



Нижегородский государственный университет
им. Н.И.Лобачевского

Факультет Вычислительной математики и кибернетики

Операционные системы: *аспекты параллелизма*

Планирование ЦП

Линёв А.В.

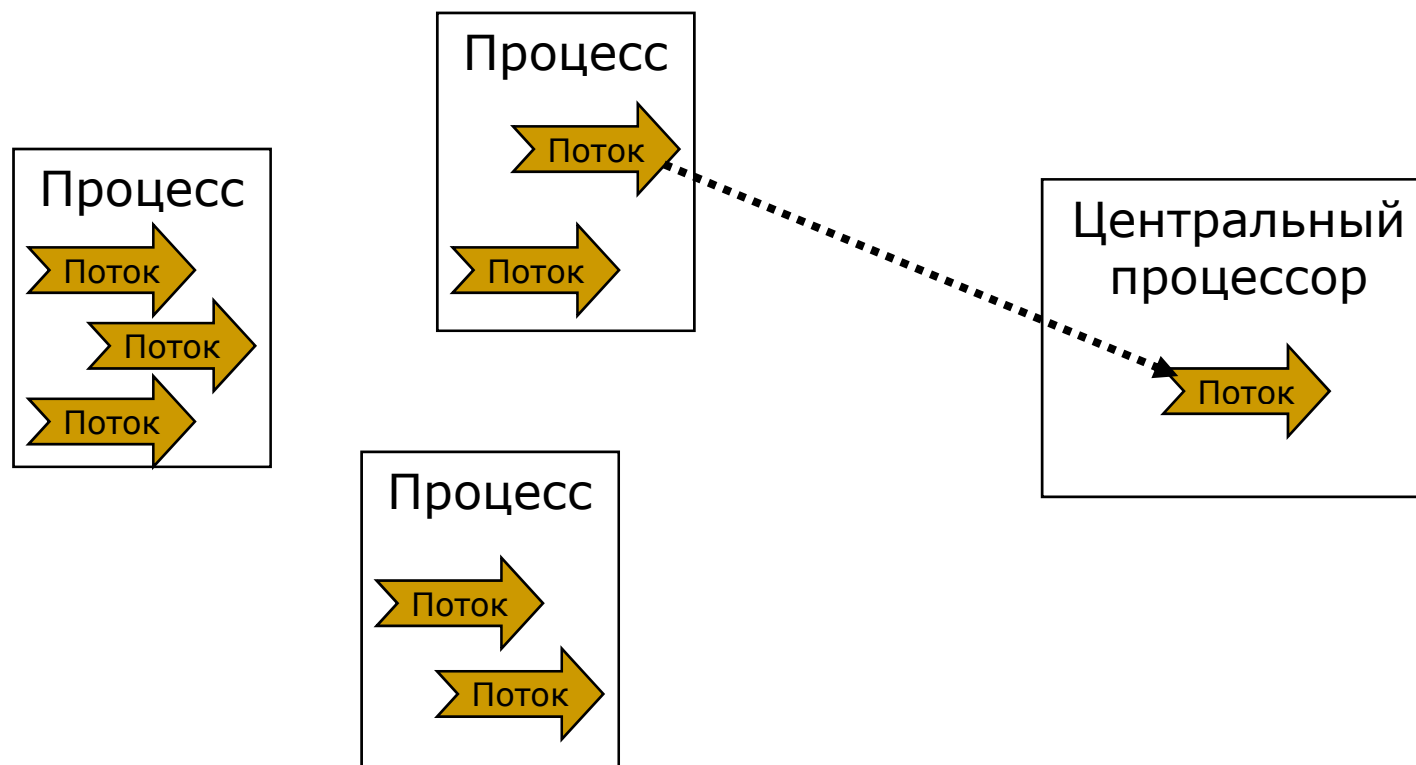
2007

Тема обсуждения

- Потокам для выполнения нужен центральный процессор (ЦП)
- Если потоков больше, чем процессоров, ОС должна обеспечить возможность поочередного использования процессоров потоками

Что такое "планирование"?

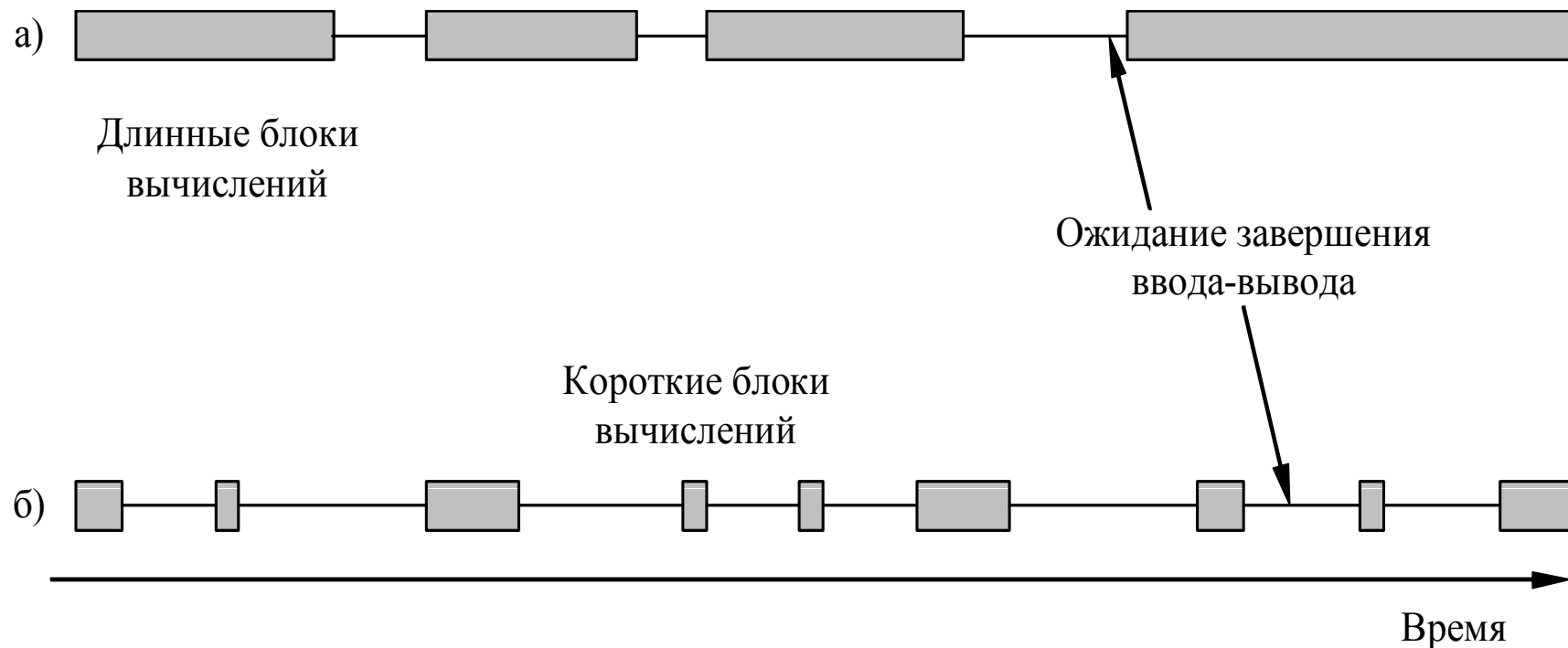
- Планирование – распределение времени центрального процессора



Что такое "планирование"?

