Topics Laundry List

CMCS22620, Spring 2004
March 29, 2004

• MiniJava
• type checking, recursive definitions, symbol tables
  – subtyping
  – structural equivalence vs. name equivalence
• representation of objects and values
  – scalars
  – arrays
  – Pascal records or C structs
  – C unions
  – discriminated unions (e.g., ML datatypes
  – objects
  – closures (e.g., ML’s first-class function values)
• stack vs. heap vs. static allocation
• translation to trees
  – high-level optimizations
  – canonicalization
• runtime environment
  – calling conventions
• parameter-passing conventions
• result-passing conventions
• activation records and stack layout
  – more stack organization
    • frames
    • without nested functions
    • with nested functions
    • with local variables that “outlive” their function invocation (e.g., because of an address-of operator or first-class functions)
    • with objects and methods

• machine instructions
  – CISC vs. RISC
  – registers
  – addressing modes

• instruction selection
  – “maximal munch”
  – dynamic programming

• basic blocks and traces

• simple local optimizations (e.g., intra-block value numbering)

• flow analysis

• global (flow-based) optimizations

• loops and dominators, loop-based optimizations

• other intermediate languages
  – static single assignment (SSA) form
  – continuation-passing style (CPS)
  – inter-procedural optimizations
- pipelining and scheduling
- branch prediction
- liveness analysis
- (graph-coloring) register allocation
  - with coalescing
- generating assembly code
- garbage collection
- parser error recovery(?)