Раздел 2. Модель распределенного вычисления

LET IT TRADE

ОРС®

УНИВЕРСИТЕТ ИТМО

Модель распределенной системы

- Множество из N независимых процессов $\{P_1, P_2, \ldots P_N\}$ и множество однонаправленных каналов $\{C_{ij}\}$ между процессами P_i и P_j
- Канал связи C_{ij} существует, если P_i может напрямую отправлять сообщение P_j
- Двунаправленный канал связи – пара разнонаправленных каналов
- Топология сети – ориентированный граф
- Асинхронные распределенные системы и асинхронный обмен сообщениями

Модель распределенной системы

- Состояние канала – совокупность отправленных по этому каналу, но еще не полученных, сообщений
- Состояние процесса – «концентрированное прошлое»; в общем случае определяется текущими значениями счетчика команд, регистров и переменных
- Модель процесса – множество возможных состояний (states), подмножество из начальных состояний (initial states) и множество дискретных событий (events)

События

- События:
  - внутренние (internal events)
  - отправки сообщения (send events)
  - получения сообщения (receive events)
- Пятерка (P, s, s', m, C)
- События каждого процесса линейно упорядочены согласно порядку их наступления!
- Обозначения:
  - e_i^x – x-ое по порядку событие в процессе P_i
  - $e_i^x(s) = s'$

Выполнение процесса

- Выполнение (run, execution) процесса P_i – последовательность событий e_i^0, e_i^1, … e_i^x, e_i^{x+1}, … в которой s_i^0 – начальное состояние процесса P_i и для каждого $x \geq 0$: $s_i^{x+1} = e_i^x(s_i^x)$
- Обозначение: $R_i = (E_i, \rightarrow_i)$, где
  - E_i – множество событий процесса P_i
  - Бинарное отношение $\rightarrow_i$ задает линейный порядок на E_i согласно порядку наступления событий в P_i
  - $e_i \rightarrow_i e_i'$: e_i предшествует e_i' в выполнении процесса P_i

Глобальное состояние

- Определяется состояниями всех процессов и всех каналов, входящих в состав РС
- Начальные глобальные состояния – состояния, в которых все процессы пребывают в своих начальных состояниях, а каналы связи – пусты
- $E = \cup_i E_i$ – множество всех событий, происходящих при выполнении РС

Допустимые события

- Событие $e \in E$, такое что $e = (P, s, s', m, C)$, считается допустимым в глобальном состоянии S тогда и только тогда, когда:
  - в глобальном состоянии S процесс P находится в состоянии s
  - если ребро C направлено к процессу P, и сообщение m находится в канале C и может быть принято из него
- Состояние $e(S)$ определено только в случае, если e допустимо в глобальном состоянии S
- Событие e влияет только на часть глобального состояния: чем $e(S)$ отличается от S ???

Выполнение распределенной системы

- Последовательность событий e^0, e^1, … e^x, e^{x+1}, … называется выполнением распределенной системы R тогда и только тогда, когда
  - событие e^x допустимо в глобальном состоянии S^x
  - S^0 – начальное глобальное состояние PC
  - для каждого $x \geq 0$: $S^{x+1} = e^x(S^x)$
- Глобальное состояние S называется достижимым в выполнении R, если для этой последовательности событий существует такое $k \geq 0$, что $e^k(S^k) = S$

Пример передачи маркера

Топология сети:

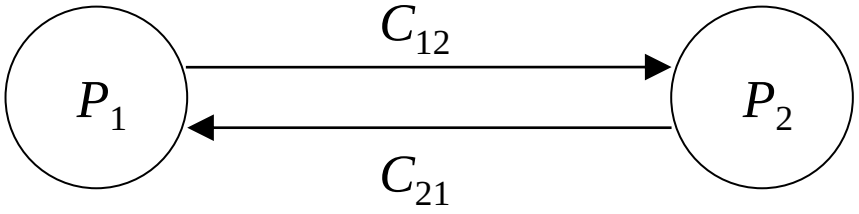
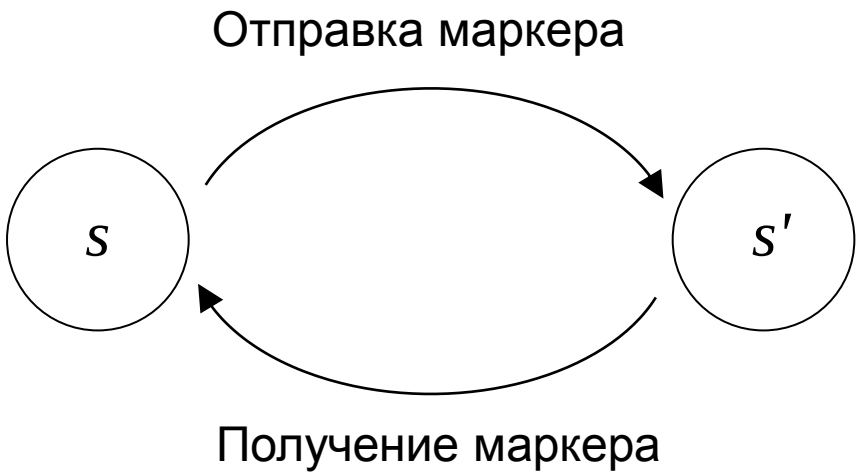
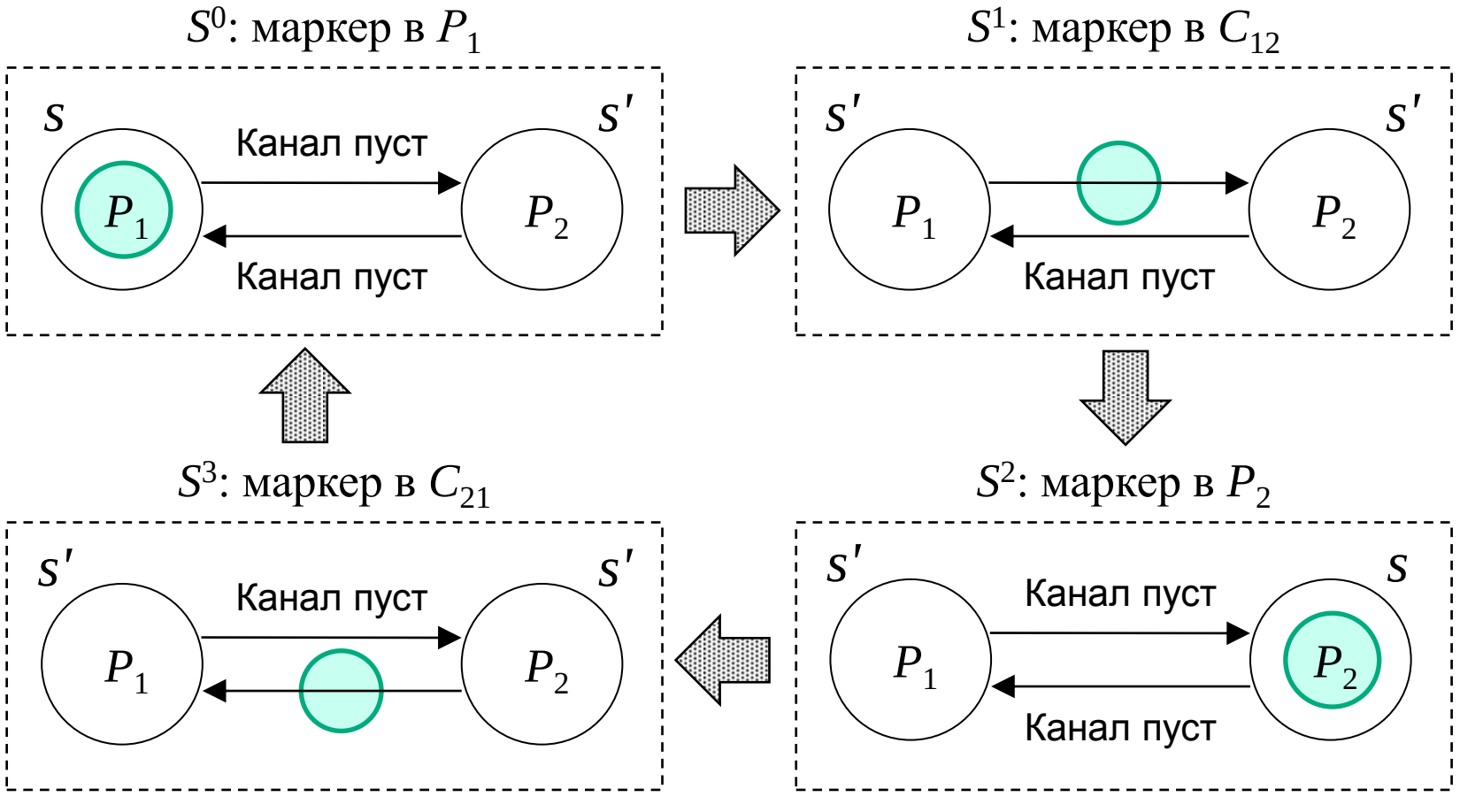


