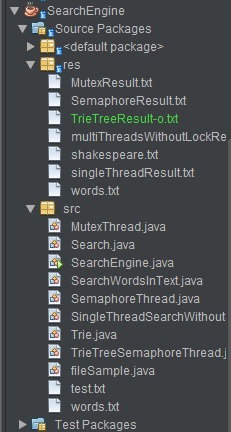
گزارش پروژه جستجو درس سیستم عامل

9712762670 محمد حسین حسینی

9722762473 محمد لکزایی

پروژه از دو بخش عمده src , res تشکیل شده است که در res فایل های متنی مورد نظر پروژه و خروجی آنها و در src کد های پروژه قرار دارند .



مطابق با خواسته صورت سوال این پروژه در 5 حالت زیر قابل اجراست :

1. single thread
2. multi thread without any lock
3. mutex lock multi thread
4. semaphore lock multi thread
5. trie tree using semaphore multi thread

برای هر یک از این موارد فایل خروجی مورد نظر در بخش res موجود است.

هر یک از متغیر های زیر بیانگر آدرس مشخص شده هستند :

textPath : مسیر فایل متنی ورودی ( در اینجا Shakespeare.txt )

keyWordsPath : مسیر فایل شامل کلماتی که باید در متن جستجو شوند.

resultPath : آدرس خروجی

برای اجرای هر یک از 5 مورد نامبرده کافیست بخش مربوطه را uncomment نماییم ؛ (نام خروجی متناسب با الگوریتم uncomment شده تغییر میکند)

خروجی کامل هر یک از الگوریتم ها در ./res/ آمده است در انتهای هریک زمان اجرای کامل در خروجی نوشته شده است :

|  |  |
| --- | --- |
| 172ms | singleThreadResult.txt |
| 139ms | multiThreadsWithoutLockResult.txt |
| 175ms | MutexResult.txt |
| 180ms | SemaphoreResult.txt |
| 192ms | TrieTreeResult-o.txt |

با بررسی داده ها میتوان موارد زیر را نتیجه گرفت :

* که از 1 ترد به چند ترد سرعت اجرا افزایش می یابد امّا خطر race condition وجود خواهد داشت.
* با کمک lock های mutex , semaphore خطر race condition برطرف می شود امّا اندکی سربار در زمان اجرا به همراه دارد.
* mutex lock اندکی عملکرد بهتری نسبت به semaphore lock دارد. (در اینجا semaphore = 1 قرار دادیم امّا اگر در پروژه ای بخواهیم تعداد همزمان افزایش یابد تنها semaphore این قابلیت را داراست و از mutex lock نمیتوانیم استفاده کنیم
* TrieTree بر خلاف انتظار در اینجا زمانی طولانی تر صرف نمود و می توان دلیل آن را اینگونه توضیح داد :

1. به این خاطر که ما در خروجی نیاز داشتیم خط پیدا شده را مشخص کنیم فلذا ناچارا باید به پردازش خط به خط متن می پرداختیم و امکان بررسی چند خط در یک لحظه را نداریم ؛ این الگوریتم با افزایش تعداد کلمات نتایج بهتری به ارمغان خواهد آورد در حالی که در Shakespeare.txt (ورودی داده شده) هر خط تنها شامل چند کلمه است و چک کردن تک تک کلمات سریعتر خواهد بود نسبت به اینکه این کلمات را در درخت مورد نظر insert کنیم و سپس کلماتی که داریم را در آن search کنیم.

با این وجود عملکرد TrieTree از آنچه انتظار داشته ایم بهتر بوده است و انتظار داریم با دادن خروجی که خط های آن طولانی تر باشد و یا امکان این را داشته باشیم که چندین نخ را همزمان بررسی نماییم ، خروجی بهتری دریافت خواهیم کرد.

با تشکر