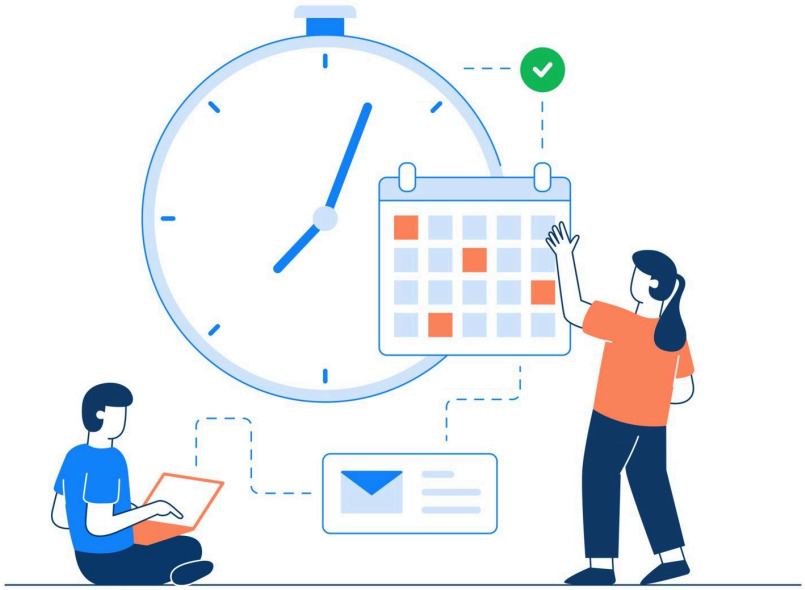
پروژه دوم درس سیستم‌های عامل

RT-Scheduling



شرح پروژه

در این پروژه ما قصد داریم سیستم الکتریکی یک خودرو را شبیه سازی کنیم. این سیستم الکتریکی تشکیل شده از زیرسیستم‌های مختلف که هر کدام ویژگی خاص خود را دارند و توسط زیرسیستم مرکزی کنترل و هماهنگ می‌شوند.
Main Thread ما زیر سیستم اصلی است که با 3 نخ دیگر که زیرسیستم‌های دیگر هستند در ارتباط است. همچنین هر زیرسیستم شامل چندین هسته پردازشی است که هر هسته نیز با یک نخ شبیه‌سازی می‌شود.
زیرسیستم تشکیل شده از تعدادی هسته پردازشی و منابع محدود که برای انجام وظیفه‌ها به آن‌ها نیاز داریم. زیرسیستم‌ها دارای دو صف Ready Queue و Waiting Queue هستند. (ممکن است صف‌ها تفاوت‌های کوچکی در هر زیرسیستم داشته باشند که در بخش مربوطه به آن اشاره خواهد شد.)

مفاهیم پروژه

صف Ready

این صف مربوط به وظیفه‌هایی می‌شوند که آماده اجرا هستند و ترتیب آن‌ها با توجه به الگوریتم‌های زمان‌بندی مطرح شده مشخص می‌گردد.

پروژه دوم درس سیستم‌های عامل

RT-Scheduling

صف Waiting

این صف مربوط به وظیفه‌هایی می‌شود که امکان اجرای آن‌ها وجود دارد، ولی منابع مورد نیاز آن‌ها موجود **نیست**. مثلاً هنگامی که یک پردازنده وظیفه‌های از صف اولویت انتخاب می‌کند ولی منابع آن موجود نیست، سیستم این وظیفه را از صف اولویت خارج کرده و در صف انتظار قرار می‌دهد.

برای جلوگیری از Starvation، باید راه‌حلی برای برگرداندن وظیفه‌ها به صف اولویت در نظر گرفته شود. در نتیجه لازم است الگوریتمی برای مرتب کردن این صف با توجه به منابع آزاد سیستم و به دست آوردن بهترین بهره‌وری از پردازنده‌ها پیاده‌سازی شود.

منابع

در سیستم شما 2 نوع منبع R1 و R2 وجود دارد که در هنگام شروع برنامه تعداد هر یک از آنها، به ازای هر زیرسیستم به شما داده می‌شود. منابع در هر زمان فقط توسط یک پردازنده قابل استفاده‌اند و به صورت **اشتراکی** استفاده **نمی‌شوند**.

وظایف

برای هر وظیفه مدت زمان مورد نیاز برای اجرای آن داده می‌شود. همچنین باید در ساختار وظیفه فیلدی برای **ذخیره وضعیت وظیفه** در نظر بگیرید که نشان‌دهنده‌ی State آن در سیستم است (Ready/Waiting/Running).

زیرسیستم اصلی

این زیر سیستم وظیفه چاپ خروجی و هماهنگ کردن زیرسیستم‌ها را دارد. (با توجه به ماهیت نخ ممکن است یک نخ زودتر کارش را انجام دهد ولی ما می‌خواهیم زیرسیستم‌ها و هسته‌های پردازشی همگی همزمان یک واحد زمانی را طی کنند.)

