CS 241 - Lecture 6

Graham Cooper

May 25th, 2015

Loaders

Basic OS code:

```
repeat:
p <- next program to run
copy p into memory, starting at 0x00 *** THIS IS THE LOADER
jalr $0
beq $0, $0, repeat
```

our loader is mips.twoints and mips.array

Problems

- OS is a program - where does it sit in memory? - Other programs in memory can't all be at address 0x00.
How can we fix this?
- Choose different starting addresses for programs at assembly time
- How does the loader know where it is supposed to put them
- What if two programs have the same load address?
- let the loader decide where to put the program

Loader: Take program P as input
- Find a location α in memory for p
- Copy P into memory starting at α
- return α to OS
OS 2.0

```
repeat:
P <- next program to run
$3 <- loader(P)
jalr $3
beq $0, $0, repeat
```

Loader PsuedoCode:
Input: words $w_1, ...w_k$ - the machine code(P)
n = k + space for stack $\leftarrow$ how much stack space? - Pick something
d = first address of n consecutive words of unused RAM.
for i = 1 ... k - $Mem[\alpha + i \times 4] \leftarrow w_i$
\$30 $\leftarrow \alpha + 4 \times n$
return a

Problems

Labels may be resolved to the wrong addresses.
Loader will have to fix the problem somehow.

What needs to change when we relocate programs?

.word id - is when we need to add α to id .word constant - no adjustment
.anything else (including beq and bne) - no adjustment

Problem

- The assmbled file is a stream of bits
- How dod we know which ones come from .word (with an id!) and which are instructions??
We can't.

The output of most assemblers is not pure machine code. - it is object code.

Object file - contains binary code AND auxiliary information needed by the loader (and later the linker)

Our object file format called MERL
(MIPS Executable Relocatable Linkable)

What do we need to put in our object file?
- the code

- which lines of code (addresses) were originally .word ID?
- other info later

0.1 MERL FORMAT

header	0x10000002 - cookie - sanity check- yes this is MERL length of the .merl file code length = header + mips
MIPS Binary	
Footer	format code = 1 - for relocation entries address - in MIPS of a relocatable word format code address ...

Note: 0x10000002 is MIPS for beq $0, $0, 2
ie. The command to skip the header.

- So MERL files can be exectuted as ordinary MIPS files (if loaded at address 0);
- Can write MERL directly.
- We would like the assembler to generate relocatable code (MERL) for us