Диаграмма состояний процесса:



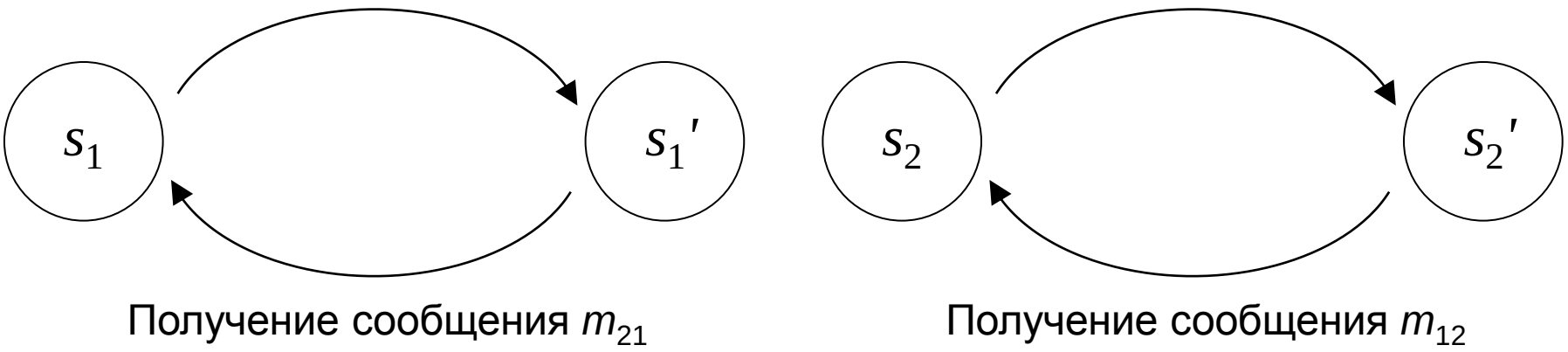
Глобальные состояния и переходы между ними



