

## 9. Пространственные базы данных

9.1. Моделирование пространства

9.2. Способы представления пространственных объектов

9.3. Вычислительная геометрия

9.4. Хранение и извлечение пространственных объектов

9.5. R-деревья

9.6. Пространственные соединения

9.7. Применение: географические базы данных

# Пространственные базы данных

## Основные характеристики:

- Представление пространственных объектов в геометрическом пространстве (обычно двух- или трехмерном)
- Форма (фигура) и расположение – неотъемлемые компоненты
- Чаще всего у координат численные значения (с определенной дискретностью и нижней и верхней границами)
- Области применения: геоинформационные системы (ГИСы), системы автоматизированного проектирования (САПРы), графический интерфейс пользователя (GUI), виртуальная реальность, компьютерные игры, анимация и т.д.

# Моделирование пространства

## 1) Объектные (object-based) модели пространства

Компоненты пространственных объектов:

- Идентификационная информация
- Описание
- Пространственная протяженность

Классификация объектов на основе размерности:

*Примечание: зависит от приложения, работающего с объектом*

а) Объекты нулевой размерности = точки

- Формы нет или знание формы объекта не требуется
- Площадь объекта очень мала в сравнении со всем рассматриваемым пространством (например, города на картах, здания на картах, пересечения дорог, и т.д.)
- Могут появляться в зависимости от масштаба карта (город – точка на мелкомасштабной карте и двухмерный объект на крупномасштабной карте)

# Моделирование пространства

## б) Одномерные объекты = линейные объекты

- Например, дороги на картах
- Основной геометрический объект – ломаная линия. Состоит из конечного множества отрезков (или сегментов или ребер), таких что любая (за исключением двух точек – начала и конца ломаной линии) из конечных точек этих отрезков принадлежит двум отрезкам
- Простая ломаная линия – нет пересечений
- Замкнутая ломаная линия – точки начала и конца ломаной линии совпадают
- Любая кривая может быть представлена с заданной точностью ломаной линией

## в) Двухмерные объекты = объекты на плоскости

- Сущности-объекты имеют не нулевую площадь
- Основной геометрический объект – полигон (многоугольник). Полигон – область, задаваемая замкнутой ломаной линией
- Выпуклый полигон  $P$ : для любых  $A, B \in P$ , отрезок  $AB$  целиком в  $P$

## г) Трехмерные объекты = объемные объекты (полиэдры=многогранники)

# Моделирование пространства

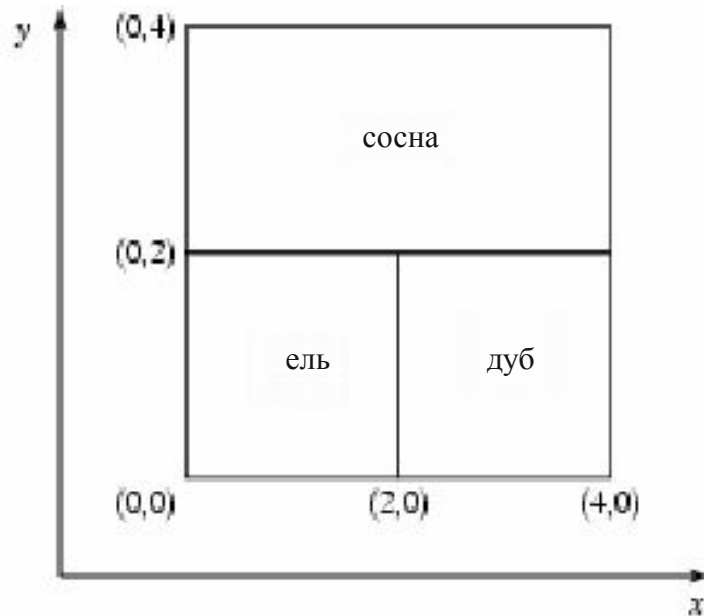
## 2) Полевые (field-based) модели пространства

- Пространственная информация задается непрерывным<sup>1</sup> полем значений, т.е. с помощью некой функции (например, по координатам  $x$  и  $y$ )
- Для каждой точки пространства может использоваться несколько атрибутов
- Примеры:
  - Температурное поле (температура в разных точках)
  - Атмосферное давление в разных точках
  - Высота над уровнем моря (на физических картах)
  - Значения уровня серого цвета на полутоновых цифровых изображениях
  - Значения красного, синего, зеленого компонентов на цветных (24-битных) изображениях

-----  
<sup>1</sup> – не в математическом смысле

# Моделирование пространства

Пример:



Объектная модель

ID	Древесная порода	Область
FS1	сосна	[(0,2), (4,2), (4,4), (0,4)]
FS2	ель	[(0,0), (2,0), (2,2), (0,2)]
FS3	дуб	[(2,0), (4,0), (4,2), (2,2)]

Полевая модель

$$f(x,y) = \begin{cases} \text{сосна} & 0 \leq x \leq 4; 2 < y \leq 4 \\ \text{ель} & 0 \leq x \leq 2; 0 \leq y \leq 2 \\ \text{дуб} & 2 < x \leq 4; 0 \leq y \leq 2 \end{cases}$$

# Способы представления пространственных объектов

## 1) Мозаичное (tessellation) представление

- Разбиение на ячейки(соты) (возможны разные формы ячеек)
- Фиксированные ячейки: одинаковые ячейки (сетка прямоугольных координат)
- Произвольные ячейки: размеры и формы ячеек различаются между собой
- Мозаика с регулярной/нерегулярной структурой
- По умолчанию:  $N \times M$  прямоугольных (обычно квадратных) ячеек, которые называются пикселями
- Естественное (дискретное) представление полевых данных
- В случае объектных данных: один пиксел для точки, набор (множество) пикселов для ломаной линии или полигона
- Более точное представление (с более мелкими ячейками) потребует больше места для хранения; обработка займет больше времени

# Способы представления пространственных объектов

## 2) Векторное представление

- Естественно для объектных моделей пространства
- Базисные элементы (примитивы): точки и ребра
- Полигон задается множеством точек, аналогично ломаная линия
- $2^n$  возможных описания полигона с  $n$  вершинами (выбор стартовой вершины, обход по/против часовой стрелки)
- Область – множество полигонов
- Представление может дополняться ограничениями (например, только простые<sup>1</sup> полигоны)
- Векторное представление полевых данных; цифровые модели местности (**digital elevation models**):
  - Значения задаются только для подмножества точек
  - Значения в остальных точках интерполируются
  - Пример: триангулированные неравномерные сети (triangulated irregular networks)

---

<sup>1</sup> – граница которого не пересекается сама с собой



# Способы представления пространственных объектов

## 3) Полуплоскостное (half-plane) представление

- Единственный используемый примитив: полуплоскость (см. математическое определение)
- Солидный математический базис
- Полуплоскость в  $d$ -мерном пространстве задается неравенством:  $a_1x_1 + a_2x_2 + \dots + a_dx_d + a_{d+1} \leq 0$
- Выпуклый полигон – пересечение конечного числа полуплоскостей
- Полигон – объединение конечного числа выпуклых полигонов
- Отрезок (ребро) линии – одномерный выпуклый полигон (пересечение двух лучей или полупрямых)
- Ломаная линия – объединение нескольких отрезков

# Вычислительная геометрия

*Алгоритмическая техника для выполнения операций в пространственных базах данных*

