# OS HW1

Operating System 106 Fall

W.J.Tsai 蔡文錦 教授

TA 鍾陳恩
倪莞雅
巫怡慧
任佳靜

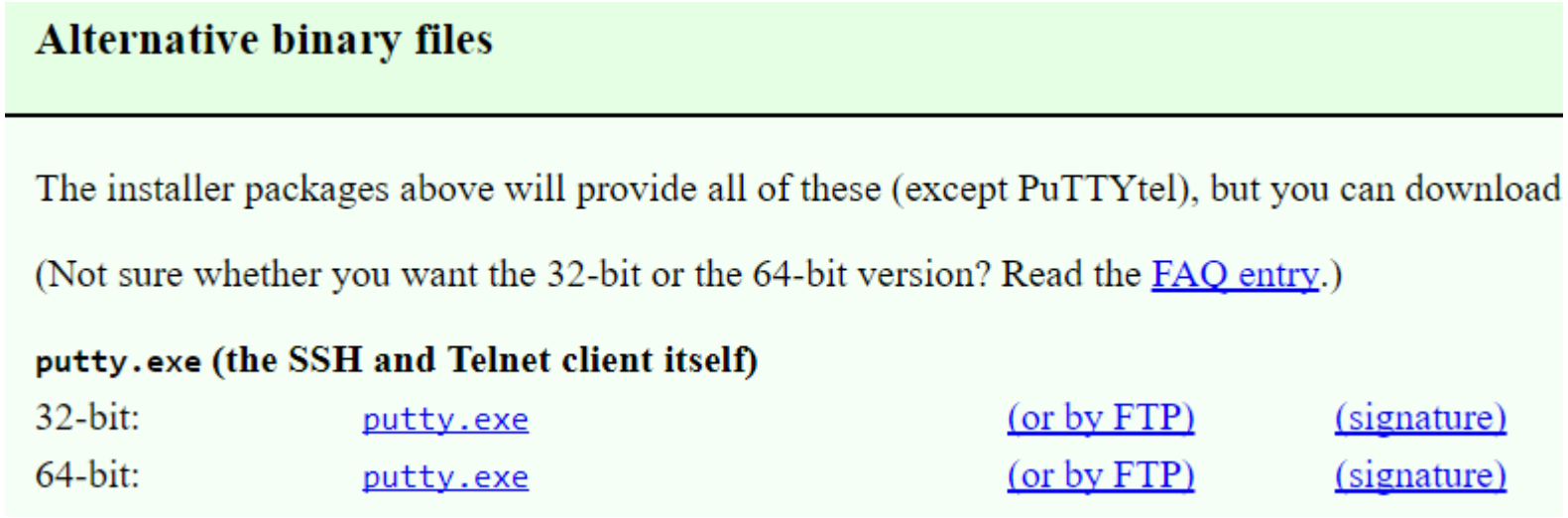
# PREWORK

# Putty

- Login Tools
  - Putty
- Editors
  - vim
- FTP Tools
  - FileZilla Client

# Putty

- Login
  - Download Putty
  - https://goo.gl/rM4Scb



**Alternative binary files**

The installer packages above will provide all of these (except PuTTYtel), but you can download

