EDA 软件设计 I

Lecture 4

Course Outline

1. **The end of first chapter**
    1. EDA入门收官
    2. 学习本门课程能给你带来什么
2. **第二章：EDA常用图算法**
    1. Intro to 图
    2. 图的基本概念回顾
    3. Rethink：图的存储
    4. BFS

EDA入门

- 通过前三学时，你大概了解了：
  1. 什么是工业软件，它参与的工业流程有哪些
  2. 什么是EDA软件，它的核心功能是什么
  3. 在电子领域、半导体产业，EDA属于**皇冠上的明珠**般的存在
  4. 芯片设计的全流程有哪些环节

EDA本身难度和学习目标

- 你了解到，EDA 软件设计**难**吗?
- 但它的难并不在于需要超高IQ，而是:
  1. **复合型的知识积累——不要用学科划分限制死了自己**
  2. **扎实的算法基础与编程能力**
  3. **强大的独立思考和迁移学习能力**
- 当你学完EDA软件设计I、II、III之后:
  1. 你不会被期待已经能够参与开发一款EDA软件
  2. 但你会已经具备行业知识、技术基础和进入EDA企业实习的能力

主流发展方式的探索方向

1. 出国留学
2. 国内保研、考研
3. 直接工作

主流发展方式的探索方向

出国留学，建议：

1. 日常学英语，早日通过语言考试
2. 保持**很好**的 **GPA**
3. 参与科研项目与发论文
4. 参加学术竞赛
5. 获取推荐渠道

主流发展方式的探索方向

国内保研、考研，建议：

1. 有目标院校和意向老师的，提早积极主动联系，争取加入实验室做项目
2. 成绩——保研硬性条件
3. 锻炼自己的综合能力
4. 在学习之余，多尝试不同事情、探索自己

主流发展方式的探索方向

直接工作，建议：

1. 多参与各方项目，学术类、工程类，丰富自己的“作品栏”
2. 最好能认清一些过于明确的目标是否是自己真正想要的
3. 多找已出入社会、在不同公司和岗位的学长、学姐交流

