iSCALARE

Лаборатория суперкомпьютерных технологий для биомедицины, фармакологии и малоразмерных структур

Параллельная симуляция
часть 1

Григорий Речистов
grigory.rechistov@phystech.edu

- Parallel Discrete Event Simulation
- Консервативные модели
- Оптимистические модели

На предыдущей лекции:

- Рассмотрены системы, состоящие из нескольких устройств
- DES: работа отдельных моделей чередуется, при этом сохраняется правильный* *порядок* событий

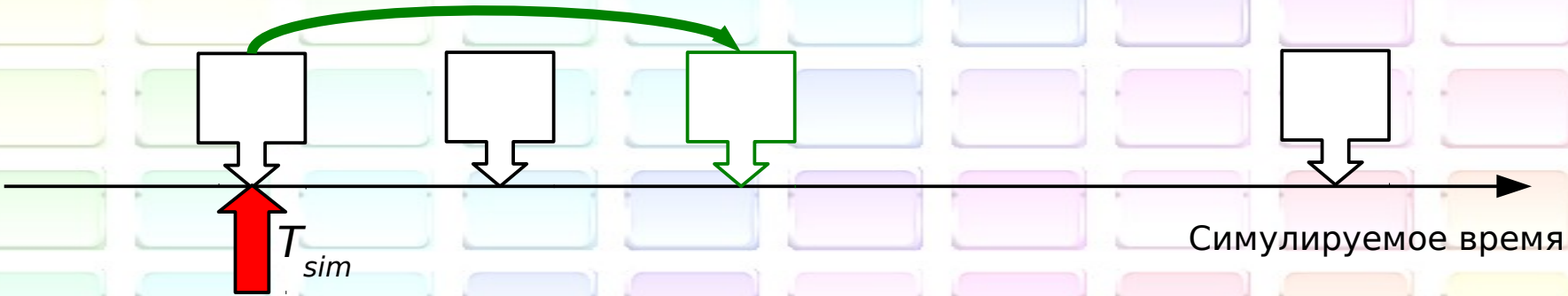
  * правильный ==

    1. совпадающий с наблюдаемым на реальной системе
    2. допустимый в реальности

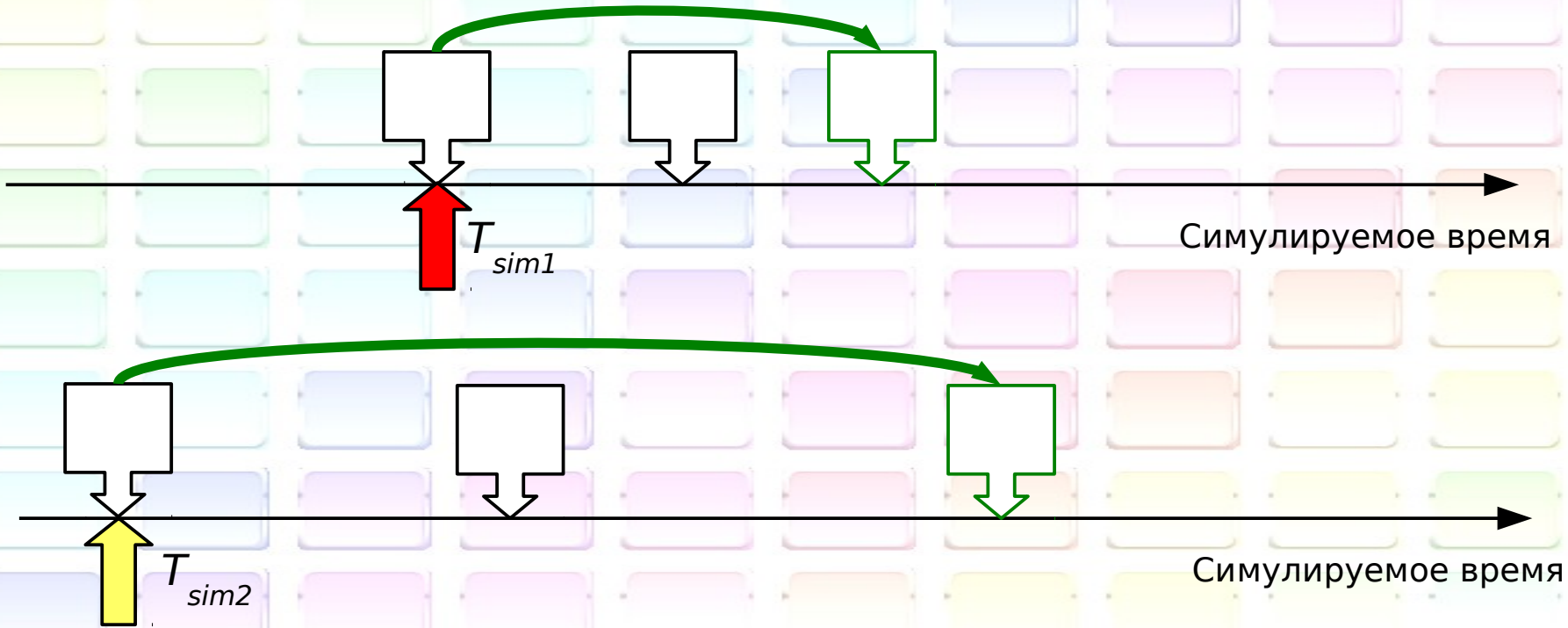
Параллельный хозяин

- Замедление моделирования N гостевых процессоров на одном хозяйском ядре: в лучшем случае 1/N
- В это время *K*-1 хозяйских ядер простаивает
- Возникает желание задействовать их в симуляции