- Планирование – распределение времени центрального процессора
- Планирование бывает
 - ❑ долгосрочное – принятие решений о запуске и откладывании запуска заданий
 - ❑ среднесрочное – принятие решений о временном вытеснении процессов на диск
 - ❑ краткосрочное (диспетчеризация) – выбор потока, которому следует предоставить ЦП
- При реализации планирования используется механизм переключения контекстов процессов и потоков

График выполнения потока

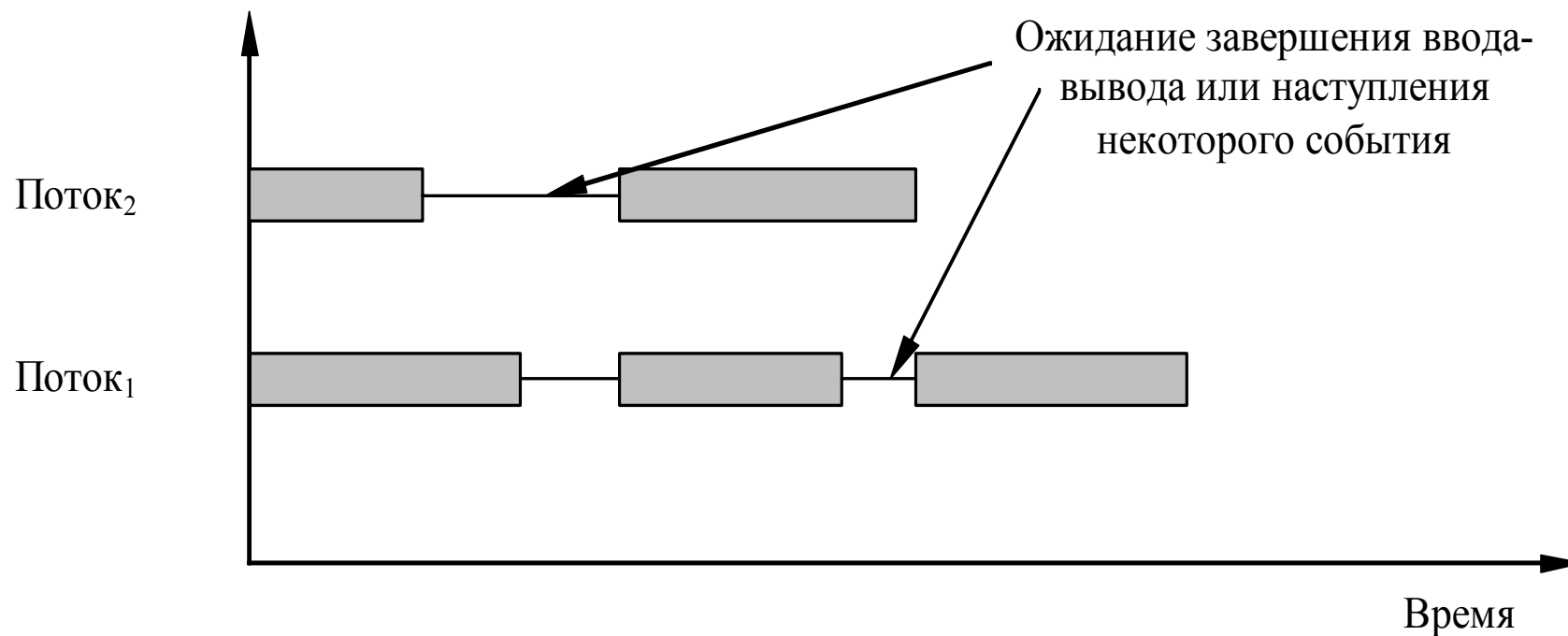


а) Потоки, ограниченные возможностями ЦП

б) Потоки, ограниченные возможностями ввода-вывода

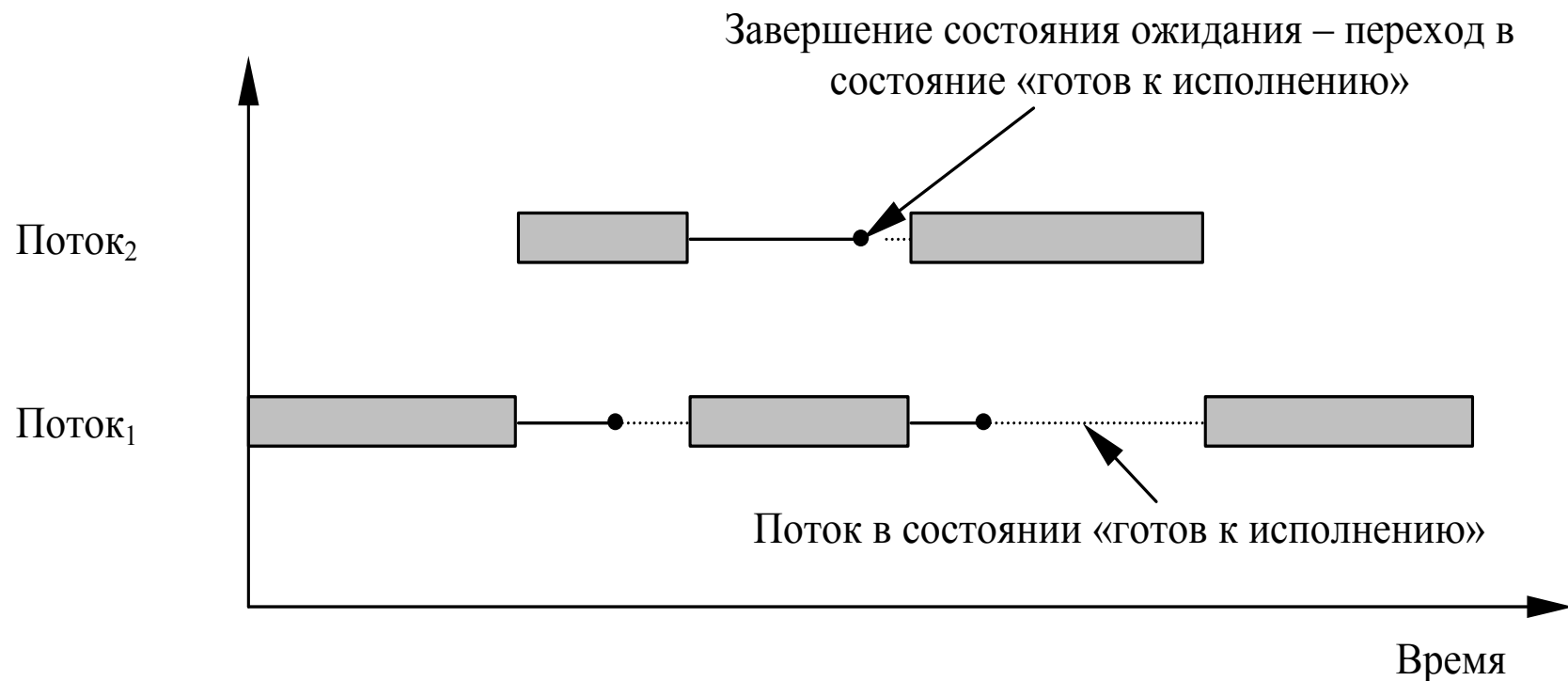
А если потоков несколько?

Для каждого потока можно построить график его выполнения в однозадачной среде



Если имеются 2 потока и 1 центральный процессор, операционная система должна определить последовательность использования центрального процессора потоками

А если потоков несколько?



