# Параллельная симуляция. Параллельная дискретная симуляция

## Курс «Программное моделирование вычислительных систем»

Григорий Речистов
grigory.rechistov@phystech.edu

19 апреля 2015 г.

# На прошлых лекциях

- Моделирование многих агентов, работающих асинхронно — DES в один поток
- Параллельное моделирование исполняющих устройств — процессоров

# Вопросы

- Что такое атомарная инструкция?

# Вопросы

- Что такое атомарная инструкция?
- Что такое инструкция-барьер?

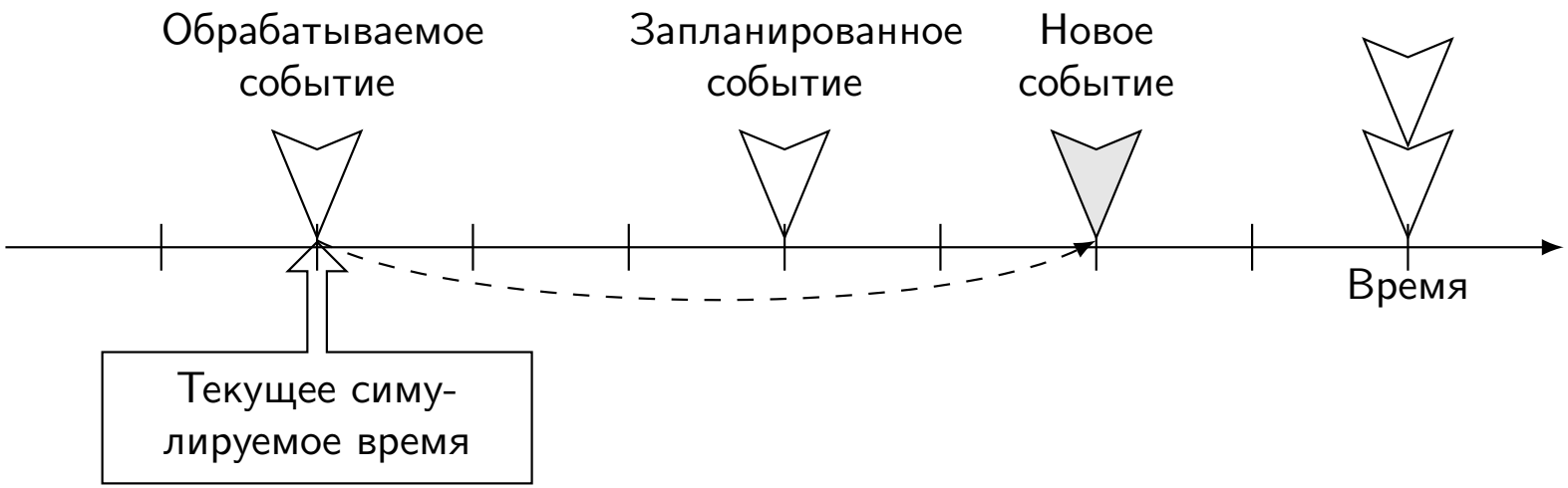
# Вопросы

- Что такое атомарная инструкция?
- Что такое инструкция-барьер?
- Верно ли, что любая атомарная инструкция является барьером?