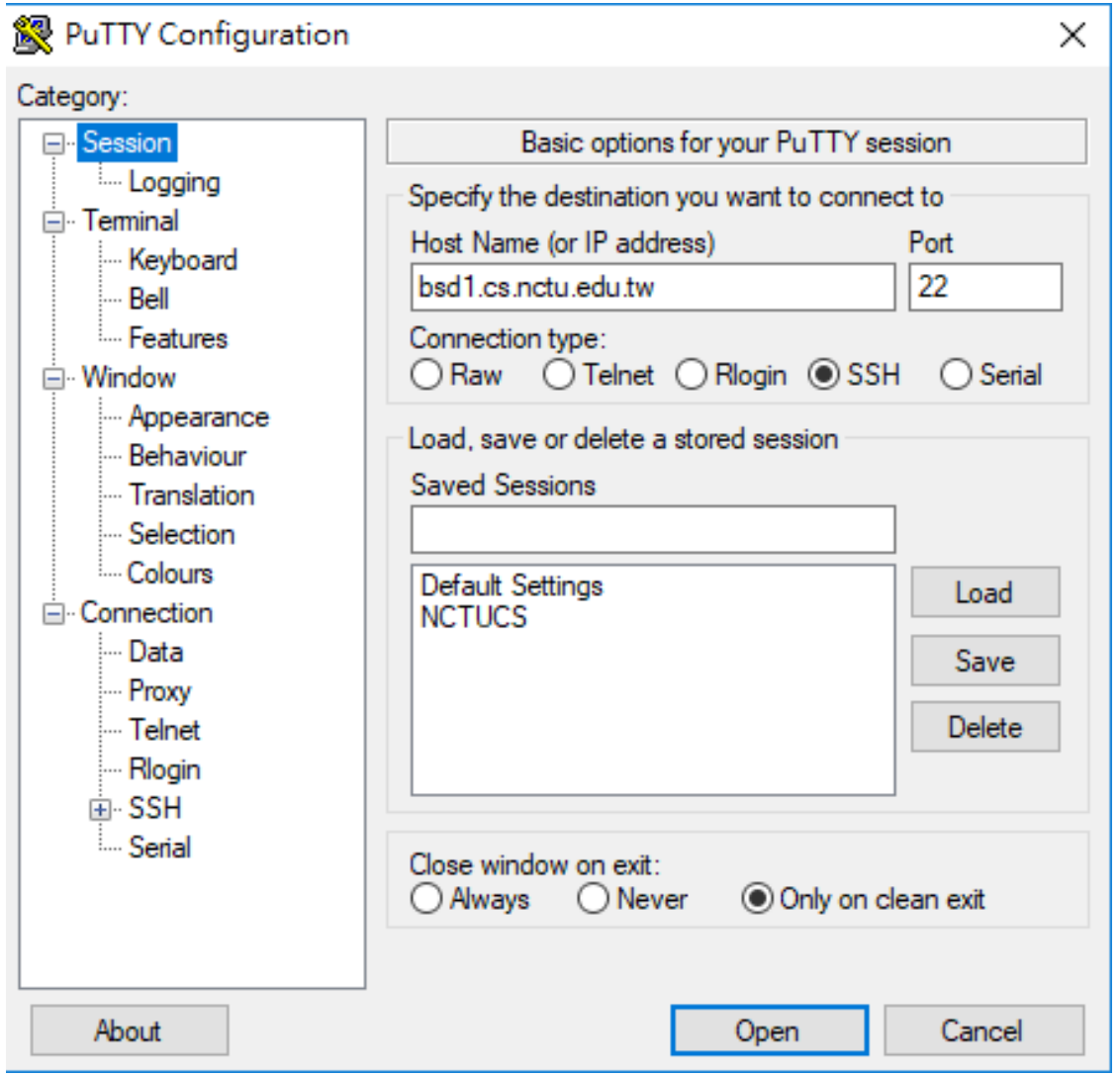
(Not sure whether you want the 32-bit or the 64-bit version? Read the FAQ entry.)

**putty.exe (the SSH and Telnet client itself)**

| | | | |
|---|---|---|---|
| 32-bit: | putty.exe | (or by FTP) | (signature) |
| 64-bit: | putty.exe | (or by FTP) | (signature) |

