CS137: Electronic Design Automation

Day 4: Jan 18, 2006
Concept Generation

Today

- Specifications/requirements/goals
- Concept Generation

Specification

- Before can start solving
  - Need to know what your solving
- Before can evaluate "goodness"
  - Need to know what trying to accomplish
  - Priorities

Quantitative Specification

- Typically need to
  - turn vague specification
  - into something quantifiable/measurable
    - Something concrete

Specifications

- More of a trick/challenge for things that don't have natural metrics
  - From:
    - Door closes easily
  - To:
    - Door latches with <5 lbs. of force…
  - From: reliable
  - To: 1 failure in 10^9 hours of operation
    - 1 undetected failure …

Our Specifications

- [talk about]

Establish Priorities

- Which requirement is king?
- Which have flexibility?
  - Can tradeoff?

Establish Targets

- Acceptable Ranges
- Ideal
- Usually Relative to alternatives/competitors

Our Targets and Priorities

- [sketch]
  - Delay in Solution
  - Area of logic in Solution
  - Energy of Solution
  - Reliability of Solution
  - Runtime of algorithm
  - Area for hardware engine to solve problem
- Your problem formulation should capture priorities

Concept Generation

Goal of Concept Generation

- Fully explore design space
  - Not miss options
  - Open minds to possibilities
  - Know what can do
    - Competitors can do

Components (Outline)

- Structured/Systematic Approach
- Partial Solutions
- Uncertainty/Unknown
- Documentation

Non-Atomic Insight

Nothing will ever be attempted, if all possible objections must be first overcome.

--- Samuel Johnson, 1759.

Uncertainty

"Do you remember what I have taught you?
Most intelligent beings prefer to live in certainty than uncertainty. Rather than accept uncertainty, they will discount the input of their own senses. It is through this mechanism that mages manipulate the perception of others."
--- Elric to Galen upon his [Elric's] death

Invoking the Darkness, Jeanne Cavelos

(This slide intentionally left [almost] blank in this copy)

Concept Generation: Steps

- Start refined problem specification

1. Clarify/decompose
2. Search
   - Externally
   - Internally / brainstorm
3. Explore Systematically
4. Reflect

Decompose

- Separate components must solve
  - Divide and conquer
- Maybe attack w/ separate passes through process
- Focus on bottleneck
  - Critical path/subproblem
  - Where most risk is
    - Most innovation needed

Search Externally

- Literature (standing on the shoulders of giants)
  - Published / library / IBID
  - Web
  - Patents
  - Pattern Catalog
- Customers
- Experts
- Benchmark related / competitive
- ➔Understand state-of-art

Search Internally/Brainstorm

- Common bugs:
  - Reject things too early
    - "Nothing…" quote
  - Latch onto few ideas
    - Get stuck in own local minima

Search Internally/Brainstorm

- Different mode of thinking
  - "Pattern" of thinking
- Generate lots of ideas
- Suspend judgment / no squashing
- Barr: grow mode / acid mode
- Find good pieces, incomplete ideas
- Wishful thinking
- Comfortable with uncertainty

Brainstorm

- Throw out ideas
- Write them down … draw pictures
- De-personalize
  - Name by characteristics not person
  - Not attacking person/idea
  - Group ownership
  - Everyone contribute to debugging
- Mix-and-match
- Good science
  - Find best ideas, avoid NIH

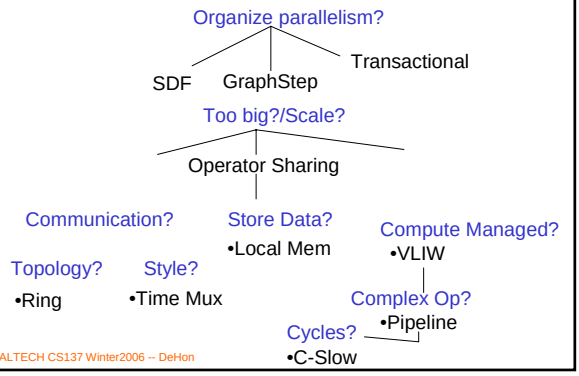
People and Brainstorming

- More brains generally good
  - Maybe limit size of group at a time
  - Different perspectives good
- Group ownership of final product
  - Involve stake holders
  - Build confidence in conclusion
  - Get buy in

Systematic Exploration

- Parameterized Design Space
- Concept combination matrix
- Classification tree
- Architecture/Pattern Tree
- Goals:
  - Find holes
  - Over-emphasis
  - Make sure think about all cross products..