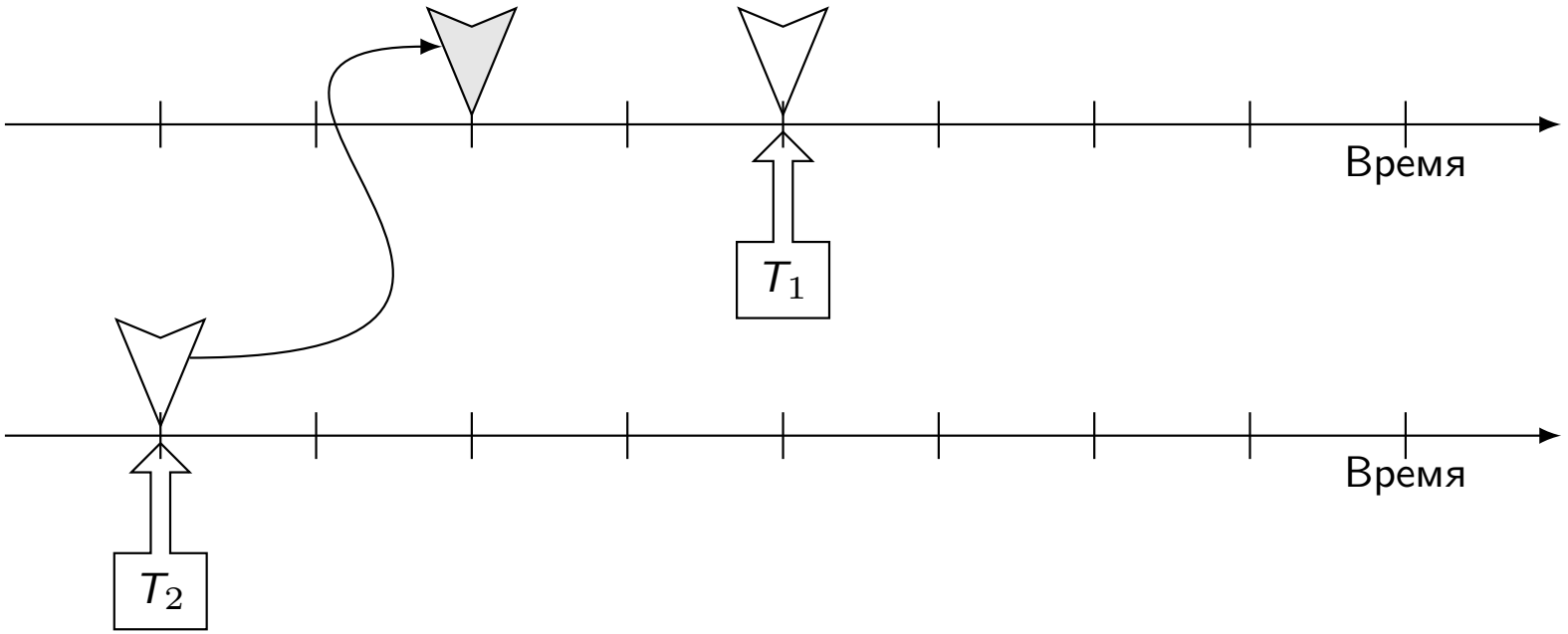
# Общая схема моделируемой системы

TODO: a mess

# DES


Обрабатываемое событие
Запланированное событие
Новое событие
Время
Текущее симу-лируемое время