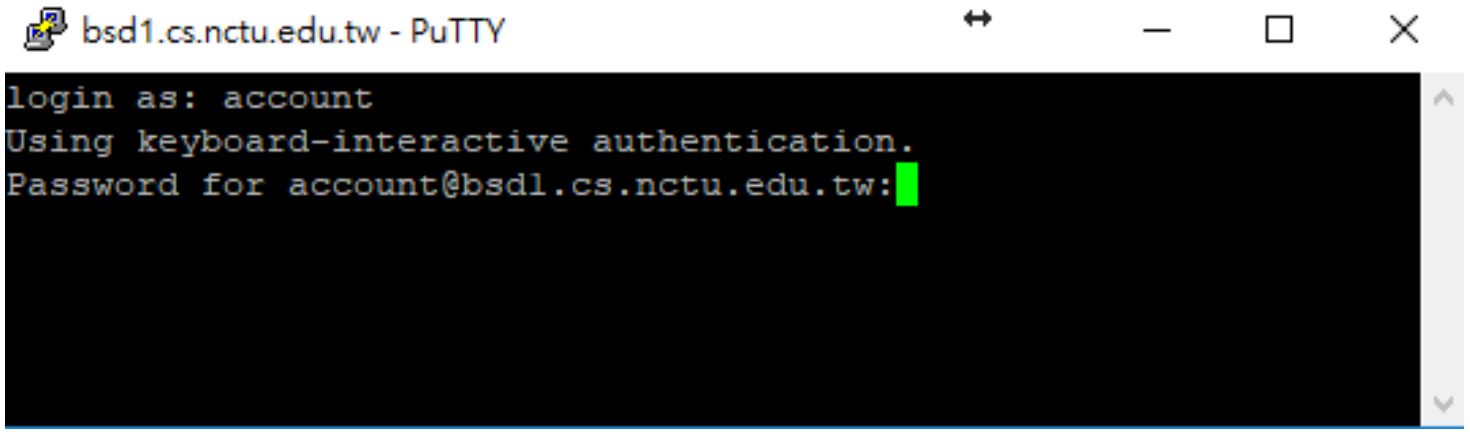
# Putty

- Login
  - Download Putty
  - https://goo.gl/rM4Scb
  - How to Use Putty
  - https://goo.gl/8AJsPL



# Putty

- Login
- The default for SSH service is port 22
  - bsd1.cs.nctu.edu.tw – bsd5.cs.nctu.edu.tw
  - linux1.cs.nctu.edu.tw – linux6.cs.nctu.edu.tw



# Putty

- Command
  - clear – clear the screen
  - ls – list directory contents
  - mv – move files or directories
  - mkdir – create directories
  - rm – remove files or directories
  - chmod – change file system modes of files or directories
  - ….

- Reference
  - http://linux.vbird.org/linux_basic/redhat6.1/linux_06command.php#filesystem

