Информационные технологии автоматизированного проектирования
Часть 1

Лекция 7

Лекция 7
АЛГОРИТМЫ И МОДЕЛИ ТРАССИРОВКИ ПРОВОДНЫХ СОЕДИНЕНИЙ В ЭА

1 Классификация алгоритмов трассировки

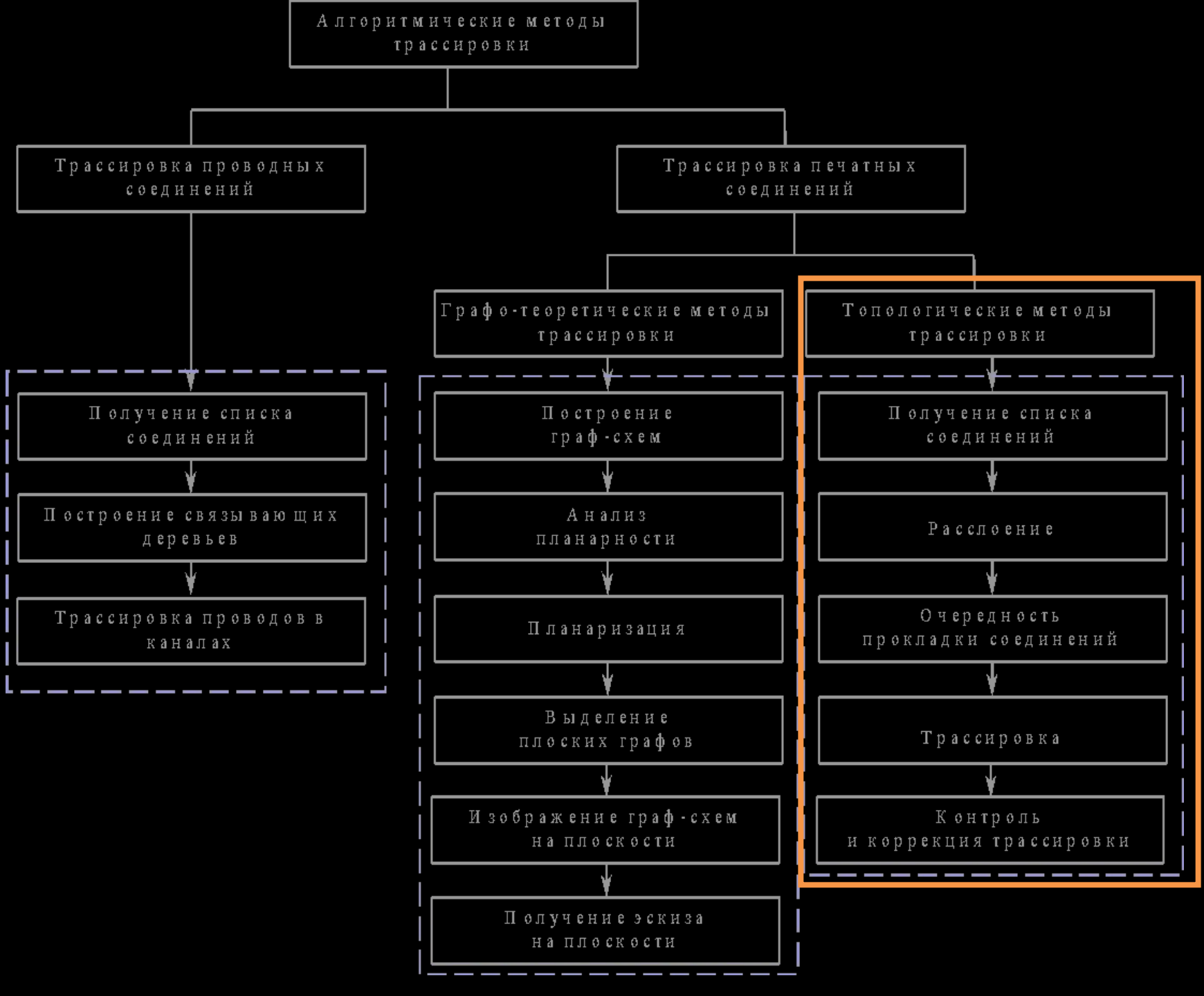
2 Формулировка задачи трассировки проводных соединений