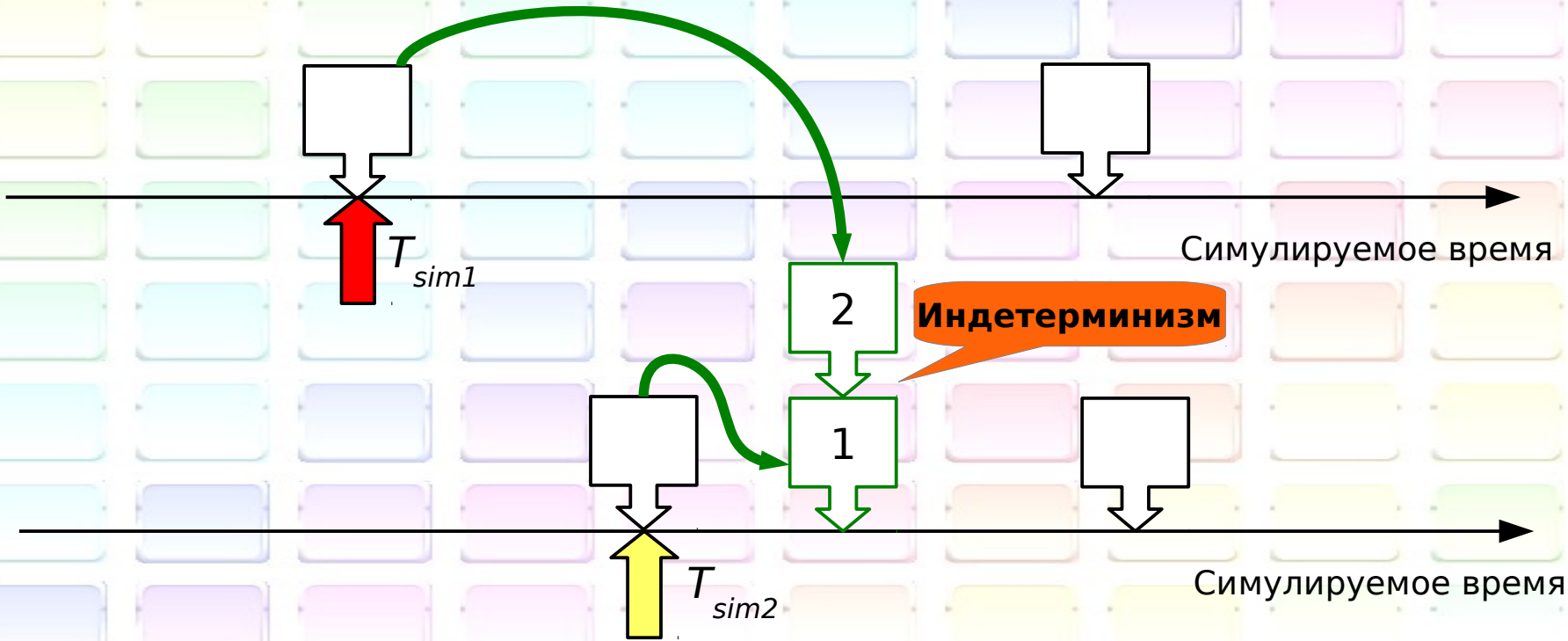
DES



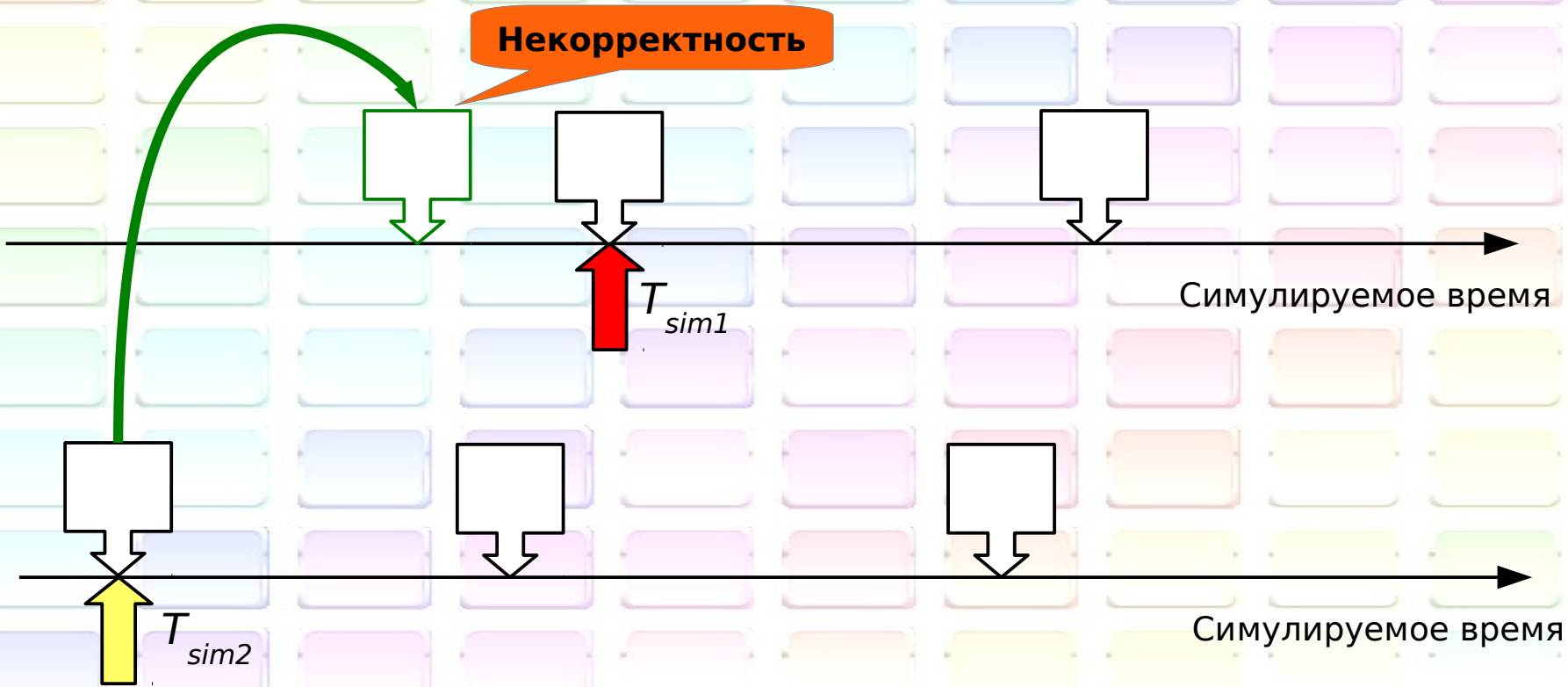
PDES



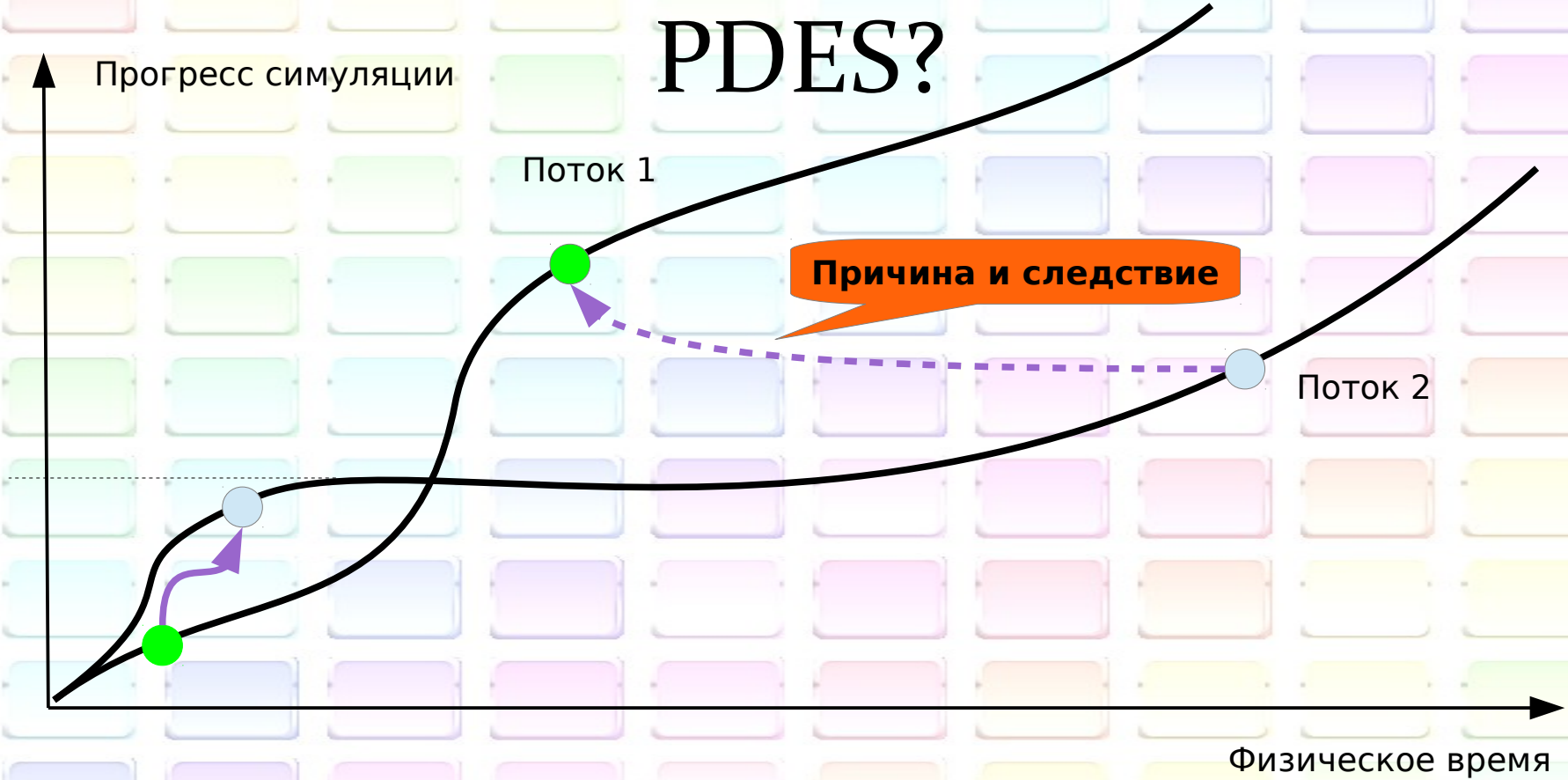
PDES?


T_{sim1}
Симулируемое время
2
Индетерминизм
1
T_{sim2}
Симулируемое время

PDES?


Некорректность
T_{sim1}
Симулируемое время
T_{sim2}
Симулируемое время

PDES?