# Наивный PDES



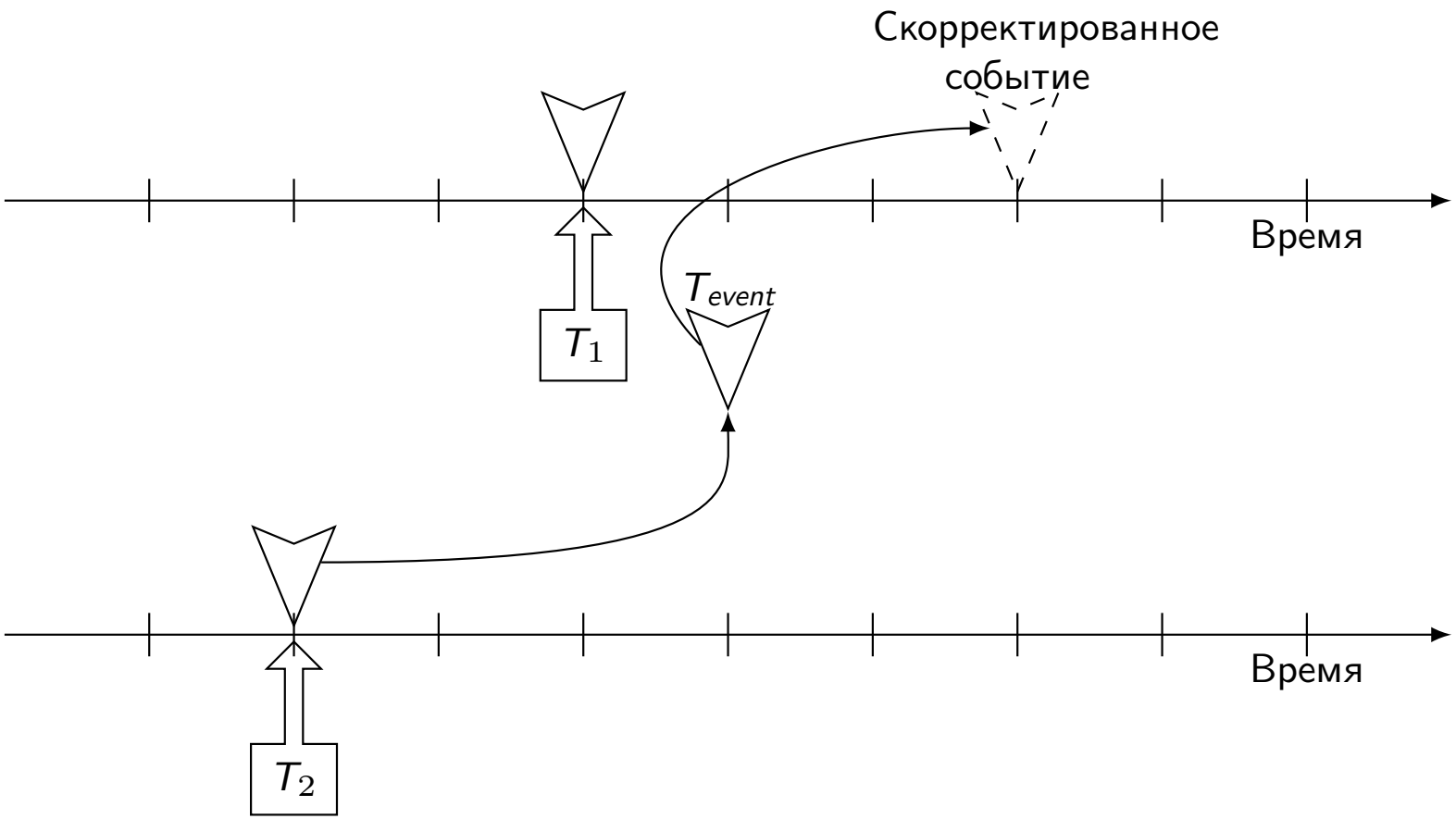
# Проблемы

- Нарушение каузальности (причинно-следственной связи)
- Недетерминизм модели
- Эффект ускорения от параллелизации не гарантирован