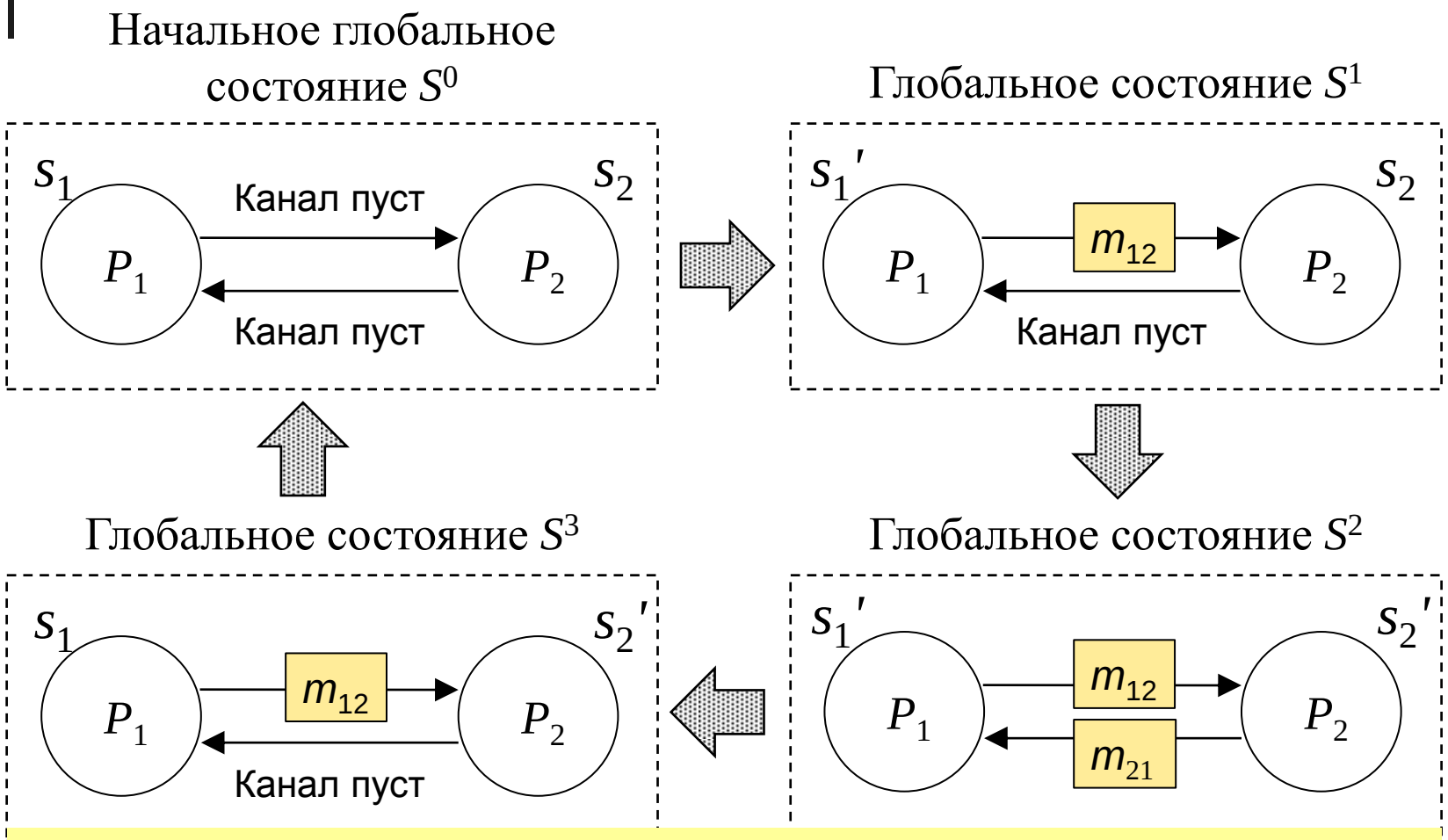
В каждом глобальном состоянии допустимо ровно одно событие!!!

Недетерминированное выполнение

Диаграмма состояний процессов P_1 и P_2;
s_1 и s_2 – начальные состояния

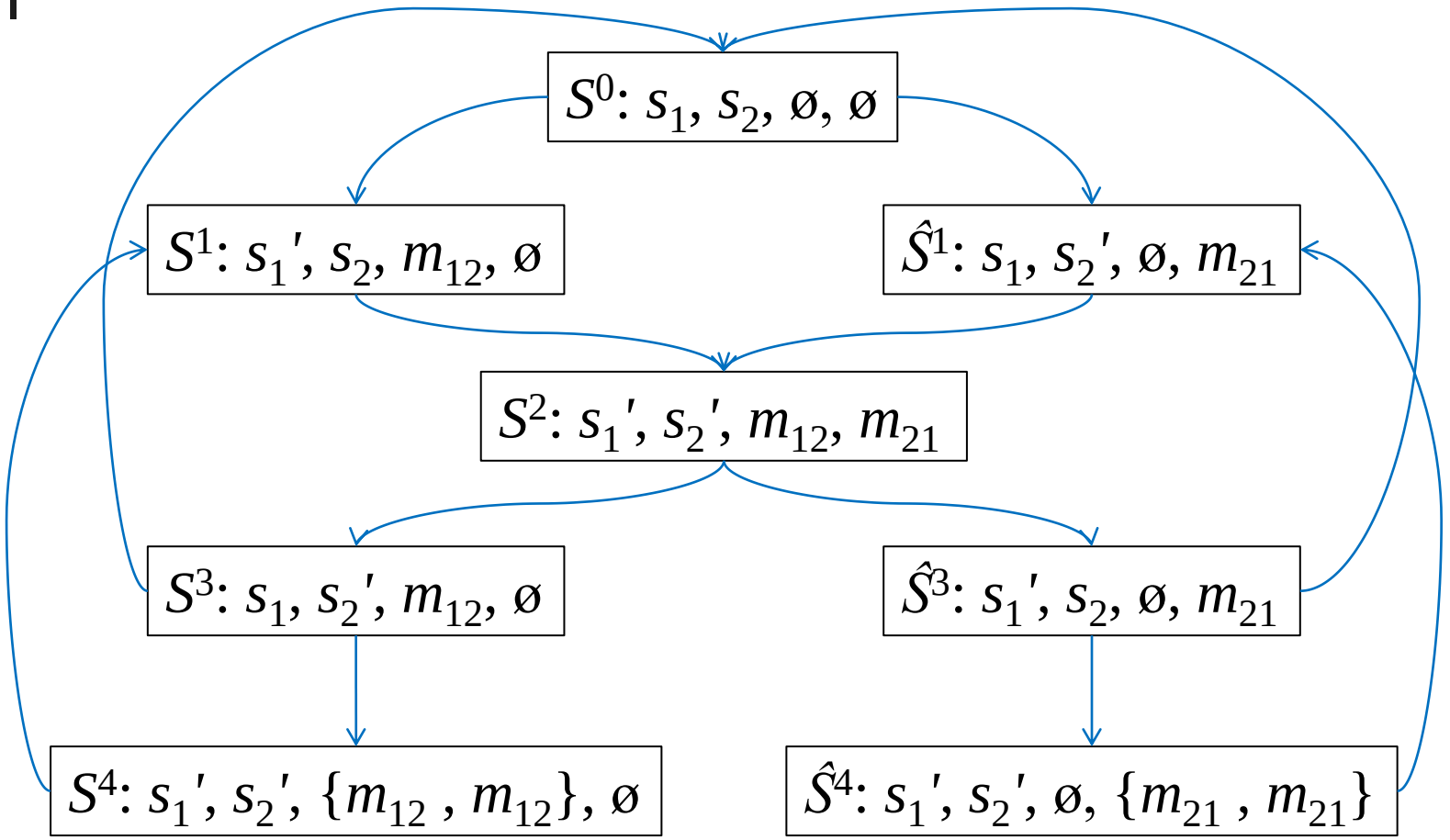


Одно из множества выполнений



М.б. другая последовательность событий и, как следствие, последовательность других глобальных состояний

Глобальные состояния и переходы между ними



Выполнение распределенной системы

while (*S* = нетерминальное состояние)

{

недетерминировано выбрать допустимое в *S* событие *e*

выполнить *e* и положить *S* = *e*(*S*)

