پروژه درس سیستم عامل

استاد درس: دکتر جوادی

گردآورنده:

پویان حسابی: ۹۷۳۱۱۲۲

امید ماهیار: ۹۷۳۱۱۰۰

تاریخ: ۲۰ بهمن ۱۳۹۹

مقدمه:

در این پروژه با تغییرات و پیاده سازی چند تابع در سیستم عامل xv6، مفاهیم درس روشن تر می شود. پس از آشنایی با تعدادی از فایل های پیاده سازی این سیستم عامل سراغ مفاهیم فراخوانی سیستم می رویم و در ۳ فایل جداگانه pid فرایند های فرزند و والد و همینطور تعداد فراخوانی آنها را انجام می دهیم. سپس دو زمانبندی پردازنده را انجام می دهیم و بعد از تست های آن زمان های انفجار، انتظار و گردش کار را بدست می آوریم.

بخش اول: سوالات

جواب هر سه سوال در قالب فایل pdf ضمیمه شده است.

بخش دوم: فراخوانی سیستمی

getParentID :

در این قسمت باید آیدی والد فرآیند در حال اجرا را به دست آوریم.با توجه به این که به صورت پیش فرض آیدی فرآیند فرآیند والد در هر فرآیند ذخیره میشود ، فقط کافی است در یک سیستم کال این آیدی را یازگردانی کنیم.

getChildrenID :

پیاده سازی این قسمت با توجه به این که به صورت پیش فرض آیدی فرزندان در والد وجود ندارد کمی پیچیده تر است.برای به دست آوردن آیدی فرزندان در زمان فراخوانی این سیستم کال یک حلقه را process table طی میکنیم و آیدی فرآیند هایی که والد آنها برابر با فرآیند در حال اجرا باشد را در آرایه children ذخیره کرده و در فایل تست آن را چاپ میکنیم.

getSyscallCounter :

بعد از درست کردن فایل تست، باید فراخوانی سیستمی برای آن در syscall.h تعریف کنیم، که وقتی آنرا صدا می کنیم تعداد فراخوانی شدن آنرا نشان دهد. بعد از اینکه در کرنل مد آنرا تعریف کردیم با تغییرات در فایل ها آنرا در سطح کاربر هم تنظیم می کنیم که به شکل فراخوانی سیستمی بتوانیم در تست از آن استفاده کنیم. در واقع در user.h و توابع پیرو آن، interface ها را قرار می دهیم.

بخش سوم: پیاده سازی الگوریتم های زمان بندی

- الگوریتم Round Robin:

همانطور که می دانیم در حالت عادی الگوریتم این سیستم عامل راند رابین می باشد، حال ما با تغییرات در کد آن قصد داریم کوانتم زمانی آنرا تغییر دهیم. که نکته ای که قابل توجه است، این است که برای هر فرایند یک کلاک در نظر می گیریم و در فایل trap.c با تغییراتی که در آن اعمال می کنیم هر گاه وقفه زمان گرفت، آن کلاک را زیاد می کنیم. غیر از آن باید شرط اینکه اگر کلاک فرایند از کوانتم زمانی بیشتر شد پردازنده آنرا قبضه کند و دوباره تابع scheduler را فراخوانی کند.

- الگوریتم زمان بندی طبق الویت:

پیاده سازی این قسمت شامل دو بخش اصلی بود.بخش اول باید سیستم کالی طراحی میشد که به هر فرآیند الویت خاصی را نسبت دهد و بخش دوم ایجاد تغییرات در تابع scheduler که در فایل proc.c واقع شده است،که این تغییرات برای انتخاب فرآیند با الویت بالاتر برای اجرا به جای روش معمول است.

بخش چهارم: کد های کمکی و ارزیابی کد های بالا

- changePolicy :

برای انتخاب الگوریتم زمان بندی احتیاج به این سیستم کال بود به این صورت که عددی را به عنوان ورودی دریافت و متغییری به نام policy را در فایل proc.c به همین مقدار تغییر میدهد و در تابع scheduler این متغییر تعیین کننده روش الگوریتم زمان بندی میشود.

- قابلیت اندازه گیری زمان:

ابتدا زمان های creation, termination, ready, running, sleeping را بدست می آوریم. که به این نحو است که پس از تعریف موارد فوق در هر فرایند، در تابعی نسبت به کلاک کل برنامه که در حال تغییر است این مقادیر را برای هر فرایند بروزرسانی میکنیم. برای هر کدام از این ۵ متغیر آرایه ای تبیین شده که بتوان آن ها را ذخیره کرد. نکته ای که حائز اهمیت است، نگهداری و برگرداندن این متغیر ها است که در تابعی که فرایند های فرزند بسته می شود این کار انجام می شود. با توجه به متغیر های بالا این پارامتر ها بدست می آید:

CPU Burst Time: running time

Waiting Time: ready time + sleeping time

Turnaround Time: termination time - creation time

- تست نویسی صف اولویت:

در این تست ۳۰ فرزند داریم که در هر کدام ۲۵۰ بار آیدی آن ها به همراه عدد چاپ می شود.

```
avg creation : 812
avg termination : 2380
avg running : 173
avg ready : 1362
avg sleeep : 26


CBT(running state): 173
Waiting time(ready + sleep): 1388
turnarround time(termmination - creation): 1568
```

- حالت عادی و پیش فرض:

در این تست ۱۰ فرزند داریم که در هر کدام ۱۰۰۰ بار آیدی آن ها به همراه عدد چاپ می شود. که از زمان بندی پیش فرض سیستم عامل استفاده شده است.

```
avg creation : 687
avg termination : 3231
avg running : 528
avg ready : 1899
avg sleeep : 115


CBT(running state): 528
Waiting time(ready + sleep): 2014
turnarround time(termmination - creation): 2544
```

- تست نویسی راند رابین:

در این تست ۱۰ فرزند داریم که در هر کدام ۱۰۰۰ بار آیدی آن ها به همراه عدد چاپ می شود. که خروجی آن در صفحات بعدی قابل مشاهده است.

```
avg creation : 611
avg termination : 3125
avg running : 526
avg ready : 1874
avg sleeep : 112


Quantum is 1

CBT(running state): 526
Waiting time(ready + sleep): 1986
turnarround time(termmination - creation): 2514
```

```
avg creation : 513
avg termination : 3104
avg running : 539
avg ready : 1940
avg sleeep : 108


Quantum is 2

CBT(running state): 539
Waiting time(ready + sleep): 2048
turnarround time(termmination - creation): 2591
```

```
avg creation : 654
avg termination : 3181
avg running : 527
avg ready : 1884
avg sleeep : 115


Quantum is 5

CBT(running state): 527
Waiting time(ready + sleep): 1999
turnarround time(termmination - creation): 2527
```

```
avg creation : 498
avg termination : 2956
avg running : 510
avg ready : 1830
avg sleeep : 116


Quantum is 10

CBT(running state): 510
Waiting time(ready + sleep): 1946
turnarround time(termmination - creation): 2458
```

```
avg creation : 510
avg termination : 3013
avg running : 523
avg ready : 1865
avg sleeep : 112


Quantum is 20

CBT(running state): 523
Waiting time(ready + sleep): 1977
turnarround time(termmination - creation): 2503
```

```
avg creation : 338
avg termination : 2847
avg running : 521
avg ready : 1861
avg sleeep : 125


Quantum is 50

CBT(running state): 521
Waiting time(ready + sleep): 1986
turnarround time(termmination - creation): 2509
```

```
avg creation : 289
avg termination : 2758
avg running : 514
avg ready : 1850
avg sleeep : 104


Quantum is 100

CBT(running state): 514
Waiting time(ready + sleep): 1954
turnarround time(termmination - creation): 2469
```

```
avg creation : 293
avg termination : 2815
avg running : 522
avg ready : 1889
avg sleeep : 109


Quantum is 150

CBT(running state): 522
Waiting time(ready + sleep): 1998
turnarround time(termmination - creation): 2522
```

```
avg creation : 375
avg termination : 2850
avg running : 516
avg ready : 1846
avg sleeep : 112


Quantum is 200

CBT(running state): 516
Waiting time(ready + sleep): 1958
turnarround time(termmination - creation): 2475
```

```
avg creation : 323
avg termination : 2773
avg running : 511
avg ready : 1824
avg sleeep : 114


Quantum is 300

CBT(running state): 511
Waiting time(ready + sleep): 1938
turnarround time(termmination - creation): 2450
```

```
avg creation : 256
avg termination : 2818
avg running : 530
avg ready : 1906
avg sleeep : 123


Quantum is 400

CBT(running state): 530
Waiting time(ready + sleep): 2029
turnarround time(termmination - creation): 2562
```

```
avg creation : 560
avg termination : 3054
avg running : 521
avg ready : 1860
avg sleeep : 111


Quantum is 600

CBT(running state): 521
Waiting time(ready + sleep): 1971
turnarround time(termmination - creation): 2494
```

```
avg creation : 452
avg termination : 2945
avg running : 520
avg ready : 1859
avg sleeep : 112


Quantum is 800

CBT(running state): 520
Waiting time(ready + sleep): 1971
turnarround time(termmination - creation): 2493
```

همانطور که مشخص است از ۶۰۰ به بعد تقریبا تغییری نمی کند چرا که کوانتم زمانی آنقدر بالا است که تقریبا مثل صف عمل می کند و سیاست آن fcfs می شود.

همانطور که از ماهیت راند رابین مشخص است، از یک زمانی به بعد تغییر آنچنان در زمان ها نمی شود همانطور که گفته شد. برای context switching از صف اولیت مقدار waiting time بیشتری دارد. ولی بن بست و قحطی با این الگوریتم هیچ وقت بوجود نمی آید.

پایان