زیرسیستم اول

این زیرسیستم دارای یک Waiting Queue و به ازای **هر پردازنده** دارای یک Ready Queue است (یعنی در کل 3 صف اولویت و یک صف Waiting داریم). همچنین دارای 3 هسته پردازشی است که توسط 3 نخ شبیه‌سازی می‌شوند و توسط نخ اصلی زیرسیستم هماهنگ می‌شوند. این سه هسته باید وظیفه‌هایی که به آن‌ها تعلق دارد را با استفاده از الگوریتم **Weighted Round Robin** اجرا کنند. توجه کنید چون ما 3 صف اولویت داریم، در ابتدای برنامه می‌توانیم مشخص کنیم هر وظیفه در ابتدا به کدام هسته پردازشی برود.

پروژه دوم درس سیستم‌های عامل

RT-Scheduling

زیرسیستم دوم

این زیرسیستم فقط یک Ready Queue دارد و **Waiting Queue در آن نداریم**. همچنین دارای 2 هسته پردازشی است که توسط 2 نخ شبیه‌سازی می‌شوند و توسط نخ اصلی زیرسیستم هماهنگ می‌شوند. این دو هسته باید وظیفه‌هایی که به آن‌ها تعلق دارد را با استفاده از **الگوریتم Shortest Remaining Time First** اجرا کنند. توجه کنید چون در این سیستم ما صف انتظار نداریم، باعث میشود که Hold and Wait وجود داشته باشد و با **DeadLock** مواجه شویم پس لازم است الگوریتمی برای مقابله با DeadLock پیاده‌سازی کنید.

زیرسیستم سوم

این زیرسیستم، یک زیرسیستم **RealTime** است که دارای یک Ready Queue و یک Waiting Queue است. همچنین دارای 1 هسته پردازشی است که توسط 1 نخ شبیه‌سازی می‌شود. این هسته باید وظیفه‌هایی که به آن تعلق دارد را با استفاده از الگوریتم **Rate Monotonic** اجرا کنند. توجه کنید که تمام وظیفه‌های این زیرسیستم **Hard RealTime** هستند و حتما باید در ددلاین‌های مشخص شده اجرا شوند.

اگر وظیفه‌ها قابل برنامه‌ریزی نبودند، این زیرسیستم به اندازه منابع مورد نیاز برای اجرا وظیفه، از زیرسیستم‌های دیگر منبع می‌گیرد و به ازای هر واحد زمانی، دو واحد زمانی از وظیفه اجرا می‌شود. به عنوان مثال یک وظیفه‌ای که 4 واحد زمانی طول می‌کشد، در دو واحد زمانی اجرا می‌شود. توجه کنید منابعی که از زیرسیستم‌های دیگر گرفته می‌شود تا زمانی در اختیار این زیرسیستم است که وظایف به ددلاین نرسند. اگر وظیفه‌ها قابل برنامه‌ریزی بودند، منابع باید به زیرسیستم خودشان پس داده شوند.

زیرسیستم چهارم (نمره اضافه)

این زیرسیستم دارای یک Waiting Queue و یک Ready Queue است. همچنین دارای 2 هسته پردازشی است که توسط 2 نخ شبیه‌سازی می‌شوند و توسط نخ اصلی زیرسیستم هماهنگ می‌شوند. این دو هسته باید وظیفه‌هایی که به آن‌ها تعلق دارد را با استفاده از الگوریتم **FCFS** اجرا کنند. توجه کنید وظیفه‌های این زیرسیستم ممکن است به یکدیگر وابسته باشند. به عنوان مثال برای اجرای وظیفه دوم، وظیفه اول حتما باید اجرا شده باشد. (ممکن است زمان ورود وظیفه‌ای که پیش‌نیاز یک وظیفه دیگر است، دیرتر از آن وظیفه باشد.) همچنین با احتمال 30%، وظیفه‌ها بعد از اجرا با خطا مواجه می‌شوند و باید دوباره اجرا شوند. (در صورت وقوع این مورد، باید در خروجی اطلاع داده شود.)

مواردی که این برنامه باید بتواند انجام دهد

1. در زیرسیستم اول چون ما چند صف اولویت داریم، با توجه مطالب آموخته شده، باید الگوریتمی برای **Load Balancing** پیاده‌سازی کنید.
2. در پایان برنامه، گزارشی از وظیفه‌های اجرا شده مانند زمان ورود و زمان پایان اجرا، مدت زمان سپری شده در صف انتظار، هسته‌ها و زیرسیستمی که آن وظیفه را اجرا کرده است باید ارائه شود.
3. به جز گزارش وظیفه‌ها در پایان برنامه، در هر واحد زمانی باید گزارشی از وضعیت سیستم ارائه شود که در ادامه به آن می‌پردازیم.

