گزارش پروژه نهایی درس ارزیابی شبکه های کامپیوتری مسئله ۲-۴

دانشجو: محمد امینی_ ۸۱۰۱۰۰۲۹۴

mohammad.amini.98@ut.ac.ir

۱ simlib WITH SIMULATION QUEUEING SINGLE-SERVER

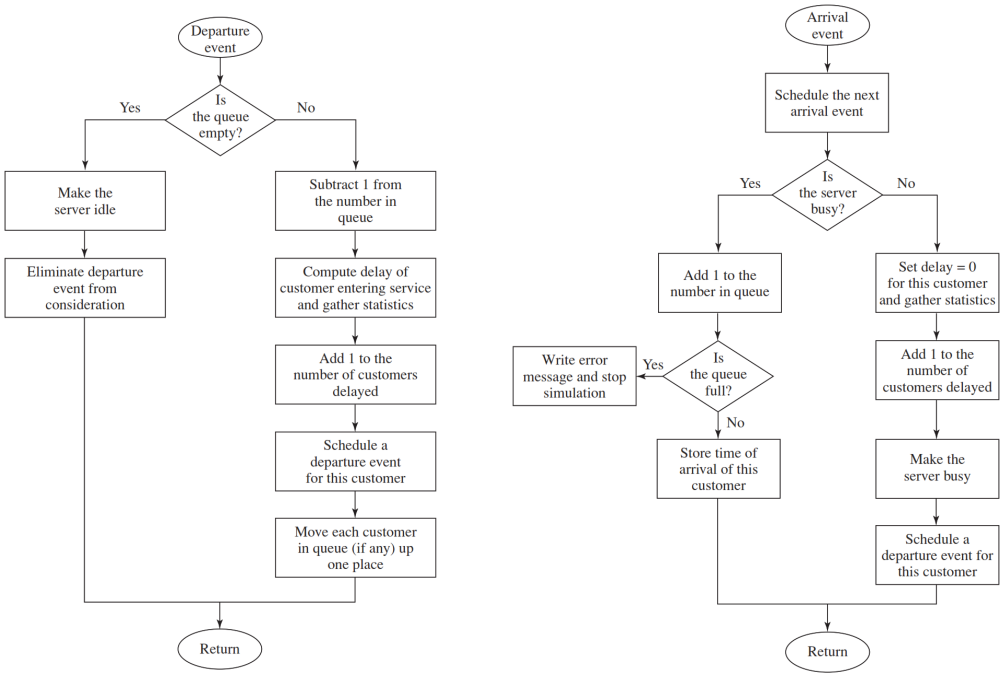
این شبیه سازی دقیقا همان شبیه سازی مسئله ۱ است با این تفاوت که با استفاده از ابزار simlib زبان c نوشته شده است به همین دلیل برای پیاده سازی آن در زبان پایتون ابتدا کتابخانه مورد نظر در پایتون پیدا سازی شده است .توضیحات کلی مسئله مانند سوال ۱ به شرح زیر است :
ـدر این مسئله به دنبال شبیه سازی مسئله mm1 هستیم که از یک سرور و یک صف تشکیل شده است . زمان ورود مشتری ها به صف یک عدد رندم نمایی با میانگین mean-interarrival است و همچنین زمانی که سرور برای پردازش هر مشتری اختصاص میدهد نیز یک زمان زندم با میانگین mean-service است.
این مسئله را به شکل event-base با دو حالت departure-event ، arrival-event بررسی میکنیم .

۱.۱ **arrival-event** :

فوچارت این event در شکل یک رسم شده است . زمانی که یک مشتری وارد صف میشود برنامه وارد این event میشود در ابتدا سیستم وضغیت server را بررسی میکند ، اگر در وضعیت آزاد باشد بلافاصله مشتری درون سرور قرار میگرد و در ادامه وضعیت سرور busy میشود و همچنین زمان اتمام پردازش مشتری محاسبه میشود .
اگر سرور از قبل در وضعیت busy باشد مشتری به صف اضافه میشود و بر تعداد مشتری های درون صف یک واحد اضافه میشود . در این برنامه برای اینکه تاخیر منتظر ماندن در صف ممکن است بی نهایت شود یک thrershold برای تعداد مشتری های درون صف در نظر گرفته میشود به همین دلیل بررسی میکنیم که تعداد حاضرین در صف بزرگتر از thrershold نباشد چناچنه این اتفاق رخ بدهد شبیه سازی پایان میابد و error نمایش داده میشود ولی در حالت نرمال زمان رسیدن مشتری ثبت میشود و مشتری منتظر ازاد شدن سرور و رسیدن به ابتدای صف برای گرفتن پردازش میشود.

