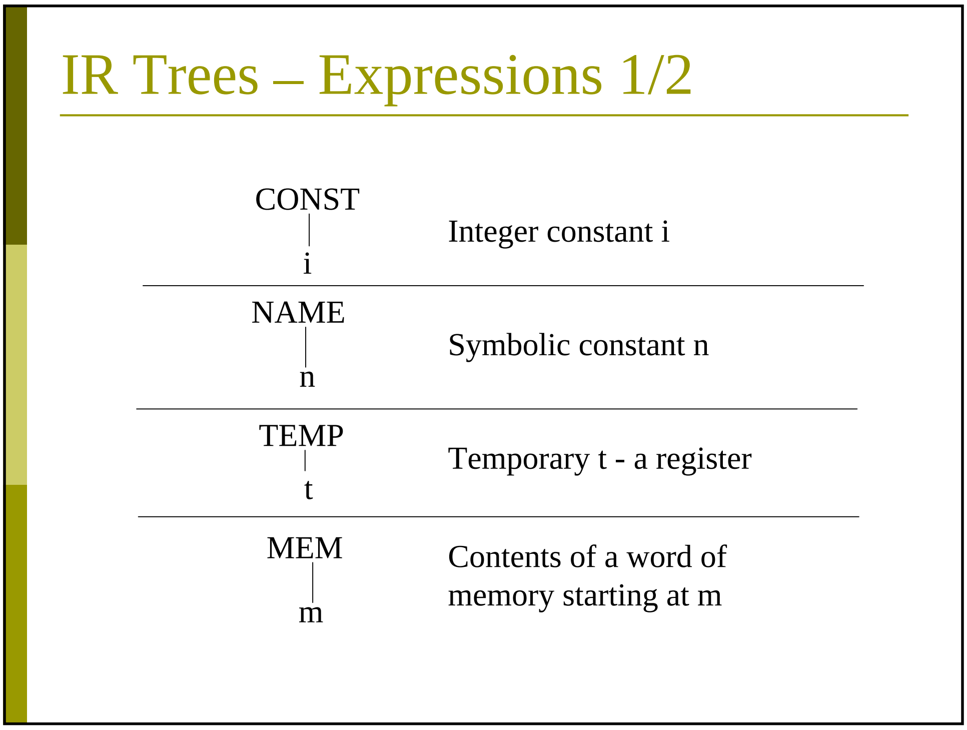
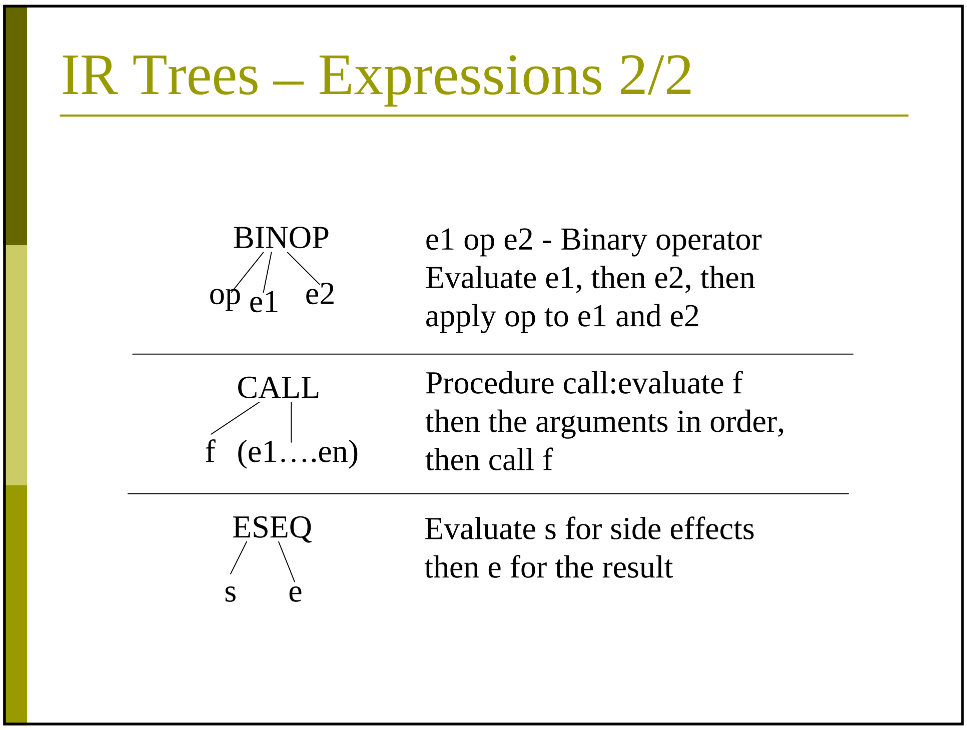


