

Address binding

- Can we load a program at any location in memory?
- This depends on how instructions and data are bound to actual memory address: Address Binding
 - Absolute addressing
 - Relative/relocatable addressing

Address binding (2)

- With relocatable addressing, we can place a program anywhere in memory.
- The instructions need to be “patched” for this relocation.

Linkers and Loaders

LEON PRESSER AND JOHN R. WHITE

University of California,
Santa Barbara, California 93106*

This is a tutorial paper on the linking and loading stages of the language transformation process. First, loaders are classified and discussed. Next, the linking process is treated in terms of the various times at which it may occur (i.e., binding to logical space). Finally, the linking and loading functions are explained in detail through a careful examination of their implementation in the IBM System/360. Examples are presented, and a number of possible system trade-offs are pointed out.

Key words and phrases binary loaders, relocating loaders, linking loaders, linkers, compilers, assemblers, relocation, program modularity, libraries

CR categories 4.11, 4.12, 4.39

1. INTRODUCTION

A computer system includes a set of software and hardware facilities which supervises its operation, insures its coordination, and facilitates its use. Such facilities are referred to as the computer's operating system. From a functional viewpoint it is justifiable to separate from the operating system

order to obtain flexibility and better utilization of main memory, translators are required to generate *relocatable* code, that is, code that can be loaded into any section of main memory for execution. Furthermore, the capability to combine subprograms into a composite program, referred to as *linking*, is of great value in modern operating sys-

Compiler, Linker, Loader

- *Compiler*: translates source code into object files.

```
cc -c main.c
```

- *Linker (linkage editor)*: “Combines and edits modules to produce a single module that can be brought into memory”

```
ld -o main main.o -lc /path/to/compiler_rt.a
```

- *Loader*: Stores a program into main memory.
 - Binary vs. relocating loaders.
 - Linking at loading time: “linking loaders”

Different address schemes

- Example progression of used addressing schemes:
 - Source code: symbolic (identifiers)
 - Object file: relative address (relocatable)
 - Executable: absolute address

Binding stages

- Note that with compile-time and load-time binding a program **cannot** be relocated after it has been loaded and started.
- Execution-time binding allows bindings to be modified during the course of program execution. This is done with hardware support (MMU), the program itself is not changed again.