Управление процессами

Проблема тупиков

Определения

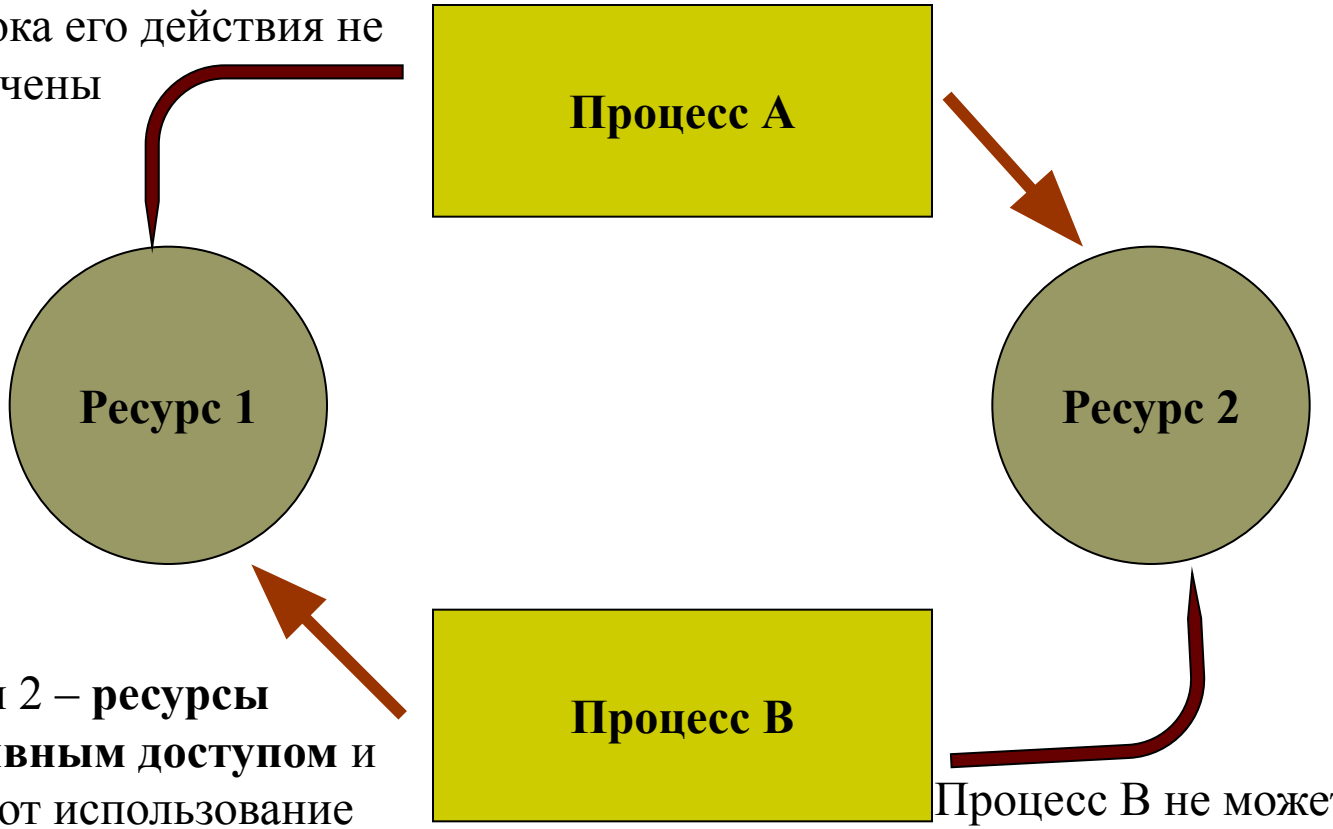
- Тупик (тупиковая ситуация, dead lock, clinch) – ситуация в системе, когда один или несколько процессов находятся в состоянии ожидания события, которое никогда не произойдет
- Бесконечное ожидание – ситуация в системе, когда запуск или возобновление работы процесса откладывается на неопределенное время в силу определенных обстоятельств. Например, если система реализует дискриминационную дисциплину планирования процессов.

Причины возникновения тупиков и бесконечного ожидания

- Реализация дискриминационной дисциплины планирования процессов (например, SJF или SRT)
- Отсутствие в системе механизмов переключения процессов
- Неэффективная политика управления ресурсами системы (например, наличие большого количества ресурсов с эксклюзивным доступом)
- Взаимоблокировки процессов
- Неэффективная реализация систем со спулингом (например, при реализации печати на принтере с использованием полной выгрузки документов в пул принтера)
- и т.д.

