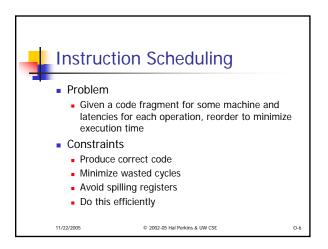
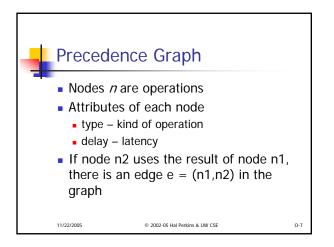
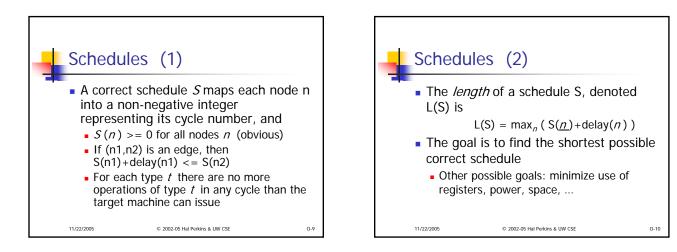


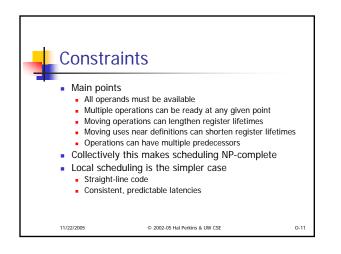
Example	: w =	w*2*x*	y*z;	
Simple schedule Loads early		ly		
1 LOAD r1	<- W	1 LOAD	r1 <- w	
4 ADD r1	<- r1,r1	2 LOAD	r2 <- x	
5 LOAD r2	2 <- X	3 LOAD	r3 <- y	
8 MULT r1	<- r1,r2	4 ADD	r1 <- r1,r1	
9 LOAD r2	2 <- y	5 MULT	r1 <- r1,r2	
12 MULT r1	<- r1,r2	6 LOAD	r2 <- z	
13 LOAD r2	2 <- Z	7 MULT	r1 <- r1,r3	
16 MULT r1	<- r1,r2	9 MULT	r1 <- r1,r2	
18 STORE w <- r1		11 STORE	w <- r1	
21 r1 free		14 r1 is free		
2 register	s, 20 cycles	3 regis	ters, 13 cycles	
11/22/2005	© 2002-05 H	al Perkins & UW CSE		0-5

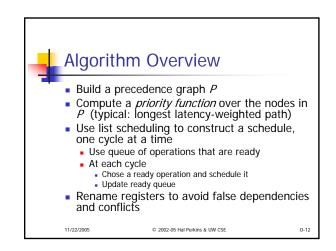




Examp	le Graph	
c LOAD d MULT e LOAD f MULT g LOAD	r1 <- r1,r1 r2 <- x r1 <- r1,r2 r2 <- y r1 <- r1,r2 r2 <- z r1 <- r1,r2 r2 <- z	
11/22/2005	© 2002-05 Hal Perkins & UW CSE	0-8







List	Scheduling Algorithm		
Cycle = 1; Ready = leaves of P; Active = empty; while (Ready and/or Active are not empty) if (Ready is not empty) remove an op from Ready; S(op) = Cycle; Active = Active \cup op; Cycle++; for each op in Active if (S(op) + delay(op) <= Cycle) remove op from Active; for each successors of op in P if (s is ready - i.e., all operands available) add s to Ready			
11/22/2005	© 2002-05 Hal Perkins & UW CSE 0-11	3	

Exam	ole	
b ADD c LOAD d MULT e LOAD f MULT g LOAD h MULT	r1 <- w r1 <- r1,r1 r2 <- x r1 <- r1,r2 r2 <- y r1 <- r1,r2 r2 <- z r1 <- r1,r2 r2 <- z r1 <- r1,r2	
11/22/2005	© 2002-05 Hal Perkins & UW CSE	O-14