Прогресс симуляции
Поток 1
Причина и следствие
Поток 2
Физическое время

Проблемы

- Нарушение каузальности (причинно-следственной связи)
- Недетерминизм модели

+

Все известные проблемы параллельного программирования

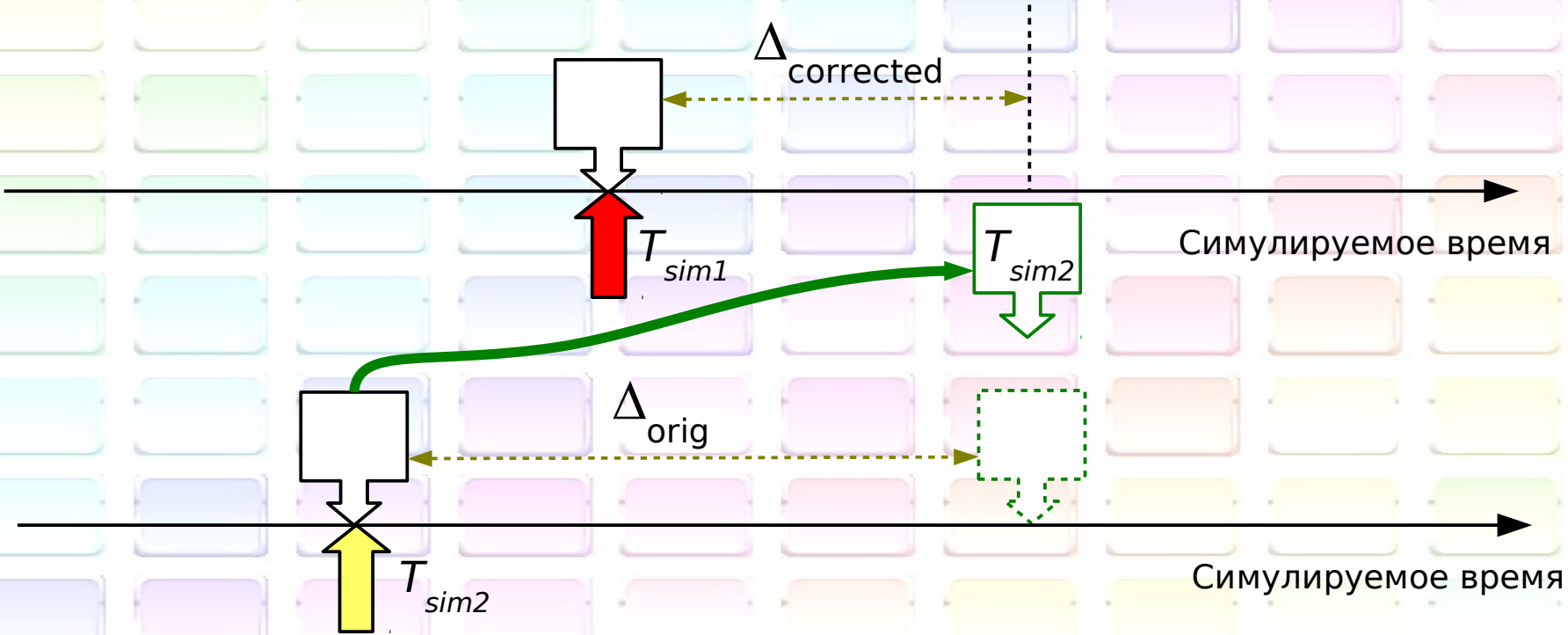
Взгляд назад

- Квотированное однопоточное моделирование MP систем: возможно нарушение порядка событий при quota > 1
  - Но: ошибка будет наблюдаться всегда => её можно отладить
  - Мы гарантированно достигаем корректности при quota → 1
- В наивном PDES мы не имеем этих возможностей

Как можно детектировать нарушения

- При пересылке сообщения добавлять к нему метку локального симулируемого времени
- По получении проверять метку и корректировать точку создания события
- При отрицательном значении корректировки — бить тревогу

Вот так:


Δ corrected
T sim1
T sim2
Симулируемое время
Δ orig
T sim2
Симулируемое время

Консервативно или оптимистично?

Теперь мы можем обнаруживать нарушения, но всё ещё не можем избавиться от их появления/последствий.

Необходимо:

1. Предотвращать их возникновение

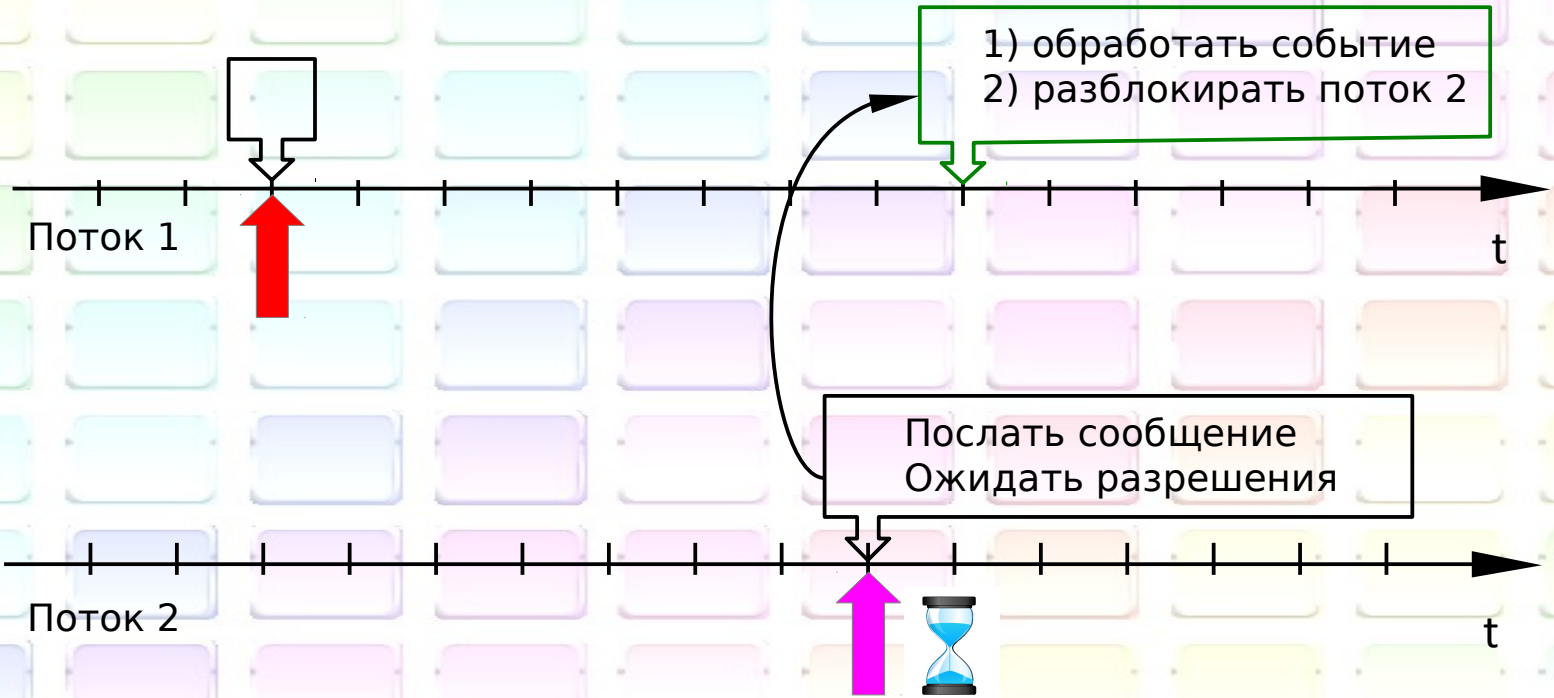
или

2. Подавлять их вредное проявление

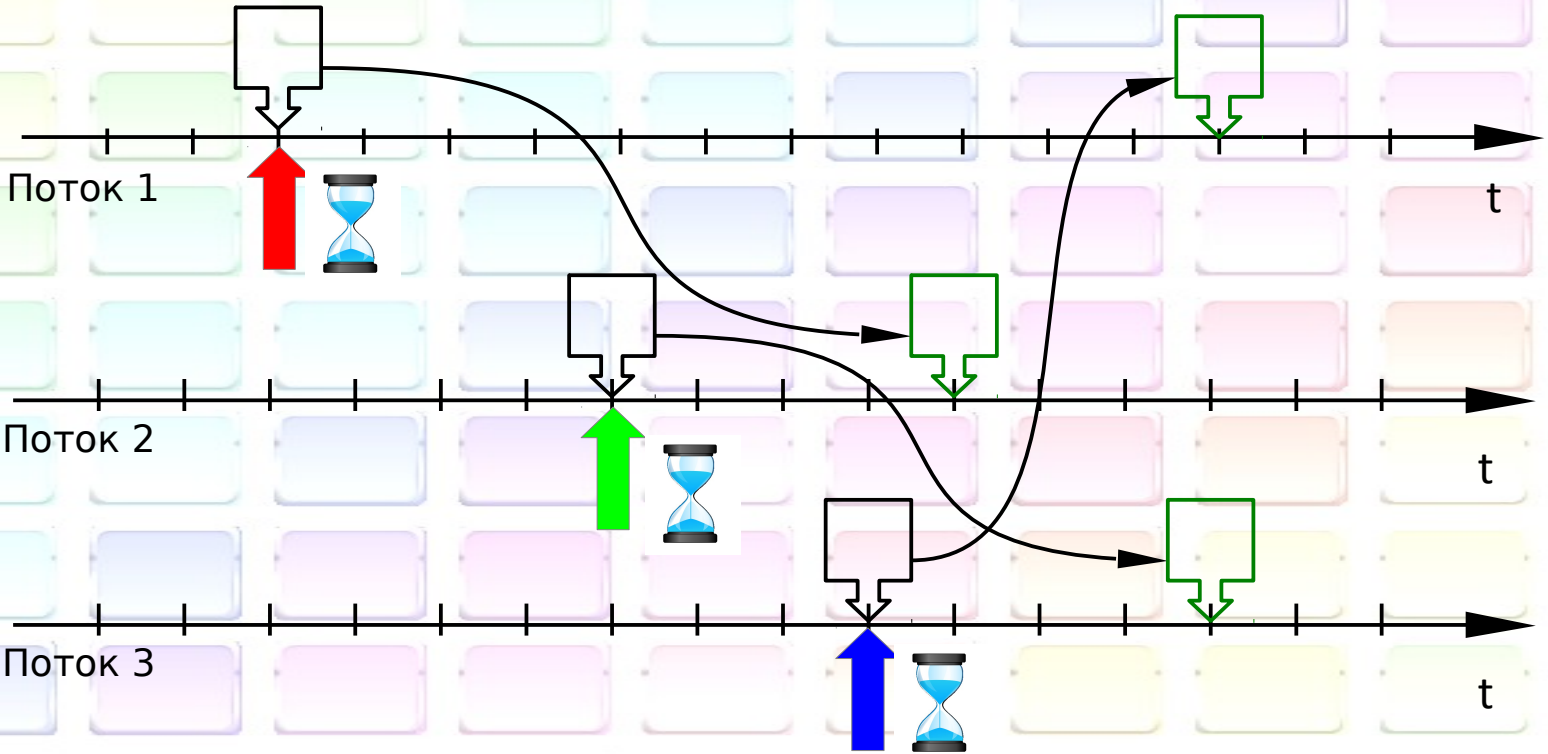
Консервативная схема 1

- При посылке сообщения блокировать отправителя до тех пор, пока получатель не обработает связанного события
- Не даём «быстрым» потокам продвигаться через этапы коммуникации
- Фактически вводим упорядоченность == последовательность при коммуникациях

Консервативная схема 2



Взаимоблокировка (дедлок)