# Как 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

# Метки времени

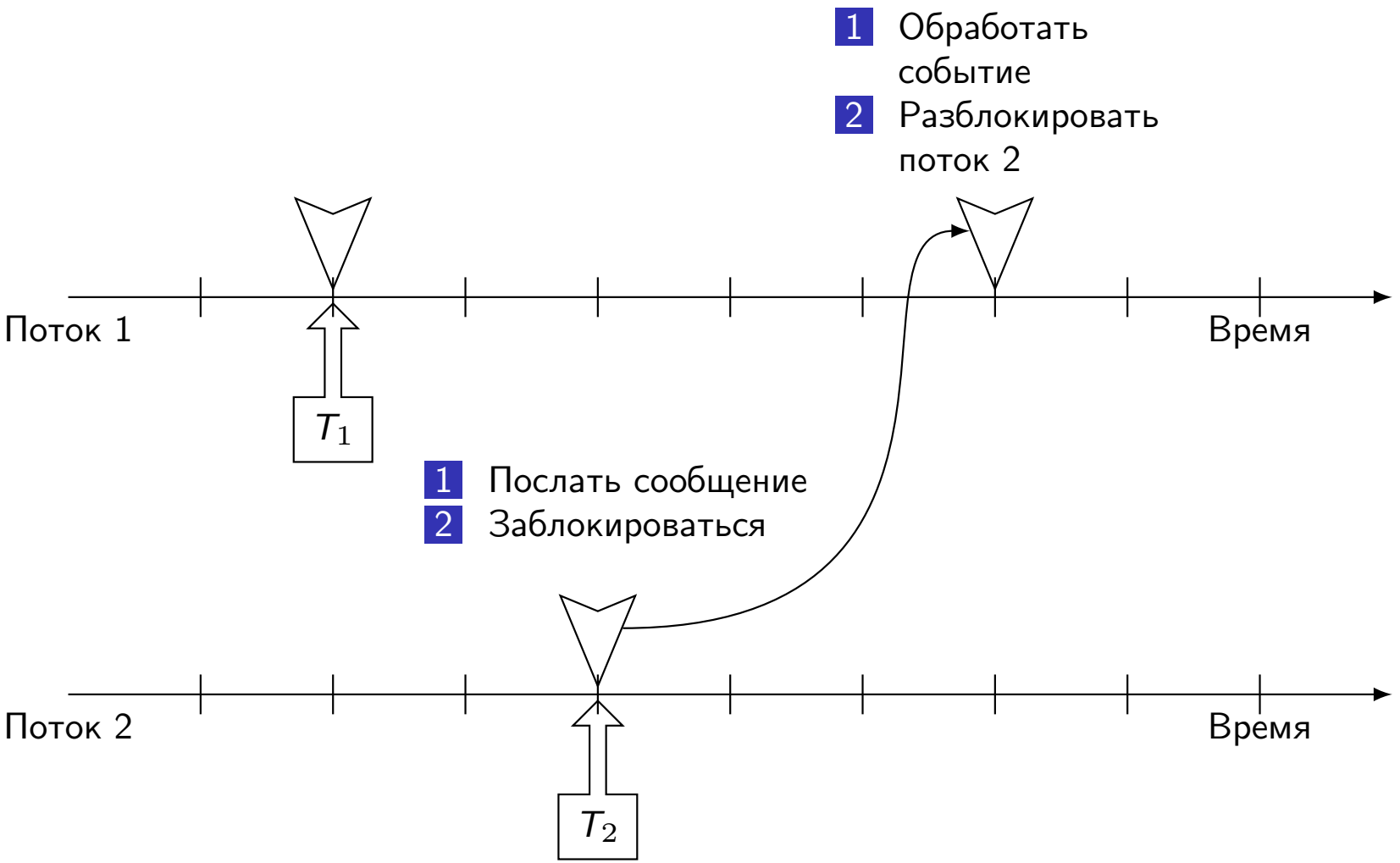


# Консервативно или оптимистично?

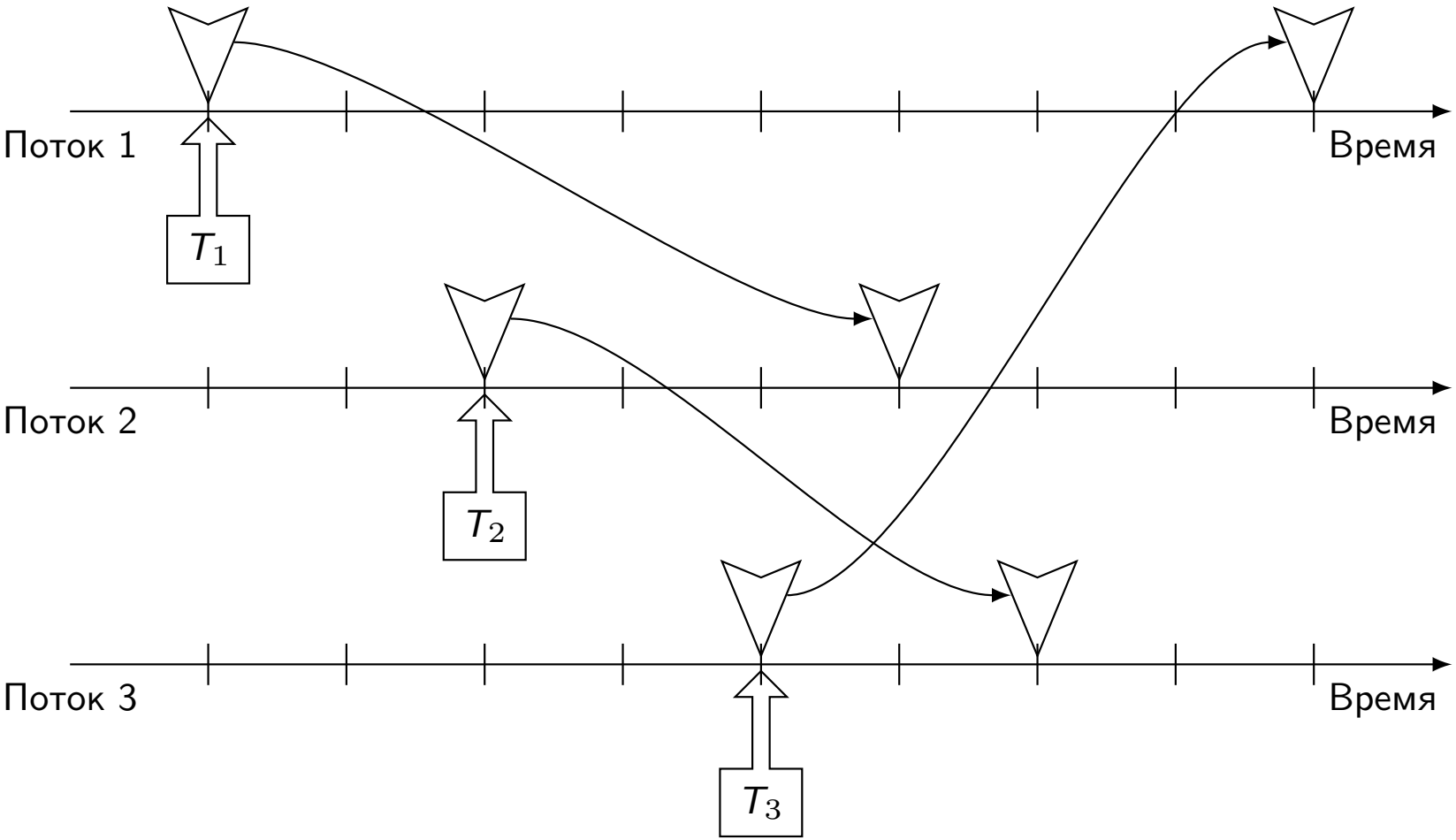
Теперь мы можем обнаруживать нарушения, но всё ещё не можем избавиться от их появления/последствий. Необходимо: 1. Предотвращать их возникновение или 2. Подавлять их вредное проявление

# 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


1 Обработать событие
2 Разблокировать поток 2
Поток 1
Время
$T_1$
1 Послать сообщение
2 Заблокироваться
Поток 2
Время
$T_2$

