Compiling Control Flow Statements

Mooly Sagiv
Tel Aviv University
sagiv@math.tau.ac.il
and
Reinhard Wilhelm
Universität des Saarlandes
wilhelm@cs.uni-sb.de

April 21, 1997

Subjects

- Syntax of control flow statements
- Sequence of statements
- Conditional Statements
- Case statements
- Iterative Statements

Syntax

Sequence of Statements $st_1; st_2; \cdots; st_n$

Conditional Statements

- **if** e **then** st **fi**
- **if** e **then** st **else** st **fi**
- **case** e **of**

  $0{:}\ st_1\ ;$

  $1{:}\ st_2\ ;$

  $\vdots$

  $k\ :\ st_k$

  end

Iterative Statements

- **while** e **do** st **od**
- **repeat** st **until** e

Sequence of statements

- $code(st_1; st_2)\rho = code(st_1)\rho; code(st_2)\rho$
  control drops through from the last instruction generated for st_1 to the first instruction generated for st_2.
  st_1 and st_2 are compiled in the same ρ

- Example: Assume that $Typ(x) = Typ(y) = int$

  $code(x := 3; y := x + 1)\ \rho$
  $= code(x := 3)\ \rho;\ code(y := x + 1)\ \rho$
  $= code_L\ x\ \rho;\ code_R\ 3\ \rho;$ **sto** i ; $code\ (y := x + 1)\ \rho$
  $=$ **ldc** a $\rho(x)$; $code_R(3)\ \rho$; **sto** i $code\ (y := x + 1)\ \rho$
  $=$ **ldc** a $\rho(x)$; **ldc** i 3; **sto** i $code\ (y := x + 1)\ \rho$
  $=$ **ldc** a $\rho(x)$; **ldc** i 3; **sto** i $code_L\ y\ \rho;\ code_R\ (x + 1)\ \rho$; **sto** i
  $=$ **ldc** a $\rho(x)$; **ldc** i 3; **sto** i **ldc** a $\rho(y)$; $code_R\ (x + 1)\ \rho$; **sto** i

Conditional Statements

Machine ressources to implement control statements: unconditional and conditional jumps

P-code Instructions for Branches

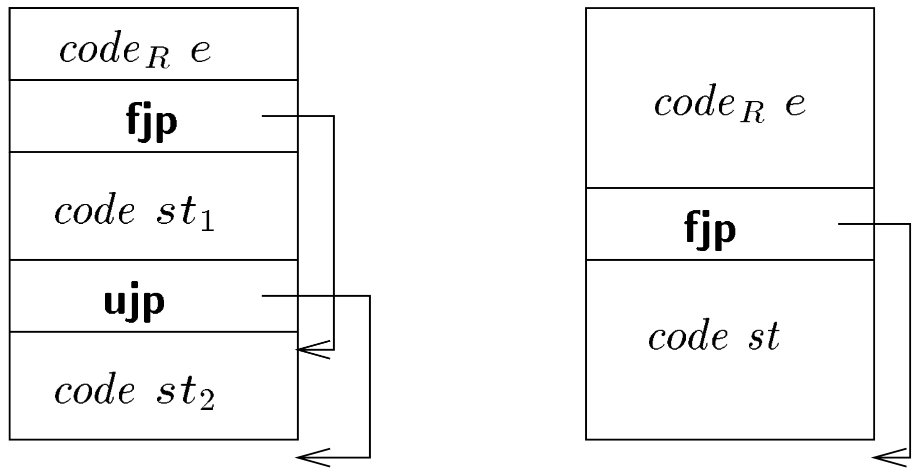
Inst.	Meaning	Cond.	Result
ujp q	$PC := q$	$q \in [0, codemax]$	
fjp q	**if** $STORE[SP] = false$ **then** $PC := q$ **fi**; $SP := SP - 1$	(b) $q \in [0, codemax]$	$()$

fjp consumes the boolean value on top of the stack.

Conditional Statements

- $code(\textbf{if } e \textbf{ then } st \textbf{ fi})\ \rho = code_R\, e\, \rho;\ \textbf{fjp}\ l;\ code\ st\ \rho;\ l:$

- $code(\textbf{if } e \textbf{ then } st_1 \textbf{ else } st_2 \textbf{ fi})\ \rho =$
  $code_R\ e\ \rho;\ \textbf{fjp}\ l_1;\ code\ st_1\ \rho;\ \textbf{ujp}\ l_2;\ l_1:\ code\ st_2\ \rho;\ l_2:$



Conditionals

doublesided onesided

Problem: Forward references to labels

Solutions:

- Generate symbolic labels in assembly code,
- 2–pass code generation,
- "Back patching" target addresses when known.

Examples

Example1:

if $a > b$ **then** $c := a$ **else** $c := b$ **fi**

Example2:

if $x < 5$ **then** **if** $y < 7$ **then** $c := a$ **fi** **fi**

Case Statement

Restricted form: Initial section of the natural numbers

Input

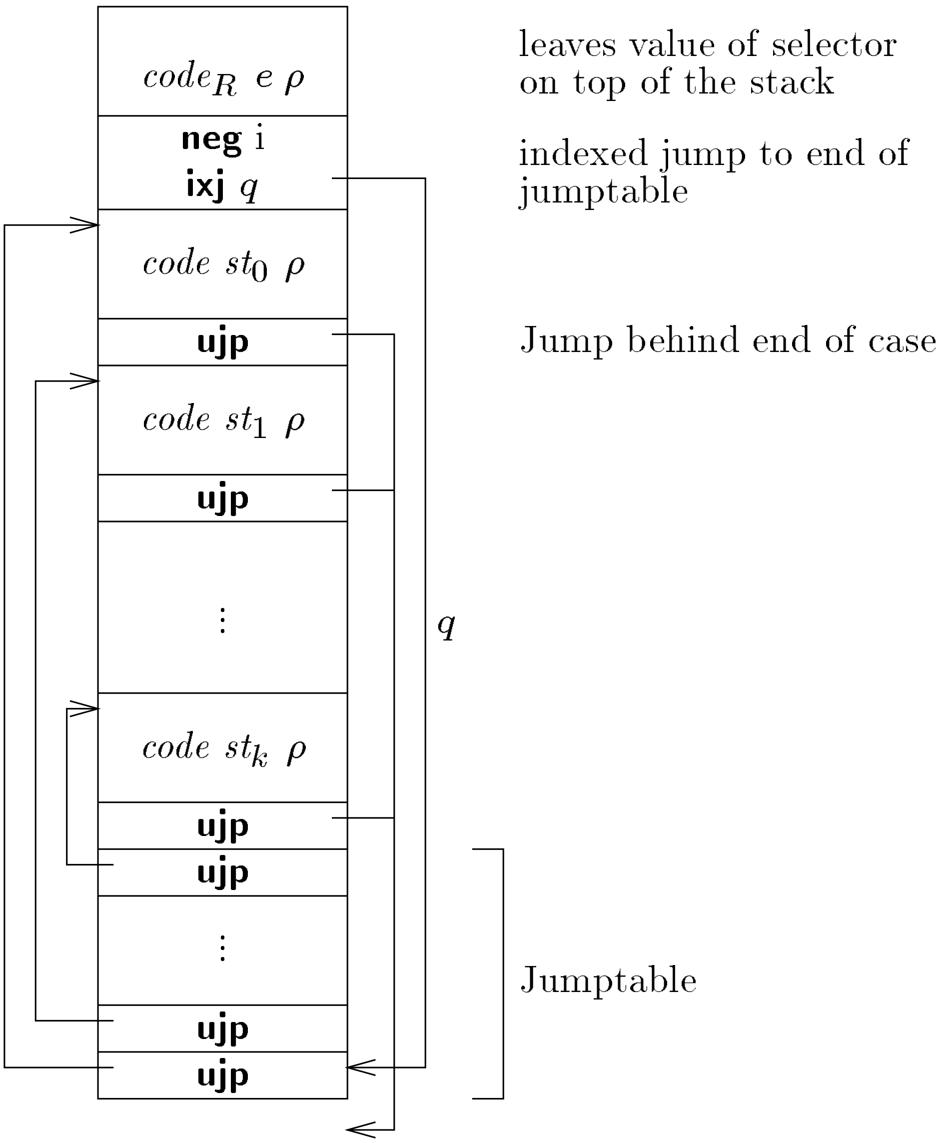
case e **of**
  0: st_1 ;
  1: st_2 ;
  $\vdots$
  k : st_k
end

Two Solutions

- Generate nested conditionals
  if $e = 0$ **then** st_1
  else if $e = 1$ **then** st_2
  $\vdots$
  else if $e = n$ **then** st_k
  else run-time error: unmatched case label **fi fi** $\cdots$ **fi**
- Jump table: $PC :=$ table(e)

P-Code Instructions for Jump Table

Instr.	Meaning	Cond.	Result
ixj q	$PC := STORE[SP] + q;$ $SP := SP - 1$	(i)	$()$



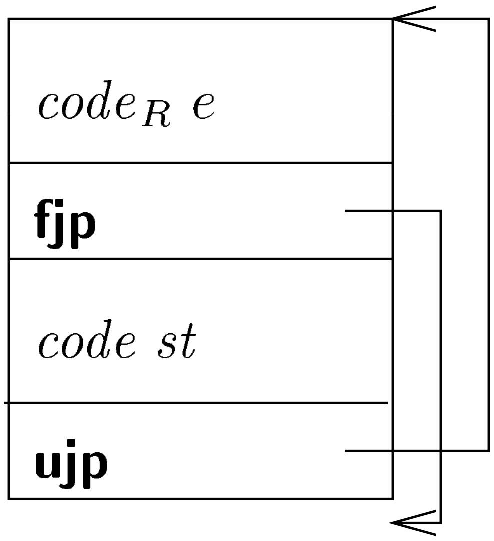
Iterative Statements

- $code\ (\textbf{while}\ e\ \textbf{do}\ st\ \textbf{od})\ \rho =$
  $l_1: \ code_R\, e\, \rho;$ **fjp** $l_2;\ code\ \ st\ \rho;$ **ujp** $l_1;\ l_2:$

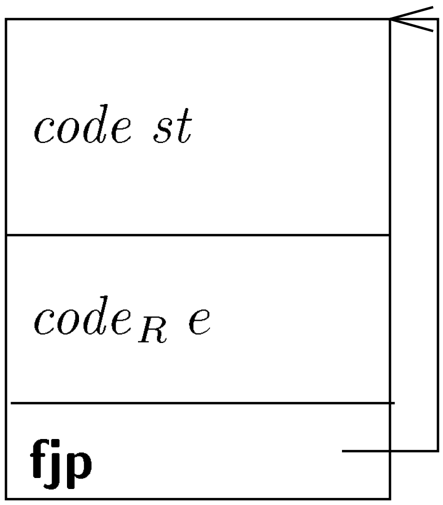
- Example:

  $x := 1;$ **while** $x < 4$ **do**
  $x := x + 1$ **od**

- $code\ (\textbf{repeat}\ st\ \textbf{until}\ e)\ \rho =$
  $l: \ code\ \ st\ \rho;\ code_R\, e\, \rho;$ **fjp** l



while e **do** st **od**



repeat st **until** e