Консервативная схема 3

- Необходимо детектировать ситуацию дедлока и разрушать его, освобождая один поток
- Лучший выбор — очередь с наименьшим значением симулируемого времени
- Система может оказаться в ситуации, когда большую часть времени почти все потоки стоят => выигрыша в скорости нет

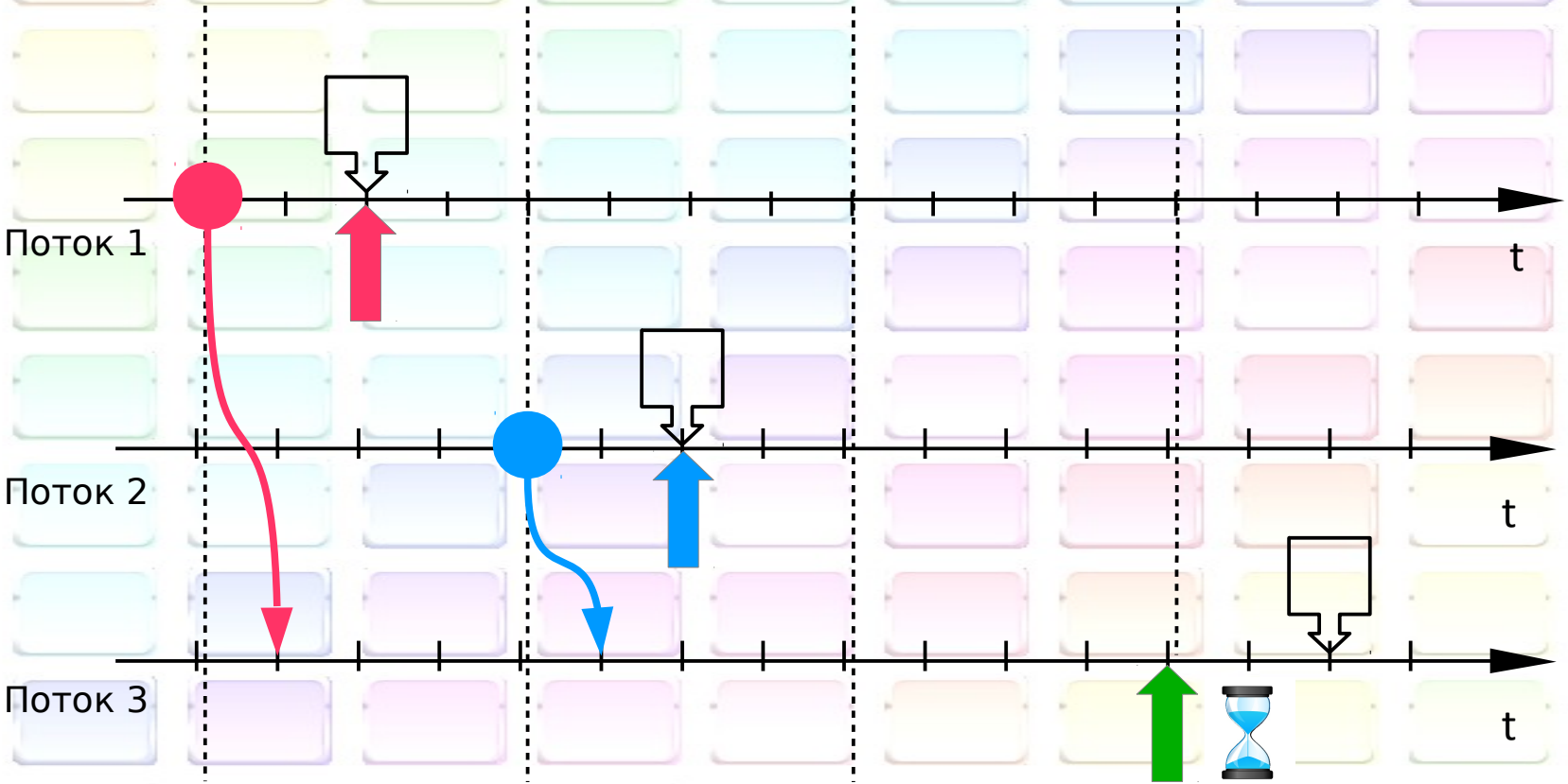
Пустые сообщения 1

- Можно ли избежать блокировок?
- Необходимость в них возникает из-за того, что отдельные потоки не знают, в какой стадии находятся остальные
- Как поток А может узнать локальное время потока В? Через временную метку, хранящуюся в каждом событии

Пустые сообщения 2

- Периодическая рассылка пустых (null) сообщений, не связанных с архитектурными событиями, но несущими метку времени
- Теперь каждый поток имеет представление о том, не слишком ли он далеко убежал в будущее, и может **сам** притормаживать/блокировать своё исполнение