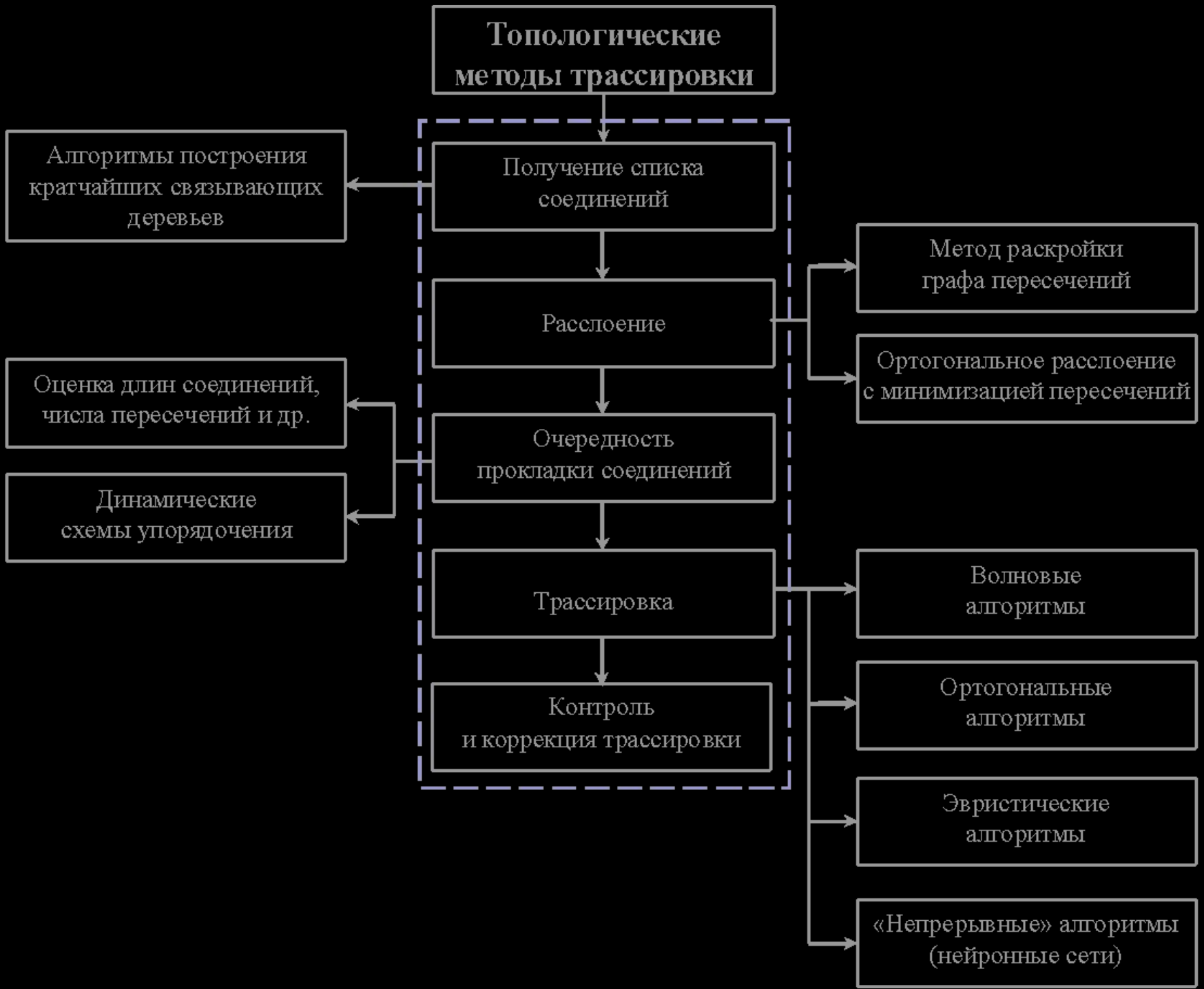
3 Алгоритм Краскала (Вайнберга – Лобермана)

4 Алгоритм Прима

5 Особенности трассировки проводов в каналах

Вопрос 1 Классификация алгоритмов трассировки


Алгоритмические методы трассировки
Трассировка проводных соединений
Трассировка печатных соединений
Графо-теоретические методы трассировки
Топологические методы трассировки
Получение списка соединений
Построение связывающих деревьев
Трассировка проводов в каналах
Построение граф-схем
Анализ планарности
Планаризация
Выделение плоских графов
Изображение граф-схем на плоскости
Получение эскиза на плоскости
Получение списка соединений
Расслоение
Очередность прокладки соединений
Трассировка
Контроль и коррекция трассировки


