ESE535:
Electronic Design Automation

Day 5: February 2, 2009
Architecture Synthesis
(Provisioning, Allocation)

Today

- Problem
- Brute-Force/Exhaustive
- Greedy
- Estimators
- Analytical Provisioning
- ILP Schedule and Provision

Previously

- General formulation for scheduled operator sharing
  - VLIW
- Fast algorithms for scheduling onto fixed resource set
  - List Scheduling

Today: Provisioning

- Given
  - An area budget
  - A graph to schedule
  - A Library of operators
- Determine:
  - Best (delay minimizing) set of operators
  - i.e. select the operator set

Exhaustive

1. Identify all area-feasible operator sets
   - E.g. preclass exercise
2. Schedule for each
3. Select best

- → optimal
- Drawbacks?

Exhaustive

- How large is space of feasible operator sets?
  - As function of
    - operator types – N
      - Types: add, multiply, divide, ….
    - Maximum number of operators of type M

Size of Feasible Space

- Consider 10 operators
  – For simplicity all of unit area
- Total area of 100

- How many cases?

Implication

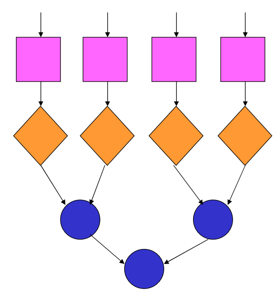
- Feasible operator space can be too large to explore exhaustively

Greedy Incremental

- Start with one of each operator
- While (there is area to hold an operator)
  – Which single operator
    - Can be added without exceeding area limit?
    - And Provides largest benefit?
  – Add one operator of that type

- How long does this run?
- Weakness?

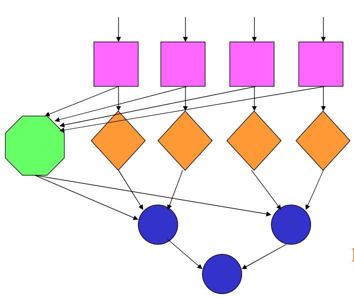
Example



Find best 5 operator solution.

original:
not quite demo.

Example



Find best 6 operator solution.

Review if this captures

Estimators

- Scheduling expensive
  – O(|E|) or O(|E|*log(|V|)) using list-schedule
- Results not analytic
  – Cannot write an equation around them

- Saw earlier bounds sometimes useful
  – No precedence → is resource bound
  – Often one bound dominates

Estimations

- Step 1: estimate with resource bound
  - O(|E|) vs. O(N) evaluation
- Step 2: use estimate in equations
  - $T=\max(N_1/R_1, N_2/R_2, \ldots.)$

Constraints

- Let A_i be area of operator type i
- Let x_i by number of operators of type i

$$\sum A_i \times x_i \leq Area$$

Achieve Time Target

- Want to achieve a schedule in T cycles
- Each resource bound must be less than T cycles:
  - $N_i/x_i \leq T$

Algebraic Solve

- Set of equations
  - $N_i/x_i \leq T$
  - $\Sigma A_i x_i \leq Area$
- Assume equality
- $N_i/x_i=T$ ➔ $x_i=N_i/T$

$$\frac{\sum A_i \times N_i}{T} \leq Area$$

Rearranging

$$\frac{\sum A_i \times N_i}{T} \leq Area$$

$$\frac{\sum A_i \times N_i}{Area} \leq T$$

Back Substitute

- $x_i=N_i/T$

$$\frac{\sum A_i \times N_i}{Area} \leq T$$

- x_i won't necessarily be integer
  - Round down definitely feasible solution
  - May have room to move a few up by 1

ILP

- Integer Linear Programming
- Formulate set of linear equation constraints (inequalities)
  - $Ax_0+Bx_1+Cx_2 \leq D$
  - $x_0+x_1=1$
  - A,B,C,D – constants
  - x_i – variables to satisfy
- Can constrain variables to integers
- No polynomial time guarantee
  - But often practical
  - Solvers exist

ILP Provision and Schedule

- Possible to formulate whole operator selection and scheduling as ILP problem

Formulation

- Integer variables M_i
  - number of operators of type i
- 0-1 (binary) variables $x_{i,j}$
  - 1 if node I is scheduled into timestep j
  - 0 otherwise
- Variable assignment completely specifies schedule
- This formulation for achieving a target time T
  - j ranges 0 to T-1

Target T → Min T

- Formulation targets T
- But do we know T?
- Do binary search for minimum T
  - How does that impact solution time?

Constraints

1. Total area constraints
2. Not assign too many things to a timestep
3. Assign every node to some timestep
4. Maintain precedence

(1) Total Area

- Same as before

$$\sum A_i \times M_i \leq Area$$

(2) Not overload timestep

- For each timestep j
  - For each operator type k

$$\sum_{o_i \in FU_k} x_{i,j} \le M_k$$

(3) Node is scheduled

- For each node in graph

$$\sum_{j} x_{i,j} = 1$$

Can narrow to sum over slack window.

(4) Precedence Holds

- For each edge from node i to node k

$$\sum_{j} j \times x_{i,j} - \sum_{j} j \times x_{k,j} \le -1$$

Can narrow to sum over slack windows.

Round up Algorithms and Runtimes

- Exhaustive Schedule
- Exhaustive Resource Bound Estimate
- Greedy Schedule
- LP on estimates
  - Particular time bound
  - Minimize time
- ILP on estimates and exact
  - Particular time bound
  - Minimize time

Admin

- Assignment 2 out
  - Programming assignment
  - Now in two pieces
- Reading on web

Big Ideas:

- Estimators
- Dominating Effects
- Reformulate as a problem we already have a solution for
  - ILP
- Technique: Greedy
- Technique: ILP