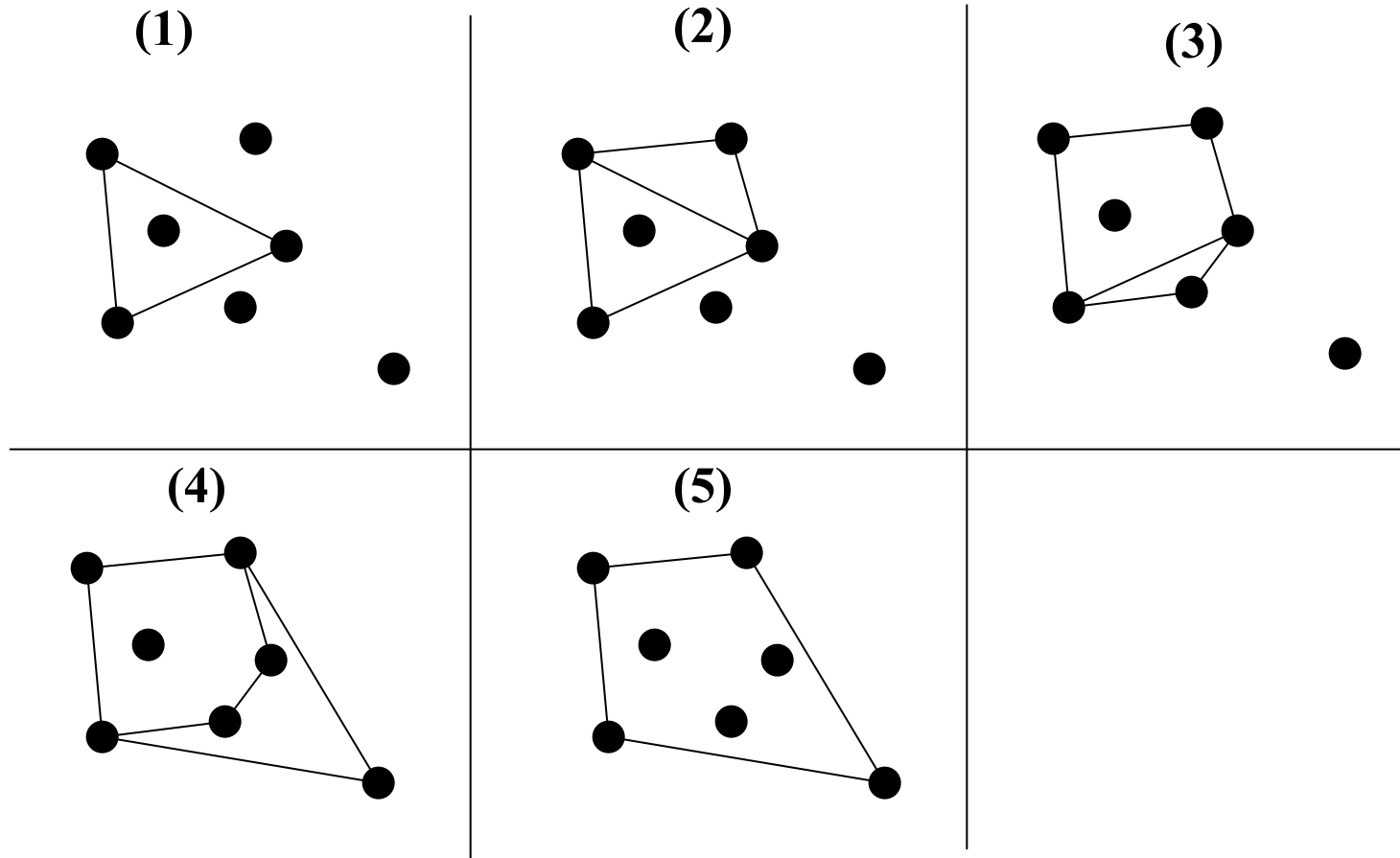
## 1) Инкрементные алгоритмы

- Решить задачу для небольшого подмножества входных данных (точек), затем решить задачу для начального множества плюс одна точка из остающихся и т.д. пока все точки не будут рассмотрены
- Пример: нахождение выпуклой оболочки для множества точек  
Простейший метод с временной сложностью  $O(n^2)$ :
  - Построить выпуклую оболочку  $H_3$  – для первых трех точек
  - Для каждой из остальных точек  $\{p_i\}, i > 3$ :
    - Если  $p_i$  внутри  $H_{i-1}$ , то  $H_i = H_{i-1}$  (проверка «внутри»: при обходе  $H_{i-1}$  по часовой стрелке,  $p_i$  остается справа)
    - Иначе, добавить  $p_i$  к  $H_{i-1}$ , возможно удалив старые точки (для  $p_i$  найти соседние такие точки  $p_a, p_b$ , чтобы угол между отрезками  $(p_a, p_i)$  и  $(p_b, p_i)$  был наибольшим)

Оптимальный алгоритм:  $O(n \log n)$ , используется предварительная сортировка точек

# Вычислительная геометрия

Иллюстрация к инкрементному нахождению выпуклой оболочки:



# Вычислительная геометрия

## 2) Стратегия «разделяй и властвуй»

- «Разделяй»: задача рекурсивно разбивается на несколько легко решаемых подзадач
- «Властвуй»: объединение снизу-вверх всех решений в одно общее решение
- Аналогия: бинарное дерево (см.следующий слайд)
- Пример: пересечение полуплоскостей

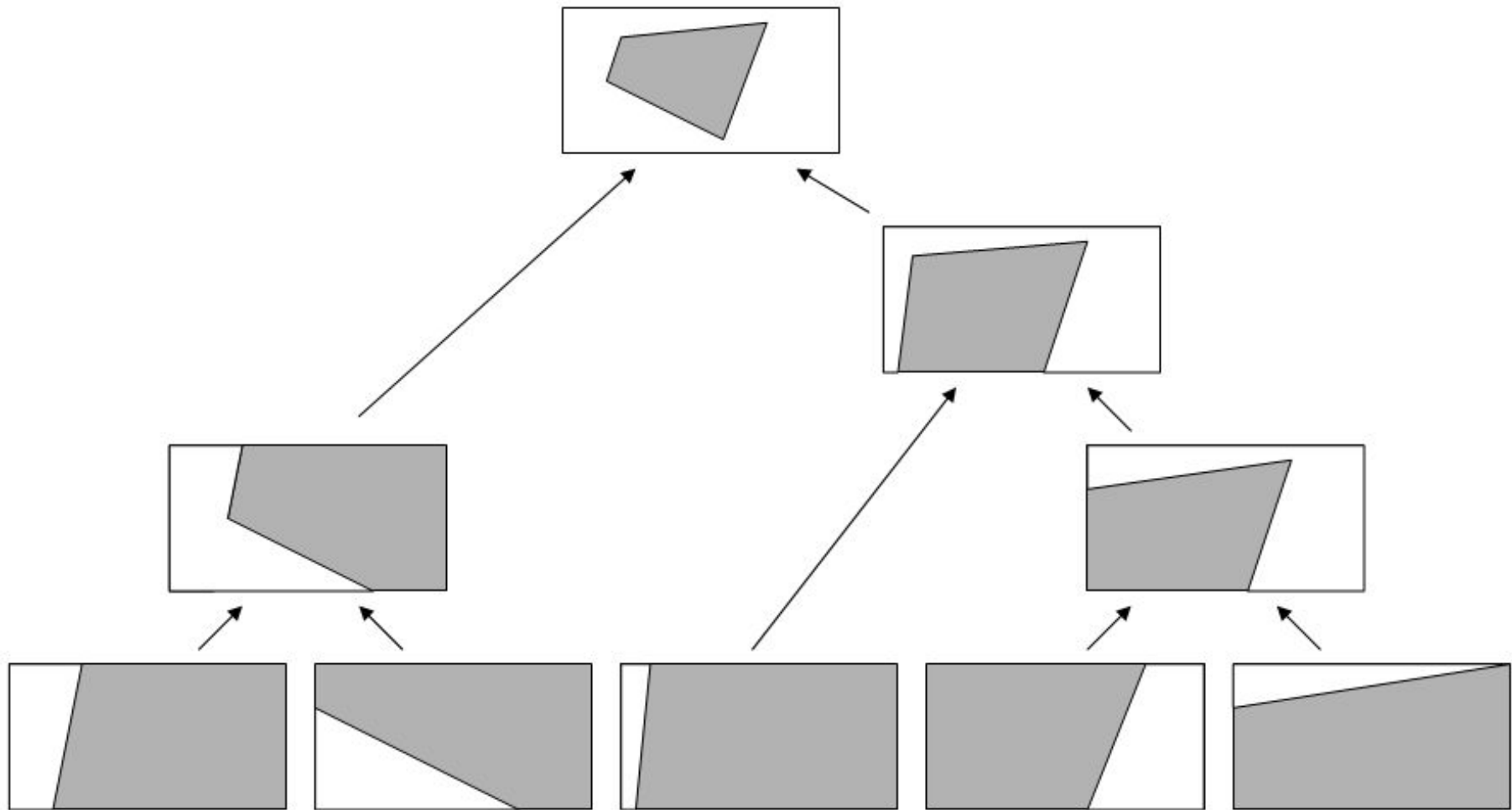
Для простоты считаем, что конечный результат – выпуклый полигон внутри прямоугольника  $R$