Хорошо ли мы распределили время
ЦП между потоками?

Критерии оценки алгоритмов планирования...

■ Общие критерии

- ❑ **Справедливость** – всем выделяется равное количество времени ЦП
- ❑ **Баланс** – обеспечение загрузки всех устройств системы
- ❑ **Накладные расходы** – процент ресурсов, потребляемых планировщиком
- ❑ **Масштабируемость** – увеличение потребления ресурсов планировщиком при увеличении числа объектов планирования

Критерии оценки алгоритмов планирования...

- Справедливость – всем выделяется равное количество времени ЦП
 - Как измерить справедливость?
 - Одинаковое время потребления ЦП? Одинаковое количество выполненных команд?
 - Честно на уровне потоков? процессов? пользователей?
 - Что если один процесс ограничен возможностями ЦП, а другой – возможностями ввода-вывода?
 - Иногда нужно быть несправедливым и явно оказывать предпочтение некоторым определенным классам потоков/запросов/событий (иметь систему приоритетов), но...
 - следует избегать **голодания** (*starvation*) – ситуации, когда некоторым потокам процессор может не выделяться в течении неограниченного времени

Критерии оценки алгоритмов планирования...

■ Системы пакетной обработки данных

- ❑ **Пропускная способность** – среднее число заданий, полностью выполняемых в единицу времени (час, сутки,...)
- ❑ **Оборотное время** – среднее время между запуском задания и завершением его обработки
- ❑ **Эффективность** – средний процент загрузки центрального процессора

Критерии оценки алгоритмов планирования...

■ Интерактивные системы

- ❑ **Время отклика** – среднее время между возникновением события и завершением его обработки
- ❑ **Время ожидания** – среднее время между переходом потока из состояния "ожидание" в состояние "готов к выполнению" и предоставлением ему ЦП
- ❑ **Соразмерность** – соответствие производительности ожиданиям пользователя

Критерии оценки алгоритмов планирования

■ Системы реального времени

- ❑ Окончание обслуживания к сроку
- ❑ **Предсказуемость** – соблюдение временных ограничений при выполнении некоторого заранее оговоренного множества системных вызовов

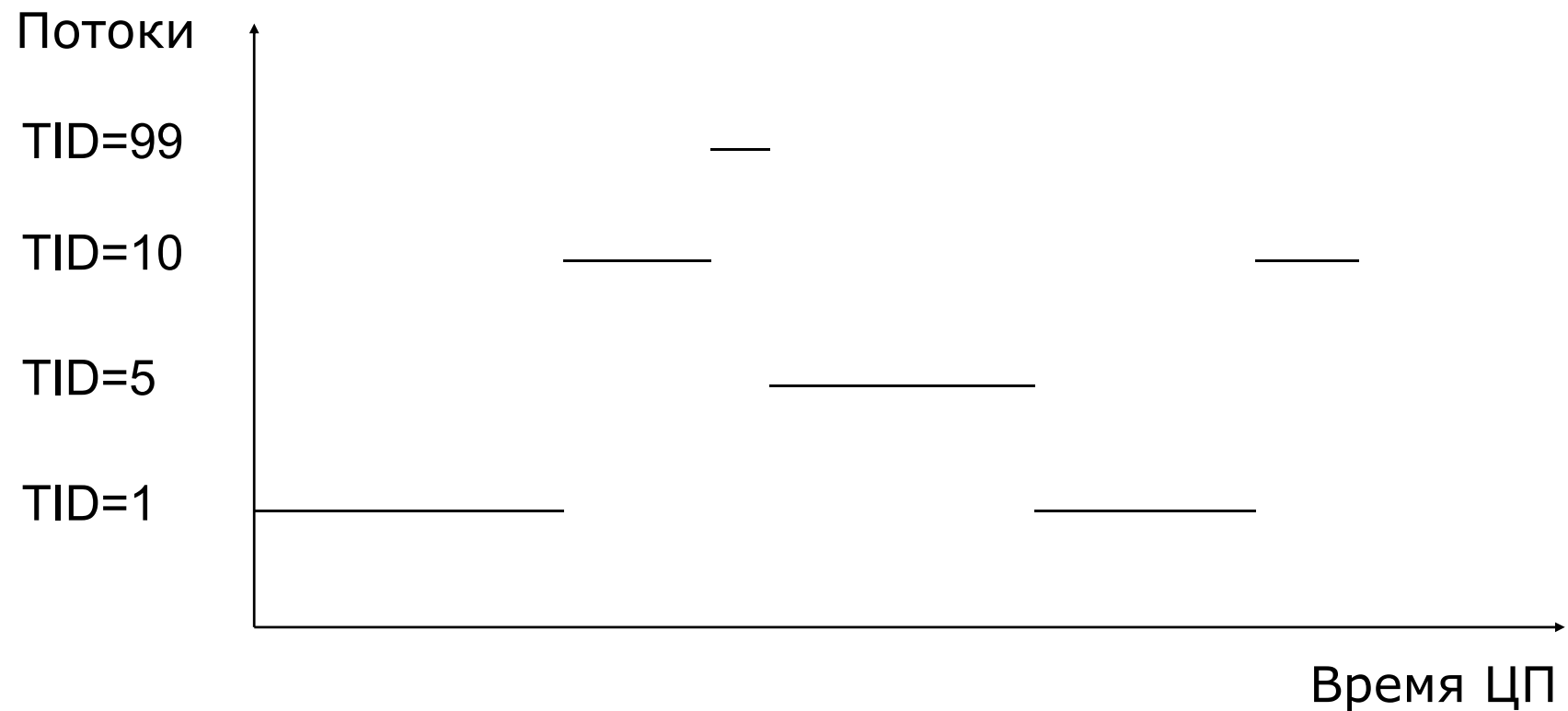
Моменты планирования

- Создание нового процесса
- Создание нового потока
- Завершение потока или процесса
- Блокирование потока (переход из состояния "выполнение" в состояние "ожидание")
- Разблокирование потока (переход из состояния "ожидание" в состояние "готов к выполнению")
- Срабатывание таймера

Невытесняющие алгоритмы планирования...

- Выбирают поток, передают ему центральный процессор и позволяют работать вплоть до блокировки либо до того момента, когда поток сам отдаст центральный процессор