Топологические методы трассировки
Получение списка соединений
Расслоение
Очередность прокладки соединений
Трассировка
Контроль и коррекция трассировки
Алгоритмы построения кратчайших связывающих деревьев
Оценка длин соединений, числа пересечений и др.
Динамические схемы упорядочения
Метод раскройки графа пересечений
Ортогональное расслоение с минимизацией пересечений
Волновые алгоритмы
Ортогональные алгоритмы
Эвристические алгоритмы
«Непрерывные» алгоритмы (нейронные сети)

Вопрос 2 Формулировка задачи трассировки проводных соединений

Исходная информация для решения задач трассировки соединений

1) список цепей
2) параметры конструктивных элементов
3) параметры монтажного поля
4) данные по размещению конструктивных элементов
5) координаты выводов элемента

Трассировка проводных соединений более проста, т.к. цепи электрически изолированы друг от друга. Глобальная оптимизация обеспечивается локальной оптимизацией отдельной цепи

Требования к трассировке соединений

1) Соединения должны соответствовать принципиальной схеме и быть кратчайшими;
2) Число пересечений трасс в монтажном поле должно быть минимальным для МПП, либо не допускается
3) Распределение цепей в монтажном поле должно приближаться к равномерному;
4) Минимум числа непроведенных соединений;
5) Минимальная протяженность параллельных участков соседних проводников;
6) Минимум числа изгибов проводников;
7) Минимум числа слоев металлизации и числа переходов из слоя в слой.

2.1 Трассировка проводных соединений по прямым, соединяющим отдельные выводы модулей (монтаж внавал)

Достоинства:

- простота выполнения
- высокая помехоустойчивость
- позволяет до минимума сократить общую длину проводников, в т.ч. протяженность параллельных
- уровень паразитных наводок и время задержки сигнала в электрических соединениях невелики

Недостатки:

✔ высока вероятность появления в процессе монтажа ошибок,

✔ сложен контроль правильности трассировки

✔ малая ремонтопригодность при высокой плотности

2.2 Трассировка проводных соединений с помощью жгутов (ленточных кабелей)

1. Получение списка соединений
2. Построение кратчайших связывающих деревьев (сетей)
3. Выполнение трассировки проводов в каналах

Недостатки :

практически неприемлем для создания высокочастотной и чувствительной к электрическим помехам аппаратуры

Достоинства :

1. более технологичен, так как позволяет разделить операции подготовки и монтажа жгутов,
2. проще процесс контроля и устранения ошибок, допущенных при монтаже

Формулировка задачи трассировки проводных соединений

В некоторой системе координат XYZ, связанной с коммутационным пространством модуля, задано местоположение множества выводов

M = {m1, m2,..., mn}.

В соответствии с электрической схемой соединений разобьем множество **M** на непересекающиеся подмножества **M(1), M(2),..., M(P)**, каждое из которых включает в себя выводы, подлежащие электрическому объединению.

Для каждого подмножества требуется определить последовательность соединения выводов и конфигурацию проводников, обеспечивающих при заданных ограничениях минимальную суммарную длину соединений (возможен учет назначения цепей)

Вопрос 3 Алгоритм Краскала (Вайнберга – Лобермана)

Алгоритм

Все известные алгоритмы построения кратчайших связывающих сетей (КСС) основаны на последовательном выборе самых коротких связей, не образующих циклов с ранее отобранными.

Пусть в некоторой системе координат XYZ задано местоположение множества точек.

M = {m1, m2,..., mn}.

1. Строим на множестве **M** полный граф **Gn(M, U)**.
2. Вычисляем длину всех ребер графа **Gn(M, U)**
3. Упорядочиваем список ребер с точки зрения их длины так, чтобы выполнялось условие

$$\forall u_i \in U\left[d(u_i) \le d(u_{i+1})\right] \qquad 1 \le i < r = \frac{n(n-1)}{2}$$

- построения кортежа с возрастанием длины каждого очередного ребра.

Алгоритм

4. Для построения дерева необходимо выбрать **n-1** ребер из кортежа, которые не образуют циклов.

Существуют 2 процедуры для решения п. 4

Вариант 1 (параллельный).