- Исходное множество из  $n$  полуплоскостей рекурсивно разбивается пополам до тех пор пока мы не получим  $n$  отдельных полуплоскостей (это дает нам бинарное дерево)
- Для каждой из полуплоскостей определяем ее пересечение с  $R$  (каждое такое пересечение - выпуклый полигон)
- Объединение результатов: рекурсивно снизу-вверх определяем попарные пересечения полигонов

Сложность:  $O(n \log n)$ , т.к. сложность нахождения пересечения выпуклых полигонов -  $O(n)$

# Вычислительная геометрия

Пересечение полуплоскостей с помощью метода «разделяй и властвуй»:



# Вычислительная геометрия

## 3) Метод заметающей прямой (sweep-line)

- Разложение пространства на вертикальные полосы, таким образом, чтобы линии, разделяющие полосы, давали нужную информацию для решения проблемы
- Процесс «заметания» заключается в перемещении вертикальной прямой слева направо, с остановками на границах вертикальных полос и сохранения/обновления информации необходимой для решения
- Используются две структуры данных:
  - Статус заметающей прямой: содержит объекты, связанные с текущей позицией прямой
  - Перечень событий: содержит границы полос, известные заранее или определяемые динамически
- Пример: найти все попарные пересечения множества прямоугольников, стороны которых параллельны координатным осям
  - Время работы в наихудшем случае  $O(n^2)$
  - Метод на основе заметающей прямой со сложностью прямо пропорциональной количеству находимых объектов (методы с такой сложностью называются output-sensitive methods)

# Вычислительная геометрия

## Алгоритм нахождения пересекающихся прямоугольников:

**begin**

Отсортировать  $2n$  нижние и верхние  $x$ -координаты  
прямоугольников и поместить результат в  $E$

Пусть  $L = \emptyset$

**while** ( $E \neq \emptyset$ ) **do**

**begin**

$p = \text{Min}(E)$

Извлечь (удалить)  $p$  из  $E$

**if**  $p$  - нижняя граница прямоугольника  $r$  **then**

**begin**

Найти (и выдать как результат) все прямоугольники из  $L$ ,  
которые пересекаются с  $r$

Вставить  $r$  в  $L$

**endif**

**if**  $p$  - верхняя граница прямоугольника  $r$  **then**

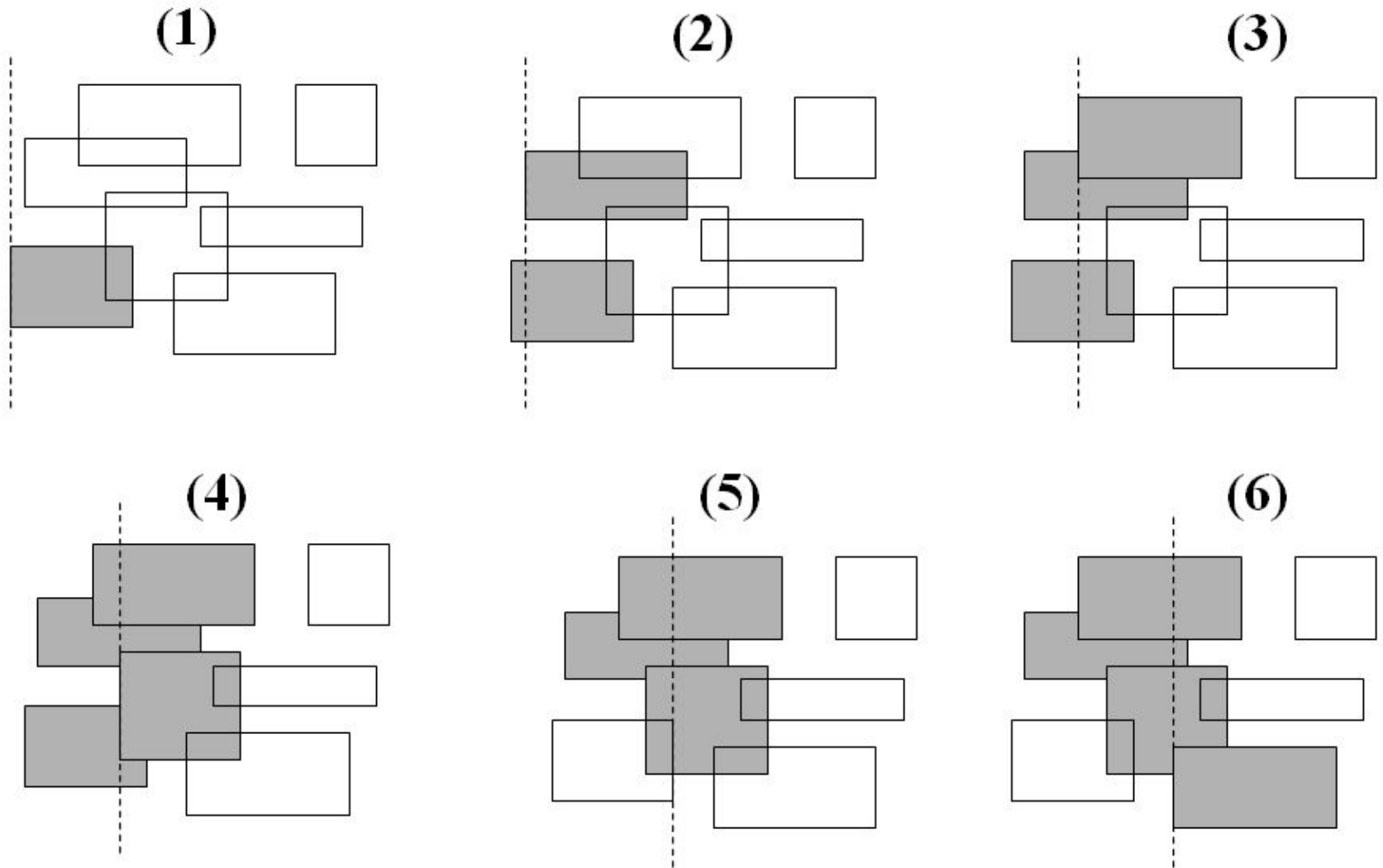
Удалить  $r$  из  $L$

**endwhile**

**end**

# Вычислительная геометрия

Метод заметающей прямой для нахождения пересекающихся прямоугольников:





# Вычислительная геометрия

Типичные задачи вычислительной геометрии:

- Расположение точки относительно полигона (внутри или вне)
- Пересечение отрезков прямых
- Пересечение ломаных линий
- Пересечение полигонов
- Отсечение с помощью прямоугольника (отсечение объекта(-ов) вне границ прямоугольного окна)
- Разбиение полигона на треугольники (триангуляция)
- Разбиение полигона на трапеции
- Представление полигона в виде нескольких выпуклых полигонов

Ограничение, накладываемые на объекты, упрощают алгоритмы; например, в случае полигонов:

- Простой полигон: граница не пересекается сама с собой
- Монотонный полигон: граница составлена из двух монотонных цепочек вершин: верхней и нижней цепочек вершин полигона (цепочка вершин монотонна, если любая вертикальная линия пересекает образуемую ломаную линию не более одного раза)
- Выпуклый полигон (было дано ранее)

# Хранение и извлечение пространственных объектов

## Общие замечания:

- Работа с произвольными фигурами затруднительна  $\Rightarrow$  Рассматривают минимальные ограничивающие прямоугольники (далее MBR<sup>1</sup>): наименьший прямоугольник, охватывающий геометрический объект на плоскости, со сторонами, параллельными координатным осям
- Значения координат отображаются на интервал  $[0, 1)$ ; пространство – гиперкуб, обозначаемый  $E^k$

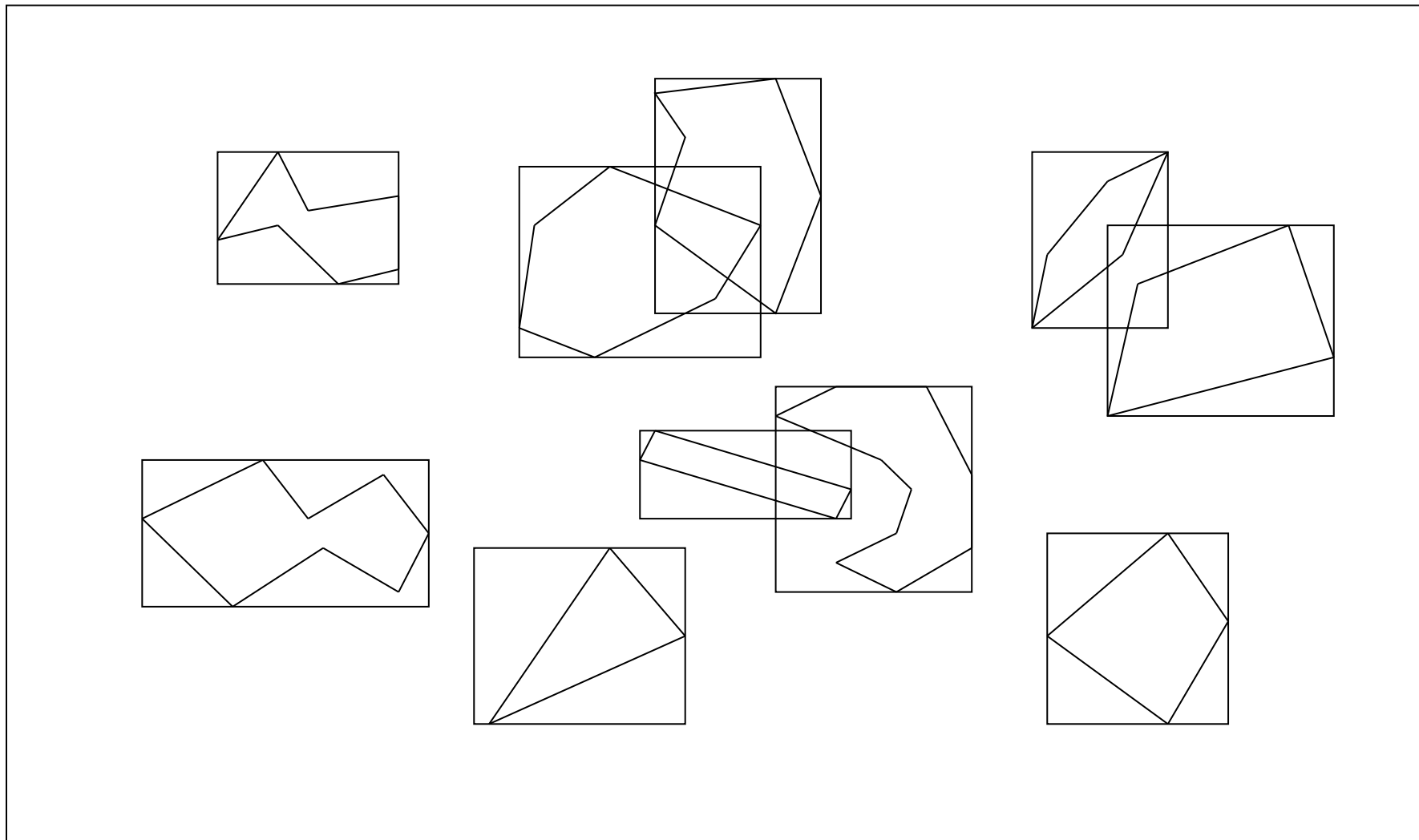
## Факторы, влияющие на производительность:

- Выбранная структура данных
- Размерность пространства
- Распределение объектов в пространстве:
  - Плотность в точке  $P$  = число прямоугольников, содержащих  $P$
  - Глобальная плотность = максимум по локальным плотностям

-----  
<sup>1</sup> - **M**inimum **b**ounding **r**ectangle или сокращенно MBR; другое название – ограничивающие блоки (bounding box)

# Хранение и извлечение пространственных объектов

Минимальные ограничивающие прямоугольники :



# Хранение и извлечение пространственных объектов

Виды запросов к пространственным объектам:

- 1) Запросы по точному совпадению: не типичны для пространственных объектов, за исключением операций вставки
- 2) Запрос по точке: для заданной точки  $P \in E^k$  найти все прямоугольники  $R$  такие, что  $P \in R$
- 3) Пересечение прямоугольников: для заданного прямоугольника  $S \subseteq E^k$  найти все прямоугольники  $R$  такие, что  $S \cap R \neq \emptyset$
- 4) Поиск «включающих» прямоугольников: для заданного прямоугольника  $S \subseteq E^k$  найти все прямоугольники  $R$  такие, что  $S \subseteq R$  ( $R$  включает в себя  $S$ )
- 5) Поиск прямоугольников «внутри»: для заданного прямоугольника  $S \subseteq E^k$  найти все прямоугольники  $R$  такие, что  $R \subseteq S$  ( $R$  внутри  $S$ )
- 6) Запрос по объему: по заданным  $v_1, v_2 \in (0, 1)$ ,  $v_1 \leq v_2$  найти все прямоугольники с объемом (площадью) в интервале  $[v_1, v_2]$
- 7) Пространственное соединение: для двух множеств  $k$ -мерных прямоугольников найти все пары, удовлетворяющие заданному условию соединения (пересечение, включение, нахождение внутри)