Пример возникновения тупика: взаимоблокировка процессов и ресурсы с эксклюзивным доступом

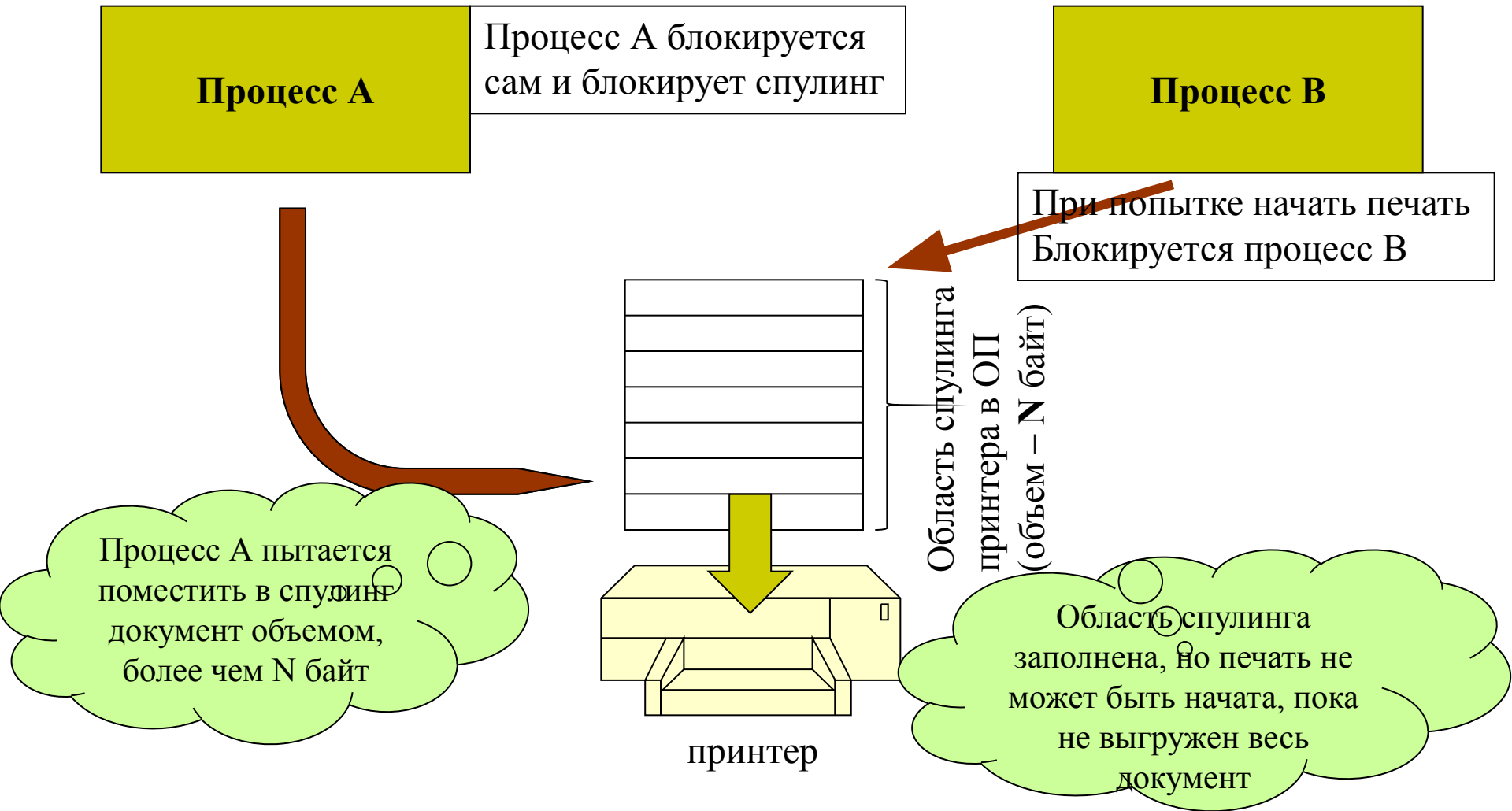
Процесс А не может освободить ресурс 1, пока его действия не будут закончены



Ресурсы 1 и 2 – **ресурсы с эксклюзивным доступом** и не допускают использование более чем одним процессом в один момент времени

