



پروژه شبیه سازی کامپیوتری

دکتر بردیا صفایی

دانشکده مهندسی کامپیوتر

دانشگاه صنعتی شریف

نیم سال اول ۰۲-۰۱

امیرحسین باقری - ۹۸۱۰۵۶۲۱

محمد رضا مفیضی - ۹۸۱۰۶۰۵۹



۱ مقدمه

در این گزارش نحوه پیاده‌سازی و نتایج شبیه‌سازی یک زمان‌بند CPU را شرح می‌دهیم.^۱ در این سیستم پردازش‌ها به‌صورت تسک‌هایی در نظر گرفته شده‌اند که بین صف‌های مختلف جابه‌جا می‌شوند. توضیحات پیاده‌سازی در ادامه آمده است.

۲ پیاده‌سازی

۱.۲ کلاس Task

کلاس Task به عنوان پردازش‌هایی که در صف‌ها قرار می‌گیرند تا به CPU برسند طراحی شده است. نمونه‌های این کلاس مقادیر X و Y و Z را برای تولید نرخ ورود (یا همان زمان $inter-arrival$)، زمان سرویس و اندازه زمان انقضا ورودی می‌گیرند. همچنین هنگام ساخت به‌صورت تصادفی و با احتمال‌های گفته‌شده یک اولویت برای تسک ساخته‌شده در نظر گرفته می‌شود.

۲.۲ کلاس JobCreator

کلاس JobCreator همان‌طور که در دستور پروژه آمده است تسک‌های ورودی سیستم را با مشخصات گفته‌شده تولید می‌کند و آن‌ها را در Priority Queue لایه اول قرار می‌دهد.

۳.۲ صف‌های لایه دوم

برای لایه دوم صف‌های First Come First Serve و Round Robin به‌ترتیب در کلاس‌های FCFS و RR پیاده‌شده‌اند. صف‌های RR مقدار کوانتوم تایم را ورودی می‌گیرند. تابع `insert` همواره تسک‌های جدید را به اول لیست اضافه می‌کند تا تسک با اولویت بیشتر (یا تسکی که زودتر آمده) آخر لیست در دسترس باشد. تابع `pop_task` برای حذف تسک از لیست و `get_task` هم برای گرفتن اولین تسک از لیست استفاده می‌شود. توابع دیگری هم برای استفاده در صف‌ها پیاده‌سازی شده‌اند.

۴.۲ کلاس Processor

کلاس Processor برای شبیه‌سازی پردازنده طراحی شده است. این کلاس تعداد پردازش‌ها، زمان شبیه‌سازی، صف‌های لایه دوم، و بقیه پارامترها را به‌عنوان ورودی می‌گیرد. تابع `job_loader` هنگامی که مجموع تعداد تسک‌ها در صف‌ها از k کمتر می‌شود k تسک با بالاترین اولویت را به اولین صف لایه دوم منتقل می‌کند.

فرایند اصلی شبیه‌سازی در تابع `dispatcher` انجام می‌شود. در این متد یکی از صف‌های لایه دوم به‌صورت تصادفی با احتمال داده‌شده انتخاب می‌شود. سپس باتوجه به سیاست صف یک تسک انتخاب می‌شود و در پردازنده اجرا می‌شود. فرایند اجرا شدن یک تسک در تابع `process` انجام می‌شود. به این صورت که واحد زمانی یک واحد به جلو برده می‌شود. سپس بررسی می‌شود که اگر تسک به‌اندازه کوانتوم زمانی در صف فعلی سپری کرده است به صف بعدی منتقل شود. همچنین اگر زمان سرویس آن به پایان رسیده است از صف حذف می‌شود.

۳ نتایج

شبیه‌سازی را به‌ازای ورودی‌ها و پارامترهای مختلف انجام می‌دهیم. ورودی‌های در نظر گرفته‌شده به‌ترتیب به‌صورت زیر هستند:

^۱ کدهای پیاده‌شده و نتایج کامل شبیه‌سازی در فایل ژوپتری که همراه با این گزارش ضمیمه شده آمده است.



۱. X-8-Y-8-Z-32-N-100-L-800-K-10-T1-2-T2-4

۲. X-8-Y-4-Z-32-N-50-L-800-K-10-T1-2-T2-4

۳. X-8-Y-6-Z-16-N-100-L-800-K-10-T1-2-T2-4

۴. X-8-Y-8-Z-32-N-100-L-800-K-10-T1-6-T2-8

۱.۳ طول صف‌ها

نمودار طول صف‌های لایه اول و دوم در شکل ۱ آمده است. در ورودی اول میانگین RR-T1 برابر با 0.30، RR-T2 برابر با 2.03، FCFS برابر با 0.87 و PQ برابر با 0.12 است. در ورودی دوم میانگین RR-T1 برابر با 0.25، RR-T2 برابر با 0.33، FCFS برابر با 0.02 و PQ برابر با 0.02 است. در ورودی سوم میانگین RR-T1 برابر با 0.26، RR-T2 برابر با 0.67، FCFS برابر با 0.13 و PQ برابر با 0.06 است. در ورودی چهارم میانگین RR-T1 برابر با 0.98، RR-T2 برابر با 2.23، FCFS برابر با 0 و PQ برابر با 0.12 است.

۲.۳ میانگین زمان صرف‌شده در صف‌ها

نمودار زمان صرف‌شده در صف‌ها در شکل ۲ آمده است.