# Хранение и извлечение пространственных объектов

## Представление пространственных объектов на основе трансформации координат

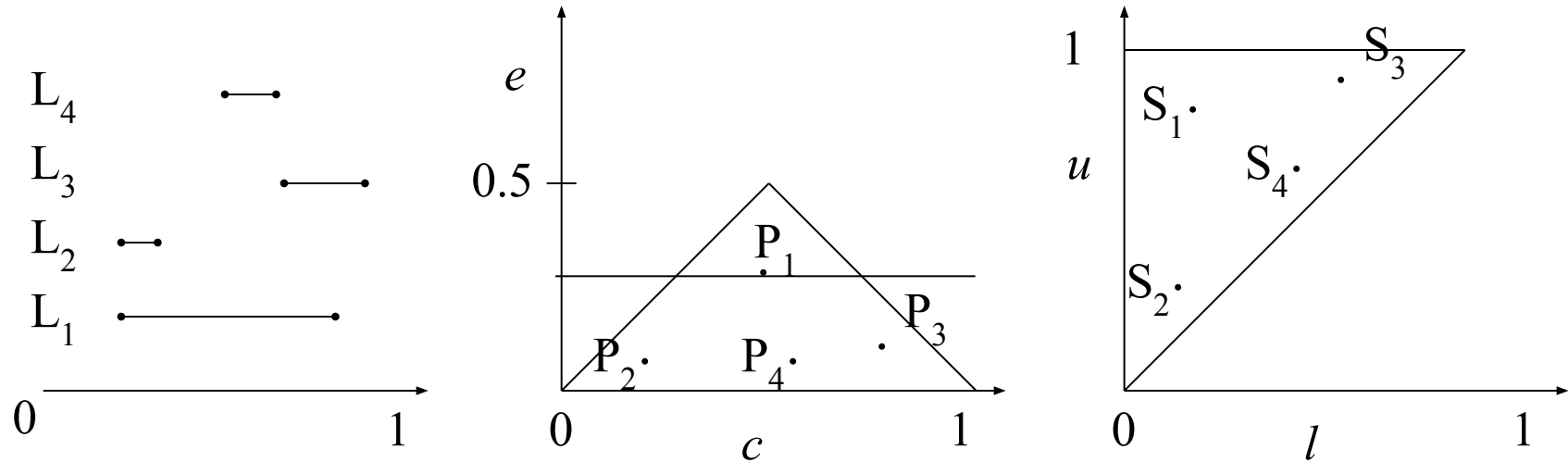
- $k$ -мерный прямоугольник можно представить  $2k$ -мерной точкой
- Возможные варианты (на примере двухмерного пр-ва):
  - a)  $(c_x, c_y, e_x, e_y)$ , где  $(c_x, c_y)$  – центральная точка, а  $e_x$  и  $e_y$  – расстояния от центральной точки до сторон прямоугольника
  - b)  $(l_x, l_y, u_x, u_y)$ , где  $(l_x, l_y)$  и  $(u_x, u_y)$  – нижняя вершина слева и верхняя вершина справа соответственно
- Достоинство варианта a): координаты расположения  $c_x$  и  $c_y$  отличны от координат протяженности  $e_x$  и  $e_y$

### Частный случай:

- Одномерное пространство  $[0, 1)$
- Прямоугольник = отрезок  $\subseteq [0, 1)$
- Варианты представления:
  - a)  $(c, e) = (\text{центр, половина длины})$
  - b)  $(l, u) = (\text{начальная точка, конечная точка})$

# Хранение и извлечение пространственных объектов

Пример представления (для одномерного пр-ва):



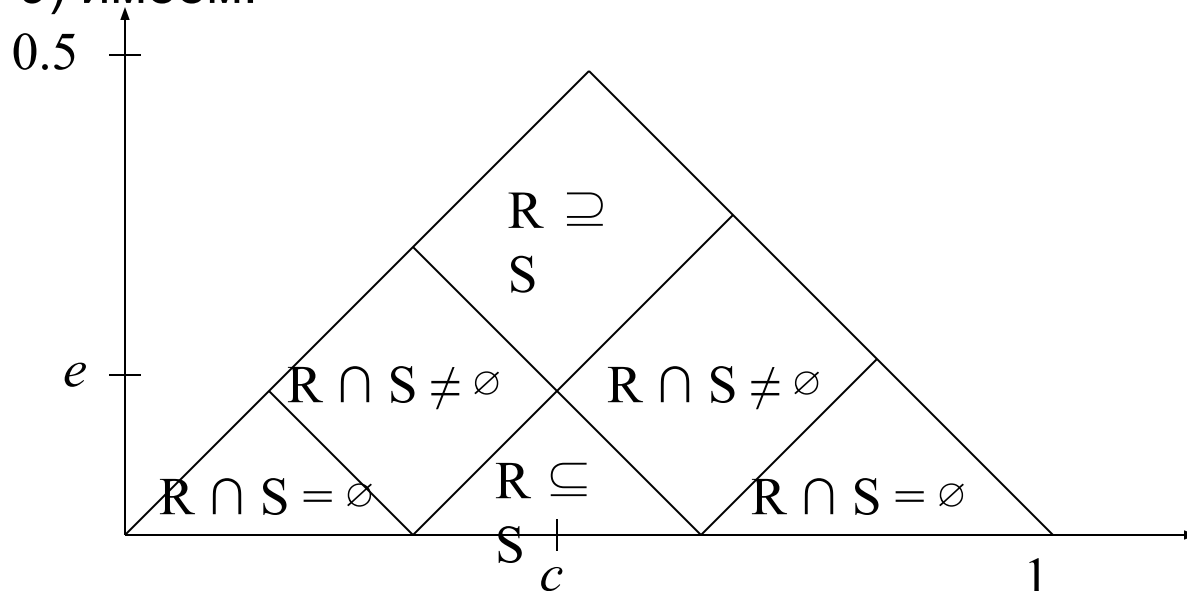
Замечания:

- В случае применения методов доступа к точечным данным возникает проблема «пустых треугольников» (или «мертвых регионов»), см. рисунок выше
- Вариант представления с координатами центра и протяженности может быть улучшен, если нам известен верхний предел размера стороны прямоугольника (тогда, например, в одномерном случае можно рассматривать только область  $[0, \text{limit}/2]$ ); в этом случае «живое» пространство будет трапецией, а «мертвые» треугольники сравнительно небольшими

# Хранение и извлечение пространственных объектов

Ответы на запросы:

- Простые геометрические вычисления укажут на области, соответствующие тому или иному типу запросов
- Пример: одномерные прямоугольники (=отрезки) могут быть представлены точками в двумерном пространстве (с помощью координат центра и протяженности); для прямоугольника в запросе  $S = (c, e)$  имеем:

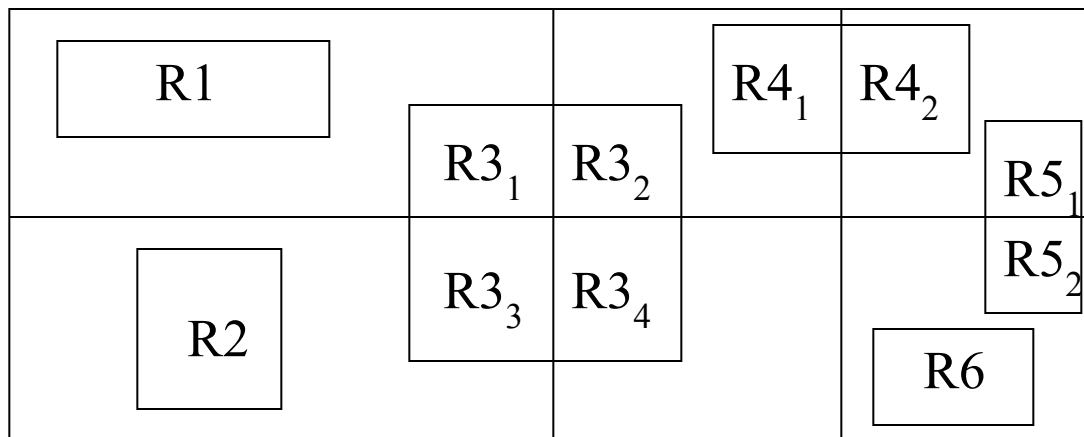


- Недостаток: близко расположенные, но различного объема, прямоугольники могут располагаться довольно далеко друг от друга в двумерном пространстве

# Хранение и извлечение пространственных объектов

## Представление пространственных объектов на основе отсечения

- Пространство разбивается на непересекающиеся прямоугольные области (также как и в большинстве методах доступа к точечным данным, см. тему 8)
- Расположение прямоугольника  $R$  может быть следующим:
  - $R$  внутри одной из областей: простая обработка (как и в методах доступа к точечным данным)
  - $R$  пересекается как минимум с двумя областями
- В случае «отсечения»: каждая область пересечения ( $R$  с областями на которые разбито пр-во) рассматривается (в том числе хранится) как самостоятельный прямоугольник, но при этом все отсеченные части указывают на один и тот же изначальный объект





# Хранение и извлечение пространственных объектов

## Достоинства:

- Отсечение может осуществляться практически напрямую с помощью любого метода доступа к точечным данным
- Точки и прямоугольники могут храниться в одном и том же месте