Процесс В не может освободить ресурс 2, пока его действия не будут закончены

Пример возникновения тупика: системы со спулингом



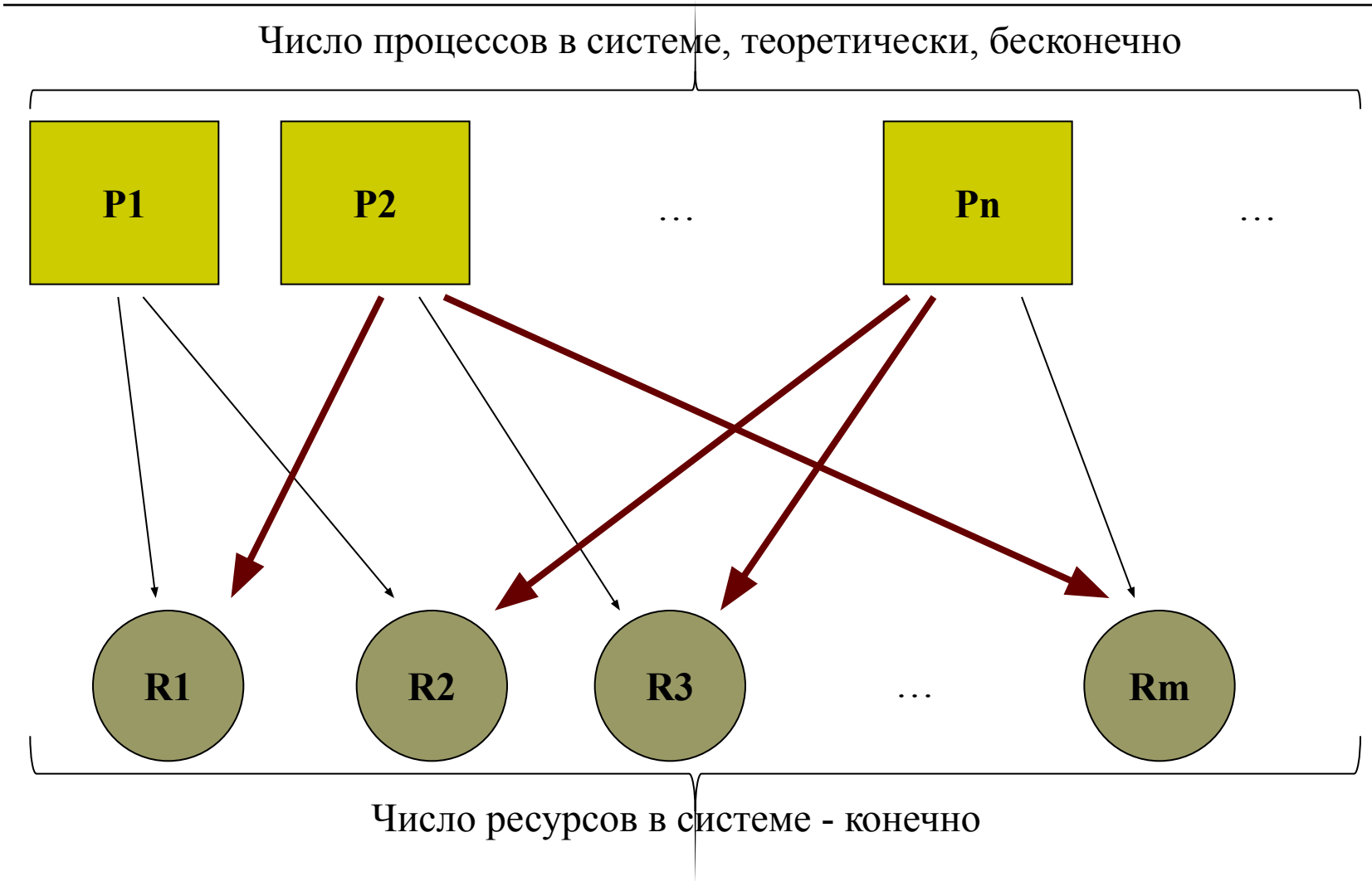
Последствия возникновения тупиков и бесконечного ожидания

- Зависание одного или более процессов
- Простой ресурсов системы
- Неэффективное управление ресурсами
- Снижение производительности работы системы
- Снижение эргономичности работы с системой с точки зрения пользователей
- Снижения доверия к системе со стороны пользователя
- Возможна потеря результатов работы системы
- Экономические убытки
- и т.д.

