ДОКЛАД

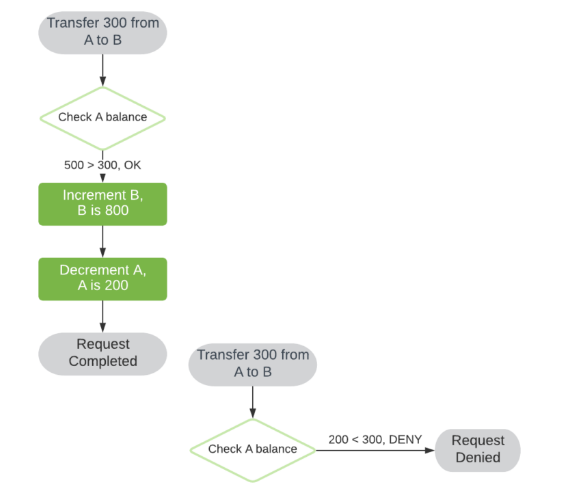
Работа с нишки - Race conditions

1. Race Condition:

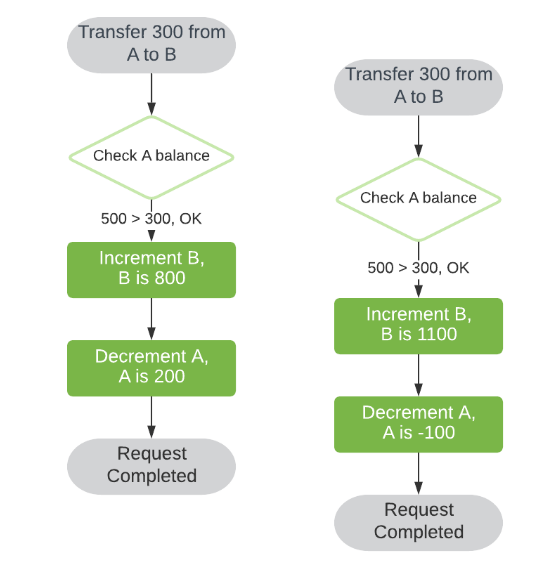
* По дефиниция Race Condition (условието за състезание) е състояние на програма, при което нейното поведение зависи от относителното синхронизиране или преплитане на множество нишки или процеси.
* Един или повече възможни резултати може да са нежелани, което да доведе до грешка. Този тип поведение е недетерминистично.
* Thread-safe е терминът, който използваме, за да опишем програма, код или структура от данни без условия на състезание, когато са достъпни от множество нишки.

ПРИМЕР:

Нека разгледаме проста функция за извършване на превод на средства между две банкови сметки:



Ако тези два опита бъдат стартирани едновременно, в различни процеси или нишки, може да наблюдаваме известно нежелано поведение:

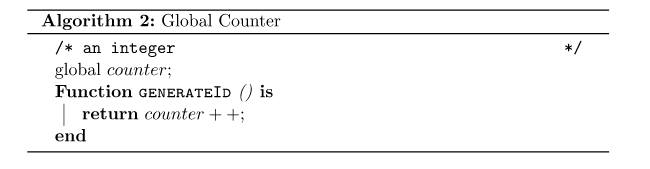


Като се има предвид непредсказуемо планиране на нишки, редът на конкретните стъпки е произволен. Попаднахме на състояние на състезание поради преплитането на нашите потоци за изпълнение.

За да се избегнат условия на състезание, всяка операция върху споделен ресурс – тоест върху ресурс, който може да бъде споделен между нишки – трябва да се изпълнява атомарно. Един от начините за постигане на атомарност е чрез използване на критични секции — взаимно изключващи се части на програмата. Друг подход е да се използват атомарни операции , за да се възползва от способността на хардуера да гарантира неделимостта.

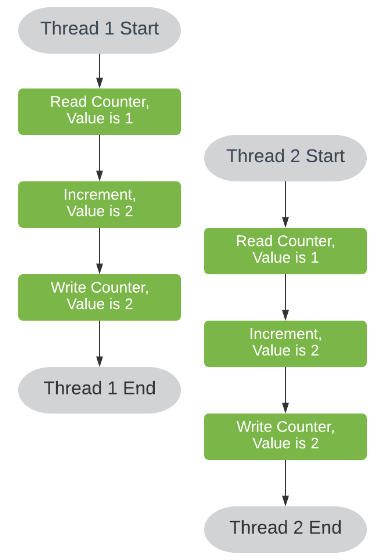
2.Четене-Промяна-Писане:

Въпреки че типът „проверете след това действайте“ условие за състезание е наистина най-често срещаният тип, който можем да срещнем в многонишкови приложения, има друг, по-лесен за разбиране тип. Помислете за следния псевдокод, който използва редовната операция за нарастване:



В повечето езици редовният оператор за нарастване представлява три последователни операции — четене, модифициране и запис.

Тъй като не сме посочили никаква атомарна гаранция за това изпълнение, ако бъде стартирано повече от едно изпълнение, може да получим същото вплитане на операции, което сме виждали преди:



Този тип състояние е тясно свързано с надпреварата за данни.

3.Елиминиране:

Има два вида подходи за борба с расовите условия:

* Избягване на споделено състояние;
* Използване на синхронизации и атомарни операции;

4. Използване на синхронизации и атомарни операции:

* Примитивите за синхронизиране, като критични секции, се използват, за да се гарантира, че определена част от програмата не може да бъде изпълнена от повече от една нишка едновременно. Заключването е механизъм за синхронизиране за налагане на поведение на критична секция на ниво нишка.
* Заключванията ни носят хит в производителността поради техните допълнителни разходи. Те също може да са трудни за справяне. Заключванията не композират, което означава, че изисква допълнителни усилия за запазване на коректността при комбиниране на базирани на заключване модули в по-голяма програма, базирана на заключване. И накрая, заключванията могат да доведат до блокиране.

Източници:

<https://www.baeldung.com/cs/race-conditions>

Изготвил:Валентин 11,,а‘‘ клас