На каждом шаге просматривают список ребер (начиная с ребра, следующего за вошедшем в решение на предыдущем шаге) и к строящемуся поддереву присоединяют то ребро, которое не образует цикла с ребрами, уже включенными в решение.

Недостатки алгоритма:

- необходимость наблюдения за различными компонентами связности,
- проверки при выборе каждого очередного ребра условия необразования цикла для всех параллельно строящихся поддеревьев.

Алгоритм

Вариант 2 (последовательный).

На каждом шаге просматривают список ребер (начиная с первого) и к строящемуся поддереву присоединяют то ребро, которое:

а) еще не включено в решение;

б) присоединяет к поддереву новую вершину (один из концов ребра должен принадлежать вершине поддерева, другой —изолированной вершине)

Недостатки алгоритма:

- необходимость на каждом шаге алгоритма начинать просмотр списка с первого ребра, причем значительная часть просматриваемых при этом ребер может не удовлетворять условиям включения их в строящееся поддерево. Это приводит к увеличению времени решения задачи.

Вопрос 4 Алгоритм Прима

Алгоритм

Позволяет организовать просмотр только тех ребер графа **Gn(M, U)**, которые связывают вершины строящегося поддерева с новыми, еще не присоединенными вершинами.

Возможно дополнительное ограничение на локальные степени вершин связывающей сети:

$$\forall m_i \in M \left[\rho(m_i) \leq k \right]$$

Шаги алгоритма:

1) любая произвольная вершина m € M соединяется с ближайшей соседней, образуя исходное поддерево.

Для определенности построения КСС можно начинать с ребра, инцидентного вершине m1.

Шаги алгоритма

2) На каждом последующем шаге к строящемуся поддереву присоединяют очередное ребро минимально возможной длины, связывающее новую, еще не присоединенную вершину **m*j*** с одной из вершин поддерева **mi** , локальная степень которой

$$\rho(m_i) < k$$

$$d(m_i, m_j) = \min_{m_f \in M^*, m_g \in M \setminus M^*} d(m_f, m_g)$$

$d(m_f, m_g)$ - длина ребра, соединяющего вершины

Детализация алгоритма

1) составляем матрицу длин, общий элемент которой **dij** равен расстоянию между i-й и j-й точками (N - число объединяемых вершин):

$$D = \left\| d_{ij} \right\|_{N \times N}$$

2) Просматриваем элементы первой строки матрицы **D** и находим минимальный из них. Пусть таким элементом оказался элемент **g**-го столбца, тогда весь первый и g-й столбцы матрицы D исключаем из рассмотрения, а первое соединение проводим между точками **m1** и **mg**.

Детализация алгоритма

3) Просматриваем первую и g-ю строки матрицы с оставшимися элементами. Из элементов этих строк находим минимальный. Предположим, что им оказался элемент, принадлежащий **k**-му столбцу. Если этот элемент находится на пересечении с первой строкой, то точку **mk** соединяем с **mi**, если же он находится на пересечении с **g**-й строкой, то точку **mk** соединяем с **mg**, после чего из матрицы D исключаем все элементы **k**-го столбца.

4) Просматриваем первую, g-ю и k-ю строки и т.д.

Детализация алгоритма

Выполнение ограничения на локальную степень вершин обеспечивается проверкой в каждой просматриваемой **i**-й строке числа уже выбранных для построения КСС элементов K(i). При **K(i) = k** все оставшиеся элементы **i**-й строки исключаются из рассмотрения.