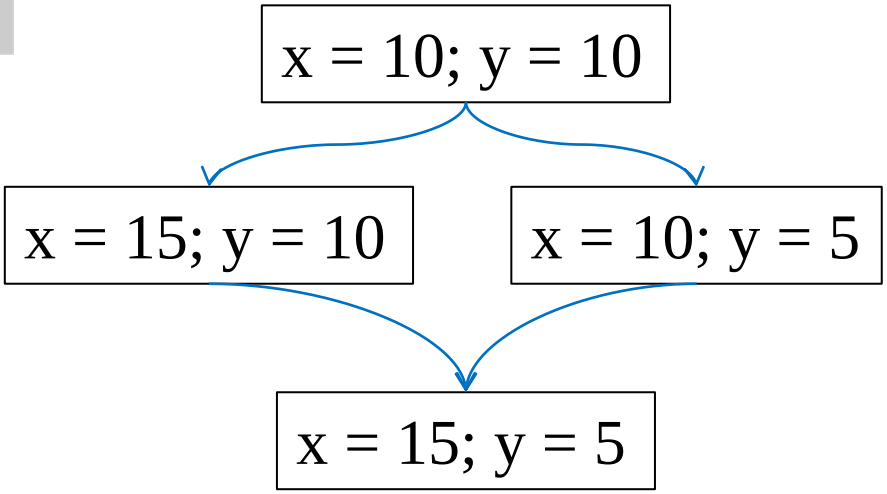
}

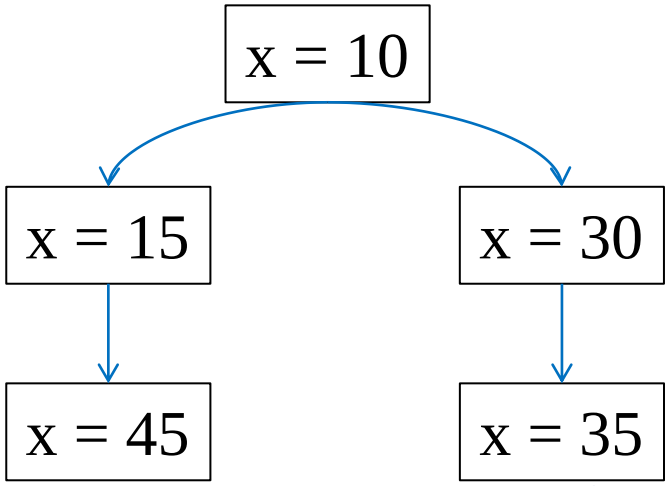
Выполнение: последовательность переходов

Недетерминизм

$x = 10; y = 10$

$x = x+5; y = y - 5$	$x = x+5; x = 3x$
Чередование	Состязание





Два типа свойств распределенных алгоритмов

- Безопасность (safety) - что-то плохое никогда не случается

«требуется, чтобы определенное утверждение оставалось истинным в каждом достижимом глобальном состоянии в любом выполнении»

- Живучесть (liveness) - что-то хорошее рано или поздно случится

«требуется, чтобы определенное утверждение становилось истинным хотя бы в одном достижимом глобальном состоянии в любом выполнении»

Справедливость

- Не все выполнения одинаково полезны! (отбросим «нереалистичные» выполнения)
- Справедливость (fairness) – если часто что-то получает возможность произойти, то оно обязательно случится
- Справедливость обычно нужна для установки живучести
- «Если в первом акте пьесы на стене висит ружьё, то в последнем акте оно непременно должно выстрелить» *А.П. Чехов*

Справедливость

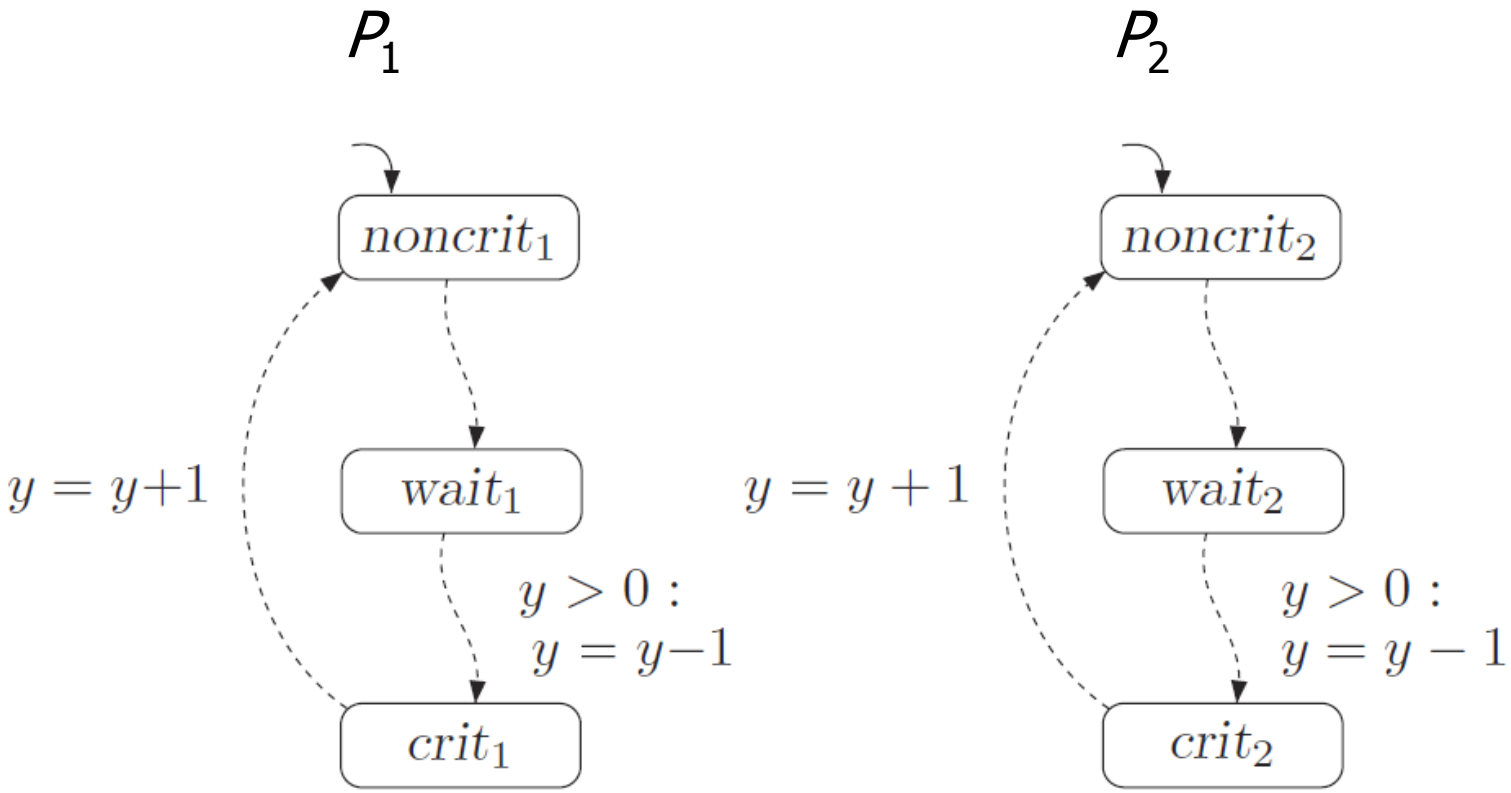
- Слабая справедливость:

«ни одно событие не может оставаться постоянно допустимым без того, чтобы, в конце концов, не произойти в виде перехода»

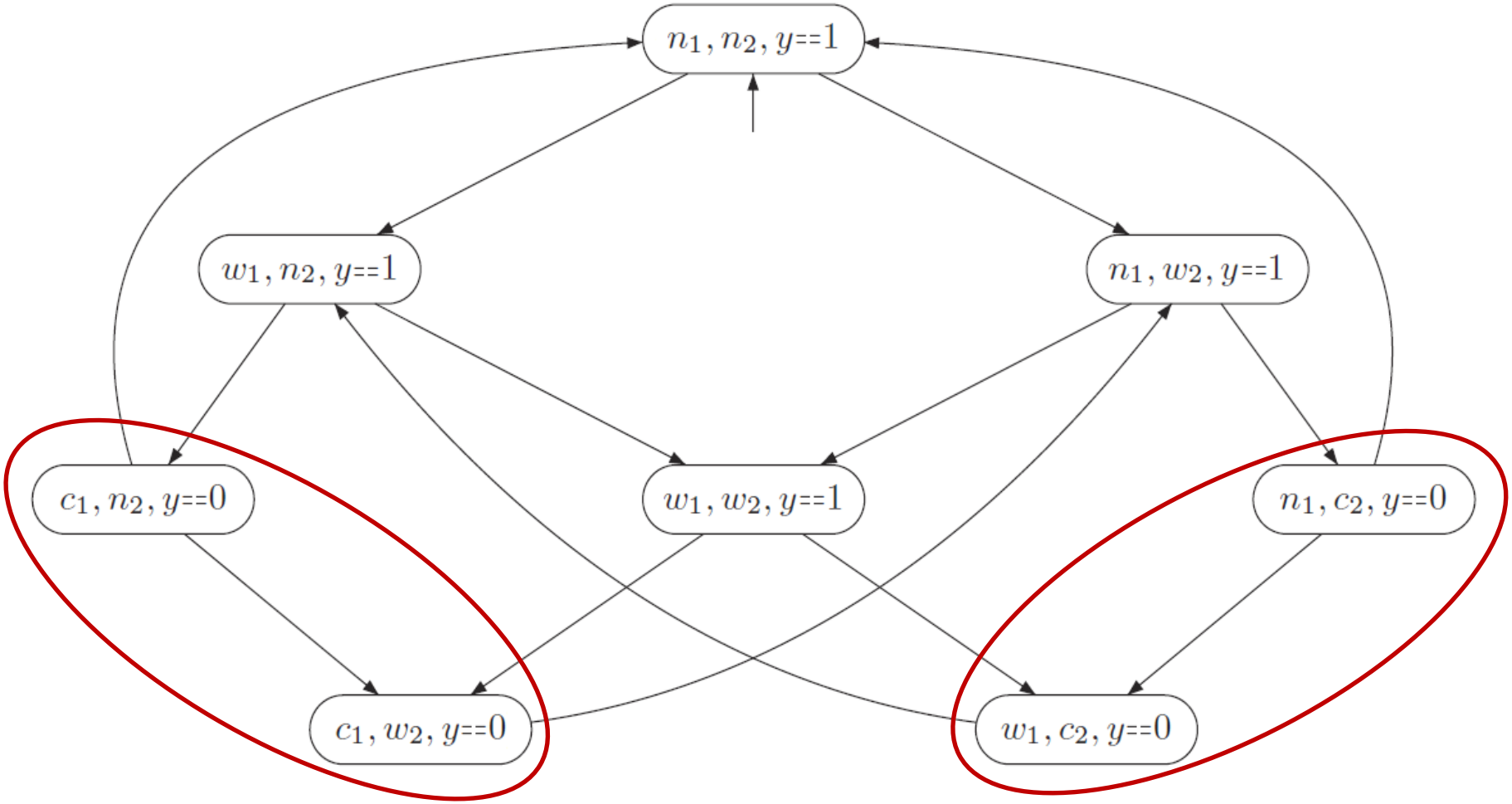
- Сильная справедливость:

«ни одно событие не может оказываться допустимым бесконечно часто без того, чтобы, в конце концов, не произойти в виде перехода»