# Взаимоблокировка



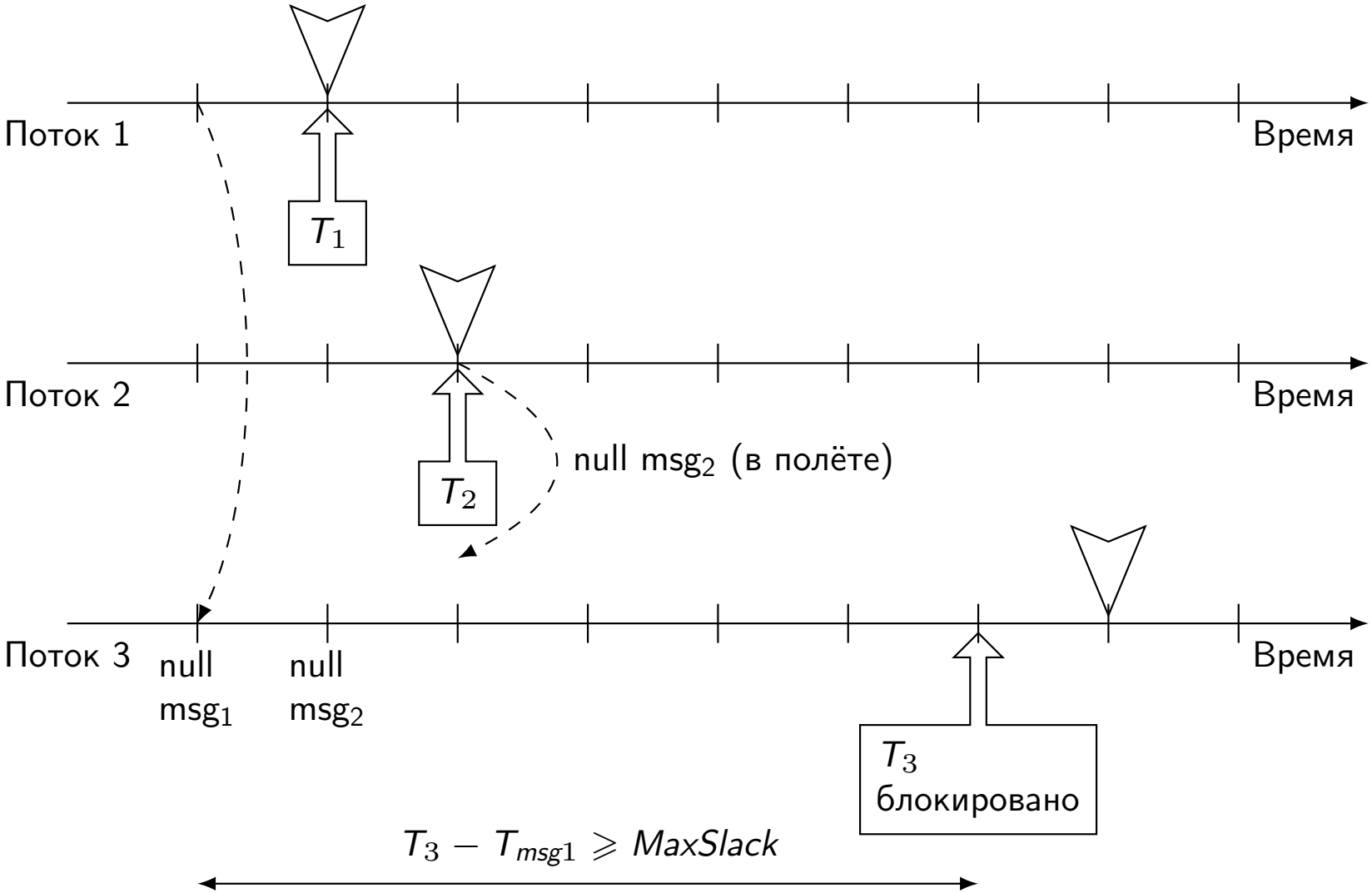
Консервативная схема 3 ☒ Необходимо детектировать ситуацию дедлока и разрушать его, освобождая один поток ☒ Лучший выбор — очередь с наименьшим значением симулируемого времени ☒ Система может оказаться в ситуации, когда большую часть времени почти все потоки стоят => выигрыша в скорости нет

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 A может узнать локальное время потока B? Через временную метку, хранящуюся в каждом событии