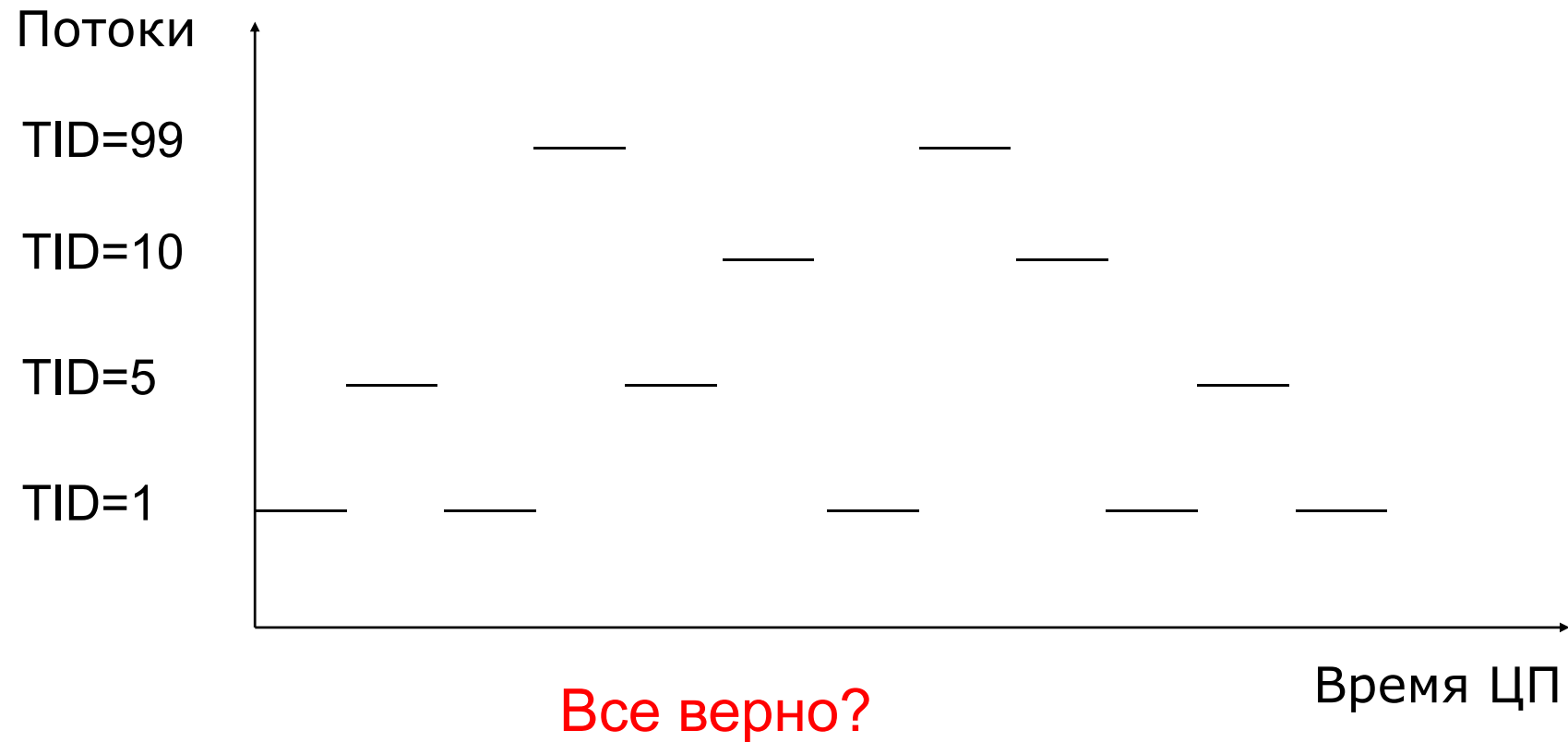
Невытесняющие алгоритмы планирования



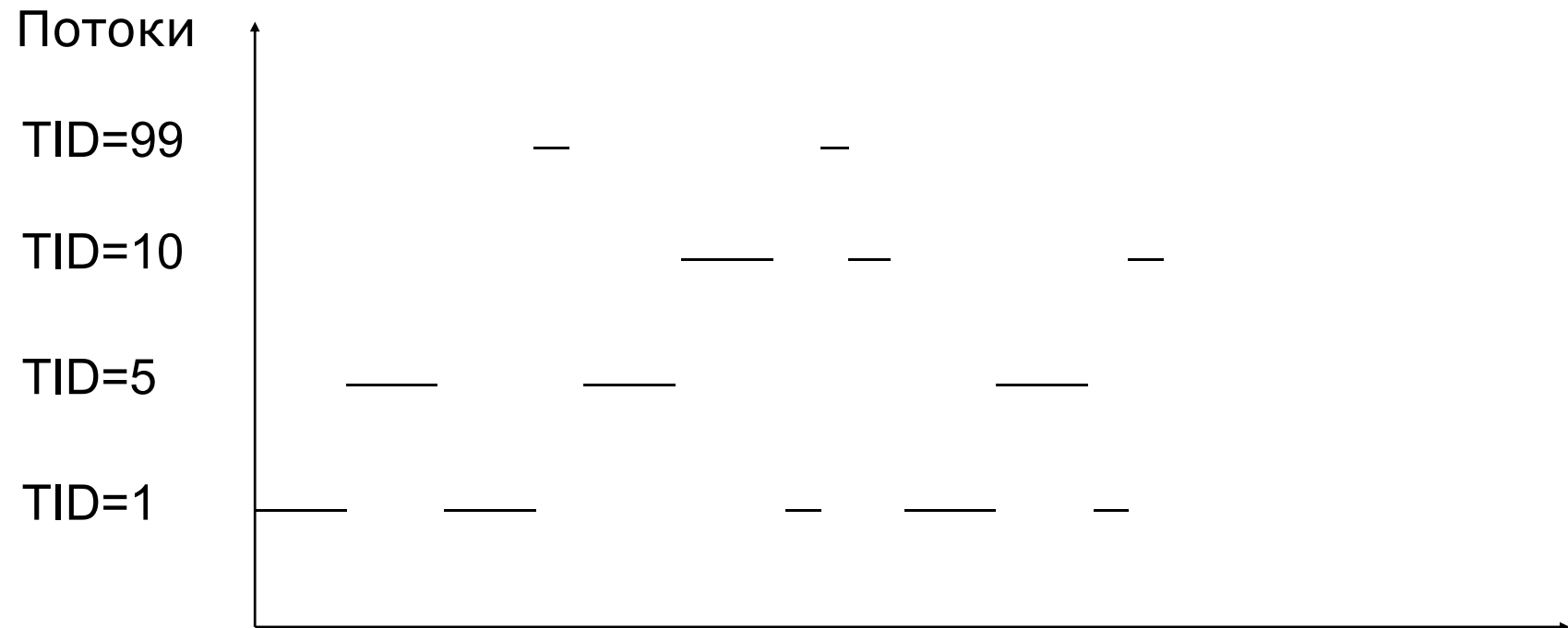
Вытесняющие алгоритмы планирования...

- Выбирают поток и позволяют ему работать некоторое максимально возможное фиксированное время. Если к концу заданного интервала времени поток все еще работает, он приостанавливается, и управление переходит к другому потоку (если существует поток, готовый к исполнению).
- Вытесняющее планирование требует прерываний по таймеру, происходящих в конце отведенного периода времени. При отсутствии таймера возможна реализация только невытесняющих алгоритмов.

Вытесняющие алгоритмы планирования...