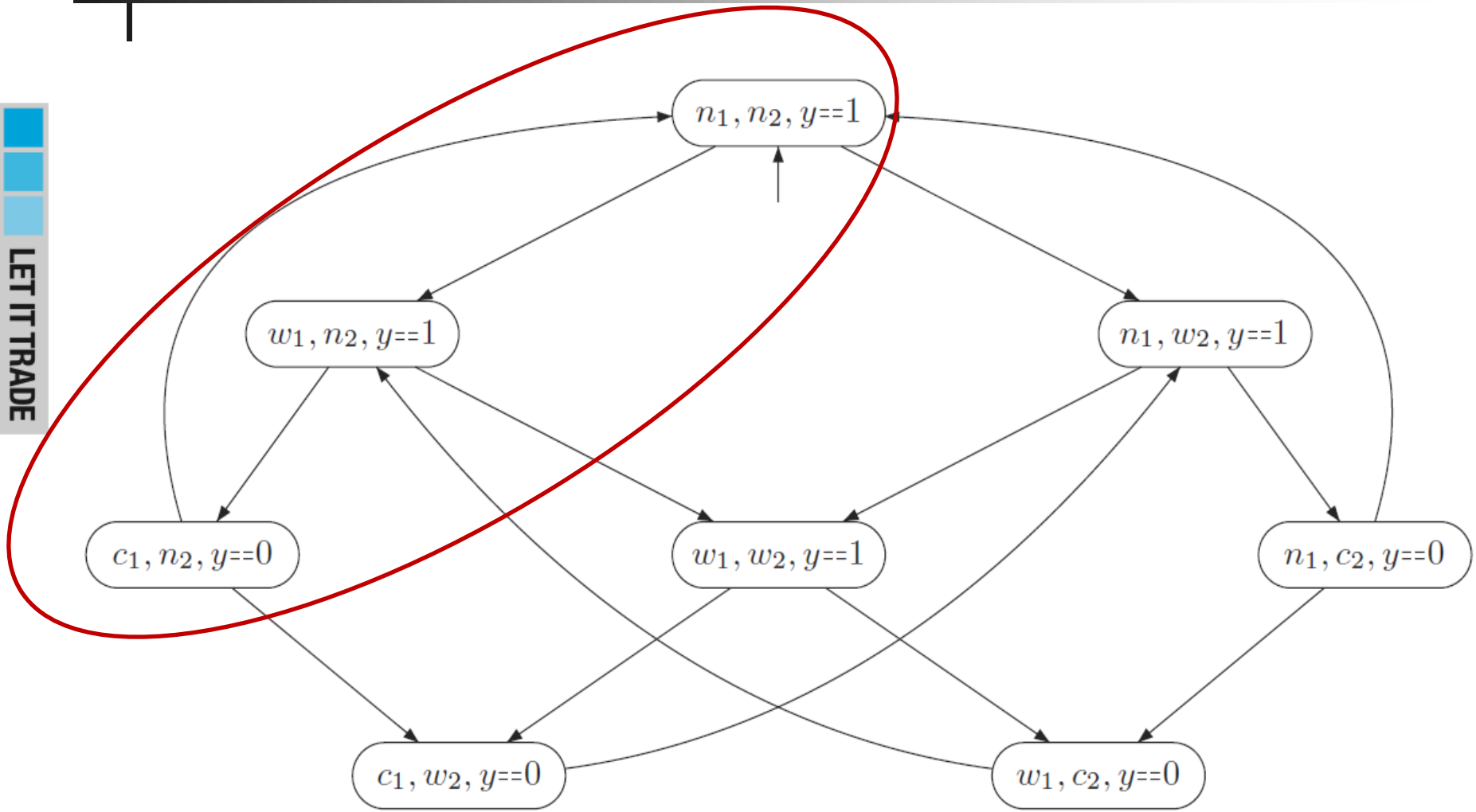
Взаимное исключение на семафоре



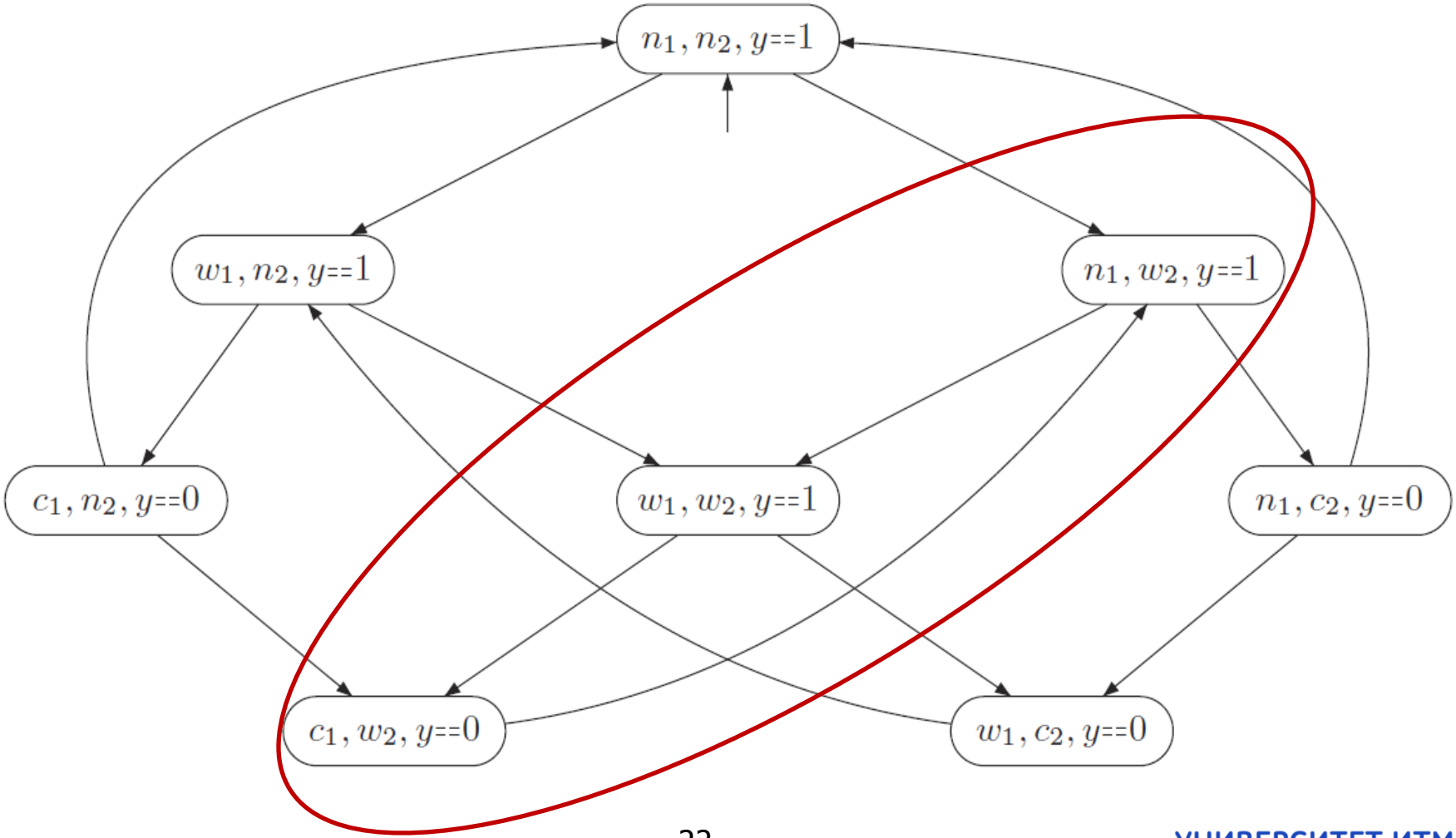
Взаимное исключение на семафоре



Взаимное исключение на семафоре



Взаимное исключение на семафоре



Несправедливость MPI

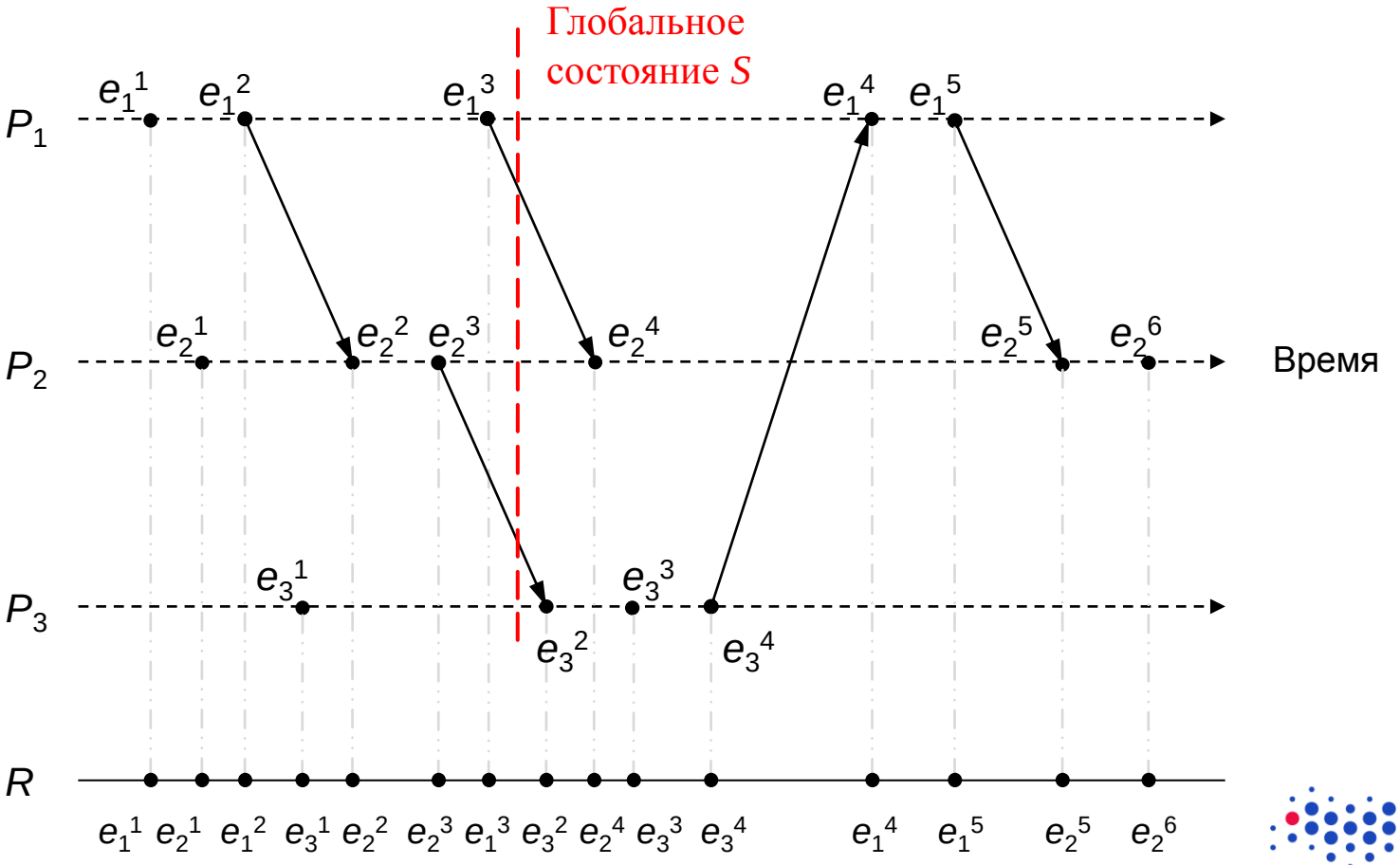
MPI makes no guarantee of *fairness* in the handling of communication.

Suppose that a send is posted. Then it is possible that the destination process repeatedly posts a receive that matches this send, yet the message is never received, because it is each time overtaken by another message, sent from another source.

Similarly, suppose that a receive was posted by a multi-threaded process. Then it is possible that messages that match this receive are repeatedly received, yet the receive is never satisfied, because it is overtaken by other receives posted at this node (by other executing threads). It is the programmer's responsibility to prevent starvation in such situations.

Причинно-следственный порядок событий

Пространственно-временная диаграмма:



Причинно-следственный порядок событий

- Порядок следования некоторых событий в выполнении *R* может быть изменен так, что это никак не отразится на последующем глобальном состоянии
