Задача 12. Разбор алгоритма поликлиники для n процессов Детальное понимание и обоснование известного алгоритма поликлиники (см. учебник Эндрюс).

Исследование провёл студент группы 22207 Гордеев Никита

Дата выполнения работы 25.12.2022 (Вариант 3)

Задачи

- **Разбор алгоритма поликлиники для n процессов**
- **Детальное понимание и обоснование известного алгоритма поликлиники (см. учебник Эндрюс).**
- **Понимание сути алгоритма и почему он обеспечивает свойство справедливости для параллельных процессов, использующих критическую секцию.**

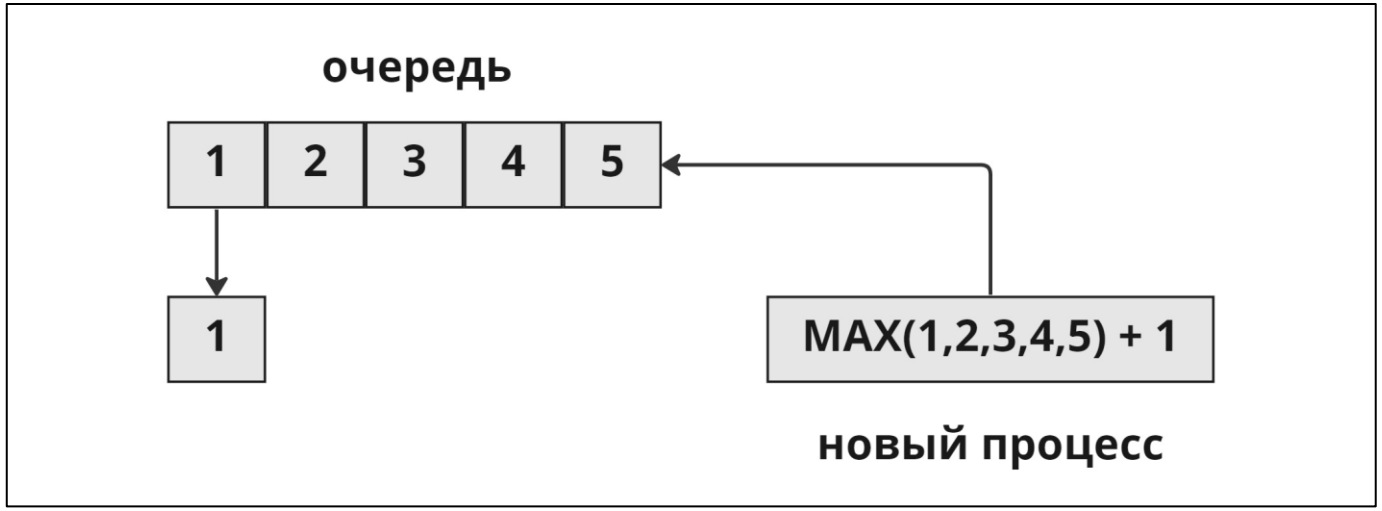
Цели

- **Понимание метода, лекция: гл.3.3.3**

Идея

Процессам присваиваются номерки по возрастанию. Процесс с минимальным номером заходит в КС, для нового процесса выбирается номерок на 1 больше максимального из имеющихся.

Схема работы №1



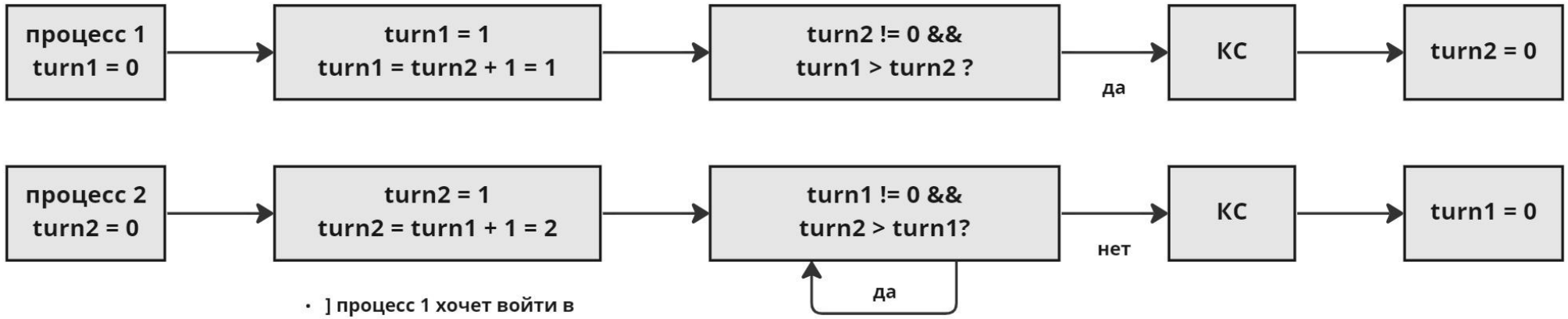
Проблема

Операция вычисления номерка для нового процесса не неделимая, поэтому не гарантируется, что у двух процессов не будет одинаковых номерков

Решение

Если у процессов одинаковые номерки, приоритет будем отдавать тому, у которого меньше номер процесса.

Схема работы №2



-] процесс 1 хочет войти в критическую секцию раньше. Чтобы избежать *состояния гонок*:
- процесс 1 — 1
- процесс 2 — 2

процесс 2 ждёт, когда условие входа будет соблюдено и входит в КС, когда turn1 меняется на 0

Алгоритма

CLINIC: (∀ i: 1 <= i <= n:
(CS[i] *в своей критической секции*) ⇒ (turn[i] > 0) ∧
(turn[i] > 0) ⇒ (∀ j: 1<= j <= n, j != i:
turn[j] == 0 ∨ turn[i] < turn[j]))

```
int turn[1:n] = ([n] 0);
## глобальный инвариант – предикат CLINIC (см. текст)

process CS[i = 1 to n] {
  while (true) {
    ⟨turn[i] = max(turn[1:n]) + 1;⟩
    for [j = 1 to n st j != i]
      ⟨await (turn[j] == 0 or turn[i] < turn[j]);⟩
    критическая секция;
    turn[i] = 0;
    некритическая секция;
  }
}
```

Вывод:

- **Каждый процесс при данном алгоритме будет иметь уникальный номер. Поэтому не менее одного процесса будут иметь наименьший номер, который и будет выполняться.**
- **Таким образом, каждый раз обнуляя номер процесса, прошедшего критическую секцию, мы позволим каждому процессу пройти критическую секцию, тем самым обеспечив свойство справедливости.**

Материалы:

- **3.3.1 Алгоритм разрыва узла // Грегори Р. Эндрюс - Основы многопоточного, параллельного и распределенного программирования (дата обращения: 11.12.2022).**
- **Взаимное исключение. Задача критической секции // StudRef URL: https://studref.com/702397/informatika/vzaimnoe_isklyuchenie_zadacha_kriticheskoy_sektsii (дата обращения: 11.12.2022).**

Изменения

- **Версия 2**
  - **Изменил оформление слайдов со схемами**
- **Версия 3**
  - **Исправил цели и задачи**