HW2

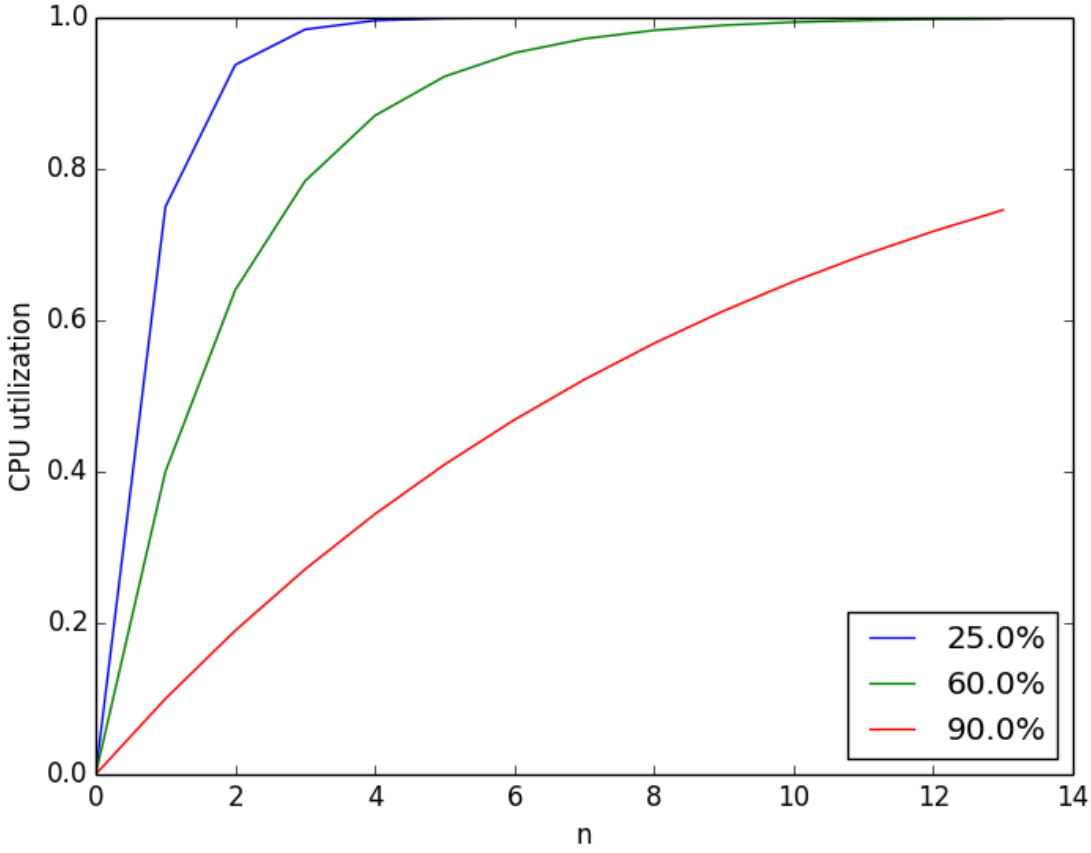
Wu Jiayao 517370910257

Ex.1

1.

The probability is p^n. The CPU utilization is $1-p^n$.

2.



3.

a)

$$\lfloor (256-96) \div 48 \rfloor = 3$$

Therefore, 3 processes can be stored in memory simultaneously

b)

$$1 - 0.9^3 = 27.1\%$$

The CPU utilization is 27.1%.

c)

Assume an average of 90% I/O waiting time.

For 256MB, the number of simultaneously running program is

$$\lfloor (512 - 96) \div 48 \rfloor = 8$$

The CPU utilization is

$$1 - 0.9^8 = 57.0\%$$

The effect can be measured as 29.9% per 256MB.

For 512MB, the number of simultaneously running program is

$$\lfloor (768 - 96) \div 48 \rfloor = 14$$

The CPU utilization is

$$1 - 0.9^{14} = 77.1\%$$

The effect can be measured as 25% per 256MB.