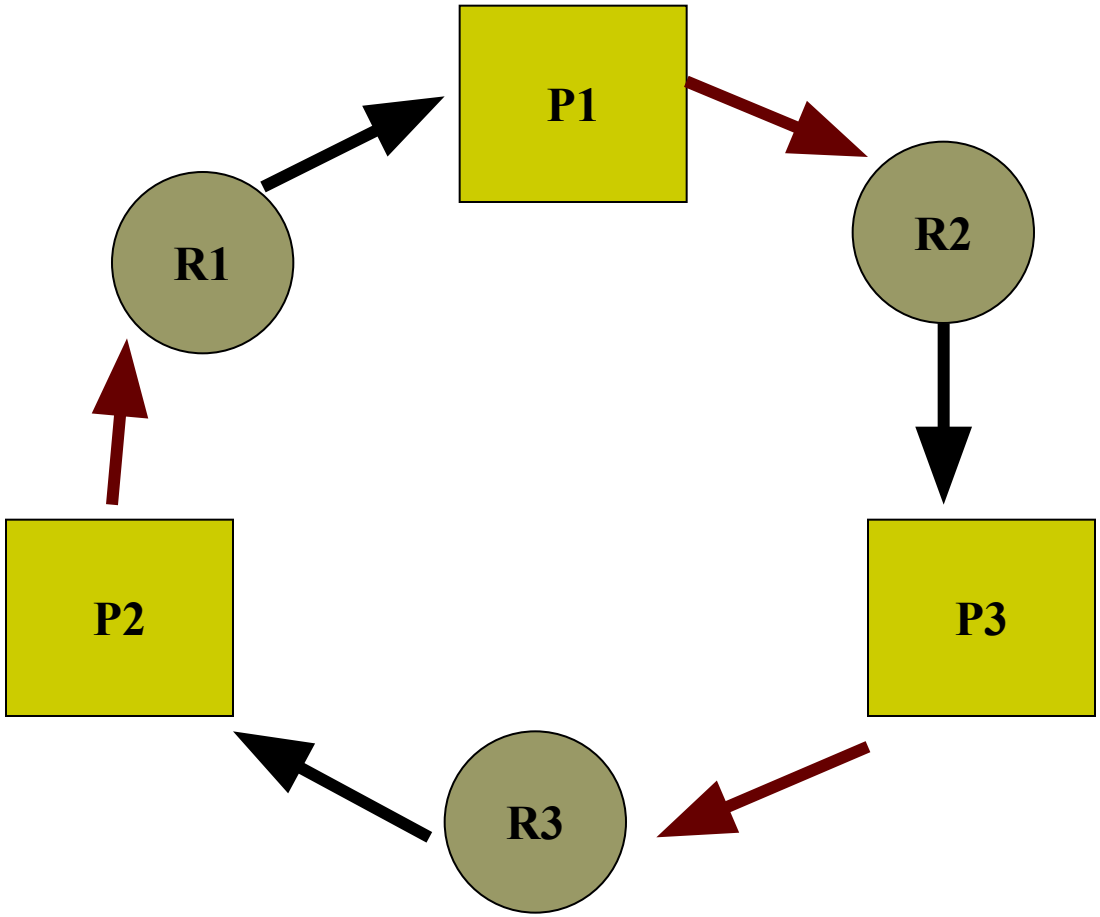
Необходимые условия возникновения тупиков

- **Условие взаимоисключения процессов при обращении к ресурсам эксклюзивного доступа**: процессы требуют предоставления им дополнительных ресурсов и права на управление ресурсами, которыми они владеют.
- **Условие ожидания**: процессы удерживают за собой уже выделенные ресурсы и в то же время для завершения работы требуют им выделения новых дополнительных ресурсов
- **Условие неперераспределимости ресурсов**: ресурсы, выделенные процессом нельзя отобрать у владельцев до тех пор, пока эти ресурсы не будут использованы процессами для завершения работы
- **Условие кругового ожидания**: В системе существует кольцо процессов, в которых каждый процесс удерживает за собой одно или несколько ресурсов необходимо следующему процессу в цепи

Наличие в системе ресурсов с эксклюзивным доступом (условия 1-3)



Условие кругового ожидания



Возникновение тупика в системе и необходимые условия

- **Тупик может возникнуть в системе только в том случае, когда <u>выполняются все четыре необходимых условия</u>**

Направления изучения тупиковых ситуаций

- **Предотвращение тупиков** – создание таких операционных систем, в которых возникновение тупика просто невозможно
- **Обходы тупиков** – создание таких ОС, которые могли бы предсказать возможный тупик и мягко обойти его
- **Обнаружение тупиков** – реализация средств ОС, позволяющих определять, что процессы находятся в состоянии тупика и/или бесконечного ожидания
- **Восстановление после тупиков** – реализация средств ОС, позволяющих разорвать тупик и восстановить работоспособность системы

Предотвращение тупиков

- Тупик может возникнуть в системе, если в ней выполняются **все** четыре необходимых условия
- Следовательно, если **нарушить хоть одно** необходимое условие, то **тупик просто не сможет возникнуть**

Методы предотвращения тупиков

- Нарушение условия взаимоисключения процессов при обращении к ресурсам с эксклюзивным доступом
- Нарушение условия ожидания
- Нарушение условия неперераспределимости ресурсов
- Нарушение условия кругового ожидания

Нарушение условия взаимоисключения процессов при обращении к ресурсам с эксклюзивным доступом

- **Запретить эксклюзивный доступ к ресурсам**
- Последствия:
  - Процессы не смогут бесконечно удерживать ресурсы – рано или поздно им придется их освободить
  - Невозможно будет реализовать некоторые системные механизмы, например, обслуживание прерываний устройств
- Этот подход никогда не может быть реализован

Нарушение условия ожидания

- Чтобы процессы не ожидали дополнительных ресурсов, необходимо выделить им все ресурсы заранее
- Не запускать процесс, если в данный момент система не может выделить ему все необходимые ресурсы