- Поэтому представление о времени как о линейном порядке на множестве *E* всех событий не совсем подходит для распределенной обработки информации !!!

Независимые события

- Пусть:
  - S – глобальное состояние РС
  - e_i и e_j – события, которые происходят в разных процессах P_i и P_j, т.е. $i \neq j$
  - оба события допустимы в глобальном состоянии S
- Тогда:
  - событие e_i допустимо в глобальном состоянии $e_j(S)$
  - событие e_j допустимо в глобальном состоянии $e_i(S)$
  - $e_i\,(e_j(S)) = e_j\,(e_i(S))$

Независимые события

- Доказательство:
  - $e_i = (P_i, s_i, s_i', m_i, C_i)$ и $e_j = (P_j, s_j, s_j', m_j, C_j)$
  - e_i и e_j не являются взаимосвязанными событиями отправки и получения одного и того же сообщения, т.е. m_i и m_j – различные сообщения
  - $S_i = e_i(S)$ отличается от S только состоянием процесса P_i и, возможно, состоянием канала C_i
  - поэтому e_j является допустимым в состоянии S_i
  - $S_{ij} = e_j(S_i)$:
    - P_i и P_j – в состояниях s_i' и s_j'
    - сообщение m_i либо добавлено в канал C_i, либо удалено из него
    - сообщение m_j либо добавлено в канал C_j, либо удалено из него
  - $S_{ij} = e_j(S_i) = e_i(S_j) = S_{ji}$