## Недостатки:

- Повышенные требования к пространству (многочисленные указатели на один и тот же объект)
- Дополнительные издержки при операциях вставки и удаления
- В случае высокой глобальной плотности необходимы избыточные страницы

## Производительность:

- Запросы по точному совпадению, по точке и поиск включающих прямоугольников потребуют доступа только к одной странице (при условии, что нет переполнения)
- Пересечение прямоугольников и поиск прямоугольников «внутри» может потребовать просмотра всех отсеченных частей прямоугольника запроса; количество false drops может быть большим

## Пример реализации:

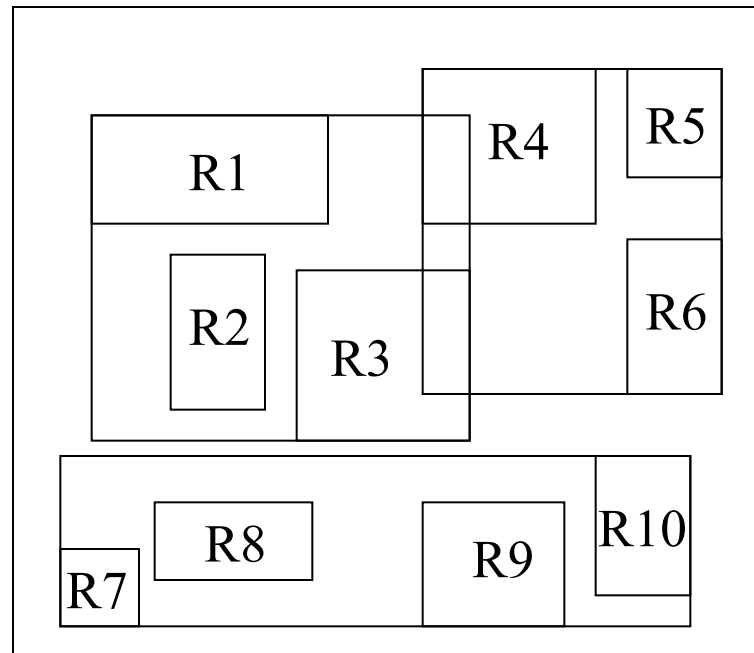
- R+-дерево [3]: сбалансированное внешнее (т.е. данные об объектах хранятся только в листьях) дерево; похоже на R-дерево (см.далее)

# Хранение и извлечение пространственных объектов

## Представление пространственных объектов на основе перекрывающихся областей

- Каждый прямоугольник представлен в базе данных только один раз (в отличие от R+-дерева)
- Прямоугольники сгруппированы по дисковым страницам
- Каждая область (образующая группу прямоугольников) задается минимальным ограничивающим прямоугольником
- Области могут перекрываться

Пример:



# Хранение и извлечение пространственных объектов

## Потенциальные недостатки:

- Высокая степень перекрытия ухудшает производительность
- Степень перекрытия MBR'ов может быть много выше степени перекрытия рассматриваемого множества прямоугольников
- Запрос по точному совпадению, вставка и удаление могут потребовать доступа к более чем одной странице
- Пересечение прямоугольников и поиск прямоугольников внутри могут требовать доступа к одним и тем же страницам, при этом поиск прямоугольников внутри дает как правило много меньшее количество результатов (т.к. каждый прямоугольник внутри также является пересекающимся)

## Обобщение:

- Области (минимальные ограничивающие прямоугольники) могут быть сами сгруппированы, образуя прямоугольники более высокого уровня
- Это позволяет построить древовидную структуру

# R-деревья

## Индекс на основе перекрывающихся областей - R-дерево [4] (rectangle tree):

- Сбалансированная динамическая внешняя древовидная структура, где узлы – страницы
- Хранит как точки так и прямоугольники
- Широко используется; например, в пространственном модуле Oracle

### Виды узлов:

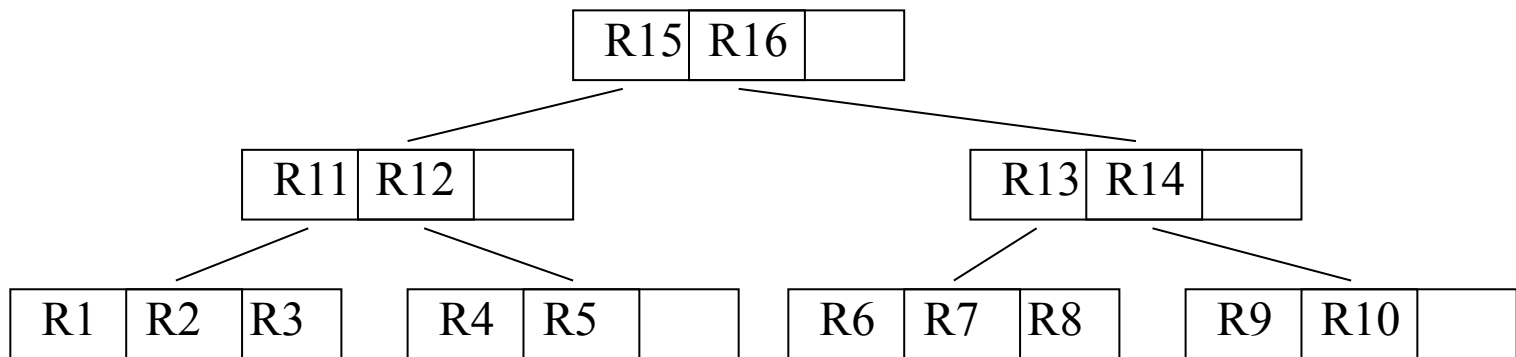
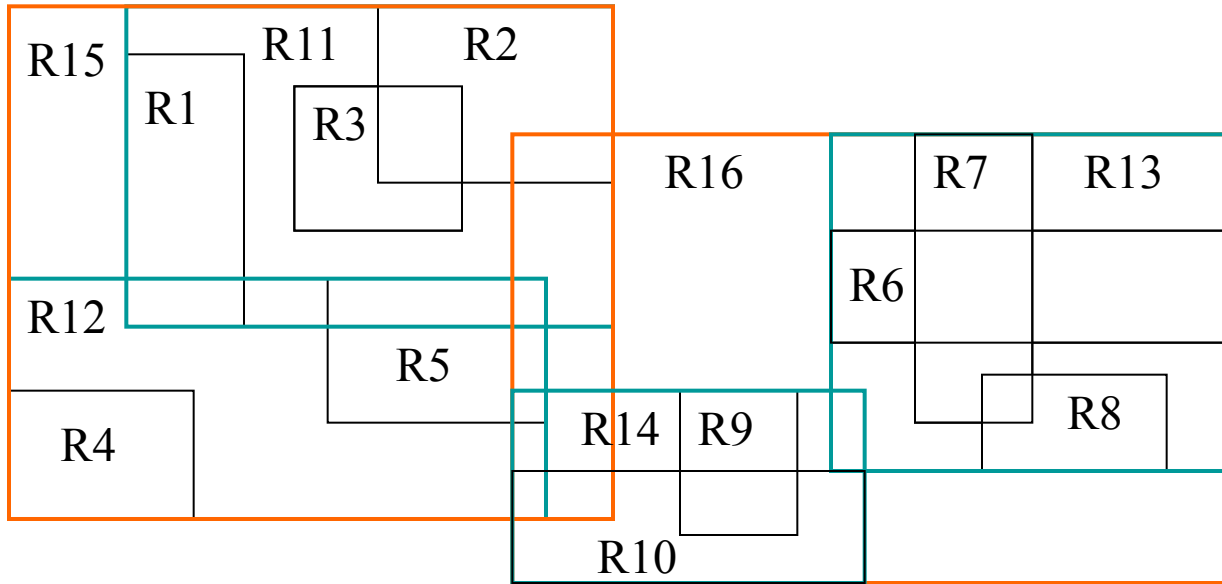
- Лист содержит пары (R, TID), где R – MBR пространственного объекта, а TID – указывает на точное описание объекта
- Внутренний узел содержит пары (R, ptr), где R – MBR прямоугольников в узле-потомке, а ptr – указатель на узел-потомок