Нарушение условия ожидания: недостатки

- Процессы, которые требуют большого количества ресурсов, могут оказаться в ситуации **бесконечного ожидания**, т.к. процессы, которым нужно меньше ресурсов, будут «выхватывать» их у тяжеловесных процессов
- Процессы с меньшими потребностями в ресурсах будут проходить в системе быстрее. Такая политика распределения ресурсов дискриминационная
- Ресурсы могут простаивать при наличии большого количества легковесных процессов
- Система не сможет запустить процесс, которому необходимо больше ресурсов, чем есть в системе
- Перестройка ПО, так чтобы программы запрашивали все необходимые ресурсы в самом начале

Нарушение условия ожидания: последствия

- Полностью устраняется проблема тупиков
- Не решена проблема бесконечного ожидания
- Ресурсы используются неэффективно
- Снижение производительности системы
- Нарушение общепринятых принципов разработки программ

Нарушение условия неперераспределимости ресурсов

- Перед запуском процессу выделяются только те ресурсы, которые ему необходимы
- Если процессу нужны дополнительные ресурсы, и система может их ему предоставить, то они выделяются процессу, и он продолжает работу
- Если процессу требуются дополнительные ресурсы и система **не** может их предоставить, то у процесса отбираются **все ресурсы**, которыми он владел, а сам процесс приостанавливается
- Процесс будет возобновлен только в том случае, если системе удастся выделить ему те ресурсы, которыми он уже владел, и все ресурсы, которые он запрашивал

Нарушение условия неперераспределимости ресурсов: недостатки

- Если процесс был приостановлен и для его возобновления требуется много ресурсов, то он может оказаться в ситуации **бесконечного ожидания** – легковесные процессы могут перехватывать, необходимые ему ресурсы
- Процессы с меньшими потребностями в ресурсах будут проходить в системе быстрее. Такая политика распределения ресурсов дискриминационная
- Ресурсы могут простаивать, если в системе выполняется много легковесных процессов

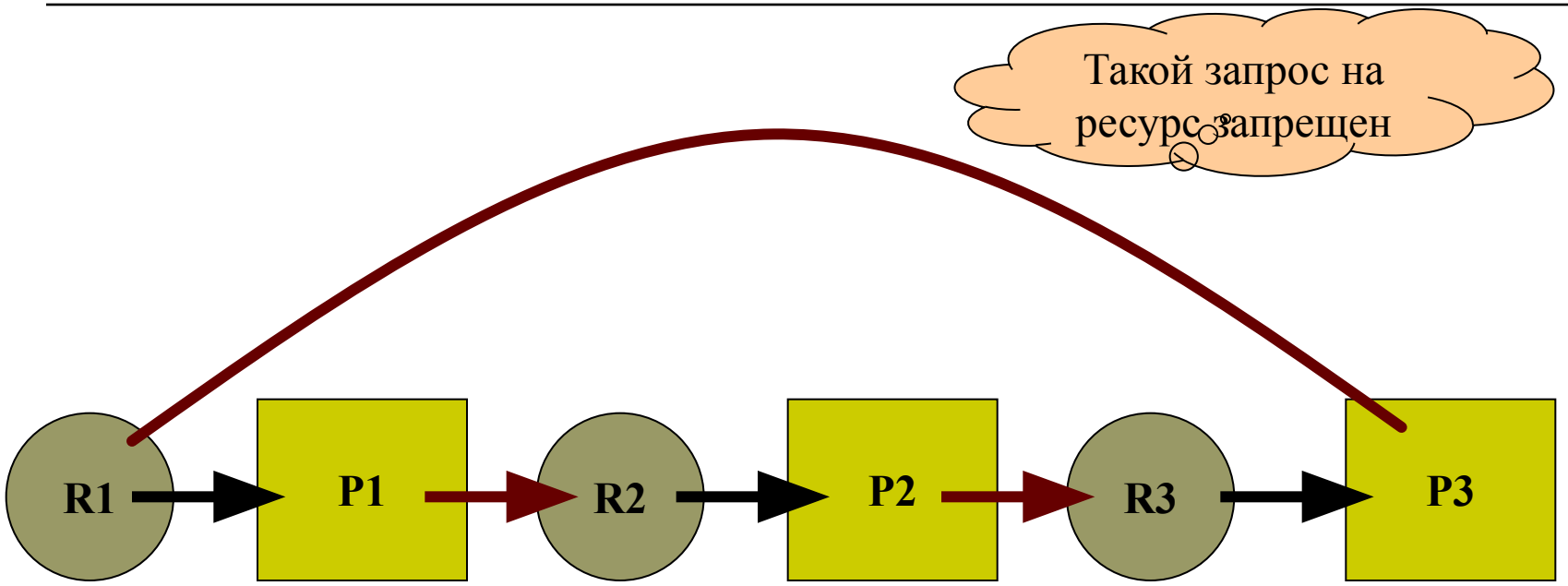
Нарушение условия неперераспределимости ресурсов: последствия

