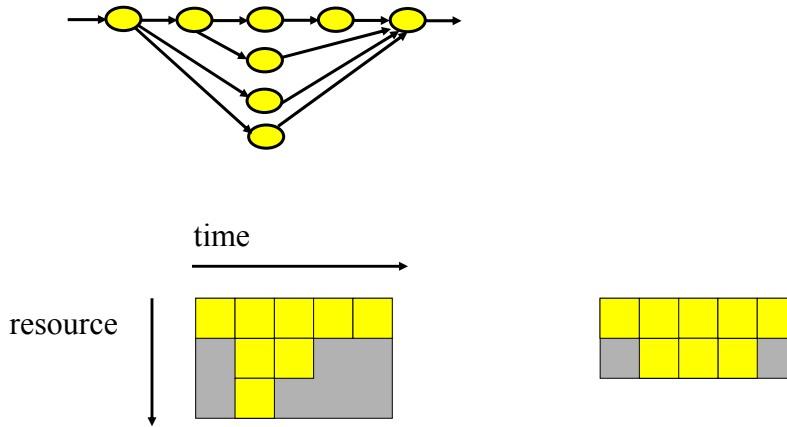
CALTECH CS137 Winter2004 -- DeHon

Sharing

- Does not have to increase delay
 - w/ careful time assignment
 - can often reduce peak resource requirements
 - while obtaining original (unshared) delay
- **Alternately:** Minimize delay given fixed resources

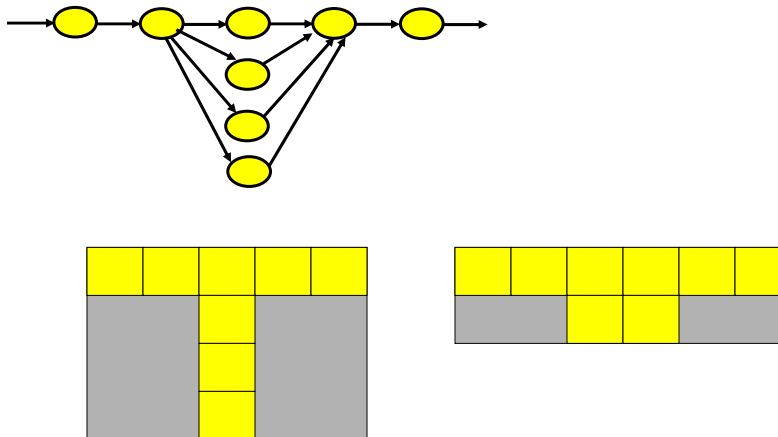
CALTECH CS137 Winter2004 -- DeHon

Schedule Examples



CALTECH CS137 Winter2004 -- DeHon

More Schedule Examples



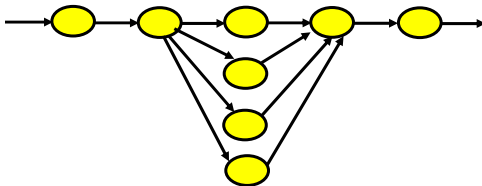
CALTECH CS137 Winter2004 -- DeHon

Scheduling

- **Task:** assign time slots (and resources) to operations
 - **time-constrained:** minimizing peak resource requirements
 - *n.b.* time-constrained, not always constrained to minimum execution time
 - **resource-constrained:** minimizing execution time

CALTECH CS137 Winter2004 -- DeHon

Resource-Time Example



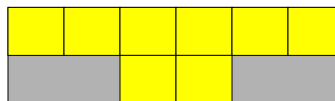
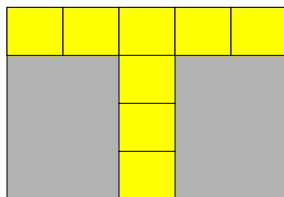
Time Constraint:

$<5 \rightarrow --$

$5 \rightarrow 4$

$6,7 \rightarrow 2$

$>7 \rightarrow 1$



CALTECH CS137 Winter2004 -- DeHon

Scheduling Use

- Very general problem formulation
 - HDL/Behavioral → RTL
 - Register/Memory allocation/scheduling
 - Time-Switched Routing
 - TDMA, bus scheduling, static routing
 - Instruction/Functional Unit scheduling
 - Routing (share channel)
 - Processor tasks

CALTECH CS137 Winter2004 -- DeHon

Two Types (1)

- **Data independent**
 - graph static
 - resource requirements and execution time
 - independent of data
 - schedule statically
 - maybe bounded-time guarantees
 - typical ECAD problem

CALTECH CS137 Winter2004 -- DeHon

Two Types (2)

- **Data Dependent**
 - execution time of operators variable
 - depend on data
 - flow/requirement of operators data dependent
 - if cannot bound range of variation
 - must schedule online/dynamically
 - cannot guarantee bounded-time
 - general case (*i.e.* halting problem)
 - typical “General-Purpose” (non-real-time) OS problem

CALTECH CS137 Winter2004 -- DeHon

Unbounded Problem

- **Easy:**
 - compute ASAP schedule
 - *i.e.* schedule everything as soon as predecessors allow
 - will achieve minimum time
 - won't achieve minimum area
 - (meet resource bounds)

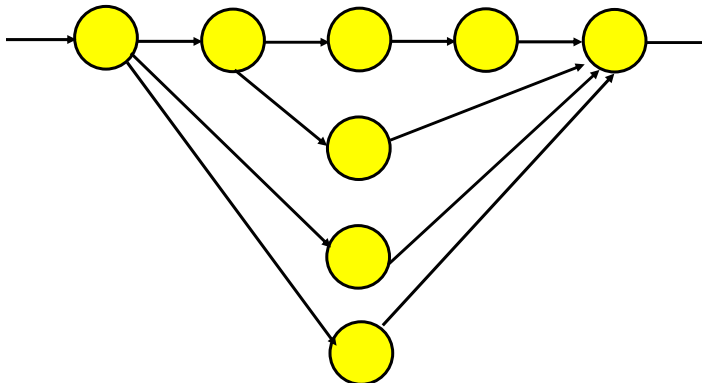
CALTECH CS137 Winter2004 -- DeHon

ASAP Schedule

- For each input
 - mark input on successor
 - if successor has all inputs marked, put in visit queue
- While visit queue not empty
 - pick node
 - update time-slot based on latest input
 - mark inputs of all successors, adding to visit queue when all inputs marked