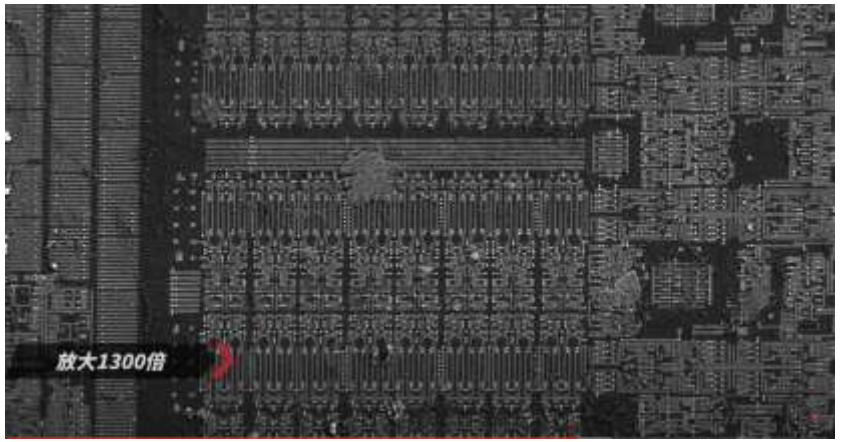
适用于所有人

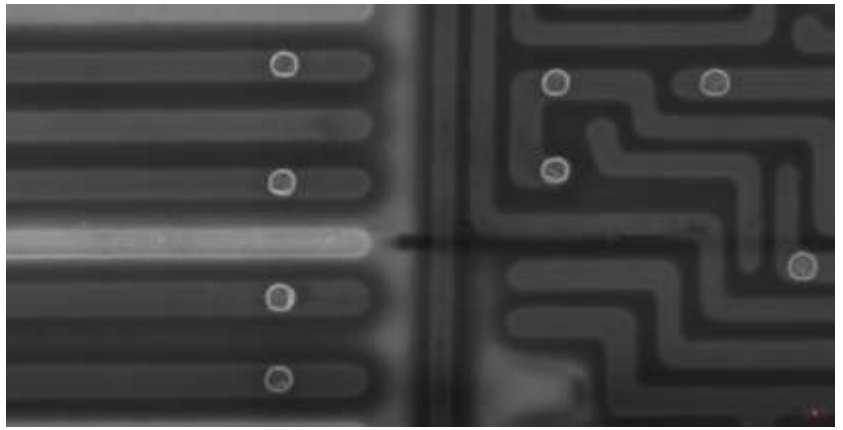
- 大学时代，多看多学，多去挖掘、了解【你不知道"你不知道"】的事情
- 少点目的性、功利性，免得你错过你完全不知道的好风景

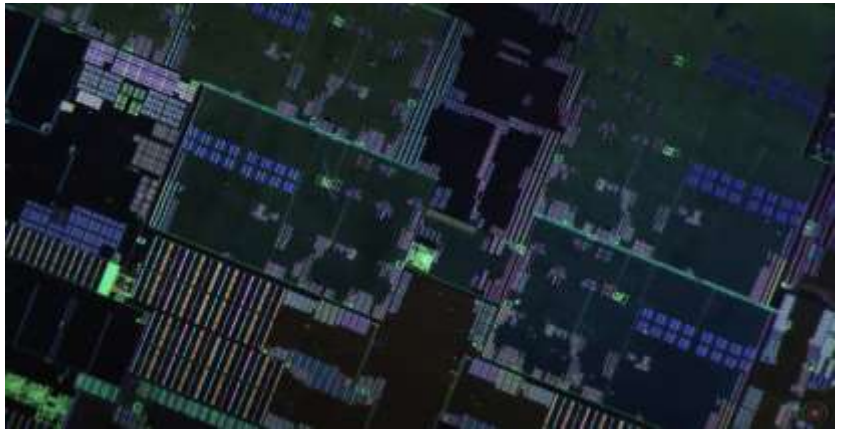
如果只能给一条建议：**Do a lot of readings and Exercise**

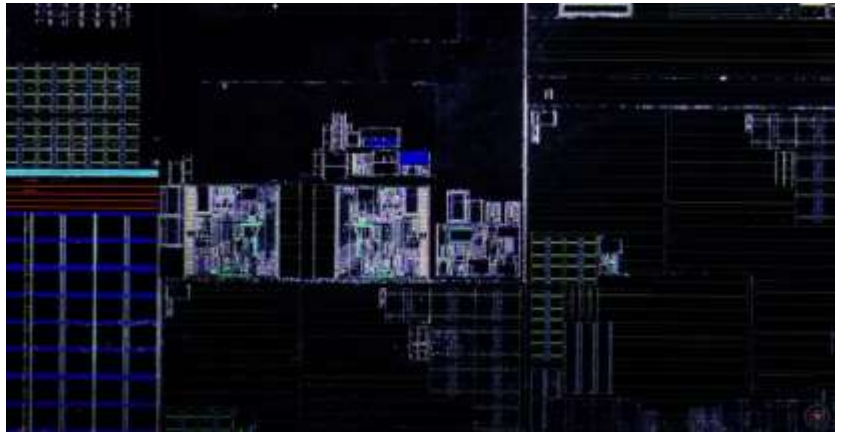
EDA 软件设计 I

第二章：基础图算法
Graph Algorithms


放大1300倍







“图”

When we talk about **Graphs**

- “图”这个字可以泛指很多东西：
  - 图像/图片：照片、计算机生成的图像、插图
  - 绘画：艺术作品中的图形
  - 示意图/图表：数据和信息的可视化表示，如柱状图、饼图、流程图等