# Пустые сообщения

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

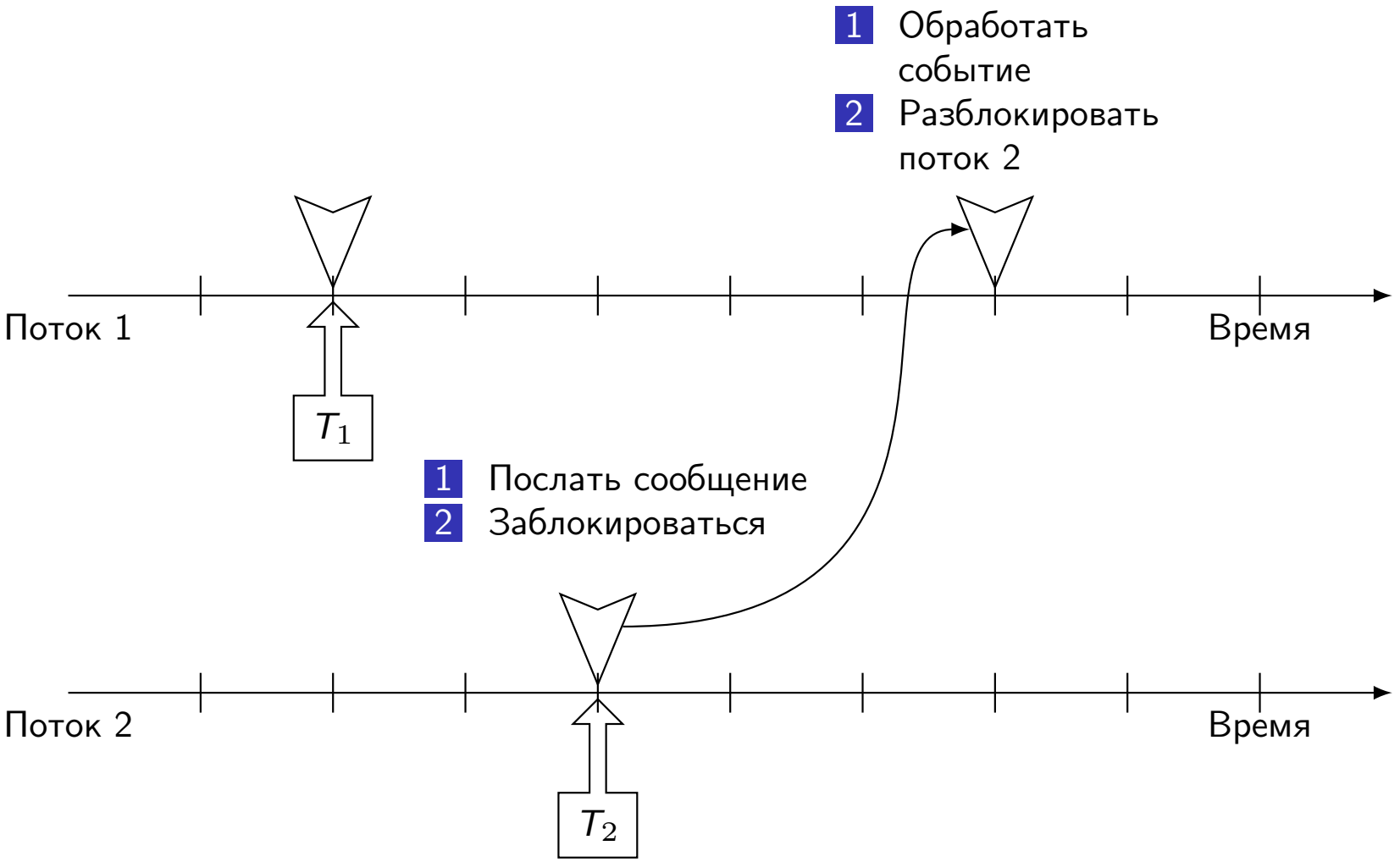
# Slack


Поток 1
Время
$T_1$
Поток 2
Время
$T_2$
null $msg_2$ (в полёте)
Поток 3
Время
null
$msg_1$
null
$msg_2$
$T_3$
блокировано
$T_3 - T_{msg1} \geqslant MaxSlack$