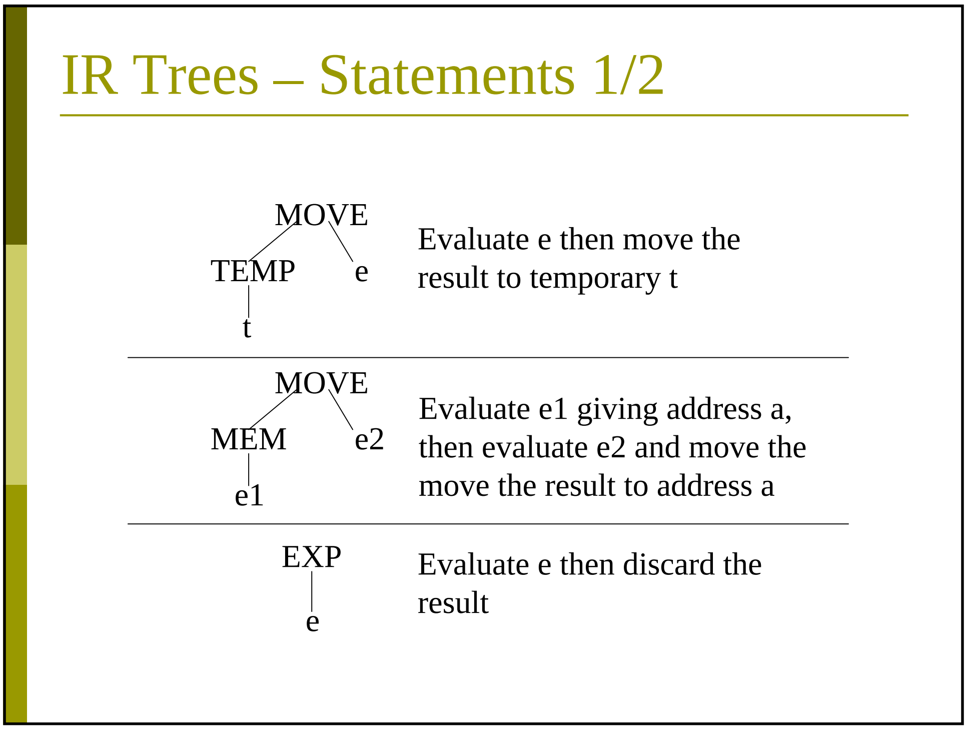
IR Trees – Expressions 1/2

CONST i	Integer constant i
NAME n	Symbolic constant n
TEMP t	Temporary t - a register
MEM m	Contents of a word of memory starting at m