### Свойства:

- Прямоугольники на пути от вершины дерева к листьям вложены друг в друга (т.е. прямоугольник узла-потомка внутри прямоугольника узла-родителя)
- Какие-либо ограничения на перекрытие прямоугольников (за исключением только что упомянутого) отсутствуют, но (!) количество перекрытий должно минимизироваться
- При емкости страницы в  $M$  записей, для количества записей на одной странице определяется нижняя граница -  $m \leq M/2$
- Для  $N$  записей, высота (дерева)  $\leq \lceil \log_m N \rceil - 1$  и количество узлов  $\leq N/(m - 1)$

# R-деревья

Пример R-дерева:



# R-деревья

Обработка запросов:

- a) Запрос по точке: найти объекты, содержащие заданную точку  
Начиная с корня, рекурсивно просматриваем все поддеревья, MBR'ы которых содержат данную точку. Дойдя до уровня листьев, получим описания объектов, каждый из которых необходимо проверить на предмет содержания точки
- b) Запрос на пересечение: найти объекты, пересекающиеся с заданным прямоугольником  
Обработка такая же как и в а), но проверяется не содержание точки, а пересечение объектов

Выполнение других типов запросов происходит похожим образом

Производительность:

- Гарантия отсутствует, т.к. может требоваться просмотр значительного количества узлов дерева
- Степень перекрытия MBR'ов, описываемых во внутренних узлах, определяет производительность
- Самая важная роль при минимизации степени перекрытия у операции вставки

# R-деревья

Вставка в R-дерево:

1. Используя процедуру **ChooseLeaf** (см.след.слайд), найти лист L для вставляемого прямоугольника R
2. Если в L есть место для R, осуществить вставку; иначе, вызвать процедуру **SplitNode**, возвращающую листья L и LL, которые совместно содержат R и старые объекты из L
3. Вызвать процедуру **AdjustTree** с входными параметрами L и возможно LL. Корректировка (дерева) ведет к увеличению ограничивающих прямоугольников в узлах-родителях, и возможно вызовет расщепление узлов
4. Если корневой узел был расщеплен на два, то создать новый корневой узел, узлами-потомками которого будут эти два образовавшихся узла

# R-деревья

## Процедура **ChooseLeaf**:

1. Начать с корневого узла (= N)
2. Если N является листом, вернуть N
3. Просмотреть пары (указывающие на поддеревья) в узле N. Выбрать ту пару, чей MBR при включении прямоугольника R увеличится наименьшим образом (в идеале, вообще не изменится). Пусть F указатель определенной таким образом пары. В спорных случаях (когда приращение MBR'ов одинаково) – выбирать прямоугольник с наименьшей площадью.
4. Переопределить N как узел на который указывает F и продолжить с шага 2



# R-деревья

## Расщепление:

Наиболее сложная задача (экспоненциальное число альтернатив)

- Происходит в листе, но может распространиться наверх
- Задача: минимизировать степень перекрытия MBR'ов
- Эвристическая процедура: попытаться минимизировать общую площадь двух прямоугольников, образующихся в результате расщепления
- Два способа (второй на след. слайде)

## **SplitNode** (квадратичное время):

1. Найти два прямоугольника  $R_1, R_2$ , которые в случае помещения в один и тот же узел, приведут к наибольшей потере пространства, т.е. для которых  $\text{Area}(\text{MBR}(R_1, R_2) - \{R_1, R_2\})$  максимальна.  $R_1$  и  $R_2$  будут «ядрами» двух формируемых групп прямоугольников
2. Остановить процедуру, если все прямоугольники распределены по своим группам. Если все остающиеся прямоугольники должны быть отнесены к одной из групп (для того чтобы было выполнено условие минимально допустимого количества записей в данной группе, см. слайд 246), то поместить прямоугольники в эту группу и остановиться
3. Для каждого из остающихся прямоугольников вычислить  $d_1$  = увеличение площади MBR, если прямоугольник отнесен к группе 1, и  $d_2$  (если к группе 2). Выбрать прямоугольник с наибольшим значением  $|d_1 - d_2|$ , и вставить его в группу для которой d-значение минимально. Перейти к шагу 2.

# R-деревья

Этапы, требующие нелинейного времени, в процедуре выше:

- Выбор «ядер» (первых элементов в группах)
- Выбор следующего прямоугольника (шаг 3)

**SplitNode** (линейное время):

- Выбор первых элементов для групп: для каждого измерения найти два прямоугольника, которые имеют наибольшую нижнюю границу (по этому измерению) и наименьшую верхнюю границу соответственно; определить максимум (по всем измерениям  $d$ ) следующего выражения:

$$\frac{|Max(\text{нижн. граница } R_1 \text{ по измерению } d) - Min(\text{верх. граница } R_2 \text{ по измерению } d)|}{\text{длина всего рассматриваемого множества прямоугольников по измерению } d},$$
 другими словами, будет выбрана пара прямоугольников с наибольшим нормализованным расстоянием между нижней и верхней гранями

- Выбор следующего прямоугольника: выбирать любой из остающихся

Квадратичная процедура работает до определенной степени лучше линейной, в некоторых случаях много лучше линейной

# R-деревья

Корректировка дерева:

- Параметры: лист  $L$  и возможно  $LL$ , если  $L$  был расщеплен
- Расширение границ прямоугольников, включающих прямоугольники листа  $L$
- Расщепление внутренних узлов при необходимости

## ***AdjustTree:***

1. Зададим  $N = L$  и, если существует,  $NN = LL$
2. Если  $N$  – корневой узел, то остановиться
3. Пусть узел  $P$  – родитель  $N$  и  $P_N$  – запись в  $P$  об узле-потомке  $N$ .  
Скорректировать MBR в  $P_N$  (MBR прямоугольников из узла  $N$ )
4. Если  $NN$  существует, то создать новую запись  $P_{NN}$ , указывающую на  $NN$  и хранящую MBR прямоугольников из узла  $NN$

Если  $P$  вмещает в себя  $P_{NN}$ , то вставить  $P_{NN}$  в  $P$ , иначе:

- Вызвать **SplitNode**, производящую  $P$  и  $PP$ , совместно содержащие  $P_{NN}$  и старые записи узла  $P$
- Переопределить  $N = P$  и  $NN = PP$ , и перейти к шагу 2

# R-деревья

Удаление (прямоугольника R) из R-дерева:

1. Найти лист L, содержащий R, путем просмотра всех поддеревьев, MBR'ы которых пересекаются с R
2. Удалить R из L
3. [подготовка к сжатию дерева] Задать  $N = L$  и  $Q = \text{empty}$  (= множество удаляемых узлов)
4. Если N – корневой узел, то перейти к шагу 7, иначе: пусть узел P – родитель узла N и  $P_N$  – запись в P об узле N
5. [проверка условия минимальной заполненности узла] Если в узле N менее чем  $m$  (см. слайд 246) записей, то удалить  $P_N$  из P and добавить узел N в множество Q, иначе скорректировать MBR в  $P_N$
6. Переопределить  $N = P$  и перейти к шагу 4
7. [передислокация записей из удаленных узлов] Заново вставить в R-дерево все записи из множества Q. Записи из удаленных листов вставляются в листы (с помощью операции стандартной вставки). В тоже время, записи из удаленных внутренних узлов вставляются во внутренние узлы так, чтобы листья, образуемых ими поддеревьях, были на том же уровне, что и листья основного дерева
8. Если у корневого узла только один узел-потомок, то сделать потомка новым корневым узлом