# Пустые сообщения

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

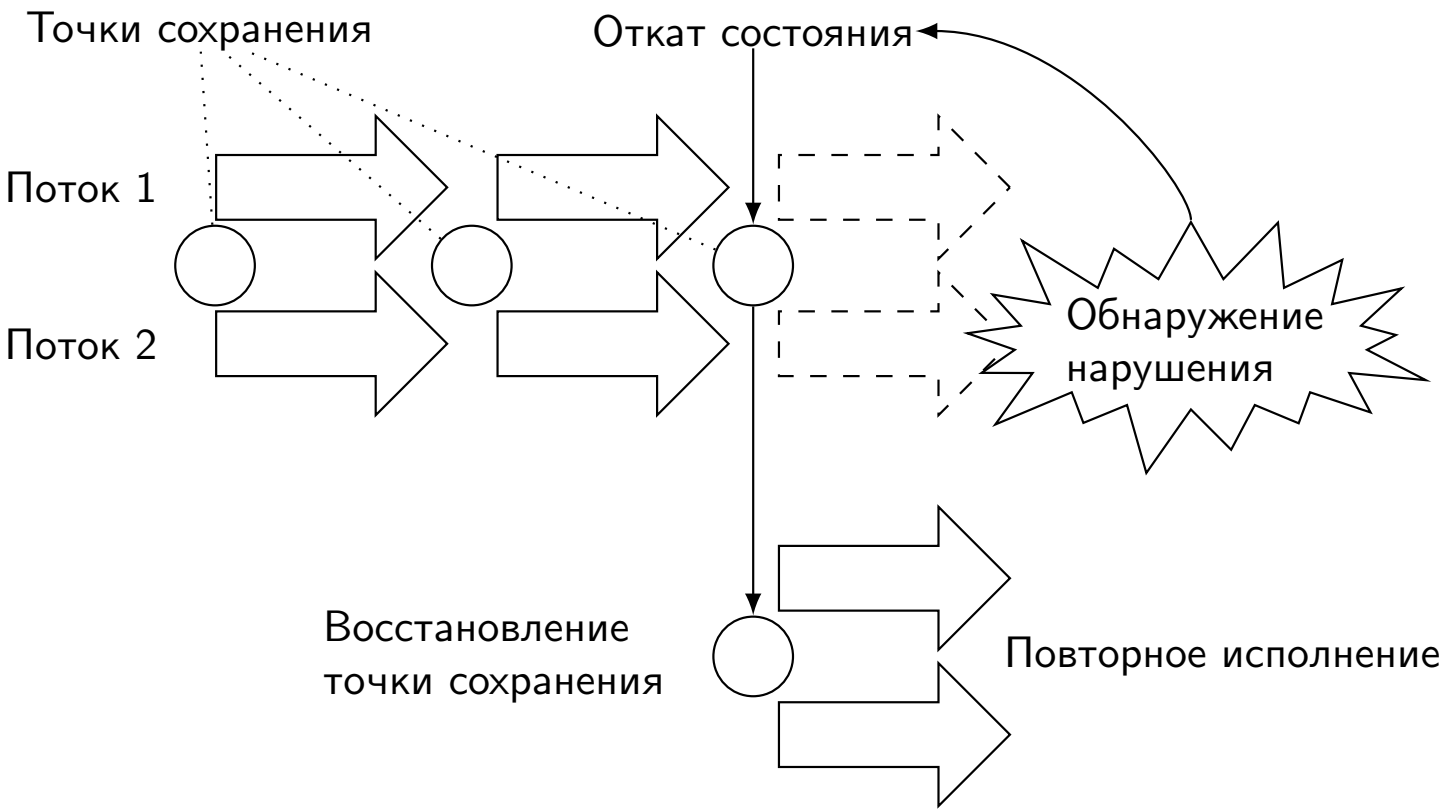
# Частный случай: домены синхронизации



# Оптимистичные схемы

Даём параллельной программе работать самой по себе
Периодически сохраняем (корректное) состояние всей системы
При обнаружении каузальных ошибок откатываемся до ближайшего сохранённого состояния Проходим проблемный участок аккуратными методами (напр. консервативно)

# Точки сохранения



# Time Warp

Сообщение — набор данных, описывающих событие, которое должно быть добавлено в одну из очередей событий. Оно характеризуется, кроме своего непосредственного содержимого, виртуальными временами отправки tsend и обработки treceive. LVT (local virtual time) — значение симулируемого времени отдельного потока, участвующего в симуляции. Для создаваемых событий их время отправки tsend равно значению LVT отправителя. В отличие от консервативных схем, эта величина может как расти в процессе симуляции, так и убывать в случае отката процесса.