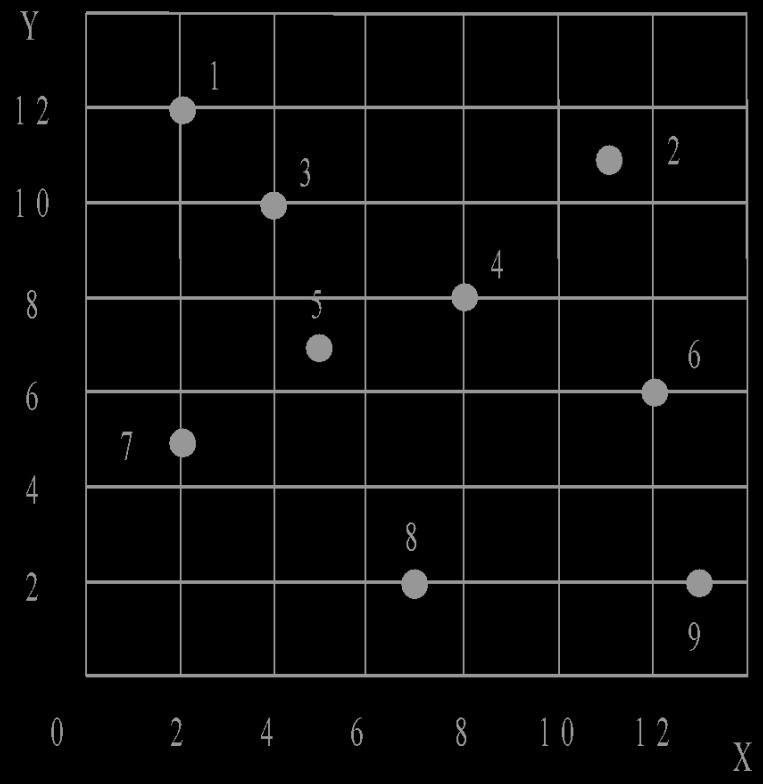
Пример использования алгоритма Прима

На плоскости в декартовой системе координат задано местоположение девяти точек (рисунок). Расстояние между любыми двумя точками **mi** и **mj** равно

$$d_{ij} = |x_i - x_j| + |y_i - y_j|$$

Требуется для заданного множества точек M определить минимальное связывающее дерево при условии:

$$\forall m_i \in M[\rho(m_i) \leq 3]$$



Пример использования алгоритма Прима

Решение.

Составляем матрицу длин:

$$D = \begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 0 & 10 & \langle 4 \rangle & 10 & 8 & 16 & 7 & 15 & 21 \\ 2 & 10 & 0 & 8 & 6 & 10 & \langle 6 \rangle & 15 & 13 & 11 \\ 3 & 4 & 8 & 0 & 6 & \langle 4 \rangle & 12 & 7 & 11 & 17 \\ 4 & 10 & \langle 6 \rangle & 6 & 0 & 4 & 6 & 9 & 7 & 11 \\ 5 & 8 & 10 & 4 & \langle 4 \rangle & 0 & 8 & \langle 5 \rangle & 7 & 13 \\ 6 & 16 & 6 & 12 & 6 & 8 & 0 & 11 & 9 & \langle 5 \rangle \\ 7 & 7 & 15 & 7 & 9 & 5 & 11 & 0 & 8 & 14 \\ 8 & 15 & 13 & 11 & 7 & 7 & 9 & 8 & 0 & 6 \\ 9 & 21 & 11 & 17 & 13 & 13 & 5 & 14 & \langle 6 \rangle & 0 \end{array}$$