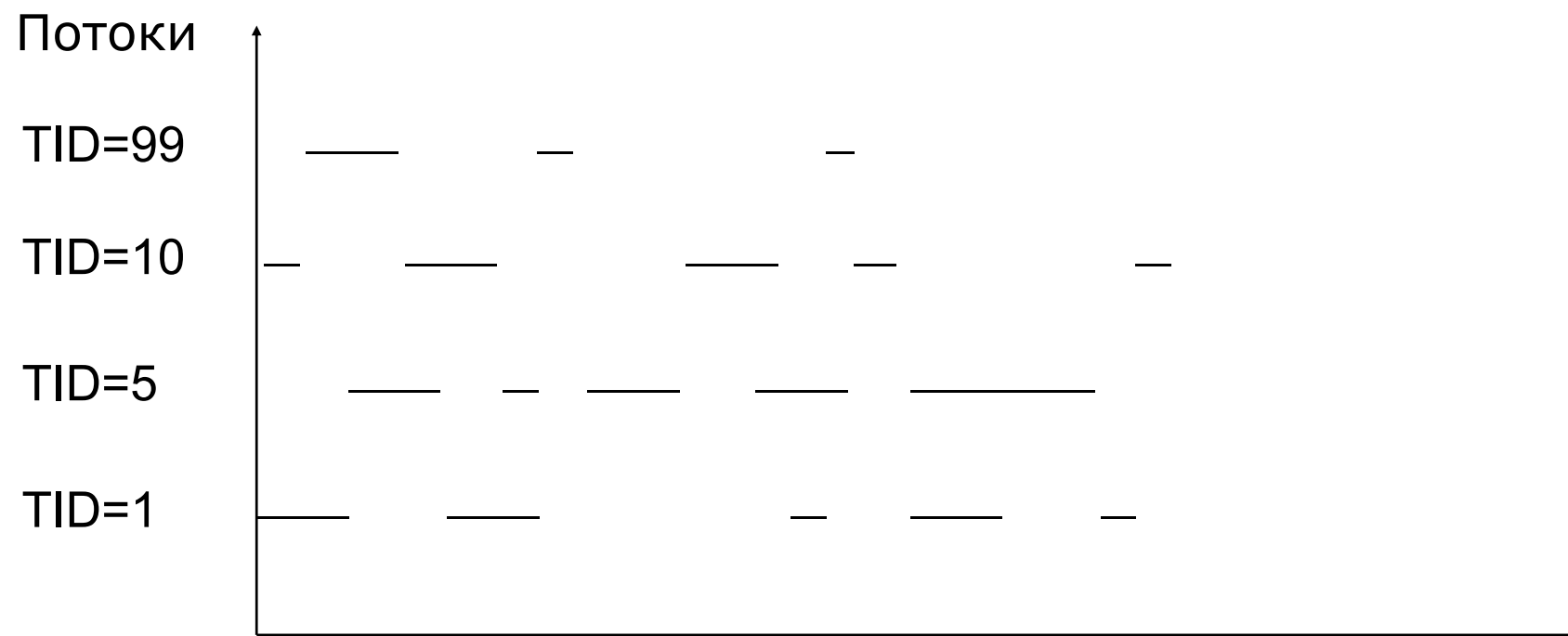


Вытесняющие алгоритмы планирования...



Часть потоков может не потреблять
каждый свой квант полностью

Вытесняющие алгоритмы планирования на многопроцессорных системах



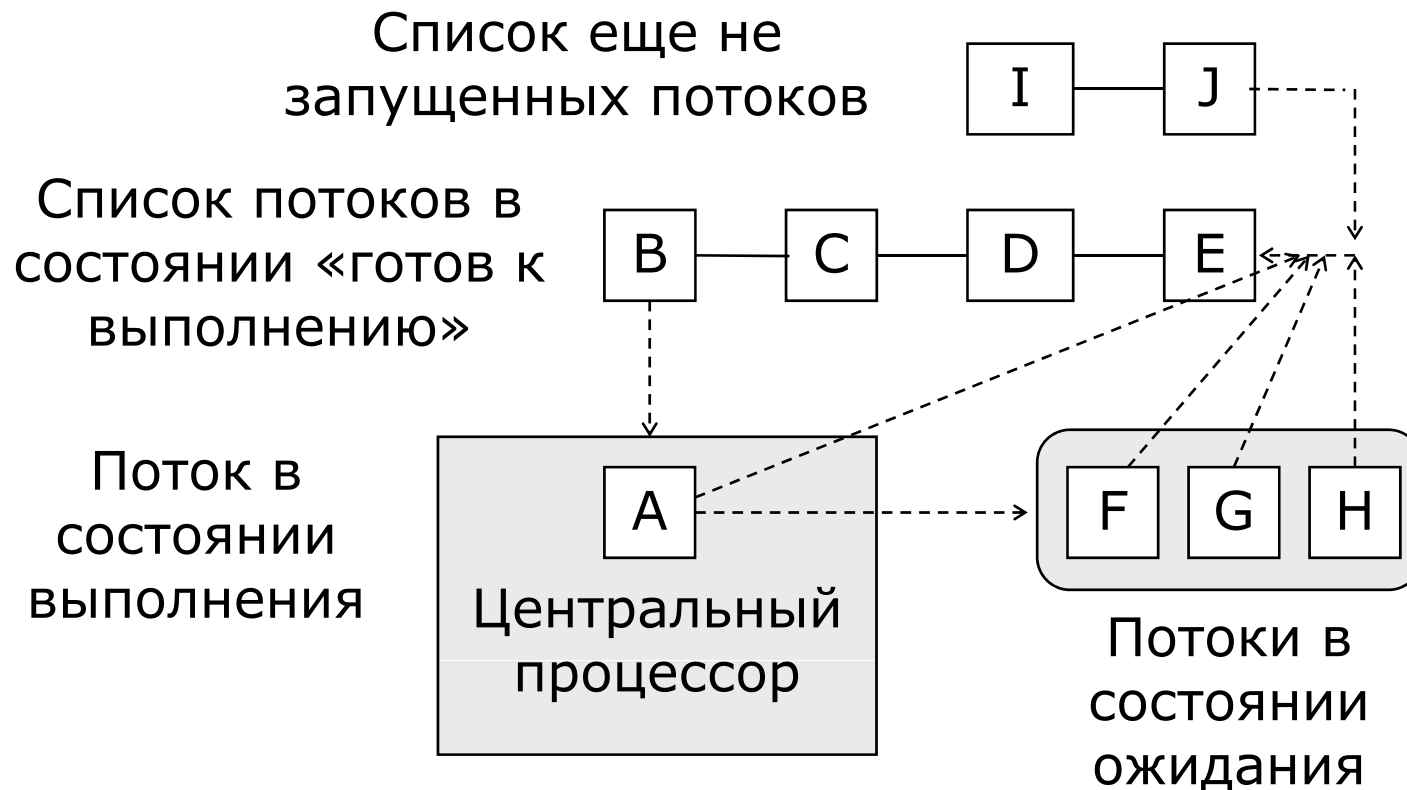
На двухпроцессорной системе
одновременно выполняются не более двух потоков



First Come – First Served (FCFS или FIFO)...

- Невытесняющий
- Планировщик поддерживает очередь готовых к выполнению потоков и предоставляет ЦП первому потоку из очереди
- При переходе в состояние "готов к выполнению" поток помещается в конец очереди

First Come – First Served (FCFS или FIFO)



Shortest Job First (SJF)