# R-деревья

## R\*-дерево [5]: улучшенная версия R-дерева

- Откладывает расщепление путем принудительной вставки:
  - Сортировка всех прямоугольников на основе расстояний между их центрами и центром соответствующих MBR'ов
  - Определенная часть наиболее удаленных прямоугольников удаляется и затем повторно вставляется
- Более сложная эвристическая процедура для расщепления:
  - См.[5]
  - Временная сложность  $O(M \cdot \log M)$  для  $M$  прямоугольников
- Превосходит R-дерево
- Хорошо работает в качестве метода доступа к точечным данным
- «Эталонная» структура данных для других структур пространственных данных (пожалуй, наиболее известный метод доступа к пространственным данным)

# R-деревья

## X-дерево [6]:

- Может хранить точечные и пространственные данные
- Превосходит  $R^*$ -деревья, TV-деревья, и ряд других структур, особенно в пространствах большой размерности
- Основное предположение: с ростом размерности пространства последовательный индекс становится все более эффективен, т.к. перекрытия становятся все больше и больше
- Решение: внутренние узлы могут быть произвольного размера; суперузел содержит более одной страницы
- Многостраничный суперузел с (физически) последовательно расположенными страницами обрабатывается быстрее, чем такое же число отдельных страниц
- Для пространств большой размерности большие суперузлы предпочтительны
- X-tree настраивается на число измерений
- X-tree – «эталонная» структура для других структур данных высокой размерности

# Пространственные соединения

- Типичная операция при обработке пространственных запросов
- Задача: для двух множеств пространственных объектов найти пары, удовлетворяющие заданному пространственному предикату, например:
  - Равенство
  - Пересечение (перекрытие)
  - Включение (асимметрично)
  - Близость
  - Другие топологические зависимости (слева от, справа от, на севере от, и т.д.)
- В силу использования MBR'ов требуются два шага:
  1. Фильтрация: найти пары MBR'ов, удовлетворяющих предикату
  2. Уточнение: для каждой из пар, найденных на шаге 1, осуществить окончательную проверку, учитывая реальную геометрию объектов

# Пространственные соединения

## Примерный сценарий:

- Оба множества объектов описываются индексом на основе R-дерева
- Условие соединения – пересечение

## Стандартный алгоритм:

Основан на обходе деревьев в глубину

- Начать с корневых узлов
- На каждом шаге рассматриваются два узла  $(N_1, N_2)$ ; вычисляются пары пересекающихся записей  $(e_1, e_2)$ , где  $e_1 \in N_1, e_2 \in N_2$
- Процедура вызывается рекурсивно для поддеревьев, задаваемых  $e_1$  и  $e_2$
- При достижении уровня листов происходит сравнение непосредственно самих объектов

## Совершенствование алгоритма:

- Проверять только пары  $(e_1, e_2)$ , в которых  $e_1$  и  $e_2$  пересекаются с  $(MBR_{N_1} \cap MBR_{N_2})$
- Метод заметающей прямой: рассматривать два множества прямоугольников (красные и синие), искать пересечения только красных с синими



# Применение: географические базы данных

## Основные понятия

Географический объект:

- Две компоненты:
  - Описательная часть с численно-текстовыми атрибутами, например, город – название, население и т.д.
  - Пространственная часть (то что мы называем пространственным объектом) описывает геометрию (расположение, форму), например, город: полигон в двумерном пространстве

Элементарные и сложные (сложно-составные) объекты:

- Сложные объекты состоят из других элементарных/сложных объектов

Тема (theme):

- Класс (тип) географического объекта
- Соответствует отношению в реляционной бд; тема задается схемой и есть экземпляры темы (класса)
- Примеры тем: реки, города, страны, дороги и т.д.

# Применение: географические базы данных

## Геоинформатические операции

Проекция темы на подмножество описательных атрибутов:

- Соответствует реляционной проекции
- Визуальный результат: часть атрибутов на карте пропадает

Выборка на основе описательных атрибутов:

- Соответствует реляционной выборке
- Остаются только те географические объекты, что удовлетворяют условиям выборки
- Визуальный результат: часть объектов пропадает

Геометрическая выборка:

- Объекты в заданном окне: выбираются объекты (возвращаются целиком), пересекающиеся с заданным прямоугольником
- Запрос по точке: выбираются объекты, геометрия которых содержит данную точку
- Отсечение по заданному окну: выбираются объекты (возвращаются только(!) пересечения, а не целые объекты), пересекающиеся с заданным прямоугольником

# Применение: географические базы данных

## Объединение тем:

- Соответствует реляционному объединению
- Объединяет две темы, имеющие одинаковые схемы

## Наложение тем:

- Рядовая операция в геоинформационных приложениях
- Пространственное соединение: вычислить пересечения
- На основе пересечений создаются новые географические объекты:
  - Описательные атрибуты берутся от обоих пересекающихся объектов
  - Пространственная компонента определяется геометрией пересечения

## Метрические операции:

- Например, расстояние между Москвой и Санкт-Петербургом

## Топологические операции:

- Например, список стран, имеющих общую границу с Россией (Украина, Белоруссия, Литва, Латвия и т.д.)
- Список городов до которых можно долететь (без дополнительной посадки) из Санкт-Петербурга

# Применение: географические базы данных

## Геопространственные СУБД

### 1) Специализированные геоинформационные СУБД

#### *ArcInfo:*

- Задумана как набор инструментальных средств разработки
- Большой выбор пространственных функций
- Подсистемы: Arc – пространственные данные, Info – описательные данные
- Представление пространственных данных: векторное, растровое (сеточное), триангуляционное

### 2) Расширения реляционных СУБД

#### *Oracle Spatial:*

- Новый пространственный тип данных
- SQL расширен операторами для манипуляций с пространственным типом данных
- Пространственное индексирование на основе Z-порядка (см. предыдущую тему)
- Оптимизация запросов, например, для пространственных соединений

# Применение: географические базы данных

## *PostgreSQL:*

- Объектно-реляционная СУБД
- Свободно распространяемая, открытый код
- Расширенные возможности:
  - Геометрические типы: точка, линия, прямоугольник, полигон, окружность и т.д.
  - Операции с геометрическими объектами: сдвиг, масштабирование и т.д.
  - Индекс на основе обобщенного R-дерева
  - Вставка геометрических объектов в виде строки координат в SQL, например, треугольник – '((1,2), (4,5), (3,1))'
- В тоже время:
  - Не поддерживаются топологические операции (например, близости)
  - Не поддерживается наложение тем
  - Не поддерживается пересечение полигонов

# Упражнения

1. Рассмотрим простой (см. слайды «Вычислительная геометрия» для определения простого полигона) полигон в двумерном пространстве, задаваемый списком точек по часовой стрелке –  $P = ((x_1, y_1), (x_2, y_2), \dots, (x_n, y_n))$ . Предложить правило (основные принципы), определяющее находится ли заданная точка  $(x, y)$  внутри  $P$ . Предложите варианты правила в случаях, если полигон: выпуклый, не выпуклый, точки на гранях полигона не внутри  $P$ .
2. Предложить способ (основные принципы) для нахождения пересечения двух треугольников в двумерном пространстве.

# Ссылки на литературу

- [1] P. Rigaux, M. Scholl, A. Voisard. Spatial Databases, with Application to GIS, Morgan-Kaufmann, 2002
- [2] Gaede and Günther. Multidimensional Access Methods. ACM Computing Surveys, 30(2), 1998
- [3] T. Sellis, N. Roussopoulos, and C. Faloutsos. The R+-Tree: A dynamic index for multi-dimensional objects. VLDB-1987, 1987
- [4] A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. SIGMOD-1984, 1984
- [5] N. Beckmann, H. Kriegel, R. Schneider, B. Seeger. The R\*-Tree: An Efficient and Robust Access Method for Points and Rectangles. SIGMOD-1990, 1990
- [6] S. Berchtold, D. Keim, H. Kriegel. The X-tree: An Index Structure for High-Dimensional Data. VLDB-1996, 1996