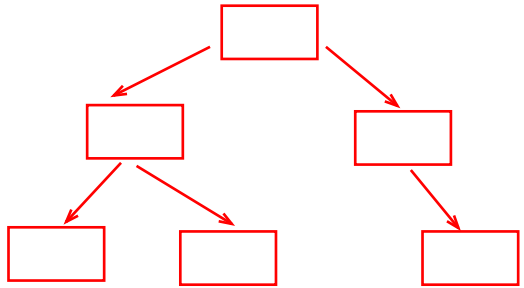
进程创建

系统允许一个进程创建新进程，新进程即为子进程，子进程还可以创建新的子进程，形成进程树结构模型。



```
#include <sys/types.h>
#include <unistd.h>
pid_t fork(void);
```

返回值：

- 成功：子进程中返回 0，父进程中返回子进程 ID
- 失败：返回 -1

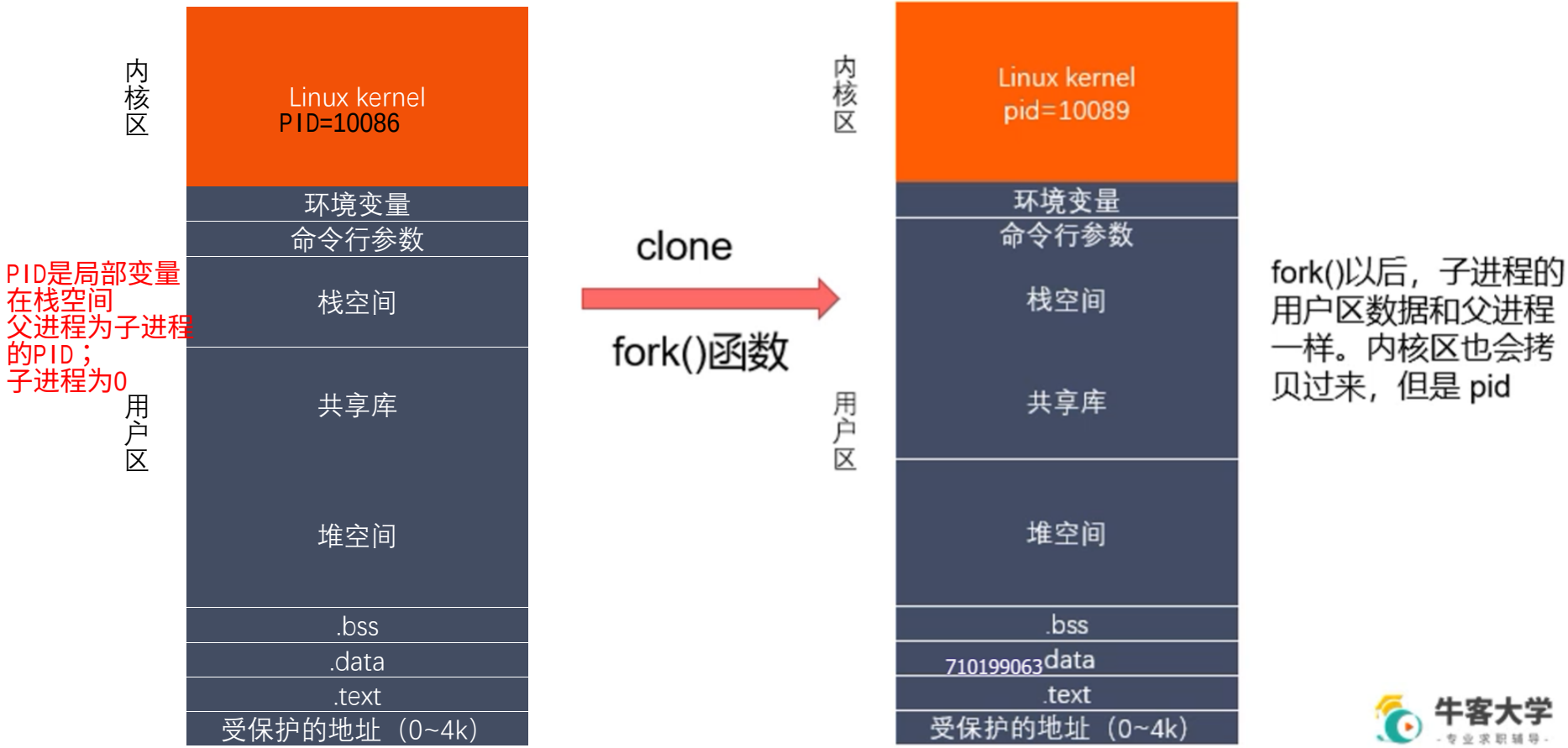
失败的两个主要原因：

1. 当前系统的进程数已经达到了系统规定的上限，这时 errno 的值被设置为 EAGAIN
2. 系统内存不足，这时 errno 的值被设置为 ENOMEM

```c
int main() {                                   父进程

    pid_t pid = fork();

    if(pid > 0) {
        printf("i am parent process, pid : %d, ppid : %d\n", getpid(), getppid());
    } else if(pid == 0) {
        printf("i am child process, pid : %d, ppid : %d\n", getpid(),getppid());
    }

    for(int i = 0; i < 3; i++) {
        printf("i : %d , pid : %d\n", i , getpid());
        sleep(1);
    }
    return 0;
}
```