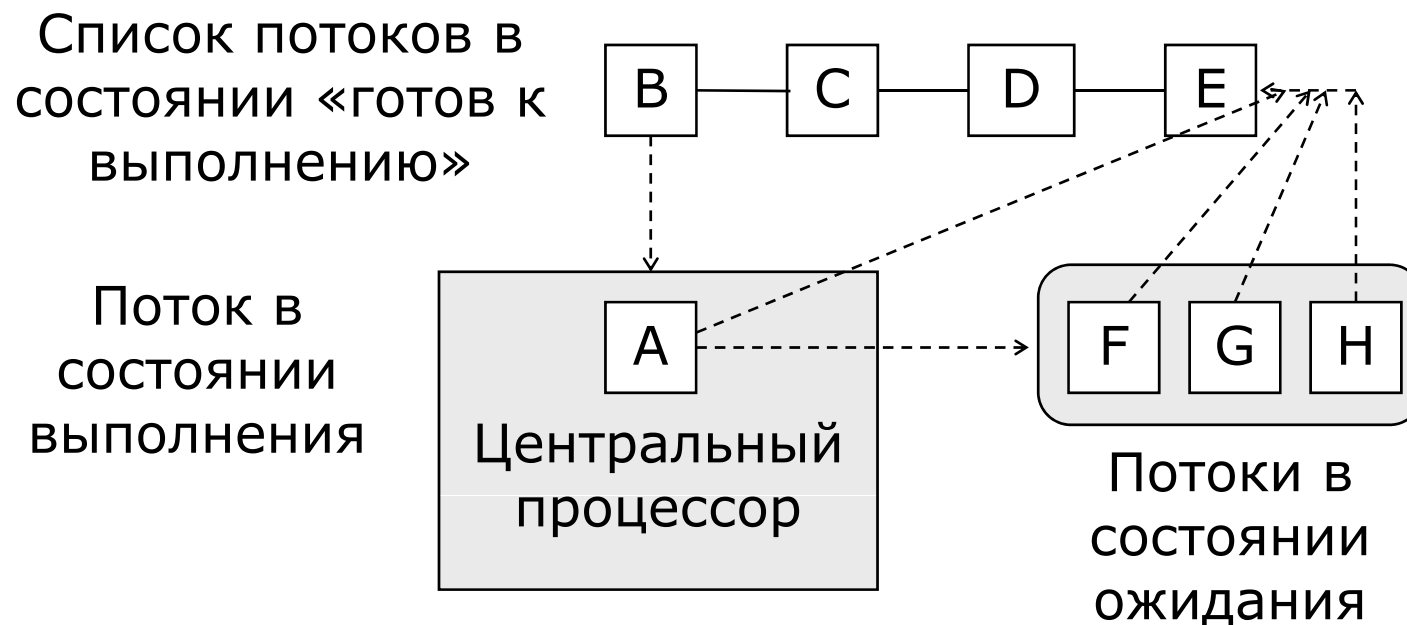
- Невытесняющий
- Планировщик выбирает на выполнение самую короткую задачу
 - Похож на FCFS, но для каждого потока вводится дополнительная характеристика – ожидаемое время выполнения, и потоки в очереди готовых к выполнению отсортированы в порядке увеличения ее значения
 - Автоматически определить время выполнения программы невозможно, следовательно, его должен задавать пользователь, запускающий программу

Shortest Remaining Time (SRT)

- Невытесняющий
- Планировщик выбирает на выполнение задачу, имеющую наименьшее планируемое время до окончания выполнения
 - У каждого потока 2 дополнительные характеристики – ожидаемое время выполнения и уже потребленное время ЦП

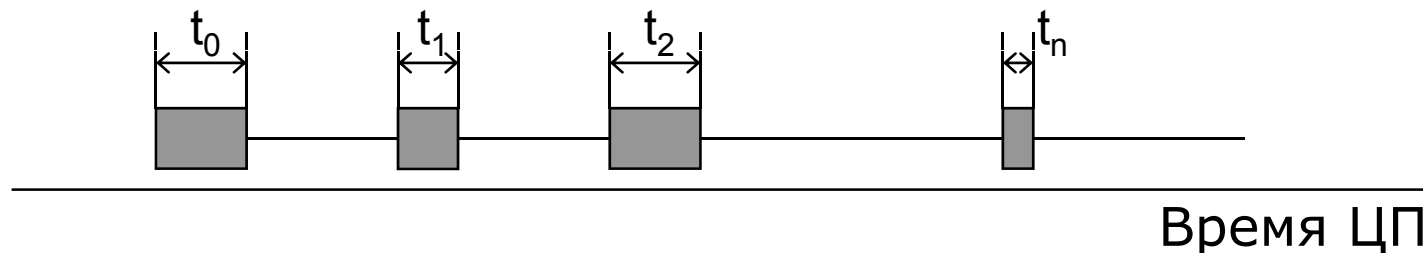
Round Robin (RR)

■ RR - вытесняющий аналог FCFS



Shortest Remaining Processing Time first (SRPT)

- Вытесняющий
- Считается, что каждый поток обрабатывает последовательность запросов
- Для каждого потока вычисляется характеристика, оценивающая время обработки одного запроса.
Например:
 $T(0)=0, T(n+1) = a*t(n) + (1-a)*T(n)$
 - $T(n)$ – значение характеристики после выполнения n -ого запроса
 - $t(n)$ – время обработки n -ого запроса
 - a – коэффициент ($0 \leq a \leq 1$)
- На выполнение выбирается поток с минимальным значением характеристики

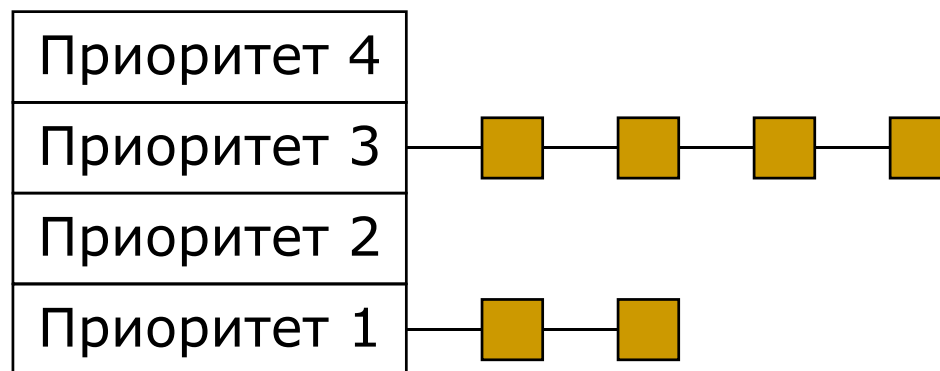


Приоритетное планирование

- Необходим дифференцированный подход к потокам
 - В системе могут одновременно существовать служебные процессы и процессы, запущенные пользователями
 - Пользователи могут различаться по статусу

Приоритетное планирование

очереди
приоритетов



самый высокий
приоритет

самый низкий
приоритет

Приоритетное планирование

- При использовании приоритетного планирования, если в системе всегда готов к выполнению хотя бы один из высокоприоритетных потоков, то потоки с низкими приоритетами никогда не будут выполнены. Поэтому, как правило, используются некоторые модификации этого алгоритма.

