

CSCI 8150  
Advanced Computer Architecture

Hwang, Chapter 2  
Program and Network Properties  
2.3 Program Flow Mechanisms

### Program Flow Mechanisms

- ❖ Conventional machines used control flow mechanism in which order of program execution explicitly stated in user programs.
- ❖ Dataflow machines which instructions can be executed by determining operand availability.
- ❖ Reduction machines trigger an instruction's execution based on the demand for its results.

### Control Flow vs. Data Flow

- ❑ Control flow machines used shared memory for instructions and data. Since variables are updated by many instructions, there may be side effects on other instructions. These side effects frequently prevent parallel processing. Single processor systems are inherently sequential.
- ❑ Instructions in dataflow machines are unordered and can be executed as soon as their operands are available; data is held in the instructions themselves. Data tokens are passed from an instruction to its dependents to trigger execution.

### Data Flow Features

- ❖ No need for
  - ❑ shared memory
  - ❑ program counter
  - ❑ control sequencer
- ❖ Special mechanisms are required to
  - ❑ detect data availability
  - ❑ match data tokens with instructions needing them
  - ❑ enable chain reaction of asynchronous instruction execution

### A Dataflow Architecture - 1

- ❑ The Arvind machine (MIT) has NPEs and an N-by-N interconnection network.
- ❑ Each PE has a token-matching mechanism that dispatches only instructions with data tokens available.
- ❑ Each datum is tagged with
  - ❑ address of instruction to which it belongs
  - ❑ context in which the instruction is being executed
- ❑ Tagged tokens enter PE through local path (pipelined), and can also be communicated to other PEs through the routing network.

### A Dataflow Architecture - 2

- ❖ Instruction address(es) effectively replace the program counter in a control flow machine.
- ❖ Context identifier effectively replaces the frame base register in a control flow machine.
- ❖ Since the dataflow machine matches the data tags from one instruction with successors, synchronized instruction execution is implicit.

### A Dataflow Architecture - 3

- ❑ An *I-structure* in each PE is provided to eliminate excessive copying of data structures.
- ❑ Each word of the I-structure has a two-bit tag indicating whether the value is empty, full, or has pending read requests.
- ❑ This is a retreat from the pure dataflow approach.
- ❑ Example 2.6 shows a control flow and dataflow comparison.
- ❑ Special compiler technology needed for dataflow machines.

### Demand-Driven Mechanisms

- ❑ Data-driven machines select instructions for execution based on the availability of their operands; this is essentially a bottom-up approach.
- ❑ Demand-driven machines take a top-down approach, attempting to execute the instruction (a *demand*) that yields the final result. This triggers the execution of instructions that yield its operands, and so forth.
- ❑ The demand-driven approach matches naturally with functional programming languages (e.g. LISP and SCHEME).

### Reduction Machine Models

- ❑ String-reduction model:
  - ⊃ each demander gets a separate copy of the expression string to evaluate
  - ⊃ each reduction step has an operator and embedded reference to demand the corresponding operands
  - ⊃ each operator is suspended while arguments are evaluated
- ❑ Graph-reduction model:
  - ⊃ expression graph reduced by evaluation of branches or subgraphs, possibly in parallel, with demanders given pointers to results of reductions.
  - ⊃ based on sharing of pointers to arguments; traversal and reversal of pointers continues until constant arguments are encountered.

### Summary

- ❑ Control flow machines give complete control, but are less efficient than other approaches.
- ❑ Data flow (eager evaluation) machines have high potential for parallelism and throughput and freedom from side effects, but have high control overhead, lose time waiting for unneeded arguments, and difficulty in manipulating data structures.
- ❑ Reduction (lazy evaluation) machines have high parallelism potential, easy manipulation of data structures, and only execute required instructions. But they do not share objects with changing local state, and do require time to propagate tokens.