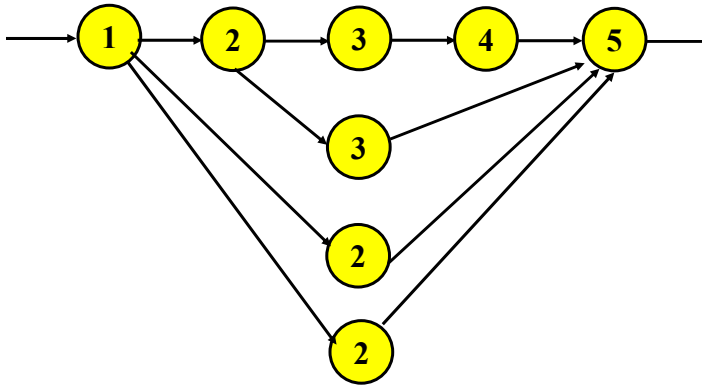
CALTECH CS137 Winter2004 -- DeHon

ASAP Example



CALTECH CS137 Winter2004 -- DeHon

ASAP Example



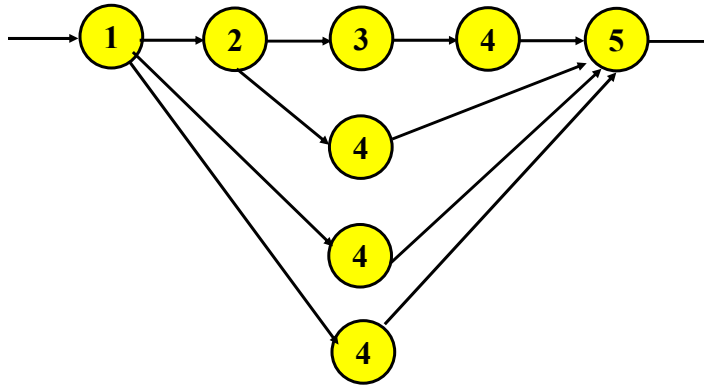
CALTECH CS137 Winter2004 -- DeHon

Also Useful to Define ALAP

- As Late As Possible
- Work backward from outputs of DAG
- Also achieve minimum time w/
unbounded resources

CALTECH CS137 Winter2004 -- DeHon

ALAP Example



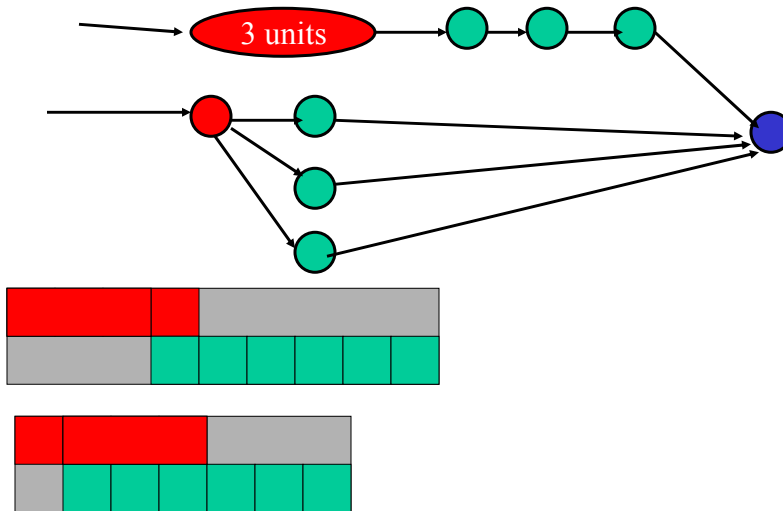
CALTECH CS137 Winter2004 -- DeHon

ALAP and ASAP

- Difference in labeling between ASAP and ALAP is slack of node
 - Freedom to select timeslot
- If $ASAP=ALAP$, no freedom to schedule

CALTECH CS137 Winter2004 -- DeHon

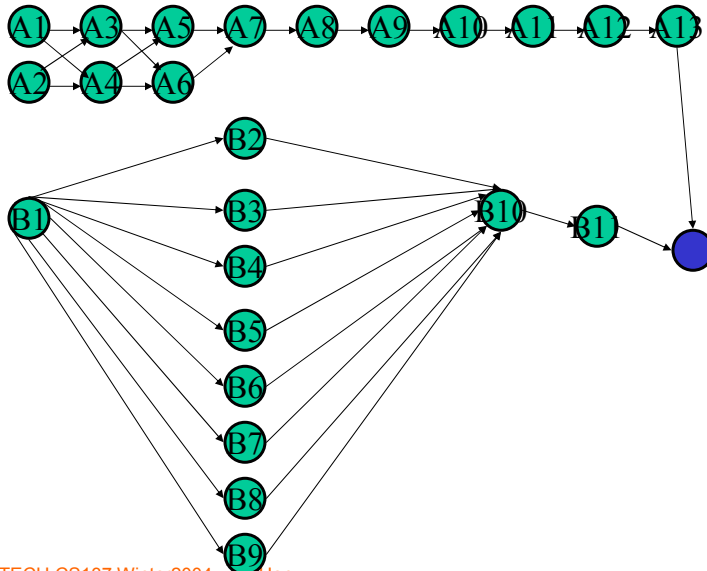
Why hard?