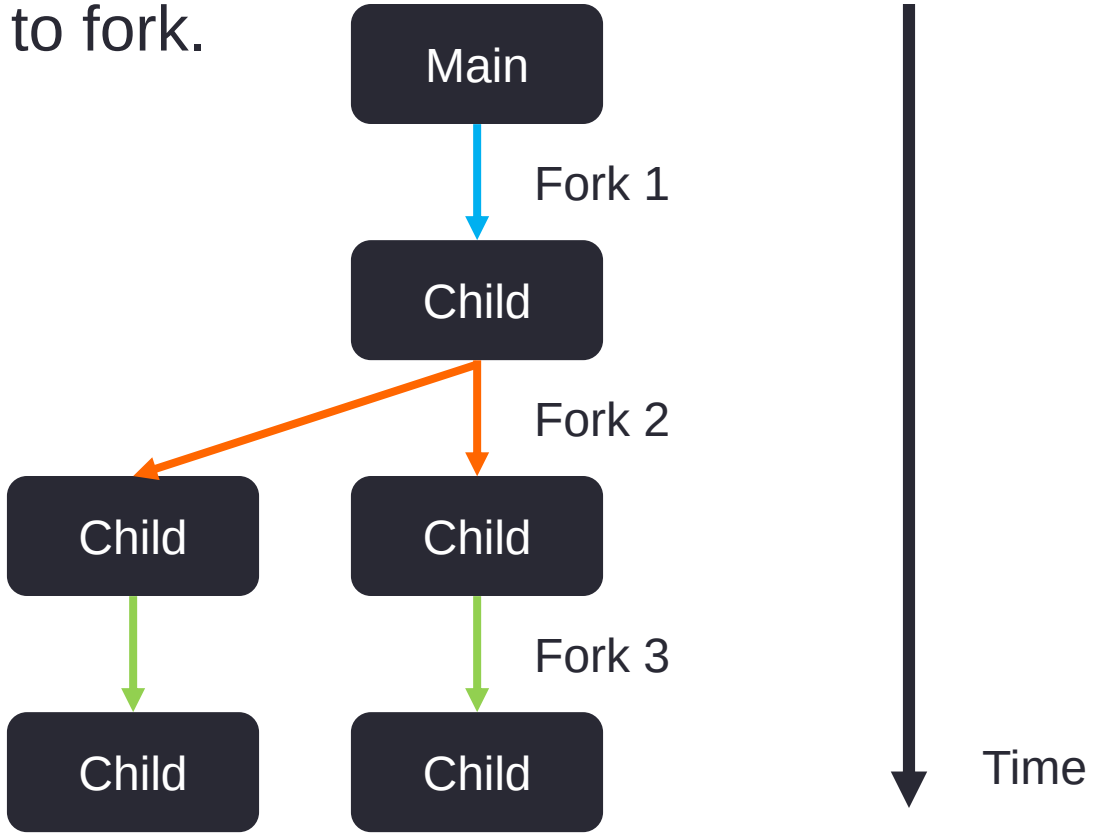
# FileZilla

- Upload File to Workstation
- Login

# HW 1-1

# Fork 3 Times

According this tree format to fork.