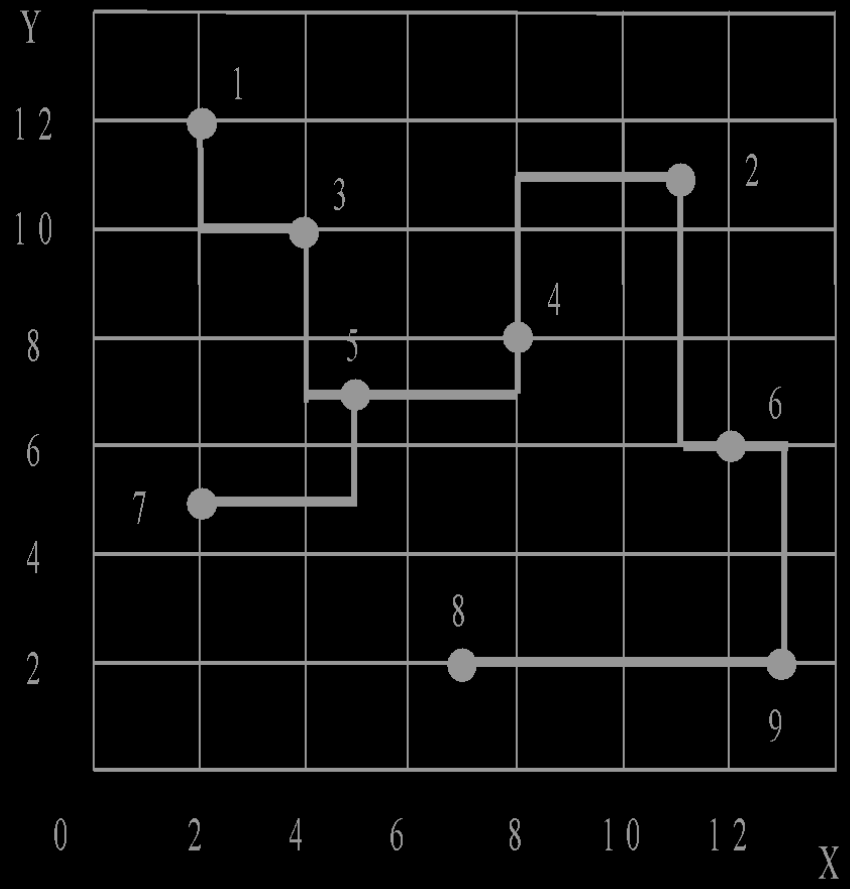
Пример использования алгоритма Прима

1) Просматриваем **1**-ю строку матрицы и выбираем элемент **d13**, являющийся минимальным в этой строке.

2) Помечаем элемент **d13**; **К(1) = К(3) = 1.**

Исключаем из рассмотрения все элементы 1-го и 3-го столбцов.

3) Просматриваем **1**-ю и **3**-ю строки. Выбираем элемент **d35; К[3] = 2, К[5] = 1**. Исключаем из рассмотрения элементы 5-го столбца.

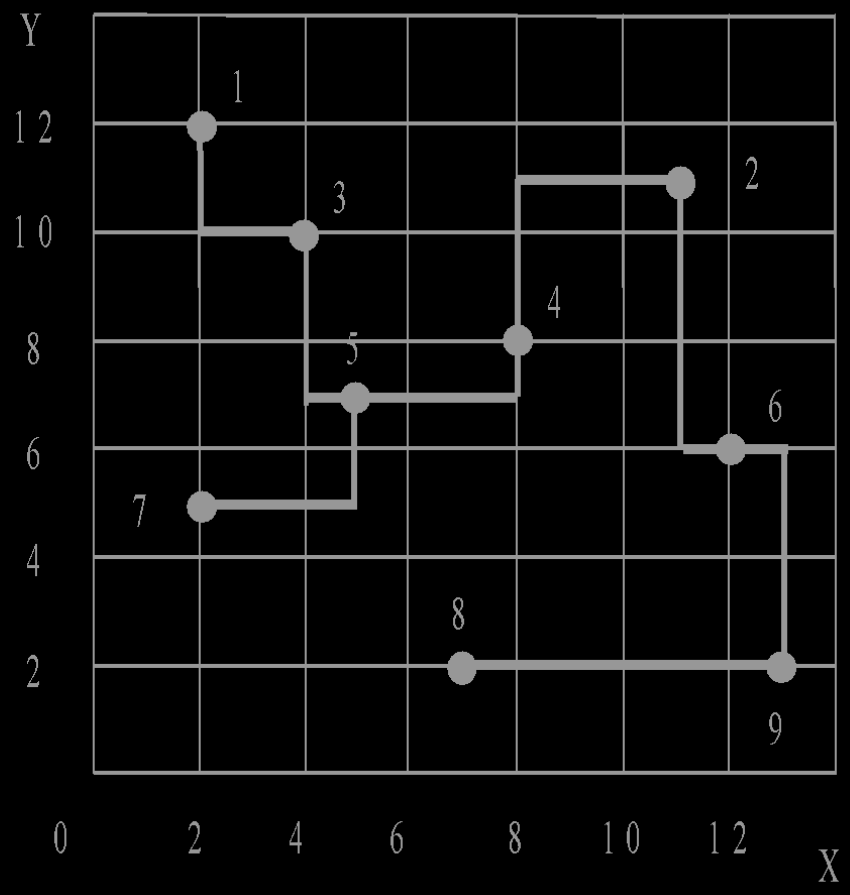


Пример использования алгоритма Прима

4) Просматриваем **1-ю, 3-ю и 5-ю** строки. Выбираем элемент **d54; К(5) = 2, К(4) = 1**. Исключаем из рассмотрения элементы **4-го** столбца.

5) Просматриваем **1-ю, 3-ю, 4-ю и 5-ю** строки. Выбираем элемент **d57; К(5) = 3, К(7) = 1.** Так как **К[5] = k,** то исключаем из рассмотрения элементы **7-го** столбца и **5-й** строки.