General

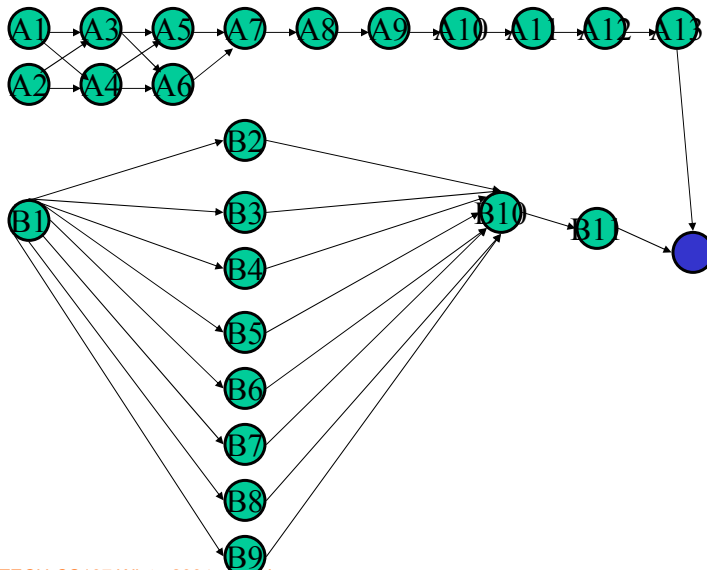
- When selecting, don't know
 - need to tackle **critical path**
 - need to run task to **enable work** (parallelism)
- Can generalize example to single resource case

Single Resource Hard (1)



CALTECH CS137 Winter2004 -- DeHon

Single Resource Hard (2)

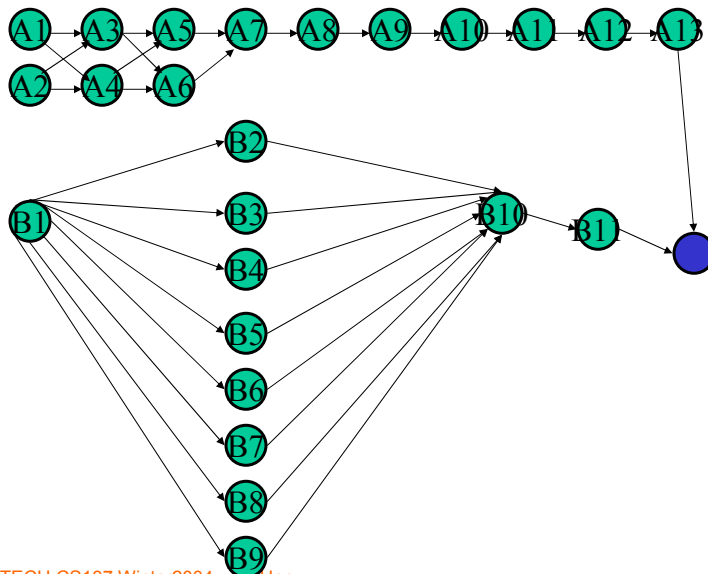


Crit. Path:

A1 A2
 A3 A4
 A5 A6
 A7 B1
 A8 B2
 A9 B3
 A10 B4
 A11 B5
 A12 B6
 A13 B7
 B8 B9
 B10
 B11

CALTECH CS137 Winter2004 -- DeHon

Single Resource Hard (3)