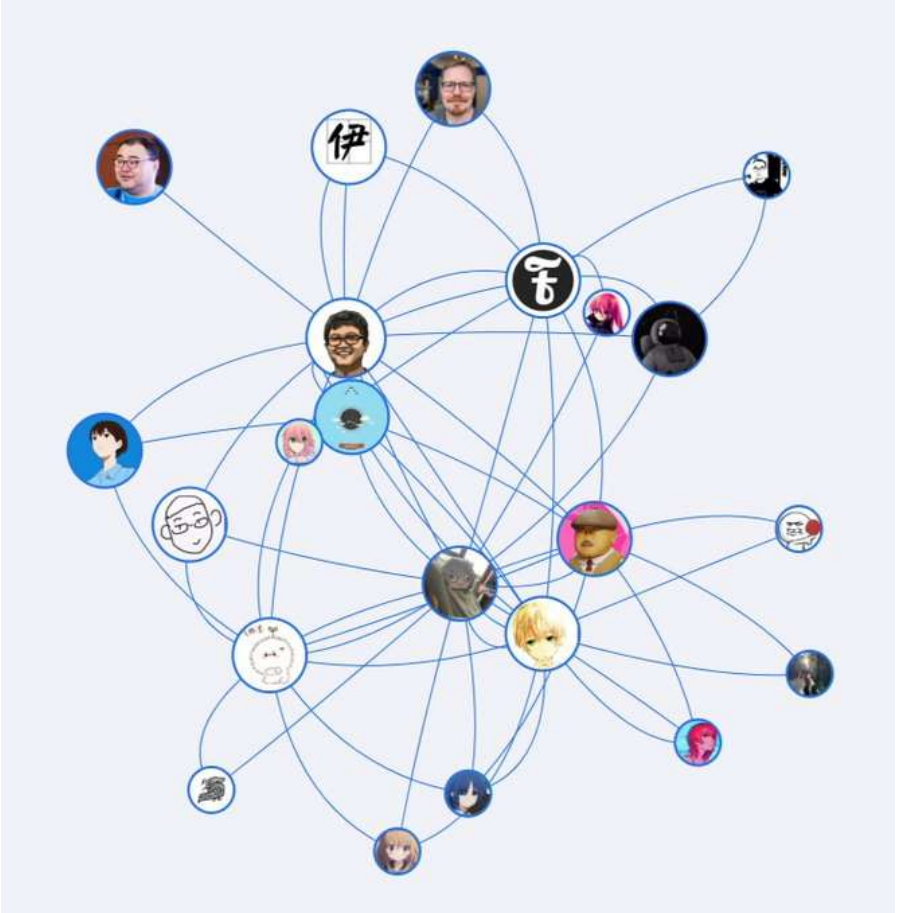
- 根据学科背景来讲：我们说的图是一种**数据结构（data structure)**