6) Продолжая процесс построения КСС аналогичным образом, выбираем элементы **d42, d26, d69, d98.**



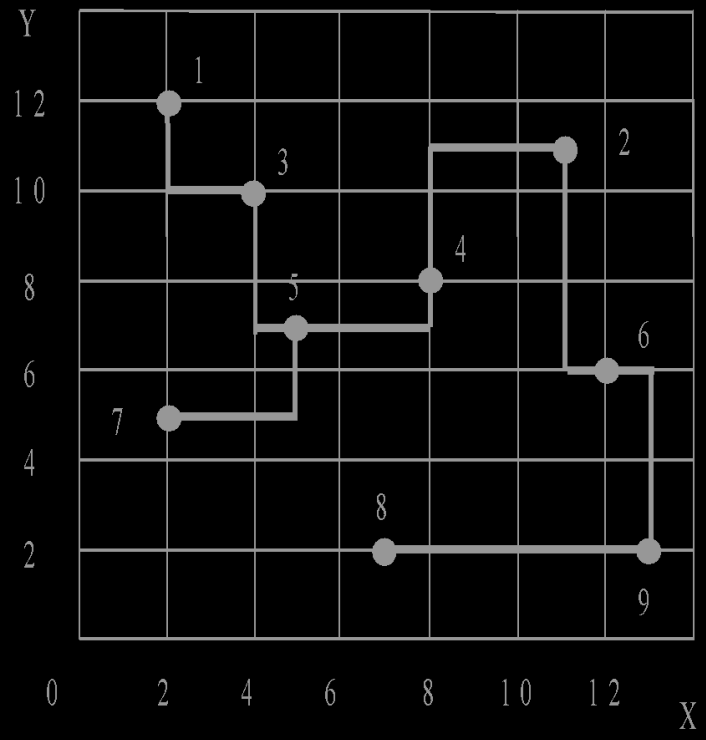
Пример использования алгоритма Прима

Суммарная длина ребер построенного дерева равна **D = 40**.

Если локальная степень вершин не должна превышать **двух**, то в результате решения будут выбраны элементы **d13, d35, d54, d42, d26, d69, d98, d17**. Суммарная длина ребер при этом равна **42**.

Алгоритм Прима находит широкое применение в САПР проводных соединений ЭА.

Например, пакет **E3**



Вопрос 5 Особенности трассировки проводов в каналах

В ЭА используется жгутовой монтаж (шлейфами), при котором проводники укладывают в нормализованные каналы, расположенные в монтажном пространстве во взаимно перпендикулярных направлениях.

Такую канальную конструкцию можно представить в виде ортогональной несимметрической сети G (A, B) со множеством узлов **A** и множеством дуг **B**.

Величина **c*ij*** - **пропускная способность дуги b*ij*** - максимальное число проводников, которое можно укладывать в соответствующем канале, интерпретируемом дугой **b*ij***.

$$\forall b_{ij} \in B \left\lfloor 0 \le x_{ij} \le c_{ij} \right\rfloor \quad (1)$$

где **xij** - дуговой поток или поток по дуге bij, равный числу проводников в канале, соединяющем точки i и j

множество узлов A :

A1 - подмножество узлов, соответствующих электрическим контактам модулей и разъемов схемы;

A2 - подмножество узлов, в которых допускается изменение направления прокладывания проводников (точки пересечения вертикальных и горизонтальных каналов).

A1: **As** - источники и **At** - стоки, определяющие, соответственно, электрические контакты модулей и разъемов

Для любого узла $a_j \in A = A_S \cup A_T \cup A_2$

сети **G (A, B)** должно выполняться условие

$$\sum_i x_{ij} - \sum_k x_{jk} = \begin{cases} -X_j, \ если \ a_j \in A_S \\ 0, \ если \ a_j \in A_2 \\ X_j, \ если \ a_j \in A_T \end{cases} \qquad (2)$$

Полный поток из A_S в A_T

$$X = \sum_{a_j \in A_T} X_j = -\sum_{a_j \in A_S} X_j \qquad (3)$$

В реальных конструкциях пропускные способности каналов ограничены, поэтому естественна постановка **задачи отыскания максимального потока в сети**, которая сводится к нахождению такого распределения потоков (проводников) по отдельным каналам, которое максимизирует целевую функцию (3) при ограничениях (1) и (2).

Решаются данные задачи методами линейного целочисленного программирования.

Вопросы по прочитанному материалу?

Спасибо за внимание!