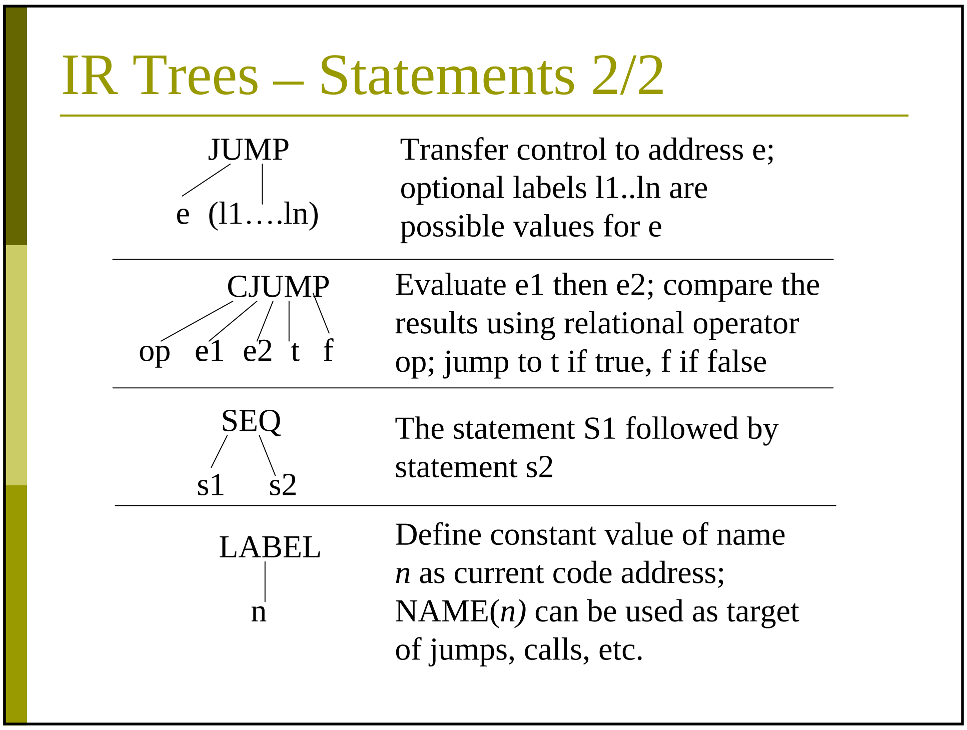
IR Trees – Expressions 2/2

BINOP op e1 e2	e1 op e2 - Binary operator Evaluate e1, then e2, then apply op to e1 and e2
CALL f (e1….en)	Procedure call:evaluate f then the arguments in order, then call f
ESEQ s e	Evaluate s for side effects then e for the result