# Output Format

Main Process ID

Total 5 Children

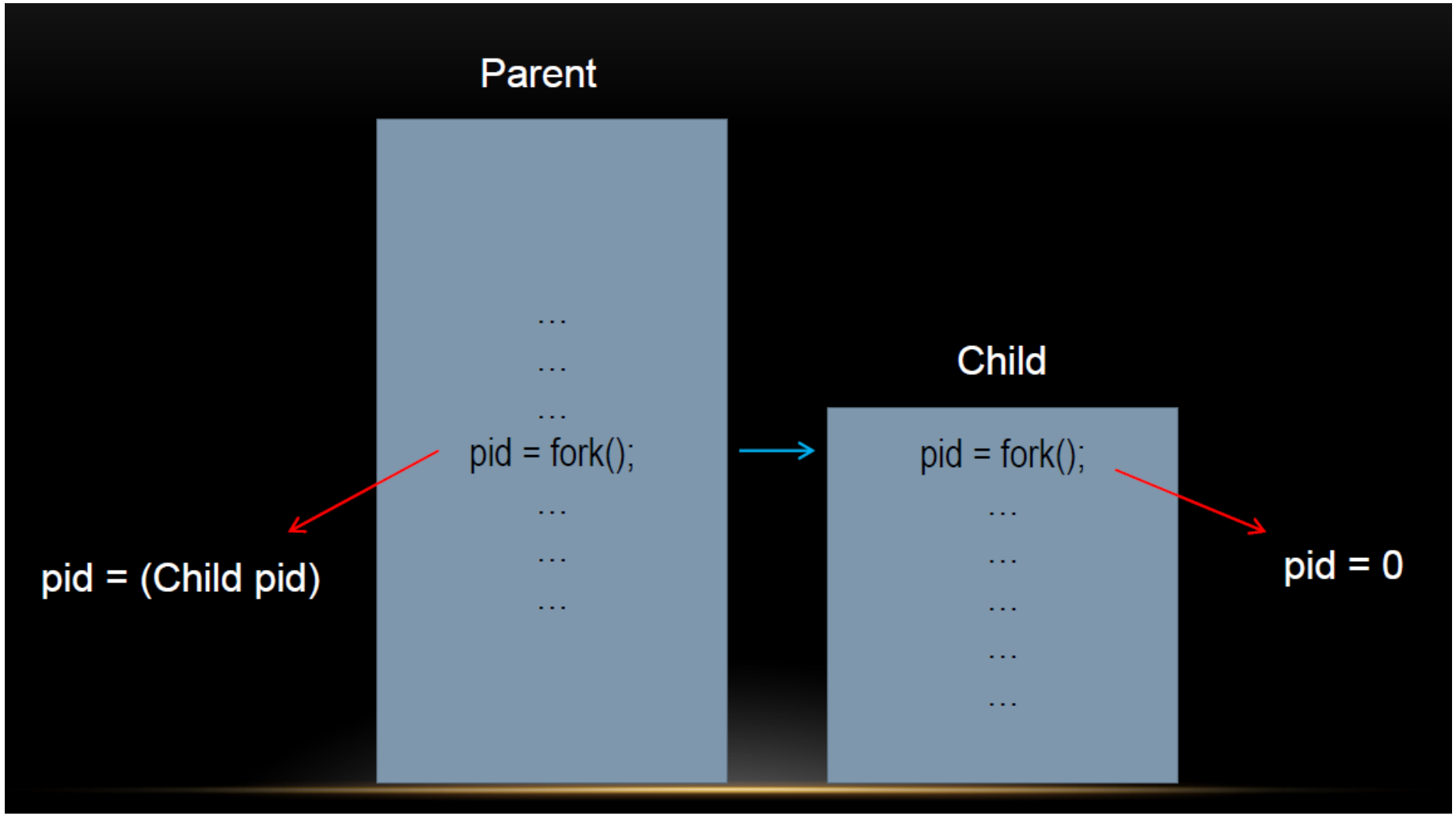
The format & fork order have to be same.
(Fork 1 2 2 3 3)

```
16:07          @bsdl [~/106_OS/hw1] >./a.out
Main Process ID : 29428

Fork 1. I'm the child 29429, my parent is 29428.
Fork 2. I'm the child 29431, my parent is 29429.
Fork 2. I'm the child 29432, my parent is 29429.
Fork 3. I'm the child 29433, my parent is 29432.
Fork 3. I'm the child 29434, my parent is 29431.
```

Hint:
Parent Process has to wait until Child Process finish, then exit.

