Process Swapping in UNIX V6

Lions' commentary chapter 14 in detail

Urs Fässler

Systems Group
ETH Zürich

20.04.2010

why writing RAM to a disc?

- small device with 16 MB RAM and a Hard disc
- want organise thousands of Mp3 on it

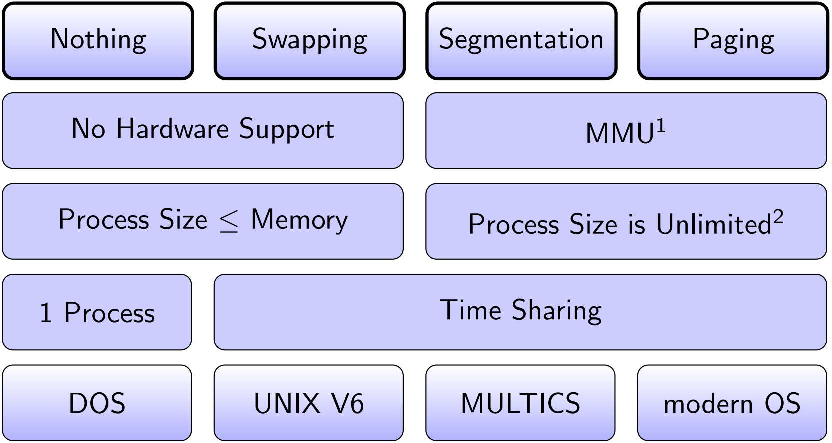
! database doesn't fit into RAM

⇒ use virtual memory

WL-HDD (www.asus.com)

OpenJukebox (openjukebox.origo.ethz.ch)

types of virtual memory



Nothing | Swapping | Segmentation | Paging

No Hardware Support | MMU[1]

Process Size $\leq$ Memory | Process Size is Unlimited[2]

1 Process | Time Sharing

DOS | UNIX V6 | MULTICS | modern OS

Since Overlays have not to be implement in the OS, they are omitted; for more information see [4] or [3]

[1] see frame 5

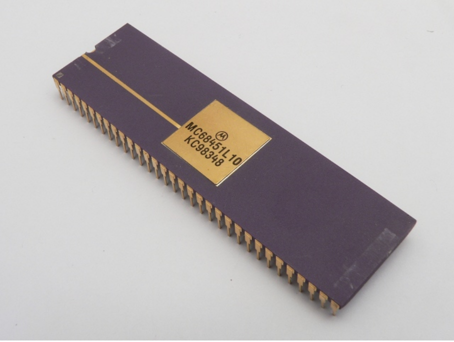
[2] it is limited by the virtual address space and the size of the swap area

swapping versus paging

swapping	**paging**
older	modern
easy	complicated
whole process	parts of the memory
proc size $<$ RAM	arbitrary proc size
no special hardware	MMU

MMU - Memory Management Unit

- paging needs a MMU
- integrated in modern CPU such as ARM, IA-32, MIPS [5]
- basic operation of paging is defined by the MMU



www.wikipedia.org

OS tasks in a paging system [4]

- creation of a process
  - create pages
  - copy text and data into pages
- set a process running
  - initializing MMU
  - switch actual page to new process
- page fault
  - find page on swap
  - find free frame in RAM
  - copy page into frame
  - restart last instruction
- after termination of a process
  - free frames in RAM
  - free pages on swap

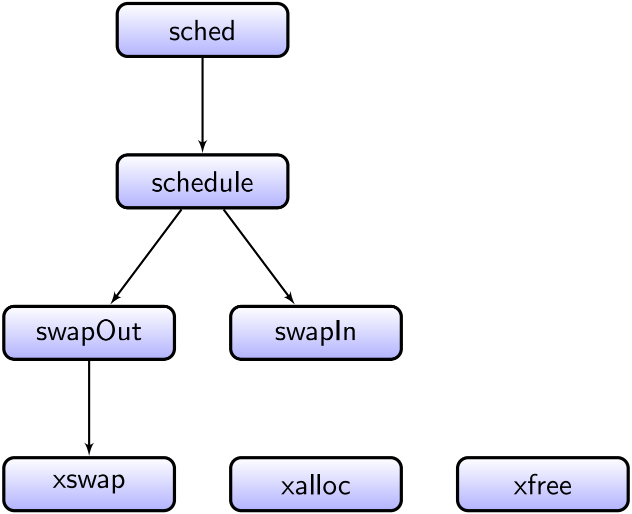
OS tasks in a swapping system (UNIX V6) [1]

- creation of a process
    - swap new process out if it doesn't fit in RAM
- process size is increased
    - swap process out if it don't fit in RAM anymore
- swapper is triggered to do something
    - swap oldest process in (if enough RAM is available)
    - otherwise swap an inactive process out