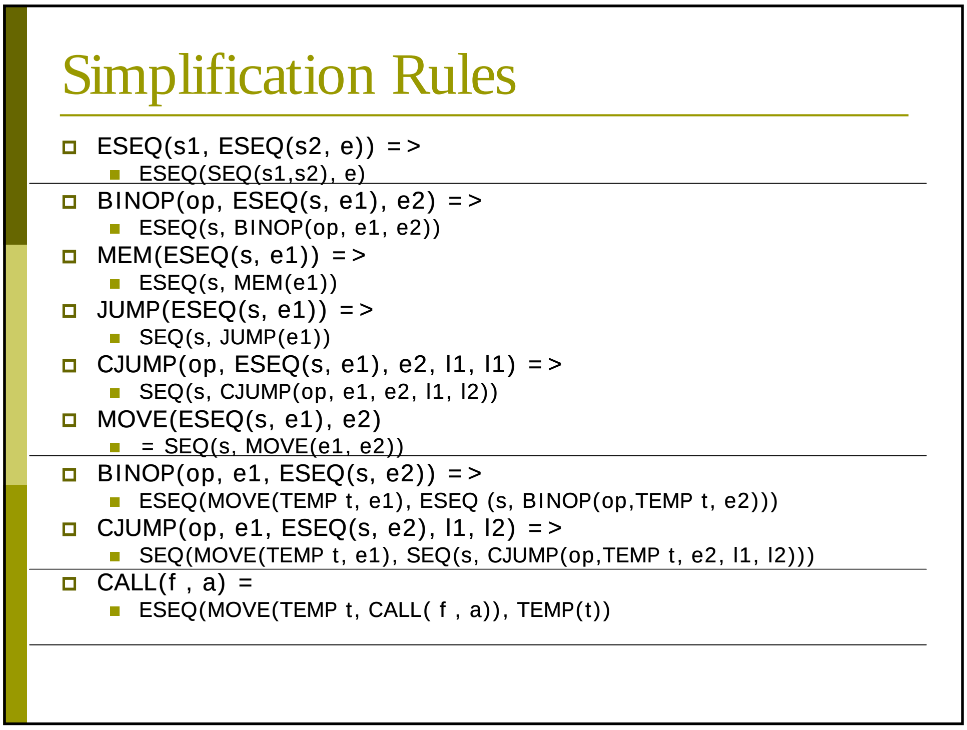
IR Trees – Statements 1/2

MOVE TEMP e t	Evaluate e then move the result to temporary t
MOVE MEM e2 e1	Evaluate e1 giving address a, then evaluate e2 and move the move the result to address a
EXP e	Evaluate e then discard the result



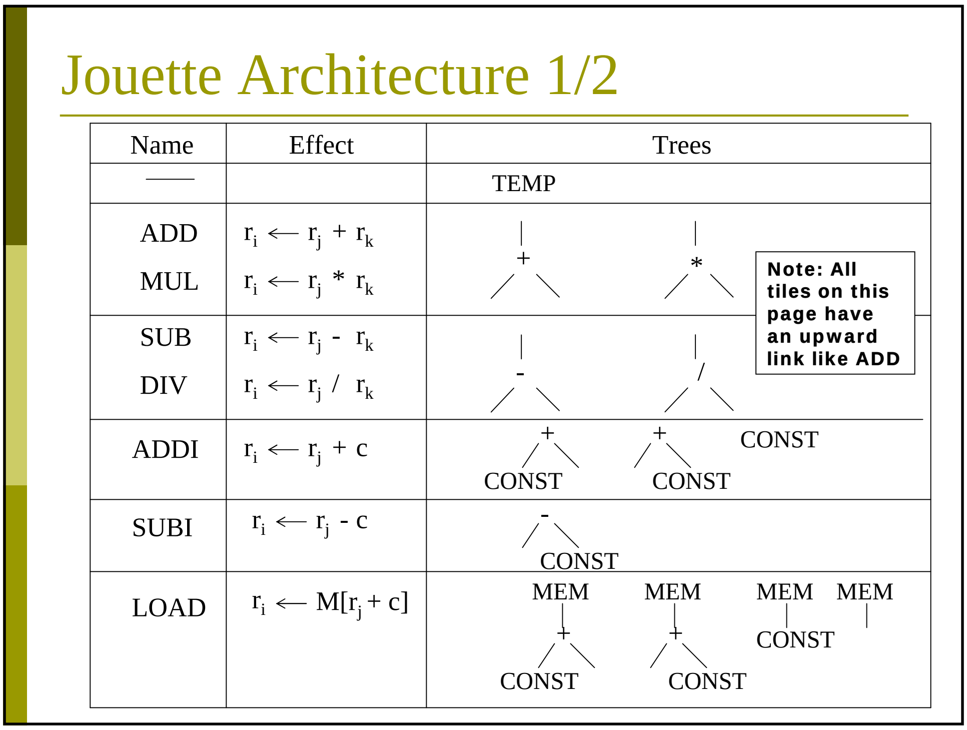
IR Trees – Statements 2/2

JUMP e (l1….ln)	Transfer control to address e; optional labels l1..ln are possible values for e
CJUMP op e1 e2 t f	Evaluate e1 then e2; compare the results using relational operator op; jump to t if true, f if false
SEQ s1 s2	The statement S1 followed by statement s2
LABEL n	Define constant value of name *n* as current code address; NAME(*n)* can be used as target of jumps, calls, etc.