- 是一个由节点和边组成的**数学模型**，用于表示对象（节点）及其相互关系（边）

图作为一种数据结构

- 是现实世界物体及其之间关系的抽象、表示

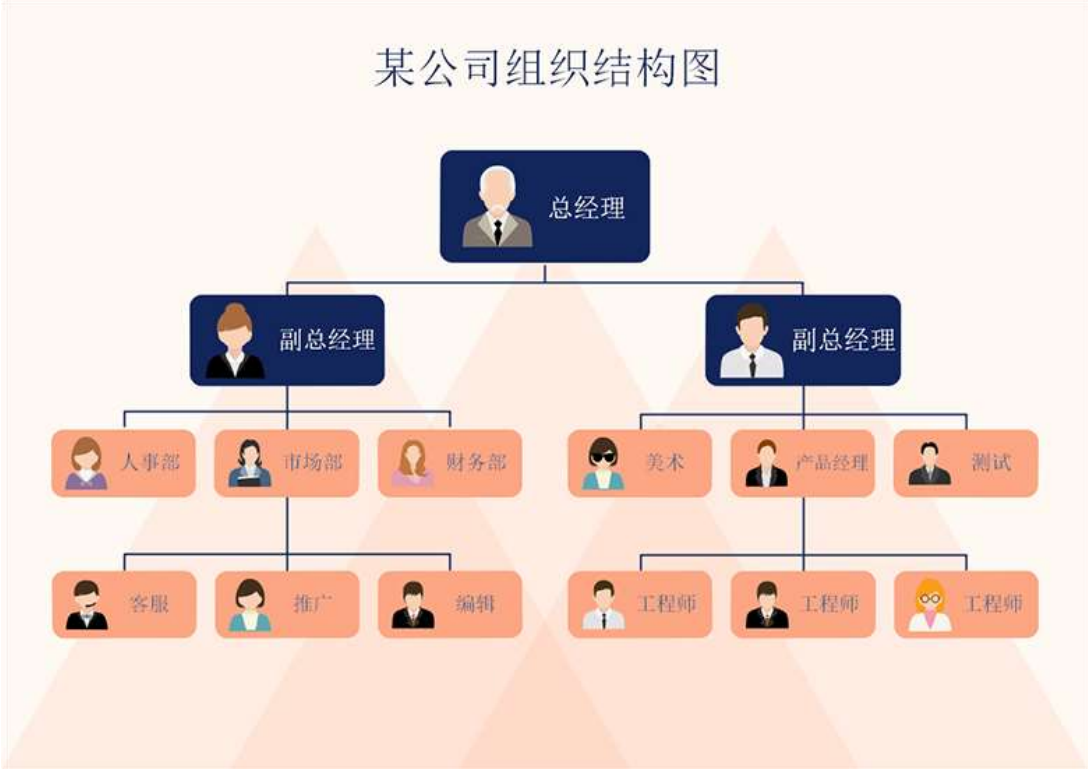
- **社交网络图**里面：
  - 节点：代表**用户**
  - 边：代表**友谊**或者**关注关系**