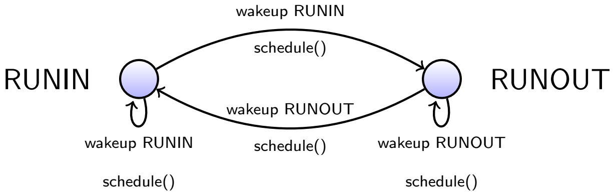
study of the code

- original code is not very readable
- automated formatted with bcpp
- refactored by hand to improve code metrics
  - # of goto: from 11 to 0
  - split sched (100 lines) into 9 functions (and moved to sched.c)
  - used strong typing
- generated documentation with Doxygen

call graph



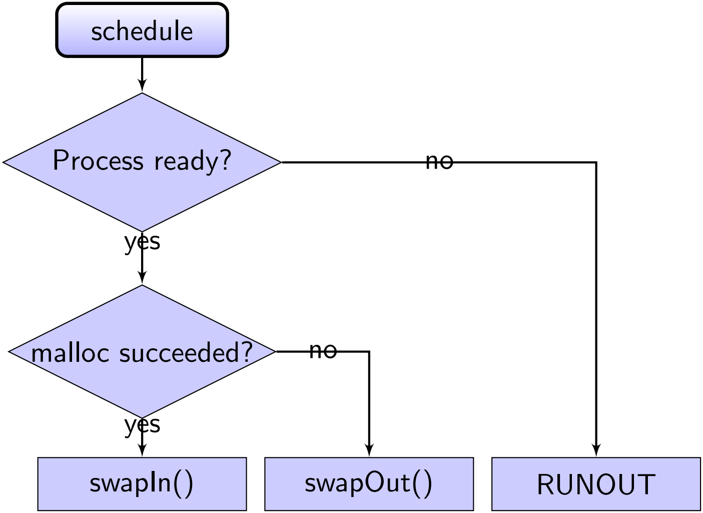
sched - waiting for changes



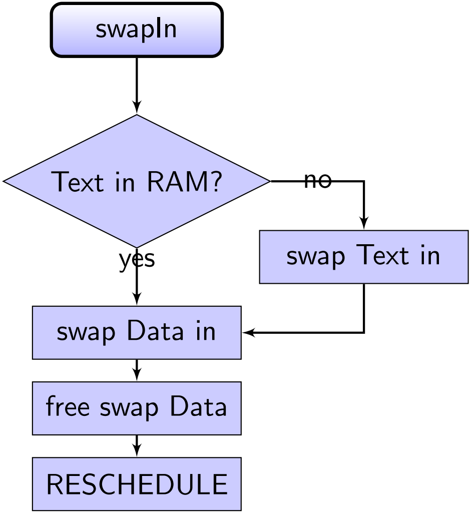
- schedule is looped until it returns something else than RESCHEDULE
  - RUNOUT: no swapped process is ready to run
  - RUNIN: process can't yet be swapped in[3]

[3] Either a process hasn't been out more than 3 seconds and/or none of the processes in RAM is inactive or has been there more than 2 seconds.

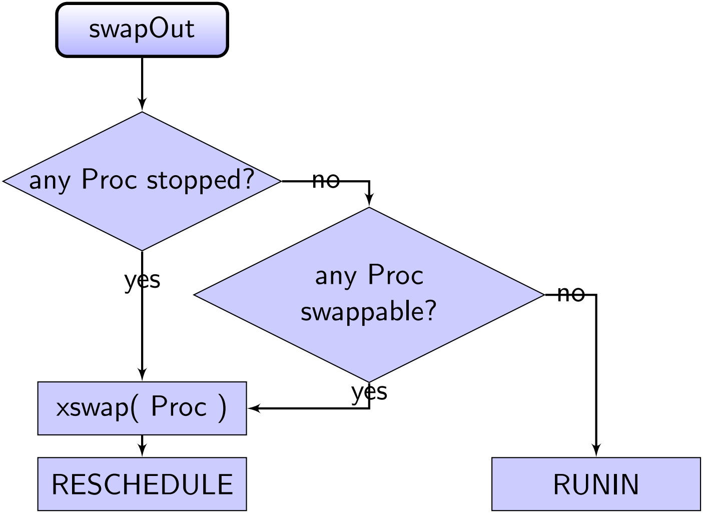
sched : schedule - process swapper



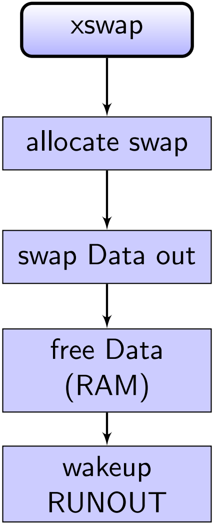
sched : schedule : swapIn



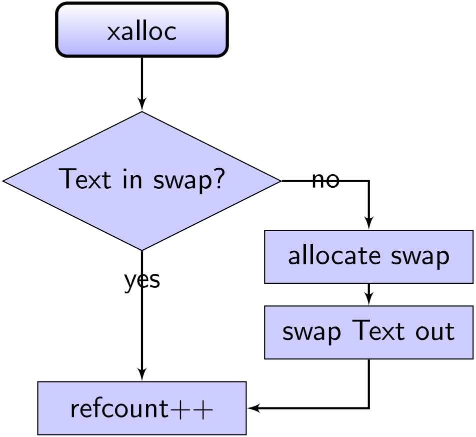
sched : schedule : swapOut



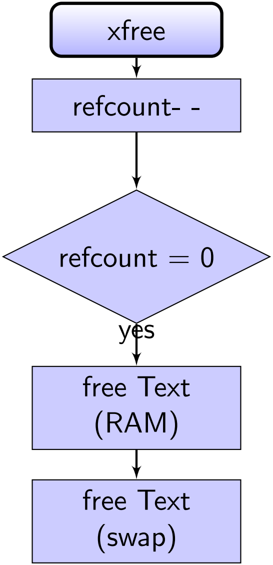
sched : schedule : swapOut : xswap - swap a process out