Design Explore for SMVM



Reflect

- …really throughout
- Comfortable solving problem?
  - Specification adequate?
- Process

Time Permitting

Design Examples

SMVM

Problem

- Compute: x=Ab

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} A_{0,0} & A_{0,1} & A_{0,2} & A_{0,3} & A_{0,4} \\ A_{1,0} & A_{1,1} & A_{1,2} & A_{1,3} & A_{1,4} \\ A_{2,0} & A_{2,1} & A_{2,2} & A_{2,3} & A_{2,4} \\ A_{3,0} & A_{3,1} & A_{3,2} & A_{3,3} & A_{3,4} \\ A_{4,0} & A_{4,1} & A_{4,2} & A_{4,3} & A_{4,4} \end{bmatrix} \times \begin{bmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \\ B_4 \end{bmatrix}$$

- Iterative: $b^i=x^{i-1}$

Parallelism?

- Compute: x=Ab

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} A_{0,0} & A_{0,1} & A_{0,2} & A_{0,3} & A_{0,4} \\ A_{1,0} & A_{1,1} & A_{1,2} & A_{1,3} & A_{1,4} \\ A_{2,0} & A_{2,1} & A_{2,2} & A_{2,3} & A_{2,4} \\ A_{3,0} & A_{3,1} & A_{3,2} & A_{3,3} & A_{3,4} \\ A_{4,0} & A_{4,1} & A_{4,2} & A_{4,3} & A_{4,4} \end{bmatrix} \times \begin{bmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \\ B_4 \end{bmatrix}$$

1. Dot products independent

$$x_i = A_i b$$

2. Multiplies within dot product independent
3. Sum of dot products associative?
   - Maybe **not** for floating point

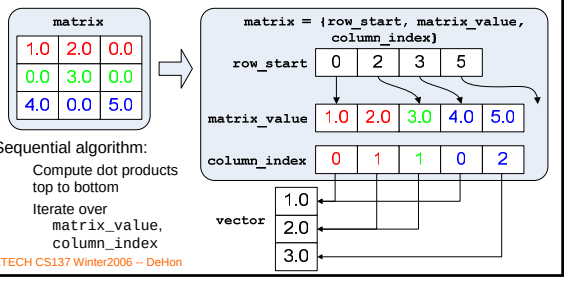
(Special) Problem Characteristics

- Matricies can be large
  - N×N with N in thousands
- Matrix A is sparse
  - *E.g.* < 100 non-zero entries in a row

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} A_{0,0} & A_{0,1} & 0 & 0 & 0 \\ A_{1,0} & A_{1,1} & A_{1,2} & 0 & 0 \\ 0 & A_{2,1} & A_{2,2} & 0 & A_{2,4} \\ 0 & 0 & 0 & A_{3,3} & 0 \\ 0 & A_{4,1} & 0 & A_{4,3} & A_{4,4} \end{bmatrix} \times \begin{bmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \\ B_4 \end{bmatrix}$$

Compressed Sparse Row (CSR) matrix representation

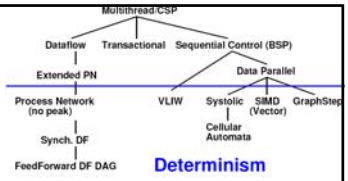
1. Remove zeros
2. Concatenate rows
3. Store column indices
4. Store row starting positions



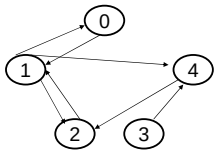
Sequential algorithm:

- Compute dot products top to bottom
- Iterate over `matrix_value`, `column_index`

What Architecture is Appropriate?