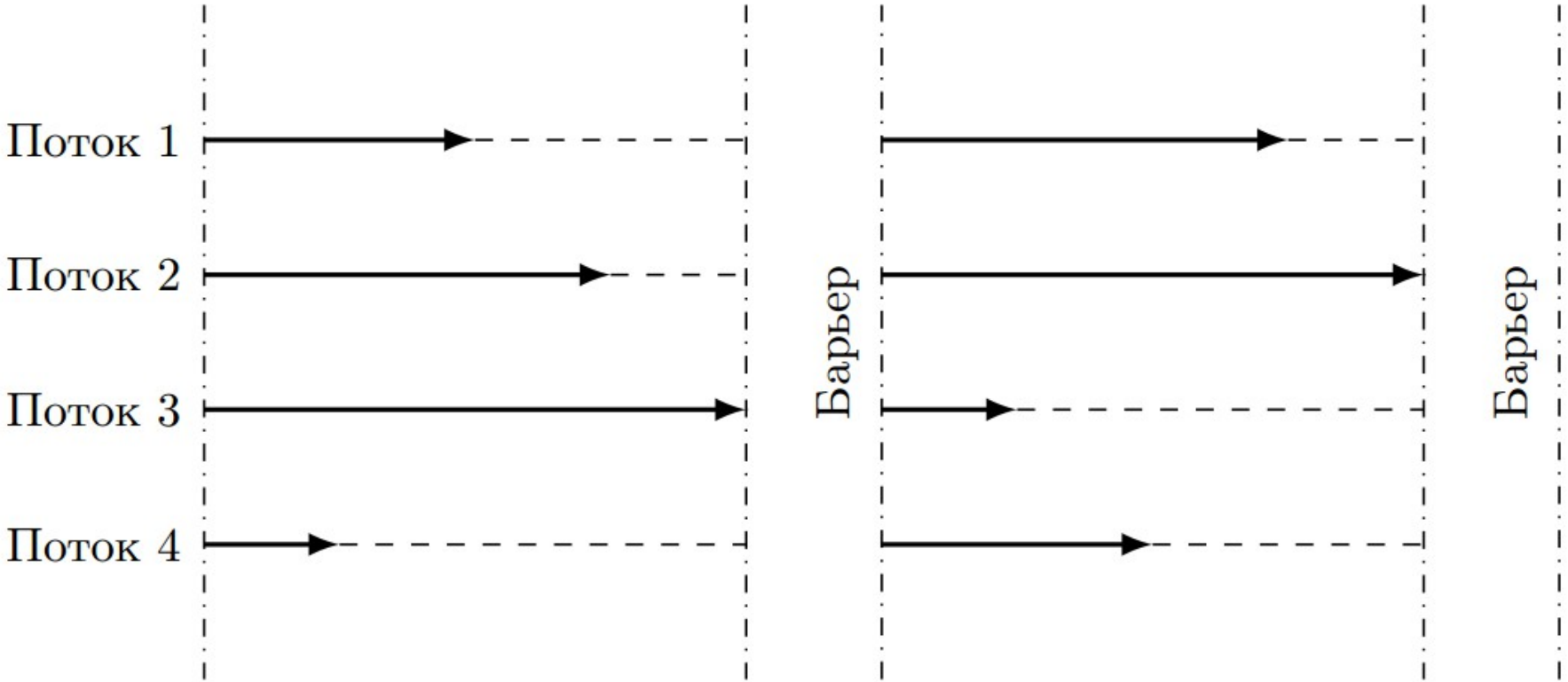
Пример



Пустые сообщения 3

- Как часто рассылать сообщения?
  - Часто => потоки могут бежать свободнее, но большой трафик
  - Редко => потоки не имеют актуальной информации о состояниях остальных и простаивают зря
- Кому рассылать?
  - Всем остальным — большой трафик
  - Не всем — дедлоки вероятны
  - Случайным адресатам — можно балансировать

Частный случай: домены синхронизации



Особенности барьерной синхронизации

- Детерминистична
- Оптимальный сценарий: частые коммуникации внутри домена, редкие – между доменами.
  - Пример: домен == SMP-машина, симуляция == группа машин, соединённых по сети
- Влияние величины кванта синхронизации на наблюдаемые внутри симуляции величины
  - Ping: 2 секунды при quant == 1 секунда

Рекомендуемая литература

Jayadev Misra

«Distributed discrete-event simulation»
ACM Computing Surveys 18 (1986)

www.cis.udel.edu/~cshen/861/notes/p39-misra.pdf

На следующей лекции:

- Оптимистичные схемы
- Детерминированность симуляции
- Что ещё можно распараллелить?
- The state of the art

Спасибо за внимание!

Все материалы курса выкладываются на сайте лаборатории:
http://iscalare.mipt.ru/material/course_materials/