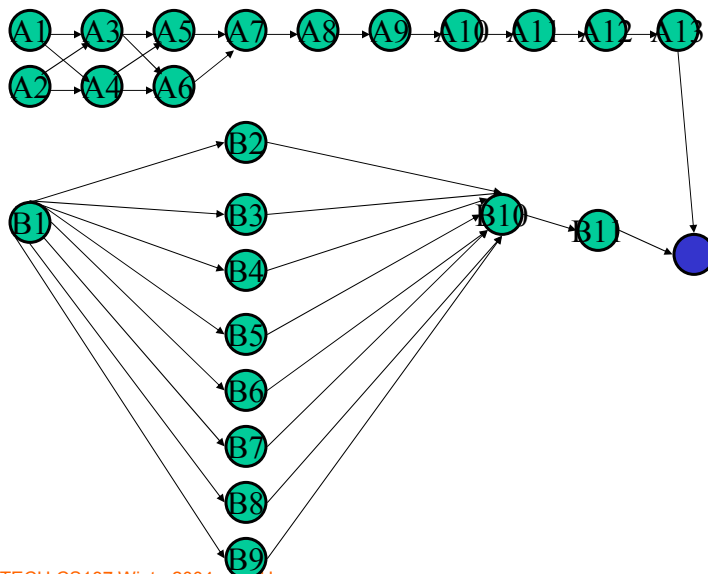


PFirst

- A1 B1
- B2 B3
- B4 B5
- B6 B7
- B8 B9
- A2 B10
- A3 A4
- A5 A6
- A7 B11
- A8
- A9
- A10
- A11
- A12
- A13

CALTECH CS137 Winter2004 -- DeHon

Single Resource Hard (4)

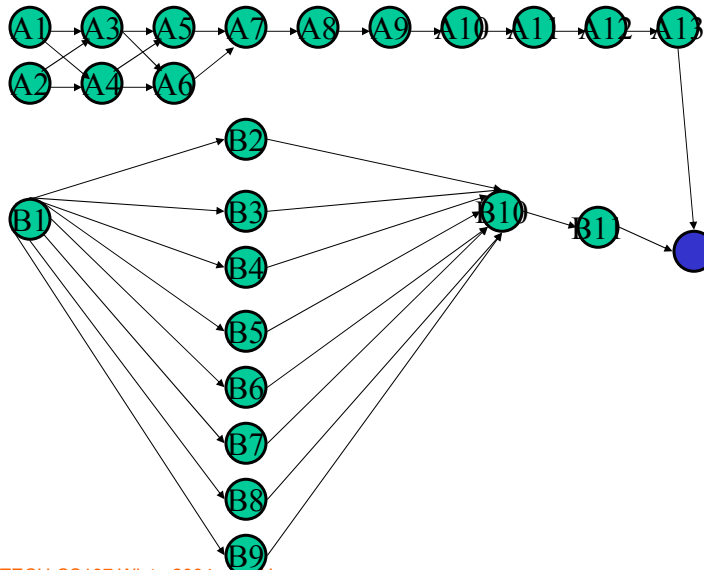


Balance1

- A1 B1
- A2 B2
- A3 B3
- A4 B4
- A5 B5
- A6 B6
- A7 B7
- A8 B8
- A9 B9
- A10 B10
- B11 B11
- A12
- A13

CALTECH CS137 Winter2004 -- DeHon

Single Resource Hard (5)