- Accept DP FP not assoc.
  - Serialize dot products



  - ➔ BSP>Data Parallel>GraphStep (SDF)

Design Explore

Organize parallelism?

SDF GraphStep Transactional

Too big?/Scale?

- Coarse-Grained Time Multiplexing
- Fine-Grained Time Multiplexing
- Common Element Sharing for Regular Graphs
- Common Operator Sharing for General Graphs
- Scheduled Operator Sharing
- Modulo Scheduling
- Multicontext (physical)
- Datapath Sizing and Serialization
- Synthesis Objective Function Tradeoffs
- Function Unit Binding
- Sequential vs. Parallel Implementation

Design Explore

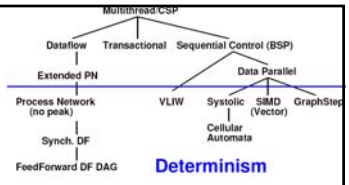
Organize parallelism?

SDF GraphStep Transactional

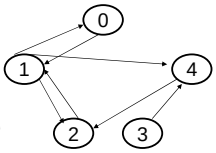
Too big?/Scale?

- Coarse-Grained Time Multiplexing
- Fine-Grained Time Multiplexing
- Common Element Sharing for Regular Graphs
- **Common Operator Sharing for General Graphs**
  - **Homogeneous FP MPY/ADD**
- Scheduled Operator Sharing
- **Modulo Scheduling**
- Multicontext (physical)
- **Datapath Sizing and Serialization**
- Synthesis Objective Function Tradeoffs
- Function Unit Binding
- Sequential vs. Parallel Implementation

Scaling?



- Accept DP FP not assoc.
  - Serialize dot products
  - Share operator for multiple dot products
  - Scaling: number of dot products/PE



Design Explore

Organize parallelism?

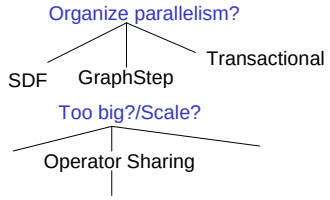
SDF GraphStep Transactional

Too big?/Scale?

Operator Sharing

Communication? Store Data? Compute Managed?

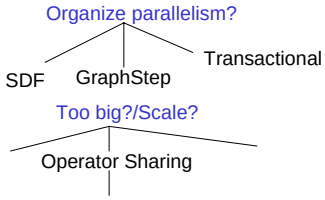
Design Explore



Communication? Store Data? Compute Managed?

- Shared vs. Distributed Memory
- On-Chip vs. Off-chip?
- …
- ➔ on chip block rams

Design Explore

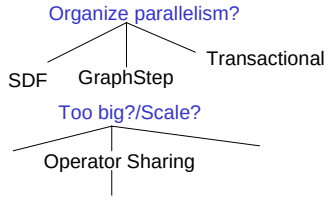


Communication? Store Data? Compute Managed?

Topology?
- Mesh
- Tree
- Ring/Bus

Style?
- Configured
- Packet Switch
- Time Mux

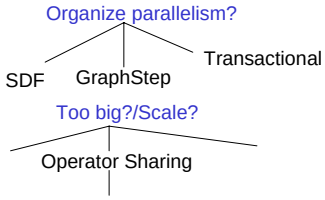
Design Explore



Communication? Store Data? Compute Managed?

Revisit Arch. Decisions
- Dataflow
- Sequenced
  - VLIW

Design Explore



Communication? Store Data? Compute Managed?

Store Data?
- Local Mem

Compute Managed?
- VLIW

Topology?
- Ring

Style?
- Time Mux

Complex Op?
- Pipeline

Cycles?
- C-Slow

Discuss

- Assumptions
- Refine/elaborate specification
- Decompose problem

Admin

- Friday
  - No class
  - Problem Formulation Due
- Monday: brainstorming session
- Wednesday: Nachiket on GraphStep programming