- Полностью решается проблема тупиков
- Не решена проблема бесконечного ожидания
- Ресурсы используются неэффективно
- Снижение производительности системы

Нарушение условия кругового ожидания

- Если организовать доступ процессов к ресурсам так, чтобы не могли образовываться циклические цепи ресурсов и процессов, то условие кругового ожидания может быть нарушено
- Необходимо ввести упорядоченность ресурсов по типам (от 1 до t)
- Если процесс владеет ресурсами типа i, то он может потребовать у системы выделения ресурсов только старших типов (от i+1 до t)
- Таким образом, процесс, находящийся в конце цепи и владеющий ресурсами типа t не может потребовать у системы выделения ресурсов типа 1 и, тем самым, замкнуть цепь

Нарушение условия кругового ожидания: схема



Нарушение условия кругового ожидания: недостатки и последствия

- ПО для таких систем должно разрабатываться с учетом ранжирования ресурсов – программы должны запрашивать ресурсы в строгом соответствии с порядком типов
- Если порядок типов изменится, то возникнет необходимость в перестройке всего программного обеспечения
- Если в системе появятся новые ресурсы (или будут удалены имеющиеся), то это приведет к изменению порядка типов ресурсов, а следовательно, к перестройке ПО
- Нарушаются общепринятые правила программирования

Предотвращения тупиков: общие выводы

- Полностью решается проблема тупиков
- В случае нарушения условия кругового ожидания решается также и проблема бесконечного ожидания
- Не удается избежать проблемы бесконечного ожидания
- Неэффективная политика использования ресурсов
- Простои системы
- Снижение производительности системы
- Увеличение времени отклика системы на запросы пользователей
- Нарушение общепринятых правил и технологий разработки программного обеспечения

Обходы тупиков

- В системе тупиковые ситуации потенциально возможны
- Система реализует специальные механизмы, которые позволяют
  - Предсказать возможность возникновения тупика
  - Мягко обойти тупиковую ситуацию

Алгоритм банкира

- **ОС** – «банкир», который владеет ресурсами
- **Процессы** – «заемщики», которые берут ресурсы «в кредит», т.е. обязаны вернуть их системе, когда они им станут не нужными

Правила алгоритма банкира

- Количество ресурсов в системе – конечно
- Количество процессов в системе – конечно
- Процессы сообщают системе о том, какое максимальное количество ресурсов им необходимо для завершения своей работы
- Система выделяет ресурсы процессам по одному
- Система обязуется в течении конечного времени удовлетворить потребность каждого процесса в ресурсах, чтобы он мог закончить свою работу
- Процесс обязуется, что вернет все ресурсы системе в течении конечного времени (по крайней мере, когда он закончится)
- Ни один процесс в системе не выполняется бесконечно

Надежное и ненадежное состояния системы

- Состояние системы называется **надежным**, если система располагает таким количеством ресурсов, чтобы выделить их одному процессу и тем самым удовлетворить его максимальную потребность в ресурсах, чтобы процесс смог завершиться
- Состояние системы называется **ненадежным**, если распределение любого количества ресурсов из имеющихся у системы любому процессу не приведет к удовлетворению максимальной потребности процесса в ресурсах и потенциально может стать **причиной тупиковой ситуации**

Общий принцип работы системы

- Система должна распределять ресурсы так, чтобы следующее ее состояние было надежным
- Если состояние системы ненадежное, то
  - Система может дождаться освобождения ресурсов каким-либо процессом
  - или Система должна констатировать возникновение тупика

Пример работы алгоритмы банкира в системе с одним типом ресурсов

- Пусть $t = 10$ – количество единиц ресурса в системе
- Пусть $m_1 = 5$, $m_2 = 3$, $m_3 = 7$ – максимальные потребности процессов P1, P2 и P3 в ресурсах
- Величины a_1, a_2, a_3 – количество ресурсов, распределенных соответствующим процессам на данном шаге работы системы
- Величины $b_i = m_i - a_i$, i=1,2,3 – количество ресурсов, в которых нуждаются процессы на данном шаге
- Величина $a = a_1+a_2+a_3$ – общее количество распределенных на данный момент ресурсов
- Величина $b = t - a$ – количество ресурсов, которыми обладает система на данный момент времени

Пример работы алгоритмы банкира в системе с одним типом ресурсов

- Состояние системы будет считаться надежным, если выполняется неравенство $b \geq b_i$ хотя бы для одного процесса i
- Состояние системы ненадежное, если все $b_i \geq b$