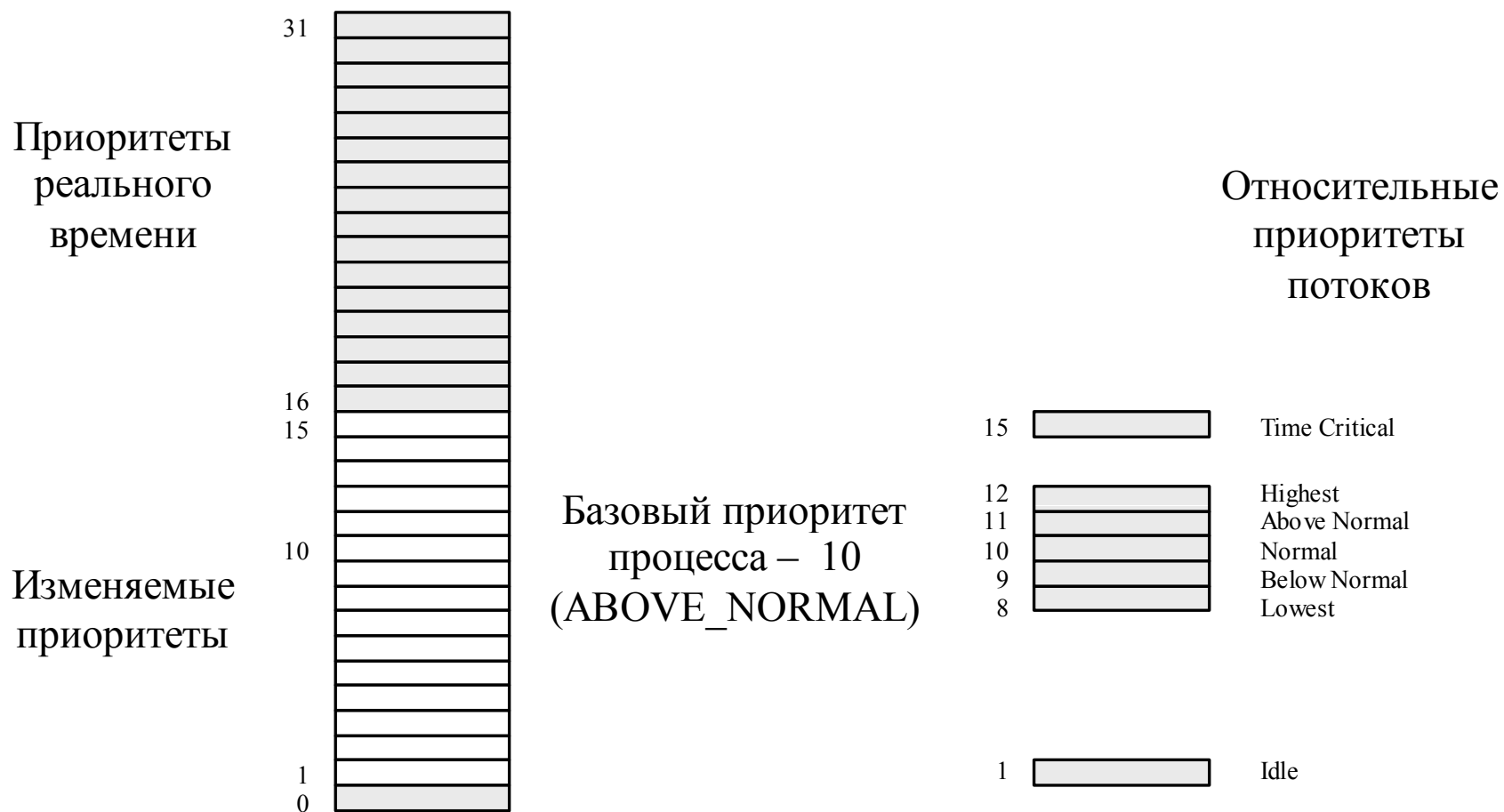
Приоритетное планирование...

- В модификациях часто разделяют базовый приоритет потока и эффективный приоритет потока
 - *Базовый приоритет* назначается потоку посредством некоторого системного вызова, например, инициированного выполнением команды пользователя
 - *Эффективный приоритет* используется планировщиком при выборе потоков на исполнение и может им изменяться в соответствии с реализованным алгоритмом планирования

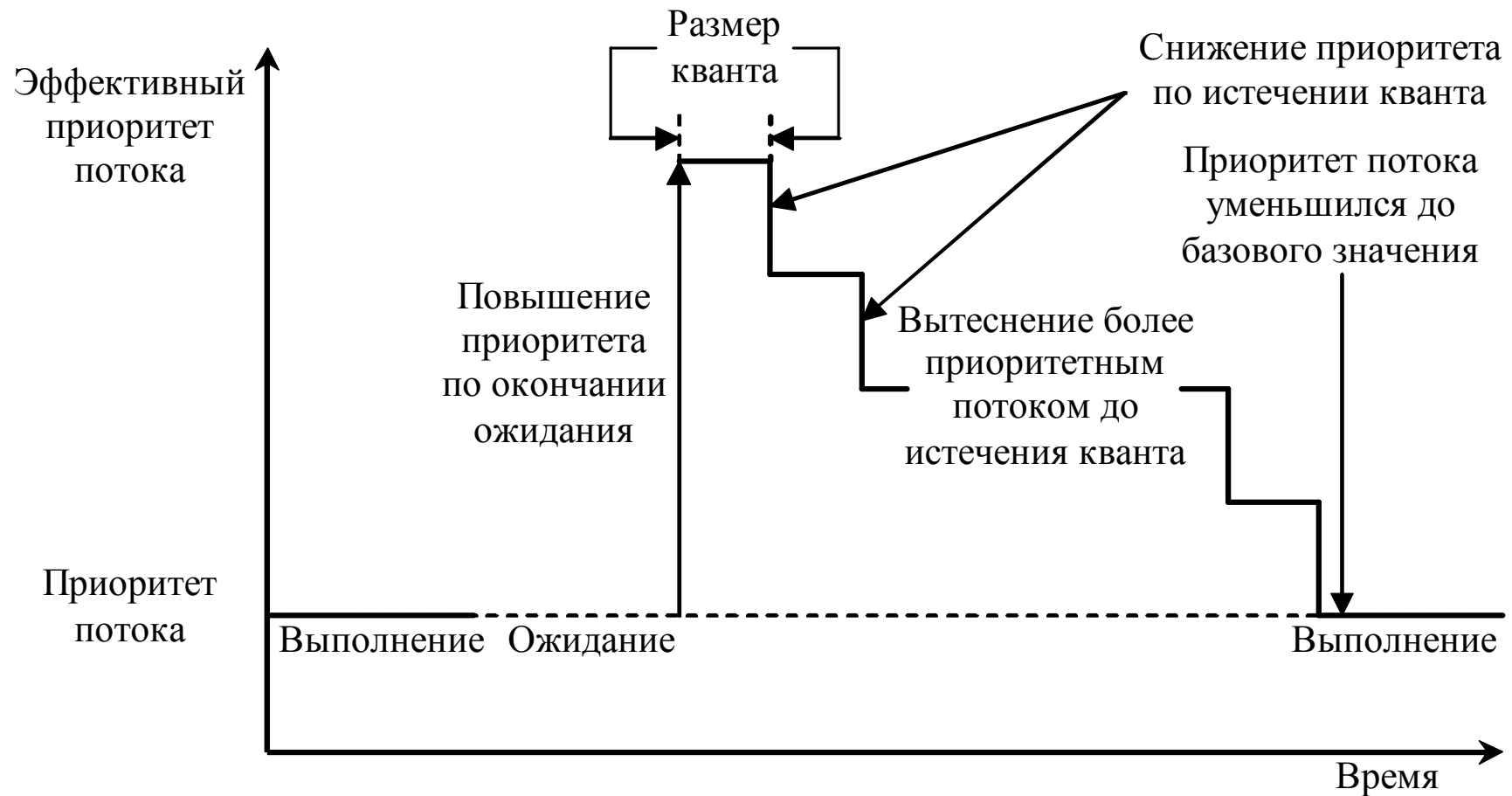
Приоритетное планирование

- Для предотвращения бесконечной работы высокоприоритетных потоков планировщик может уменьшать с каждым тактом таймера эффективный приоритет выполняющегося потока
- Для предоставления низкоприоритетным потокам центрального процессора планировщик может проверять время, в течение которого поток находился в состоянии готовности к исполнению
- Возможно адаптивное и динамическое изменение приоритетов
 - адаптивное изменение – коррекция эффективных приоритетов на основании набора правил
 - динамическое изменение – регулярный пересчет эффективных приоритетов на основании значений некоторой совокупности параметров

Планирование в Windows...



