

- Everything covered in class since the midterm in detail
- The assumption is you know everything covered before that
- Questions about how is such-and-such done in Tiger compiler are legit
  - I won't ask you to write ML code though
- List on following slides IS NOT complete
  - everything covered in class is fair game

## new topics:

- Intermediate Representation
- Instruction Selection
- Basic Blocks and Traces
- Optimizations
- Liveness
- Dataflow analysis
- Register allocation
- Instruction scheduling

## old topics:

- Compilers
- Lexical analysis
- Parsing
- Abstract Syntax Tree
- Symbol Tables
- Type checking and Semantic Analysis
- Stack Frames
- Procedure Calling Conventions

## Intermediate Representation

- what and why? where is the line between source code/IR/machine insts.
- expressions, statements and conditionals
- simple variables to function calls
  - what's interesting
  - e.g., - give intermediate tree representation for array creation
- conditionals, special cases, & and | operations...
- while, for loops; case statements
- function calls, external functions, procedure `prologue/epilogue`

## Basic Blocks

- getting rid (pulling) of SEQ/ESEQ, CALL  
... why? what is the objective of steps in canon?
- $\text{BINOP}(op, e1, (\text{ESEQ}(s, e2))) \implies$   
     $\text{ESEQ}(\text{MOVE}(\text{TEMP } tnew, e1),$   
     $\text{ESEQ}(s, (\text{BINOP}(op, \text{TEMP } tnew, e2)))$
- true label/false label ... generating optimal traces...

## Optimizations

- what to optimize? scope?
- optimize a piece of code
- steps to ensure constant propagation (or other optimization)
- peephole optimization

## Instruction Selection

- why tiles? optimal vs. optimum tiling
- maximal munch
- dynamic programming
- tree grammars
- given a IR tree, find optimum/al tiling
- RISC vs. CISC inst sets
  - pros and cons of having CISC options

## Liveness

- what info are we gathering
- outline algorithm
- static vs. dynamic info
- halting problem
- graph construction and move instruction
- example – construct graph and do analysis

## Data Flow Analysis

- sample DFAs and reasons to perform analysis
- backward/forward; pessimistic/optimistic – what does this mean
- given a property what do DFA equations look like
- example for reaching defs, available exprs...

## Register Allocation

- reg. alloc == graph coloring
- understand algorithm
- actual vs. potential spill
- coalescing strategy -> objective...
- spilling decision
- precoloring nodes – how and why
- solve reg. alloc for sample code
- 'local' reg. allocation
- implementation issues

## Instruction Scheduling

- understand problem
- understand solution approach
  - without resource bounds
  - with resource bounds
- dealing with conditionals, branch prediction
- compiler vs. architecture solution
- loop rewriting
- solve problem

- Final:
  - Wed., Dec. 8, 2:50-5:40pm
  - closed book, closed notes
  - will have grade for final exam by Fri afternoon, can email you upon request
  - will have final grade for class sometime Sunday
- Project grades:
  - IR promised tomorrow ready
  - IS ...
  - final
- Office hours
  - next week: Mon. after 1pm, Tue. 12-1
  - Brad will have regular office hours this week