For 1024MB, the number of simultaneously running program is

$$\lfloor (1280 - 96) \div 48 \rfloor = 24$$

The CPU utilization is

$$1 - 0.9^{24} = 92.0\%$$

The effect can be measured as 16.2% per 256MB.

Hence, 256MB is the most beneficial and worth the investment.

Ex.2

Note: Two lines of * comments in the file to modified is to help the grader to locate the change in related file more quickly, which are actually not added into the file

3 files need to be modified.

1. The first is dmp.c in minix/usr/src/servers/is/

```
#include "inc.h"
#include <minix/vm.h>

struct hook_entry {
  int key;
  void (*function)(void);
  char *name;
} hooks[] = {
  { F1,   proctab_dmp, "Kernel process table" },
  { F3, image_dmp, "System image" },
  { F4, privileges_dmp, "Process privileges" },
  { F5, monparams_dmp, "Boot monitor parameters" },
  { F6, irqtab_dmp, "IRQ hooks and policies" },
  { F7, kmessages_dmp, "Kernel messages" },
  { F8, vm_dmp, "VM status and process maps" },
  { F10,  kenv_dmp, "Kernel parameters" },
  { SF1,  mproc_dmp, "Process manager process table" },
  { SF2,  sigaction_dmp, "Signals" },
  { SF3,  fproc_dmp, "Filesystem process table" },
  { SF4,  dtab_dmp, "Device/Driver mapping" },
  { SF5,  mapping_dmp, "Print key mappings" },
  { SF6,  rproc_dmp, "Reincarnation server process table" },
  //**************************************
  { SF7,  procNum_dmp,"Display the number of currently running processes" },
  //**************************************
  { SF8,  data_store_dmp, "Data store contents" },
  { SF9,  procstack_dmp, "Processes with stack traces" },
};
```

To map shift+F7, function name and description about SF7 is added into the **dmp.c**

2. The second is proto.h in minix/usr/src/servers/is

```
/* dmp_kernel.c */
void proctab_dmp(void);
void procstack_dmp(void);
void privileges_dmp(void);
void image_dmp(void);
void irqtab_dmp(void);
void kmessages_dmp(void);
void monparams_dmp(void);
void kenv_dmp(void);
//**************************************
void procNum_dmp(void);
//**************************************
```

Declare the function **procNum_dmp(void)** for Shift+F7 into header file **proto.h**

3. The third is dmp_kernel.c in minix/usr/src/servers/is

```
/*===========================================================================*
 *                                procNum_dmp                                *
 *===========================================================================*/
void procNum_dmp()
{
  register struct proc *rp;
  int r;
  if((r=sys_getproctab(proc))!=OK)
  {
    printf("IS: warning:couldn't get copy of process table: %d\n",r);
  }
  int procNum = 0;
  for(rp =BEG_PROC_ADDR;rp < END_PROC_ADDR;rp++)
  {
    if(isemptyp(rp))
      continue;
    procNum++;
  }
  printf("The number of currently running process is %d\n",procNum);
}
```

Add the implementation of function **procNum_dmp(void)** into **dmp_kernel.c**

The implementation reference is from **dmp_kernel.c** itself.

4. Rebuild the kernel

Reference from offical wiki.

```
cd /usr/src/releasetools
make hdboot
```

Press Shift+F7.

```
# The number of currently running process is 42
```

Appendix

Plot code with python in Ex.1--2

```
from pylab import *
import numpy as np
X = np.linspace(0, 13, 14, endpoint=True)
p = [0.25, 0.6, 0.9]
A = [0, 0, 0]
for i in range(0, 3):
    A[i] = 1 - p[i]**X
    tmp = str(p[i]*100) + '%'
    plot(X, A[i], label=tmp)
xlabel('n')
ylabel('CPU utilization')
legend(loc='lower right')
show()
```