پروژه دوم درس سیستم‌های عامل

RT-Scheduling

فرمت ورودی و خروجی

فرمت ورودی اولیه:

در ابتدا تعداد منبع برای هر زیرسیستم وارد می‌شود.

تعداد منبع ۲	تعداد منبع ۱
تعداد منبع ۲	تعداد منبع ۱
تعداد منبع ۲	تعداد منبع ۱
تعداد منبع ۲	تعداد منبع ۱

سپس وظیفه‌های هر زیرسیستم به ترتیب وارد می‌شوند. (هر سطر مربوط به فرمت وظیفه یک زیرسیستم است و وظیفه‌های زیر سیستم‌های متفاوت با **$** از هم جدا می‌شوند.)

	شمارنده پردازنده مقصد	زمان ورود	مصرف منبع۲	مصرف منبع ۱	مدت زمان	نام
		زمان ورود	مصرف منبع۲	مصرف منبع ۱	مدت زمان	نام
تعداد دفعات تکرار	Period	زمان ورود	مصرف منبع۲	مصرف منبع ۱	مدت زمان	نام
	نام وظیفه پیشنیاز	زمان ورود	مصرف منبع۲	مصرف منبع ۱	مدت زمان	نام

مثال:

```
3 3
2 5
4 10
2 2
T11 4 1 0 0 1
T12 10 0 1 0 2
T13 20 2 0 0 3
$
T21 4 2 3 5
$
T31 2 2 3 4 10
$
T41 2 2 3 5 T42
T42 4 2 2 7 -
```

پروژه دوم درس سیستم‌های عامل

RT-Scheduling

فرمت خروجی اولیه:

```
Time: 1

Sub1:

        Resources: R1: 0 R2: 2

        Waiting Queue []
        Core1:

                Running Task: T11
                Ready Queue: []

        Core2:

                Running Task: T12
                Ready Queue: []

        Core3:

                Running Task: T12
                Ready Queue: []

Sub2:

        Resources: R1: 2 R2: 5

        Ready Queue: []

        Core1:

                Running Task: idle (یعنی هسته بیکار است)

        Core2:

                Running Task: idle


Sub3:

        Resources: R1: 4 R2: 10

        Waiting Queue []

        Ready Queue: []

        Core1:

                Running Task: idle



Sub3:

        Resources: R1: 2 R2: 2

        Waiting Queue []

        Ready Queue: []

        Core1:

                Running Task: idle

        Core1:

                Running Task: idle
```

پروژه دوم درس سیستم‌های عامل

RT-Scheduling

این خروجی به ازای هر واحد زمانی باید چاپ شود. علاوه بر این، موارد ذکر شده در صورت پروژه در پایان برنامه نیز باید چاپ شوند.

نمره اضافه

---

1. طراحی محیطی گرافیکی که به مانیتورینگ Utilization هر پردازنده پرداخته و آن را به صورت گرافیکی نشان دهد.
2. پیاده سازی زیرسیستم چهارم
3. خلاقیت (پیاده سازی مواردی که در صورت پروژه ذکر نشده)

توضیحات تکمیلی

---

1) امکان استفاده از sleep در هیچ جای پروژه **ممکن نیست**.

2) پروژه در گروه‌های 2 نفره قابل انجام است. (آپلود توسط یکی از اعضا کافی‌ست.)

گروه‌ها **می‌بایست** از گیت استفاده کنند.

3) استفاده از زبان‌های Java, Python, C, CPP مجاز است. (در صورت استفاده از C/C++ از makefile استفاده شود.)

4) مراحل پیاده‌سازی و نحوه‌ی اجرای برنامه‌ی خود را حتماً در فایل readme.md **به صورت کامل** توضیح دهید.

5) هنگام تحویل، هر دو عضو گروه باید **تسلط کامل** داشته باشند.

6) یک نمونه ورودی و خروجی از برنامه خود را در قالب دو فایل in.txt و out.txt ذخیره داشته باشید.

7) فایل نهایی (شامل کد، فایل readme.md، فایل in.txt و فایل out.txt) را به‌فرمت "RT-Scheduler_<Student.IDs>_<Student.names>.zip" در Vu بارگذاری کنید.

8) ***در صورت مشاهده هرگونه شباهت میان گروه‌ها، اگر درصد شباهت بیشتر از سی درصد (±3%) باشد؛ نمره طرفین به صورت*** $100 - 2a$ ***خواهد بود. (a به‌معنای درصد تشابه است.)***.

مهلت تحویل

جمعه 28 دی ماه 1403 شمسی؛ ساعت 23:59.

پروژه دوم درس سیستم‌های عامل

RT-Scheduling

موفق باشید