

CC4301 Arquitectura de Computadores

Auxiliar 7

Prof. Aux.: Gaspar Pizarro V.

12 de octubre de 2012

1. P1 Control 2 Año 2005

```
.globl Q
Q:
    pushl %ebp
    movl %esp, %ebp
    pushl %esi
    pushl %ebx
    movl 8(%ebp), %edx # 1er param
    movl 12(%ebp), %ecx # 2do param
    movl 16(%ebp), %ebx # 3er param
    movl (%edx,%ecx,4), %esi
    movl (%edx,%ebx,4), %eax
    movl %eax, (%edx,%ecx,4)
    movl %esi, (%edx,%ebx,4)
    popl %ebx
    popl %esi
    popl %ebp
    ret

    # si %ebx>=%esi goto .L10
    cmpl %esi, %ebx
    jge .L10
.L8:
    # si (%edi,%ebx,4)>=0 goto .L6
    cmpl $0, (%edi,%ebx,4)
    jns .L6
    addl $1, %ebx
    jmp .L3
.L6:
    subl $4, %esp
    pushl %esi # 3er. arg
    pushl %ebx # 2do. arg
    pushl %edi # 1er. arg
    call Q
    subl $1, %esi
    addl $16, %esp
.L3:
    cmpl %esi, %ebx # si %ebx<%esi goto .L8
    jl .L8
.L10:
    movl %ebx, %eax # valor de ret.
    leal -12(%ebp), %esp # %esp= %ebp-12
    popl %ebx
    popl %esi
    popl %edi
    popl %ebp
    ret

.globl P
P:
    pushl %ebp
    movl %esp, %ebp
    pushl %edi
    pushl %esi
    pushl %ebx
    subl $12, %esp
    movl 8(%ebp), %edi
    movl 12(%ebp), %ebx
    movl 16(%ebp), %esi
```

2. Recursión

Considérese la siguiente estructura:

```
struct node {
    int value;
    node *left;
    node *right;
}
```

Decompile los siguientes programas en assembler x86 de 32 bits, sabiendo sabiendo que todos los punteros apuntan a `struct node`.

2.1.

```
.globl f
f:
    pushl %ebp
    movl %esp, %ebp
    pushl %esi
    pushl %ebx
    movl 8(%ebp), %ebx
    movl 12(%ebp), %esi
    addl %esi, (%ebx)
    movl 4(%ebx), %eax
    testl %eax, %eax
    je .L2
    pushl %esi
    pushl %eax
    call f
.L2:
    movl 8(%ebx), %eax
    testl %eax, %eax
    je .L4
    pushl %esi
    pushl %eax
    #movl %esi, 4(%esp)
    #movl %eax, (%esp)
    call f
.L4:
    popl %ebx
    popl %esi
    movl %ebp, %esp
    popl %ebp
    ret
```

2.2.

```
.globl f
f:
    pushl %ebp
    movl %esp, %ebp
    pushl %ebx
    subl $20, %esp
    movl 8(%ebp), %eax
    movl 12(%ebp), %ebx
    testl %eax, %eax
    je .L5
    movl (%eax), %ecx
    movl (%ebx), %edx
    cmpl %edx, %ecx
    jne .L3
    movl 4(%ebx), %edx
    movl %edx, 4(%eax)
    movl 8(%ebx), %edx
    movl %edx, 8(%eax)
    jmp .L5
.L3:
    cmpl %edx, %ecx
    jle .L4
    movl %ebx, 4(%esp)
    movl 4(%eax), %eax
    movl %eax, (%esp)
    call f
    jmp .L5
.L4:
    cmpl %edx, %ecx
    jge .L5
    movl %ebx, 4(%esp)
    movl 8(%eax), %eax
    movl %eax, (%esp)
    call f
.L5:
    addl $20, %esp
    popl %ebx
    popl %ebp
    ret
```