# Hint



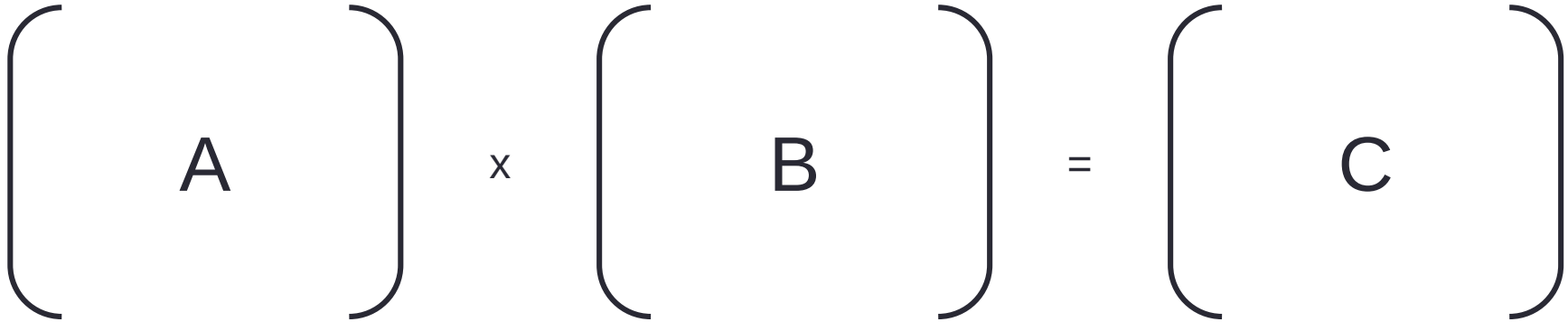
# HW 1-2

Original Author: Prof. Li-Ping, Chang

# Multi-Process Matrix Multiplication using Shared Memory

Original Author: Prof. Li-Ping, Chang

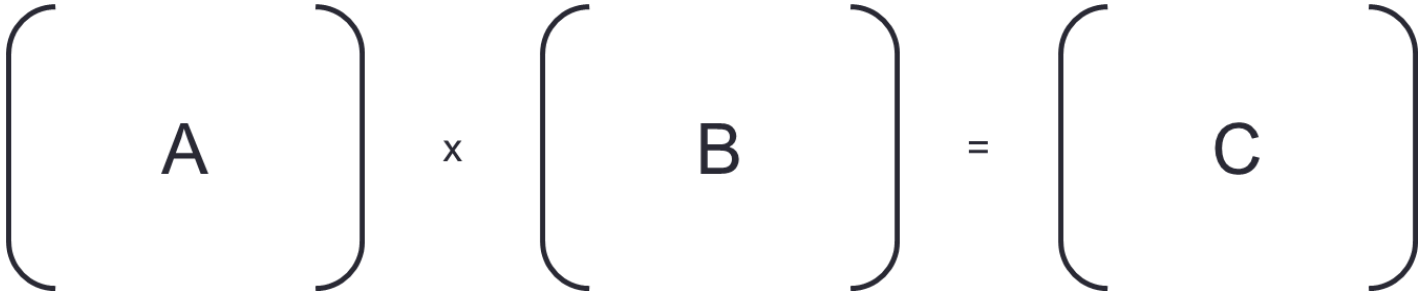
- Matrix multiplication using multiple processes
  - Parallel processing; faster on multicore machines
- Input: The dimension of the square matrices A & B
  - E.g., 100 → A, B, and C are 100 x 100 square matrices
- Output: Execution Time and a Checksum(Sum of all elements of Matrix C)