Отношение причинно-следственной зависимости

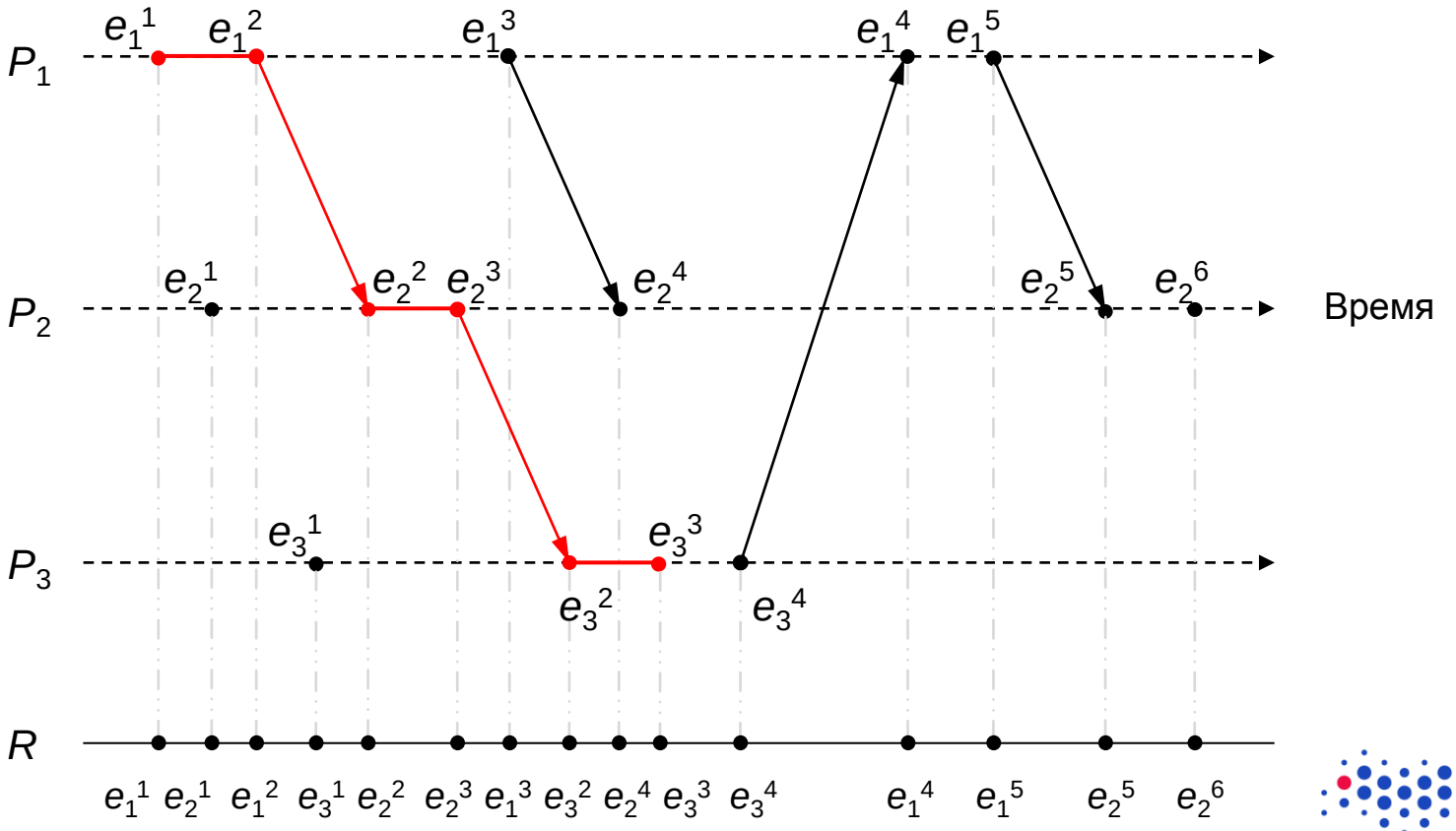
- События не могут произойти в другом порядке только в двух случаях:
  - $i = j$, т.е. события e_i и e_j происходят в одном и том же процессе
  - e_i и e_j – взаимосвязанные события отправки и получения одного и того же сообщения
- При перестановке событий система будет проходить через разные глобальные состояния, т.е. переходы $S \rightarrow S_i \rightarrow S_{ij}$ отличаются от переходов $S \rightarrow S_j \rightarrow S_{ji}$ т.к. $S_i \neq S_j$, хотя $S_{ij} = S_{ji}$!!!

Отношение причинно-следственного порядка

- e_i и e_i' – два разных события одного и того же процесса P_i и при этом $e_i \rightarrow_i e_i'$, то $e_i \rightarrow e_i'$
- e_i и e_j' – взаимосвязанные события отправки и получения одного и того же сообщения, то $e_i \rightarrow e_j'$
- отношение $\rightarrow$ транзитивно, т.е. если $e_i \rightarrow e_j'$ и $e_j' \rightarrow e_k''$, то $e_i \rightarrow e_k''$

Отношение причинно-следственного порядка

Пространственно-временная диаграмма:



Отношение причинно-следственного порядка

- Читаем $e_i \rightarrow e_j'$:
- e_i произошло раньше e_j'
- событие e_i потенциально влияет на событие e_j'
- событие e_j' потенциально зависит от e_i
- событие e_j' знает о событии e_i
- например, e_2^6 знает о наступлении всех других событий на рисунке

Зависимые и независимые события

- если $e_i \not\rightarrow e_j'$, то e_j' не знает не только о e_i, но и обо всех других событиях, произошедших после e_i в том же процессе P_i
- для любых двух e_i и e_j': $e_i \not\rightarrow e_j' \not\Rightarrow e_j' \not\rightarrow e_i$
- для любых двух e_i и e_j': $e_i \longrightarrow e_j' \Rightarrow e_j' \not\rightarrow e_i$
- если $e_i \not\rightarrow e_j'$ и $e_j' \not\rightarrow e_i$, то e_i и e_j' называются параллельными (concurrent) или независимыми (independent): $e_i \parallel e_j'$
- $\parallel$ нетранзитивно: $(e_i \parallel e_j') \wedge (e_j' \parallel e_k'') \not\Rightarrow e_i \parallel e_k''$

Отношение причинно-следственного порядка

- Иррефлексивно
- Задает отношение частичного порядка на множестве E
- Для любых двух событий e_i и e_j', происходящих при выполнении распределенной системы, либо $e_i \rightarrow e_j'$, либо $e_j' \rightarrow e_i$, либо $e_i \parallel e_j'$

Спасибо за внимание!