xalloc - copy text from disc to swap



xfree - remove link to text



comparison to modern coding style

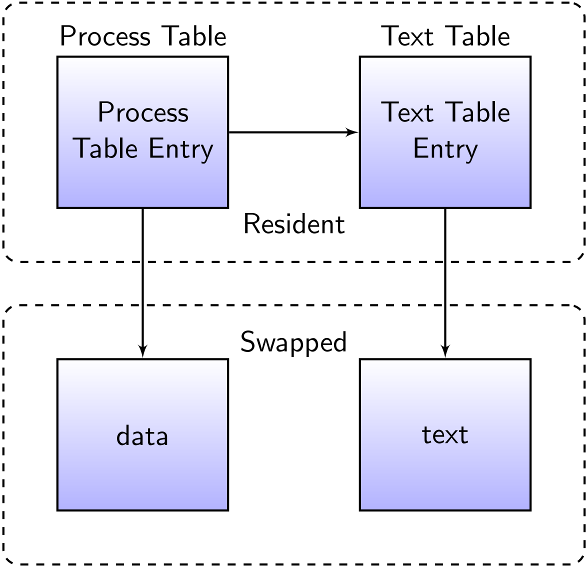
- don't use goto (at most for exceptions)
- write small subroutines for specific functionality
- use meaningful variable names
- use an ANSI C compiler
  - don't optimize (let the compiler do it)
  - use strong typing
  - define a interface (headerfile) to the implementation

about

- find the slides, handout, refactored source and more on `http://n.ethz.ch/~ursf/`

Appendix

data structure



K&R C [2]

```c
struct                /* no name needed           */
{
  int  answer;
};

int printf();         /* prototype declaration    */

main()                /* return value is int      */
{
  int *var;           /* var is a pointer to int  */
  var->answer = 42;   /* now it is a pointer to
                         the unnamed struct       */
  printf( *var );     /* arguments aren't checked */
}
```

/* You are not expected to understand this */

> *... "You are not expected to understand this" was intended as a remark in the spirit of "This won't be on the exam," rather than as an impudent challenge. The real problem is that we didn't understand what was going on either...*

`http://cm.bell-labs.com/cm/cs/who/dmr/odd.html`

OpenJukebox: /proc/meminfo

```
root@OpenWrt:~# cat /proc/meminfo
        total:    used:    free:  shared: buffers:  cached:
Mem:  14688256 14282752   405504        0   548864  4472832
Swap: 139788288  1298432 138489856
MemTotal:        14344 kB
MemFree:           396 kB
MemShared:           0 kB
Buffers:           536 kB
Cached:           3988 kB
SwapCached:        380 kB
Active:           1296 kB
Inactive:         3624 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:        14344 kB
LowFree:           396 kB
SwapTotal:      136512 kB
SwapFree:       135244 kB
```

Unfortunately, swap isn't used that much. But there were problems with not enough memory eighter with GMediaServer or mpd during building the database.

OpenJukebox: df -h

```
root@OpenWrt:~# df -h
Filesystem                Size      Used Available Use% Mounted on
rootfs                  243.1M     28.9M    201.6M  13% /
df: /jffs/rom: No such file or directory
tmpfs                     7.0M     40.0k      7.0M   1% /tmp
/dev/mtdblock/4           1.7M    528.0k      1.2M  31% /jffs
mini_fo:/jffs             1.7M    528.0k      1.2M  31% /jffs
/dev/ide/host0/bus0/target0/lun0/part2
                        243.1M     28.9M    201.6M  13% /
/dev/ide/host0/bus0/target0/lun0/part3
                         36.3G     28.9G      5.6G  84% /mnt/data
```

implementation details of paging

- swap area (Partition or File)
- time of saving text to swap (if ever)
- strategy of loading pages
- time of duplicating pages
- algorithm to find pages

Bibliography

- John Lions.
  Commentary on the Sixth Edition UNIX Operating System.
  1977.

- Dennis M. Ritchie.
  The Development of the C Language.
  1996.

- Abraham Silberschatz, Peter Baer Galvin, and Greg Gagne.
  Operating System Concepts.
  6 edition, 2003.

- Andrew S. Tanenbaum.
  Moderne Betriebssysteme.
  2 edition, 2001.

- Wikipedia.
  Memory management unit — wikipedia, the free encyclopedia, 2010.
  [Online; accessed 16-April-2010].