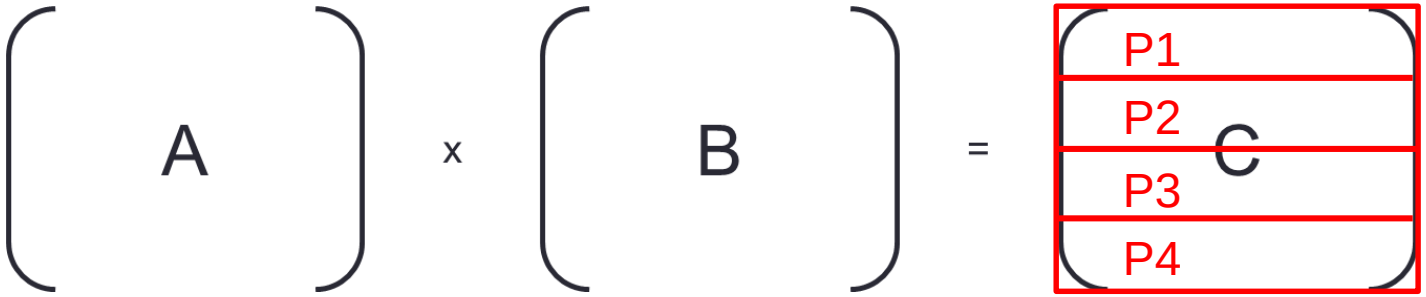
# Task Partition

Original Author: Prof. Li-Ping, Chang

- 1-process matrix multiplication



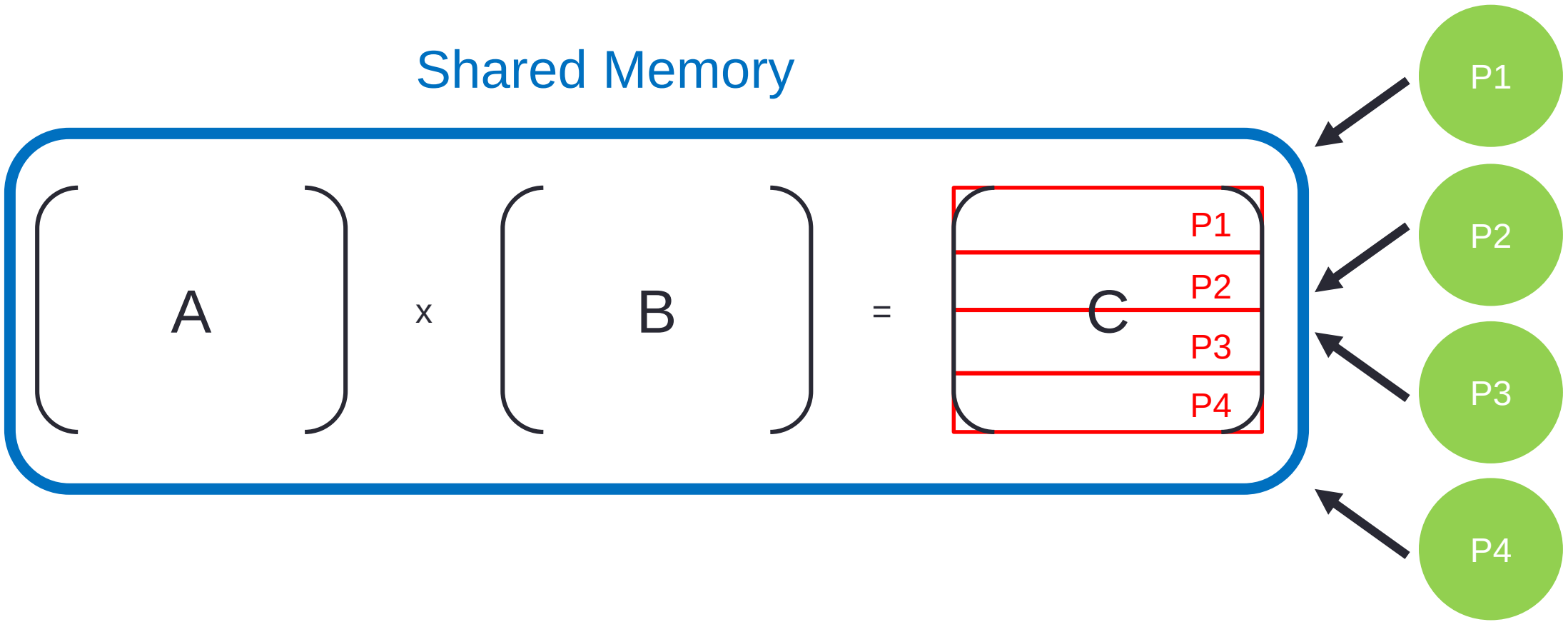

- 4-process matrix multiplication



# Shared Memory

Original Author: Prof. Li-Ping, Chang

- Matrices A, B, and C are stored in a shared memory
  - No memory locking is required since sub-matrix multiplications are mutually independent



# Matrix Initial Values

Original Author: Prof. Li-Ping, Chang

- Matrix elements in A and B are initialized as follows
- (suppose the dimension is 8x8)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 |
| 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 |
| 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 |
| 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 |
| 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 |

Checksum = 2016

# Requirements

- Use fork() to create worker processes.
- Do NOT declare any of the matrices as global arrays.
  Put all the matrices in the shared memory.
- Print the time and checksum of 1 & 4 process numbers.
- The final checksum must be correct
  - The TAs will test any matrix dimension between 100 and 800
  - Elements & Checksum are 32-bit unsigned integers
- 4-process version must be noticeably faster than 1-process version
- Violating any of the requirements will receive a score penalty