# GVT

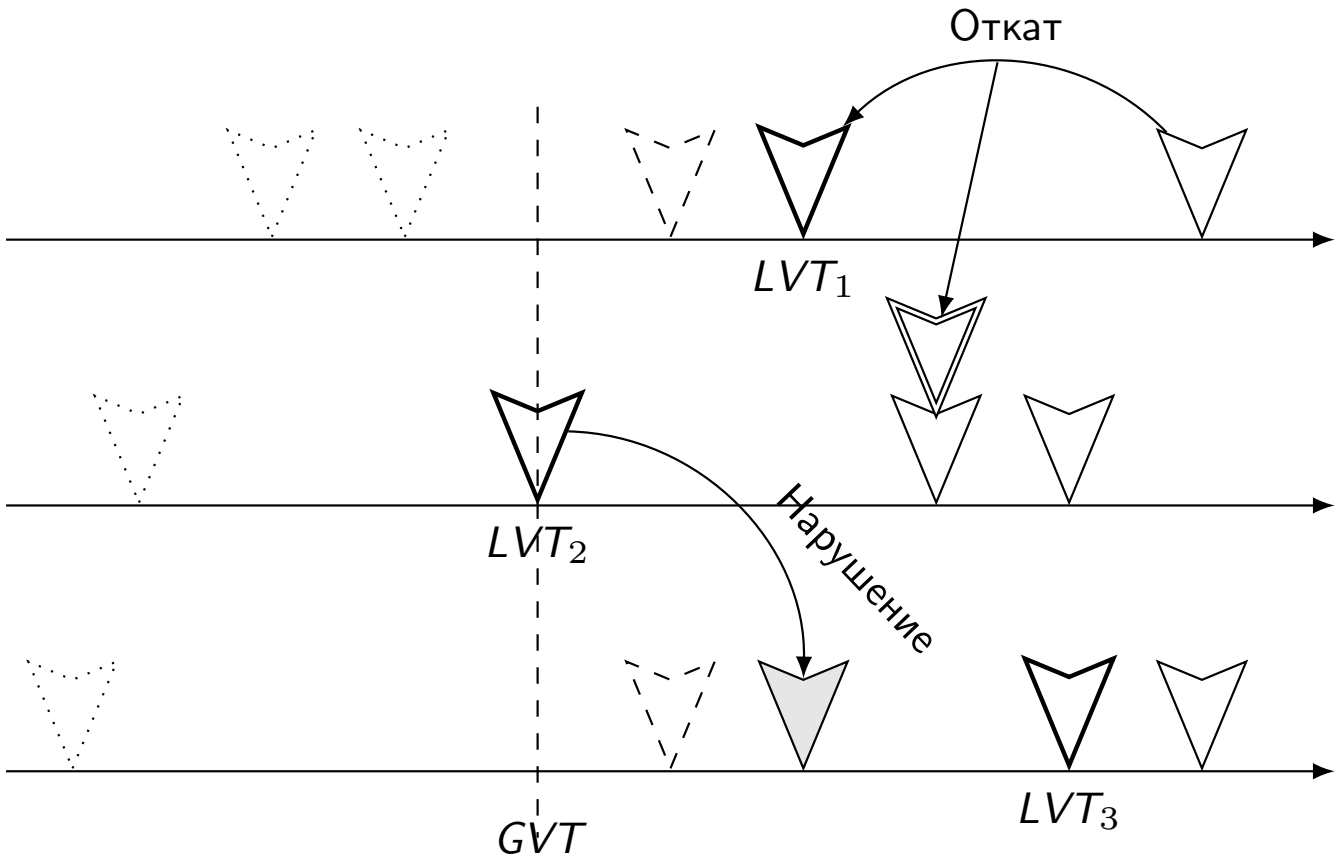
GVT (global virtual time) — глобальное время для всей симуляции, определяющее, до какой степени возможно её откатывать. Глобальное время всегда монотонно растёт, всегда оставаясь позади локального времени самого медленного потока, а также оно меньше времени отправки самого раннего ещё не доставленного события:

$$GVT \leq \min\left(\min_i LVT_i, \min_k t_k^{send}\right).$$

# Straggler, antimessage

Отставшее сообщение (straggler) — событие, пришедшее в очередь с меткой времени treceivestraggler, меньшей, чем LVT получателя. Его обнаружение вызывает откат текущего состояния, при этом LVT уменьшается, пока не станет меньше, чем treceivestraggler, после чего оно может быть обработано. После этого возобновляется прямая симуляция.
Антисообщение (antimessage). Каждое антисообщение соответствует одному ранее созданному сообщению, порождённому в интервале симулируемого времени [tstraggler, LVTi] и вызывает эффект, обратный его обработке (т.е. возвращает состояние в исходное).

# Работа Time Warp



# Fossil Collection

Освобождение места, занятого сообщениями, расположенными левее GVT

# Рекомендуемая литература I


- Fujimoto Richard M. Parallel discrete event simulation // Commun. ACM. — 1990. — Окт. — Т. 33, No 10. — С. 30–53. http://doi.acm.org/10.1145/84537.84545
- Liu Jason. Parallel Discrete-Event Simulation. — 2009. http://www.cis.fiu.edu/~liux/research/papers/pdes-eorms09.pdf
- Jayadev Misrsa. Distributed discrete-event simulation //ACM Computing Surveys 18 1986 www.cis.udel.edu/~cshen/861/notes/p39-misra.pdf

# На следующей лекции

Эффективная современная виртуализация

# Спасибо за внимание!

Слайды и материалы курса доступны по адресу
http://is.gd/ivuboc

*Замечание*: все торговые марки и логотипы, использованные в данном материале, являются собственностью их владельцев. Представленная здесь точка зрения отражает личное мнение автора, не выступающего от лица какой-либо организации.