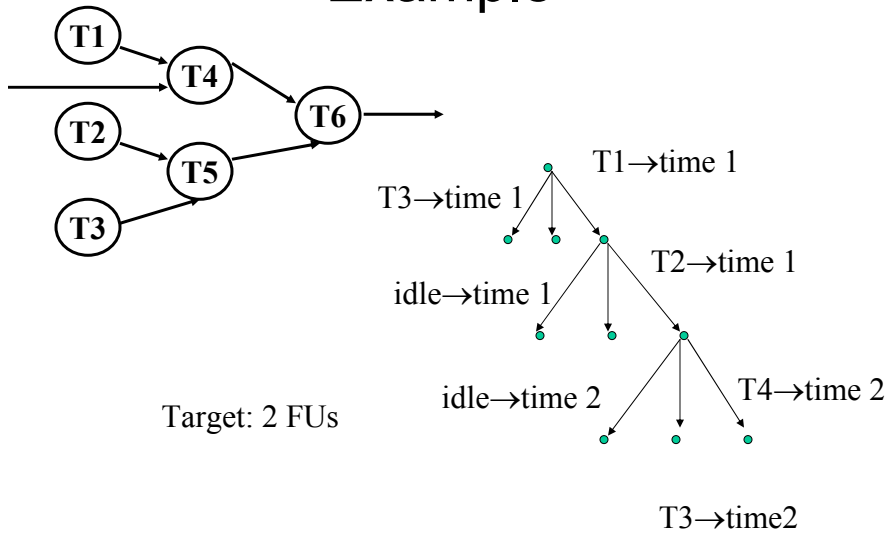
CALTECH CS137 Winter2004 -- DeHon

Brute-Force

- 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

CALTECH CS137 Winter2004 -- DeHon

Example



CALTECH CS137 Winter2004 -- DeHon

Branching Search

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

CALTECH CS137 Winter2004 -- DeHon

Reducing Work

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

CALTECH CS137 Winter2004 -- DeHon

“Equivalent” Schedules

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

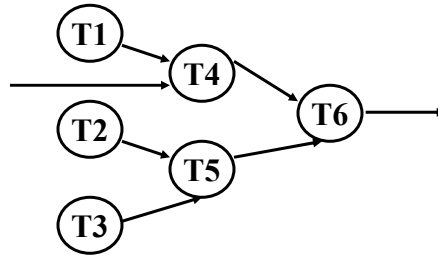
T1	T3
T2	

T2	T3
T1	

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

CALTECH CS137 Winter2004 -- DeHon

“Equivalent” Schedule Prefixes

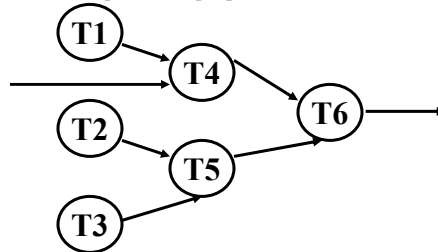


T1	T3
T2	T4

T1	T2
T3	T4

CALTECH CS137 Winter2004 -- DeHon

“Non-Equivalent” Schedule Prefixes



T1	T3
T2	

T2	T1
T3	

CALTECH CS137 Winter2004 -- DeHon

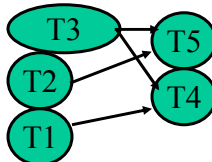
Pruning Prefixes?

- I'm not sure there is an efficient way (general)?
- Keep track of schedule set
 - walk through state-graph of scheduled prefixes
 - unfortunately, set is power-set so 2^N
 - ...but not all feasible, so shape of graph may simplify

CALTECH CS137 Winter2004 -- DeHon

Dominant Schedules

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



T1	T3	T4
T2		T5

T3	T4	
T1	T2	T5

CALTECH CS137 Winter2004 -- DeHon

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:
 - $LB = \text{current schedule time} + \text{lower_bound_estimate}$
 - if LB greater than existing solution, prune

CALTECH CS137 Winter2004 -- DeHon

Pruning Techniques

Establish Lower Bound on schedule time

- Critical Path (ASAP schedule)
- Aggregate Resource Requirements
- Critical Chain

CALTECH CS137 Winter2004 -- DeHon

Critical Path Lower Bound

- ASAP schedule ignoring resource constraints
 - (look at length of remaining critical path)
- Certainly cannot finish any faster than that

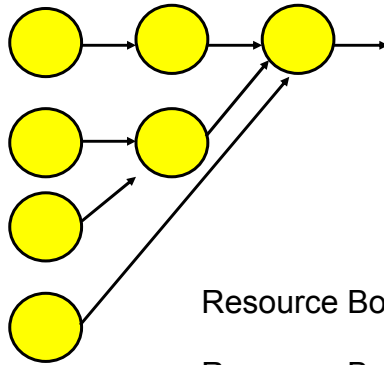
CALTECH CS137 Winter2004 -- DeHon

Resource Capacity Lower Bound

- Sum up all capacity required per resource
- Divide by total resource (for type)
- Lower bound on remaining schedule time
 - (best can do is pack all use densely)
 - Ignores schedule constraints