۲.۱ **departure-event** :

فلوچارت این event در شکل ۲ رسم شده است . زمانی که عملیات پردازش در سرور تمام میشود ابتدا وضعیت صف بررسی میشود اگر صف خالی باشد سرور در حالت idle قرار میگرد و اگر مشتری در صف باشد ، پردازش اولین مشتری در صف شروع میشود و از تعداد مشترین حاضر در صف یک واحد کاسته میشود.



arrive (آ) departe (ب)

شکل ۱: فلوچارت رسیدن و خروج

۲ شبیه سازی مسئله در نرم افزار python

ـاین شبیه سازی به زبان python نوشته شده است و برای فهم بهتر از متغییر های کد c استفاده شده است . در ادامه کد های مربوط به شبیه سازی را توضیح میدهیم.

- معیار اتمام شبیه سازی در این کد متغییر num-delayed-requried است که برابر است مقدار ۱۰۰۰ تنظیم شده است ، کد اصلی درون یک while نوشته شده که همواره num-cuts-delay با این مقدار مقایسه میشود و هنگاهی که متغییر num-cuts-delay ازین مقدار بیشتر شد شبیه سازی پایان میابد و نتیجه نمایش داده میشود.

ـدر این شبیه سازی زبان بندی مسئله به وسیله تابع timing کتابخانه simlib انجام میشود همچنین با رسیدن یک مشتری جدید به وسیله تابع simlib.list-file در صورتی که شرایط ادامه شبیه سازی وجود داشته باشد مشتری به صف اضافه میشود و در مقابل پس از پایان پردازش در سرور به وسیله simlib.list-remove از لیست حذف میشود.

ـتابع arrive

طبق کد زیر با رسیدن یک مشتری ایتدا توسط simlib.event-schedule زمان رسیدن مشتری بعدی به صورت random پیش بینی میشود سپس با استفاده از simlib.list-size[LIST-SERVER] == ۱ وضعیت سرور بررسی میشود اگر شرط برقرار باشد به معنی شلوغ بودن سرور است که در نتیجه آن زمان رسیدن ثبت و مشتری به اخر صف اضافه میشود و در غیر این صورت مشتری درون سرور قرار میگیرد و زمان پردازش آن توسط سرور به صورت رندم محاسبه میشود.

```python
def arrive():
    global num_custs_delayed, mean_interarrival, mean_service
    simlib.event_schedule(simlib.sim_time + simlib.expon(mean_interarrival, STREAM_INTERARRIVAL), EVENT_ARRIVAL)

    if simlib.list_size[LIST_SERVER] == 1:

        simlib.transfer[1] = simlib.sim_time
        simlib.list_file(simlibdefs.LAST, LIST_QUEUE)
    else:
        simlib.sampst(0.0, SAMPST_DELAYS)
        num_custs_delayed += 1
        simlib.list_file(simlibdefs.FIRST, LIST_SERVER)
        simlib.event_schedule(simlib.sim_time + simlib.expon(mean_service, STREAM_SERVICE), EVENT_DEPARTURE)
```

شکل ۲: کد ۱

ـتابع depart

در این تابع سرور وضعیت صف را با استفاده از simlib.list-size[LIST-QUEUE] == ۰ بررسی میکند چنانچه شرط برقرار باشد به معنی خالی بئدن صف انتظار سرور است که در این صورت سرور با خالی گذاشتن تعداد مشتری در حال پردازش در حالت ایده آل است و در صورت برقرار نبودن شرط فوق سرور از اول صف پردازش یک مشتری را آغاز میکند و با استفاده از زمانsimlib.event-schedule به اتمام رسیدن پردازش را محاسبه میکند.

```python
def depart():
    global num_custs_delayed, mean_service
    if simlib.list_size[LIST_QUEUE] == 0:

        simlib.list_remove(simlibdefs.FIRST, LIST_SERVER)

    else:
        simlib.list_remove(simlibdefs.FIRST, LIST_QUEUE)
        simlib.sampst(simlib.sim_time - simlib.transfer[1], SAMPST_DELAYS)
        num_custs_delayed += 1
        simlib.event_schedule(simlib.sim_time + simlib.expon(mean_service, STREAM_SERVICE), EVENT_DEPARTURE)
```

شکل ۳: کد ۲

۳ نتایج

ـدر انتها نیز نتایج شبیه سازی را نمایش میدهیم که کاملا با نتایج شبیه سازی کتاب مرجع مطابقت دارد.

Delays in queue, in minutes:

variable number	sampst Number of Average values	Maximum	Minimum
1	0.524871	1000.00	5.63310	0.00000

Queue length (1) and server utilization (2):

File number	Time average	Maximum	Minimum
1	0.540076	8.00000	0.00000
2	0.510693	1.00000	0.00000

شکل ۴: نتایج

منابع

[1] De Smet R. Simulation modeling and analysis: Averill M. Law and W. David Kelton McGraw-Hill, Inc., New York, 1991, xxii+ 759 pages,£ 31.05, ISBN 0 07 036698 5.