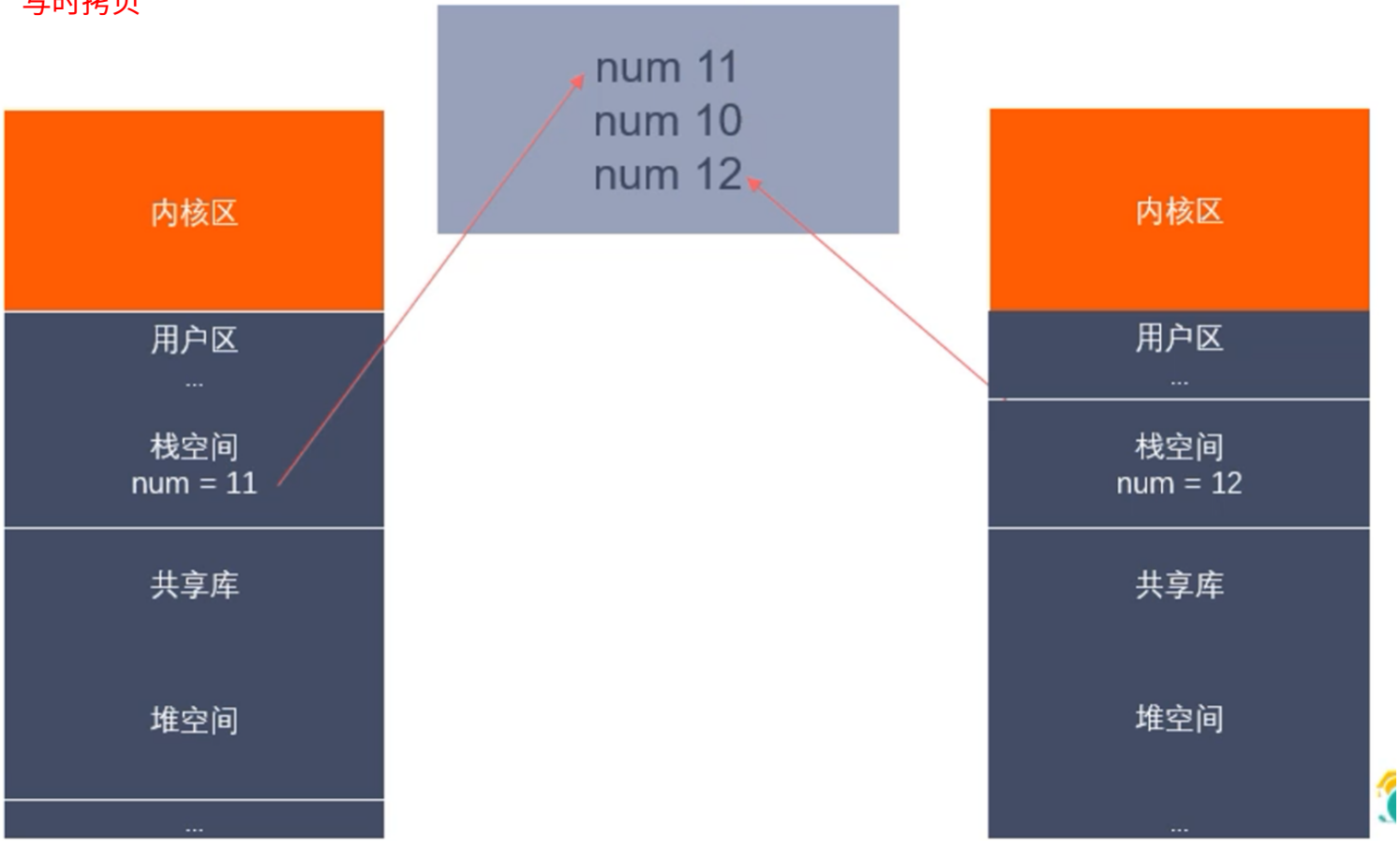
```c
int main() {                                   子进程

    pid_t pid = fork();

    if(pid > 0) {
        printf("i am parent process, pid : %d, ppid : %d\n", getpid(), getppid());
    }else if(pid == 0) {
        printf("i am child process, pid : %d, ppid : %d\n", getpid(),getppid());
    }

    for(int i = 0; i < 3; i++) {
        printf("i : %d , pid : %d\n", i , getpid());
        sleep(1);
    }
    return 0;
}
```

02 / 父子进程虚拟地址空间



02 / 父子进程虚拟地址空间

写时拷贝



使用 GDB 调试的时候，GDB 默认只能跟踪一个进程，可以在 fork 函数调用之前，通过指令设置 GDB 调试工具跟踪父进程或者是跟踪子进程，默认跟踪父进程。

show follow-fork-mode 查看当前调试的是什么进程

设置调试父进程或者子进程：set follow-fork-mode [parent（默认）| child]

设置调试模式：set detach-on-fork [on | off]

默认为 on，表示调试当前进程的时候，其它的进程继续运行，如果为 off，调试当前进程的时候，其它进程被 GDB 挂起。

查看调试的进程：info inferiors

切换当前调试的进程：inferior id

使进程脱离 GDB 调试：detach inferiors id

牛客大学
-专业求职辅导-

THANKS

关注【牛客大学】公众号
回复“牛客大学”获取更多求职资料