CALTECH CS137 Winter2004 -- DeHon

Example

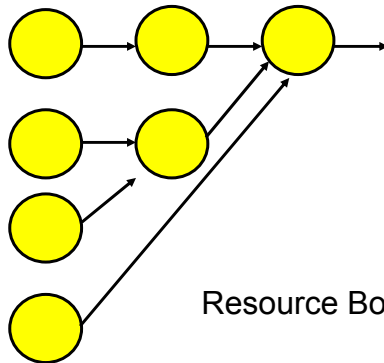


Critical Path

Resource Bound (2 resources)

Resource Bound (4 resources)

Example



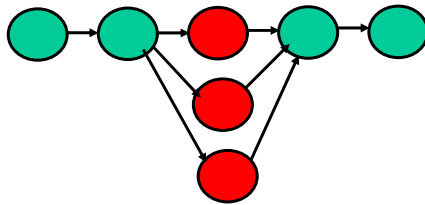
Critical Path 3

Resource Bound (2 resources) $7/2=4$

Resource Bound (4 resources) $7/4=2$

“Critical Chain” Lower Bound

- Bottleneck resource present coupled resource and latency bound

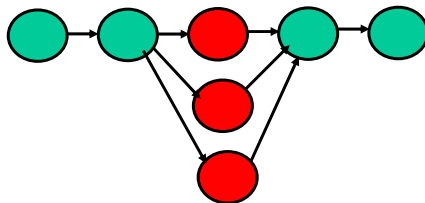


Single red resource

CALTECH CS137 Winter2004 -- DeHon

“Critical Chain” Lower Bound

- Bottleneck resource present coupled resource and latency bound



Single red resource

Critical path 5
Resource Bound (1,1) 4
Critical Chain (1,1) 7

CALTECH CS137 Winter2004 -- DeHon

Lower Bound Pruning

- Typically
 - make move (schedule)
 - recalculate lower bound
 - if best case completion greater than current best
 - prune branch

CALTECH CS137 Winter2004 -- DeHon

Search Ordering

- Performance sensitive to ordering
 - e.g. if find best path first, easy to prune
 - paths in order of decreasing length, cannot
- Depth first
 - get complete time for bound
 - may find long paths first
- Breadth first
 - don't have complete path for bound
 - lots of intermediate data to keep
 - ultimately finds short paths before long

CALTECH CS137 Winter2004 -- DeHon

Search Ordering

- Mixture of depth and breadth
- Use heuristics to select nodes for expansion
- If heuristics help find good solutions
 - helps prune
 - focus attention in search

CALTECH CS137 Winter2004 -- DeHon

Alpha-Beta Search

- Generalization
 - keep both upper and lower bound estimates on partial schedule
 - 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)

CALTECH CS137 Winter2004 -- DeHon

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”

CALTECH CS137 Winter2004 -- DeHon

Upper Bounds

- Determine upper bounds by heuristics/approximations
 - talk about next time

CALTECH CS137 Winter2004 -- DeHon

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
- Not sure we've covered enough techniques today to make scheduling (in general) polynomial time in practice

CALTECH CS137 Winter2004 -- DeHon

Adapt Time Constrained

- Vary resources and run as-is
- Binary search for minimum resources achieving time target
- Use lower bounds to define region of interest to search
- Use lower bounds to prune/exit search early if won't meet target

CALTECH CS137 Winter2004 -- DeHon

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

CALTECH CS137 Winter2004 -- DeHon

Summary

- Resource sharing saves area
 - allows us to fit in fixed area
- Requires that we schedule tasks onto resources
- General kind of problem arises
- Branch-and-bound search
 - “equivalent” states
 - dominators
 - estimates/pruning

CALTECH CS137 Winter2004 -- DeHon

Admin

- Assignment #4 Due Friday
- Reading
 - Today/tomorrow (received W)
 - Next Monday Gerez
- Next/final three classes all meet

CALTECH CS137 Winter2004 -- DeHon

Big Ideas:

- Exploit freedom in problem to reduce costs
 - (slack in schedules)
- Use dominating effects
 - (constrained resources)
- Use dominators to reduce work
- Technique: Search
 - Branch-and-Bound
 - Alpha-Beta

CALTECH CS137 Winter2004 -- DeHon