Пример надежного состояния

Процессы	m_i	a_i	b_i
P1	5	3	2
P2	3	2	1
P3	7	4	3
$t = 10, b = 1$		$a = 9$	

Пример ненадежного состояния

Процессы	m_i	a_i	b_i
P1	5	3	2
P2	3	1	2
P3	7	5	3
$t = 10$, $\underline{b = 1}$		$a = 9$	

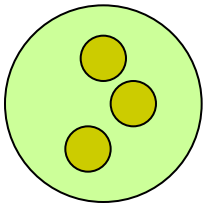
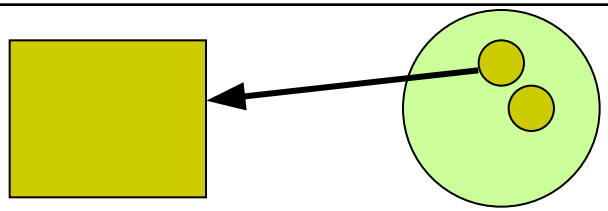
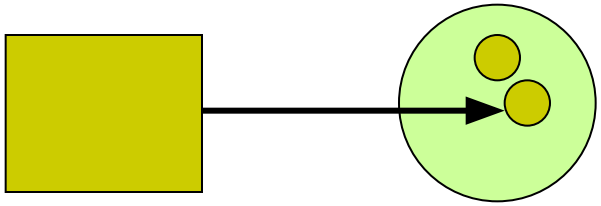
Обнаружение тупиков

- Построение и анализ графовых моделей, отражающих связи «процессы-ресурсы»
- Использование сканеров процессов (ОС Microsoft Windows)
- Использование процессов-демонов (ОС UNIX/Linux)

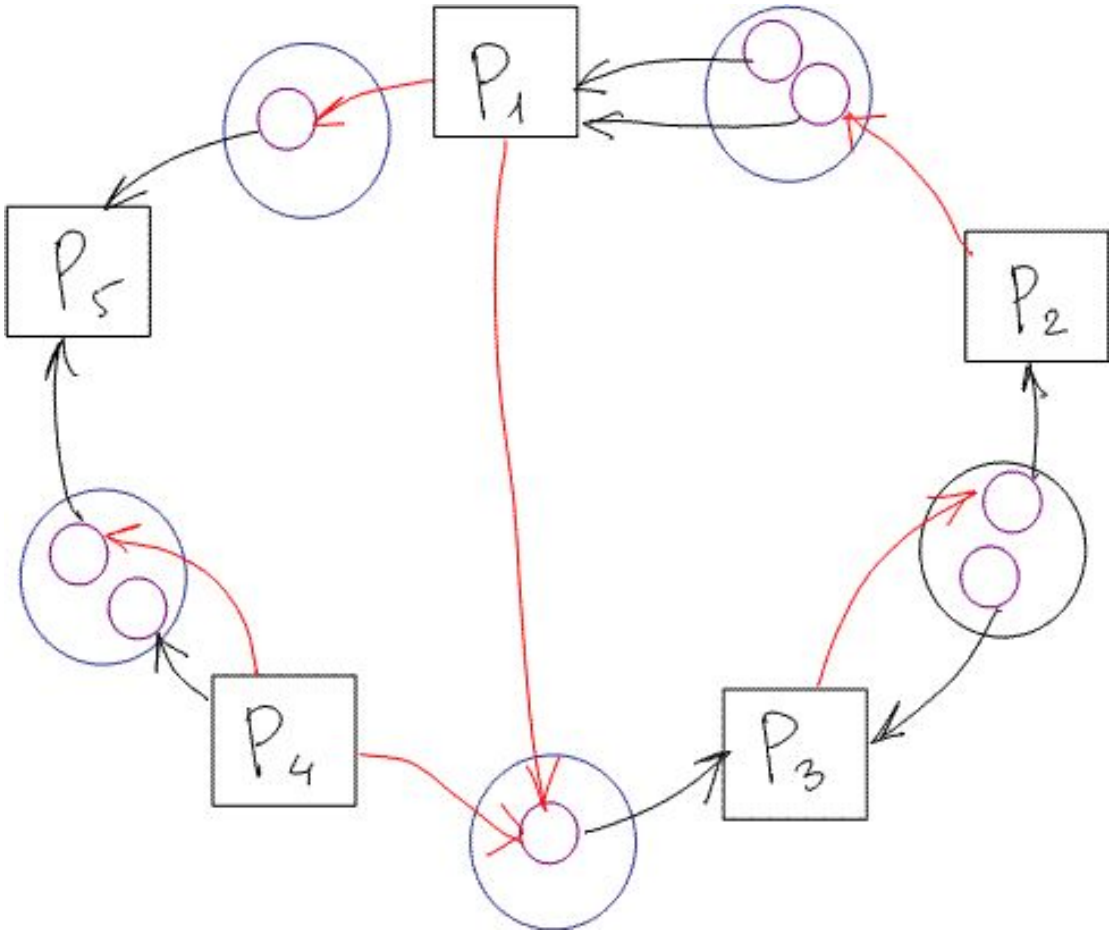
Графовые модели в обнаружении тупиков

- Строится граф, отражающий
  - процессы
  - ресурсы
  - связи, показывающий какие ресурсы уже распределены процессам
  - связи, показывающие на какие ресурсы претендуют процессы
- Для проверки наличия тупика граф редуцируется по специальным правилам
- Если после редукции графа не остается ни одной дуги, то тупика нет