图作为一种数据结构

- 是现实世界物体及其之间关系的抽象、表示

- **组织结构图**里面：
  - 节点：代表**公司或组织里面的人员**
  - 边：代表**上下级关系**或者**部门合作关系**



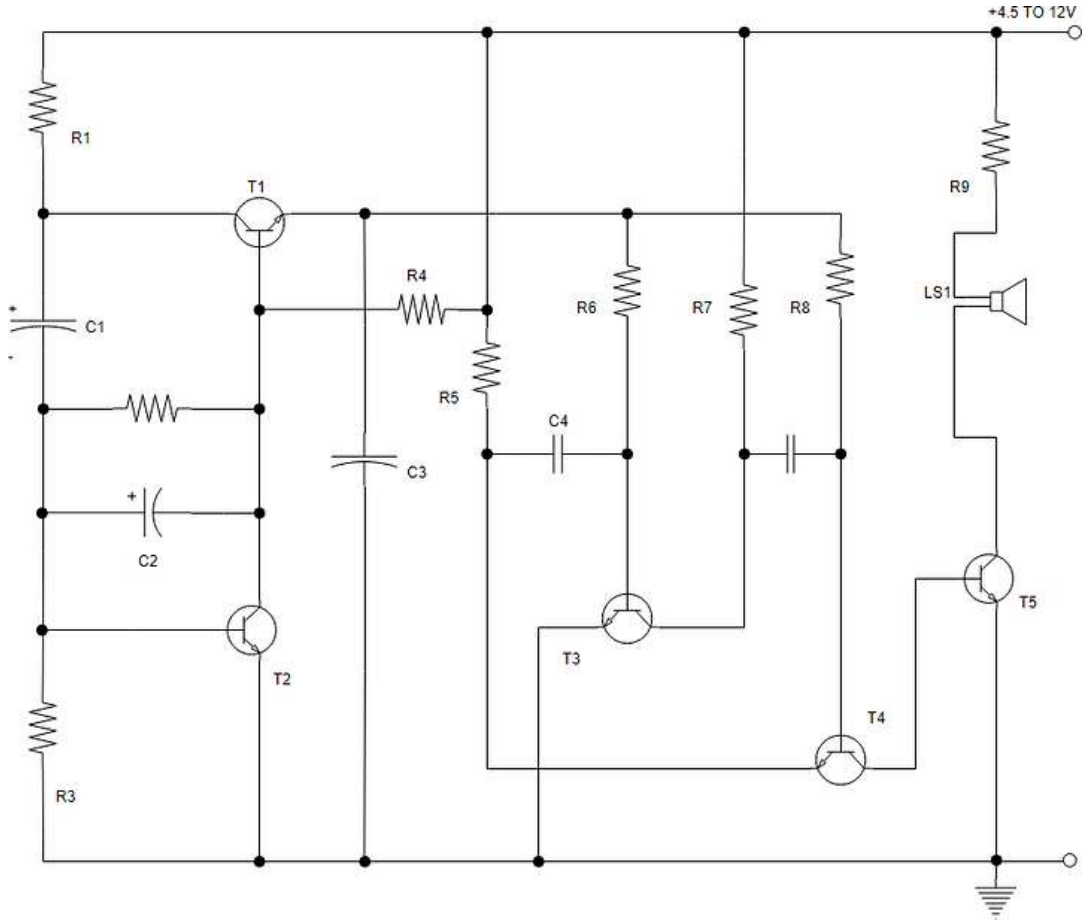
图作为一种数据结构

- 是现实世界物体及其之间关系的抽象、表示

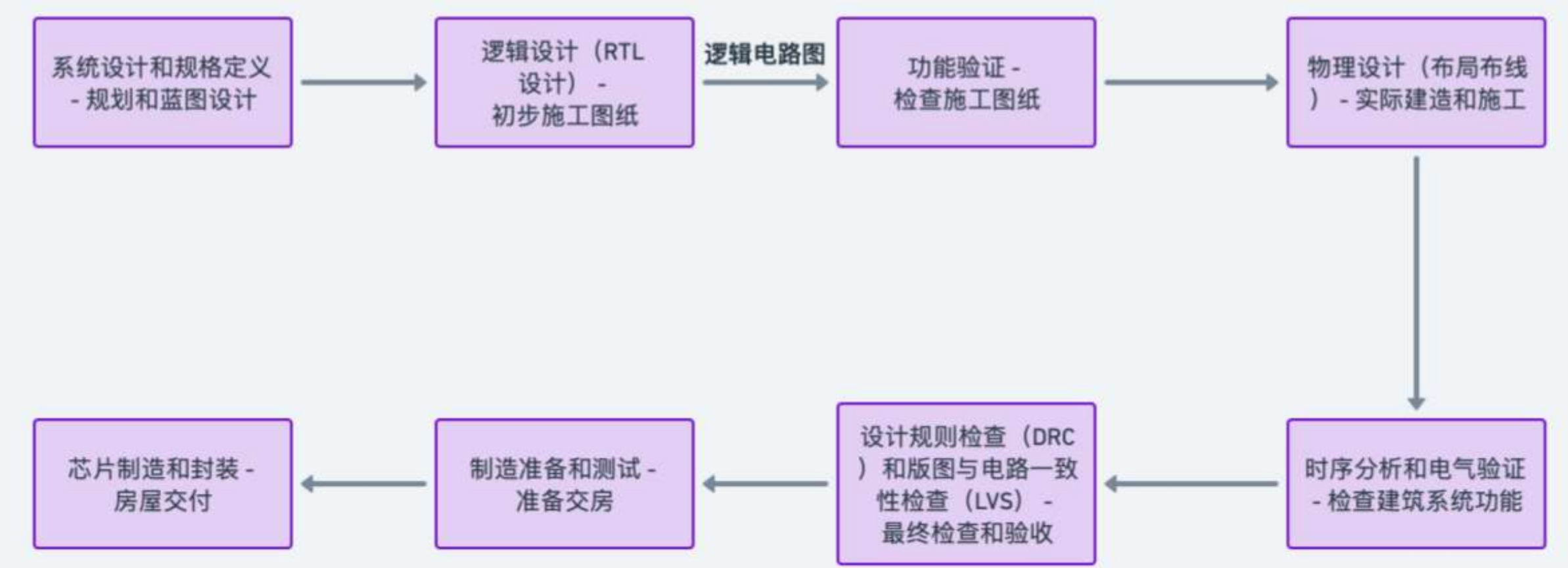
- **轨道交通网络图**里面：
  - 节点：代表**车站或交叉路口**
  - 边：代表**地铁线路。**
    - 边的权重表示可表示**距离、时间或费用**