Simplification Rules

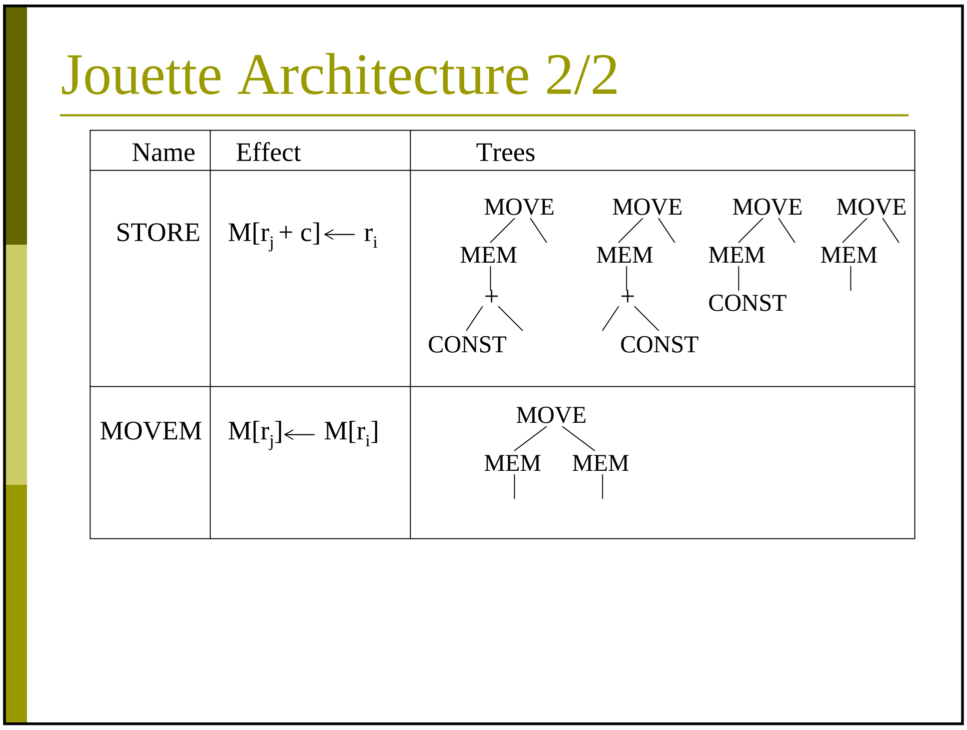
- ESEQ(s1, ESEQ(s2, e)) =>
  - ESEQ(SEQ(s1,s2), e)
- BINOP(op, ESEQ(s, e1), e2) =>
  - ESEQ(s, BINOP(op, e1, e2))
- MEM(ESEQ(s, e1)) =>
  - ESEQ(s, MEM(e1))
- JUMP(ESEQ(s, e1)) =>
  - SEQ(s, JUMP(e1))
- CJUMP(op, ESEQ(s, e1), e2, l1, l1) =>
  - SEQ(s, CJUMP(op, e1, e2, l1, l2))
- MOVE(ESEQ(s, e1), e2)
  - = SEQ(s, MOVE(e1, e2))
- BINOP(op, e1, ESEQ(s, e2)) =>
  - ESEQ(MOVE(TEMP t, e1), ESEQ (s, BINOP(op,TEMP t, e2)))
- CJUMP(op, e1, ESEQ(s, e2), l1, l2) =>
  - SEQ(MOVE(TEMP t, e1), SEQ(s, CJUMP(op,TEMP t, e2, l1, l2)))
- CALL(f , a) =
  - ESEQ(MOVE(TEMP t, CALL(f , a)), TEMP(t))



Jouette Architecture 1/2

Name	Effect	Trees
——		TEMP
ADD MUL	$r_i \leftarrow r_j + r_k$ $r_i \leftarrow r_j * r_k$	+ *
SUB DIV	$r_i \leftarrow r_j - r_k$ $r_i \leftarrow r_j / r_k$	- /
ADDI	$r_i \leftarrow r_j + c$	+ CONST + CONST CONST
SUBI	$r_i \leftarrow r_j - c$	- CONST
LOAD	$r_i \leftarrow M[r_j + c]$	MEM + CONST MEM + CONST MEM CONST MEM

Note: All tiles on this page have an upward link like ADD



Jouette Architecture 2/2

Name	Effect	Trees
STORE	$M[r_j + c] \leftarrow r_i$	MOVE MEM + CONST MOVE MEM + CONST MOVE MEM CONST MOVE MEM
MOVEM	$M[r_j] \leftarrow M[r_i]$	MOVE MEM MEM