Обозначения в графовой модели

	Процесс
	Ресурс определенного типа Маленькие кружочки показывают единицы ресурса
	Единица ресурса распределена процессу
	Процесс запрашивает выделение ресурса

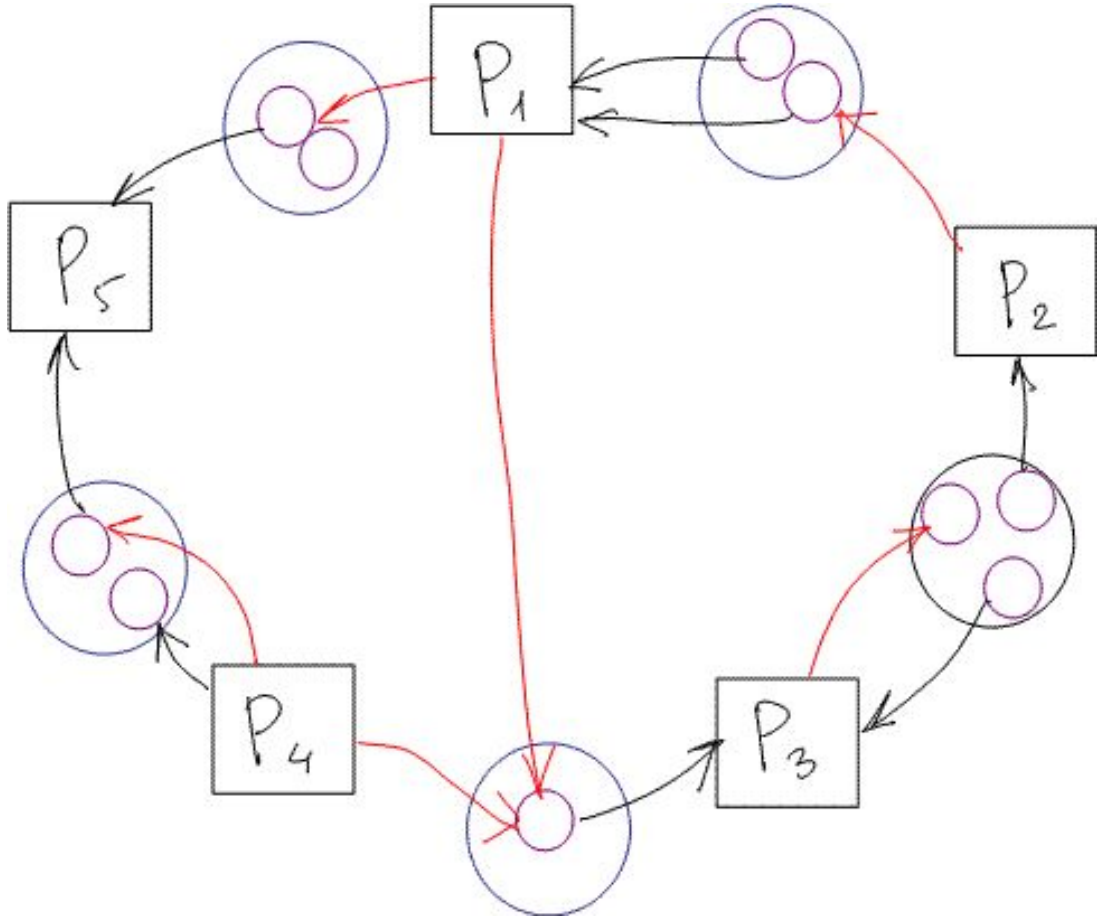
Пример графа



Правила редуцирования графа

- Удаляются все дуги запросов, которые могут быть в данный момент удовлетворены
- Если у процесса есть только дуги распределения ресурсов, то удаляются они
- Повторить до тех пор пока это возможно

Пример редуцируемого графа



Выход из тупика и восстановление работы системы

- Выход из тупика всегда сопровождается завершение хотя бы одного процесса, участвующего в тупике
- При завершении процесса его работа будет утеряна, возобновление работы процесса с точки зависания невозможно
- Экономически более выгодно потерять работу одного или даже нескольких процессов, чем позволить системе простаивать
- Современные ОС используют более мягкие средства выхода из тупика (например, повышение блокированного приоритета в Windows), но для этого требуется реализация очень сложной дисциплины планирования процессов и политики управления ресурсами