# Output

```
17:13          @bsdl [~/106_OS/hw1] >./a.out
Dimension: 800
1-process, checksum = 561324032
elapsed 5.524694  s
4-process, checksum = 561324032
elapsed 2.871483  s
```

# RULES

# Rules

- 0. Use NCTU CS Workstation as your programming environment
- 1. Do NOT create ZOMBIE process, otherwise you will get -10 pts
- 2. Use only C/C++, OTHER LANGUAGES WILL GET 0 POINT!
- 3. Filename format: StudentID_hw1-1.c & StudentID_hw1-2.c (or .cpp)
- 4. Put two .c files into compressed file, and name it StudentID_OS_hw1.zip
- 5. Incorrect filename format will get -5 pts
- 6. Total score: 100pts. PLAGIARISM WILL GET 0 POINT!
- 7. Deadline: 2017/10/15 (Sun) PM11:59
- 8. DELAYED SUBMISSION WILL GET 0 POINT!

- If you have any question, just send email to TAs.

# APPENDIX – HW 1-2

Original Author: Prof. Li-Ping, Chang

# Appendix – Header Files

- unistd.h
- sys/ipc.h
- sys/shm.h
- sys/wait.h

- sys/time.h

# Appendix – APIs

Original Author: Prof. Li-Ping, Chang

- shmget() – create a block of shared memory
- shmat() – attach shared memory to the current process's address space
- shmdt() – detach shared memory from the current process's address space
- shmctl() – control shared memory

- gettimeofday()

# Appendix – shmget

Original Author: Prof. Li-Ping, Chang

- **`int shmget(key_t` *`key`*`, size_t` *`size`*`, int` *`shmflg`*`);`**

- create a block of shared memory
- Return the ID of the request shm of size equals to the value of *`size`*
- *`key`* : 0/IPC_PRIVATE for new allocate shm
- *`size`* : size in bytes
- *`shmflg`* : IPC_CREAT | mode_flags(9 bits)

  e.g. IPC_CREAT | 0600  for read only shm

# Appendix – shmat

Original Author: Prof. Li-Ping, Chang

- **`void *shmat(int` *`shmid`*`, const void *`*`shmaddr`*`, int` *`shmflg`*`);`**

- Attach shared memory to the current process's address space
- Return the address of the attached shared memory segment identified by *`shmid`*
- *`shmaddr`* : NULL for system to choose suitable address

- *`shmflg`* : 0 for read/write, **`SHM_RDONLY`** for read only

# Appendix – shmdt

Original Author: Prof. Li-Ping, Chang

- **`int shmdt(const void *`*`shmaddr`*`);`**

- Detach the shared memory segment located at the address *`shmaddr`* from the address space of the calling process
- *`shmaddr`* must equal to the value returned by **`shmat()`**

# Appendix – shmctl

Original Author: Prof. Li-Ping, Chang

- `int shmctl(int shmid, int cmd, struct shmid_ds *buf);`

- Control shared memory
- Perform control operation
  - `IPC_STAT`
  - `IPC_SET`
  - `IPC_RMID`
  - …

- **`IPC_RMID`: Marking a shared memory to be deleted. The share memory will be destroyed on when the last process detach the memory from its address space. Must be called by the creator of the shared memory.**

# Appendix – gettimeofday

Original Author: Prof. Li-Ping, Chang

```
struct timeval start, end;
gettimeofday(&start, 0);

//do something

gettimeofday(&end, 0);
int sec = end.tv_sec – start.tv_sec;
int usec = end.tv_usec – start.tv_usec;

printf(“elapsed %f ms”, sec*1000+(usec/1000.0));
```

# Appendix – API Reference

- http://blog.csdn.net/guoping16/article/details/6584058
- http://man7.org/linux/man-pages/man2/shmget.2.html
- http://man7.org/linux/man-pages/man2/shmat.2.html