Планирование в Windows



Параметры пересчета приоритетов

- Каким пользователем запущен процесс или сформировано задание
- Насколько важной является поставленная задача, т. е. каков приоритет ее выполнения
- Сколько процессорного времени запрошено пользователем для решения задачи
- Каково соотношение процессорного времени и времени, необходимого для осуществления операций ввода-вывода
- Какие ресурсы вычислительной системы (оперативная память, устройства ввода-вывода, специальные библиотеки и системные программы и т. д.) и в каком количестве необходимы заданию
- Сколько времени прошло со времени выгрузки процесса на диск или его загрузки в оперативную память
- Сколько оперативной памяти занимает процесс
- Сколько времени ЦП было уже предоставлено потоку

Планирование с использованием многоуровневых очередей

планирование
внутри очереди

очереди
приоритетов

RR

Приоритет 4

RR

Приоритет 3

FCFS

Приоритет 2

RR

Приоритет 1

самый высокий
приоритет

самый низкий
приоритет



Multi-Level Feedback Queue (MLFQ)

При использовании многоуровневых очередей с обратной связью поток не постоянно приписан к определенной очереди, а может мигрировать из очереди в очередь в зависимости от своего поведения



Параметры алгоритма MLFQ

- Количество очередей для потоков, находящихся в состоянии готов к исполнению
- Алгоритм планирования, действующий между очередями
- Алгоритмы планирования, действующие внутри очередей
- Правила помещения родившегося потока в одну из очередей
- Правила перевода потоков из одной очереди в другую

Пример...

- Количество очередей: 4
- Алгоритм планирования, действующий между очередями: приоритетное планирование (приоритеты 1-4, наивысший приоритет - 4)
- Алгоритмы планирования в очередях
 - Пр.4. RR с размером кванта 8
 - Пр.3. RR с размером кванта 16
 - Пр.2. RR с размером кванта 32
 - Пр.1. FCFS
- Новый поток поступает в очередь с наивысшим приоритетом

Пример...

Создание
потока

→ Пр.4 – RR с квантом 8

Пр.3 – RR с квантом 16

Пр.2 – RR с квантом 32

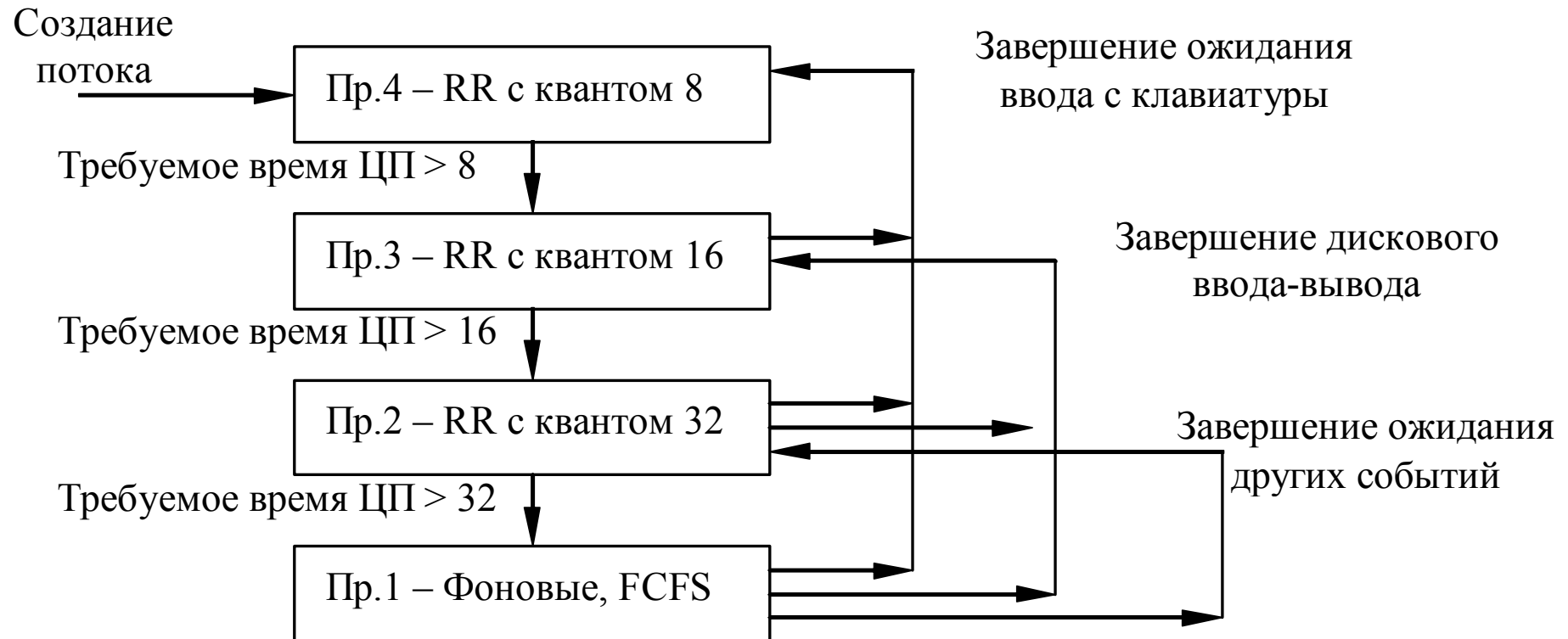
Пр.1 – Фоновые, FCFS



Пример...

- Правила перевода потоков из одной очереди в другую
 - если поток из очереди Пр.4. не завершил обработку до завершения кванта, он переводится в очередь Пр.3
 - если поток из очереди Пр.3. не завершил обработку до завершения кванта, он переводится в очередь Пр.2
 - если поток из очереди Пр.2. не завершил обработку до завершения кванта, он переводится в очередь Пр.1
 - после завершения ожидания ввода с клавиатуры потоки из очередей Пр.1,2,3 переводятся в очередь Пр.4
 - после завершения операций дискового ввода-вывода потоки из очередей Пр.1,2 переводятся в очередь Пр.3
 - после завершения ожидания всех других событий потоки из очереди Пр.1 переводятся в очередь Пр.2

Пример



Заключение

- Существует несколько уровней планирования
- Существуют различные алгоритмы планирования
- Определены критерии оценки алгоритмов планирования (зачастую противоположные)
- В реальных системах реализуются гибридные алгоритмы, использующие, к тому же, адаптивные или динамические приоритеты

Вопросы для обсуждения

- Какие критерии оценки алгоритмов планирования вы считаете более важными в настоящее время?
- Windows использует адаптивное планирование. Попробуйте предположить условия, при которых адаптивно изменяется приоритет потока.

Задание для самостоятельной работы

- Предложите свой алгоритм планирования
- Найдите описание проблемы инверсии приоритетов и изучите способы ее решения [1]
- Напишите программу, изменяющую статические составляющие приоритета своего потока и потока другого процесса (Win32 или UNIX)

Литература

1. Таненбаум Э. Современные операционные системы. 2-е изд. - СПб.: Питер, 2002.
2. Карпов В.Е., Коньков К.А. Введение в операционные системы. Курс лекций. 2-е изд. - М.: ИНТУИТ.РУ, 2005.

Тема следующей лекции – Синхронизация-1

- Необходимость синхронизации
- Задача взаимного исключения и ее решения