۳.۳ میزان بهره‌وری پردازنده

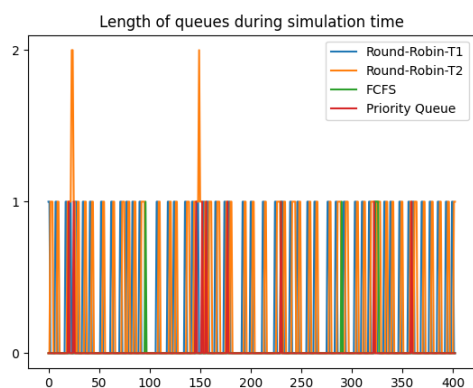
میزان بهره‌وری پردازنده در ورودی اول 100٪، در ورودی دوم 78.0٪، در ورودی سوم 80.2٪ و در ورودی چهارم 98.5٪ است.

۴.۳ بهبود میانگین زمان صرف‌شده در صف‌ها

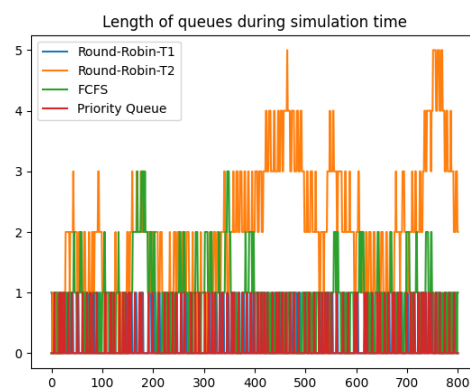
تودو

۵.۳ درصد پردازش‌های منقضی‌شده

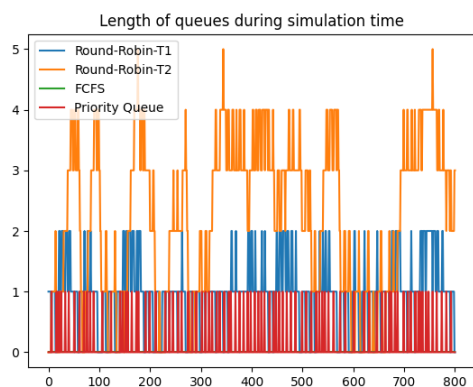
درصد پردازش‌های منقضی‌شده در ورودی اول 20.0٪، در ورودی دوم 0.0٪، در ورودی سوم 1.0٪ و در ورودی چهارم 28.0٪ است.



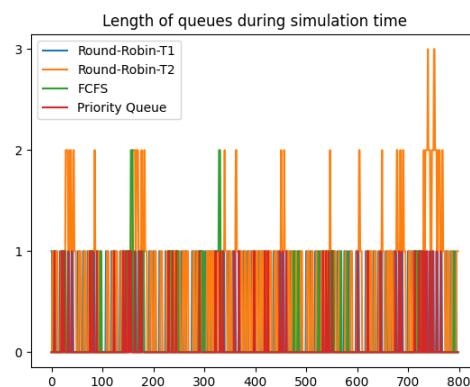
(ب) ورودی دوم



(آ) ورودی اول

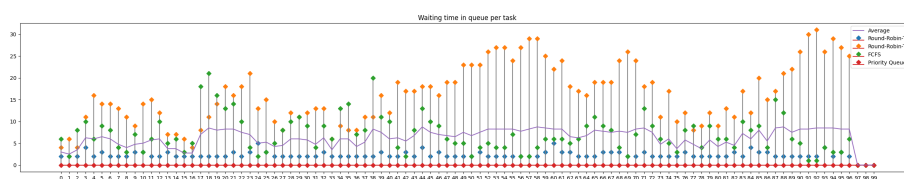


(د) ورودی چهارم

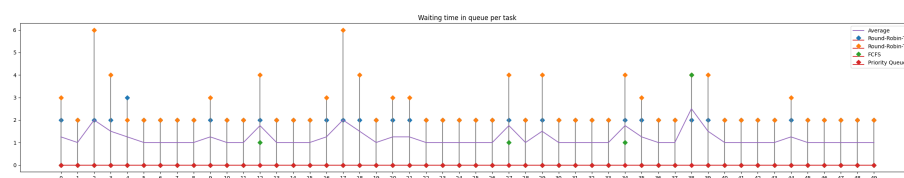


(ج) ورودی سوم

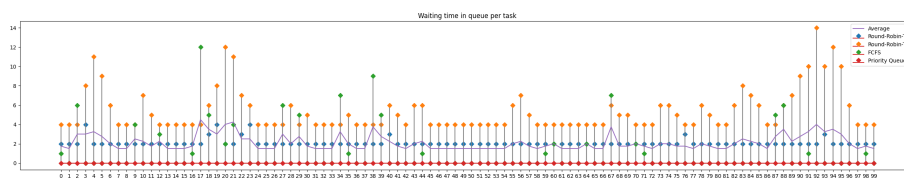
شکل ۱: طول صف‌های لایه اول و دوم به‌ازای ورودی‌های مختلف



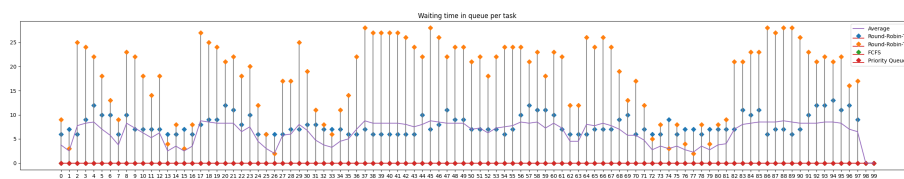
(آ) ورودی اول



(ب) ورودی دوم



(ج) ورودی سوم



(د) ورودی چهارم

شکل ۲: زمان صرف‌شده در صف‌های مختلف به‌ازای ورودی‌های مختلف