图作为一种数据结构

- **EDA背景下**
- **逻辑电路图**
  - 表示**数字电路**中的逻辑门的连接和功能
  - ➢节点：代表**逻辑门（或门、与门和非门）**
  - ➢边：代表**逻辑门之间的连接（信号线）**



Review in Time:芯片设计全流程



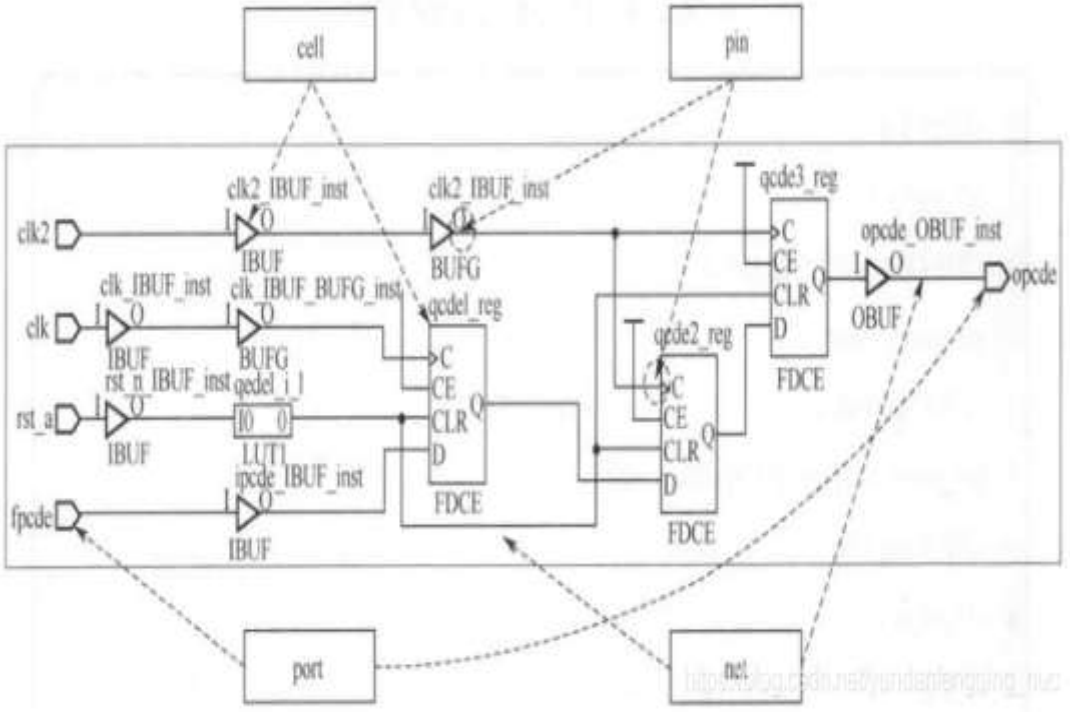
逻辑电路是如何运作的

- 程序逻辑：
  - if 【今天天气好】(A)，return 【出门打球】
  - else，if【家里有游戏机】（B），return 【就在家里打游戏】
  - else，return 【躺着】

- 逻辑电路的实现思路：
  1. **if 天气好**：用一个AND门连接输入为A和X1。当A为真（即天气好）时，输出为X1（出门打球）
  2. **else if 家里有游戏机**: 在天气不好时，检查B, 使用AND门连接输入为B和X2。当B为真时（家里有游戏机），输出为X2（在家打游戏）
  3. **else**: 结合NOT A和NOT B，得到输出X3 (躺着)

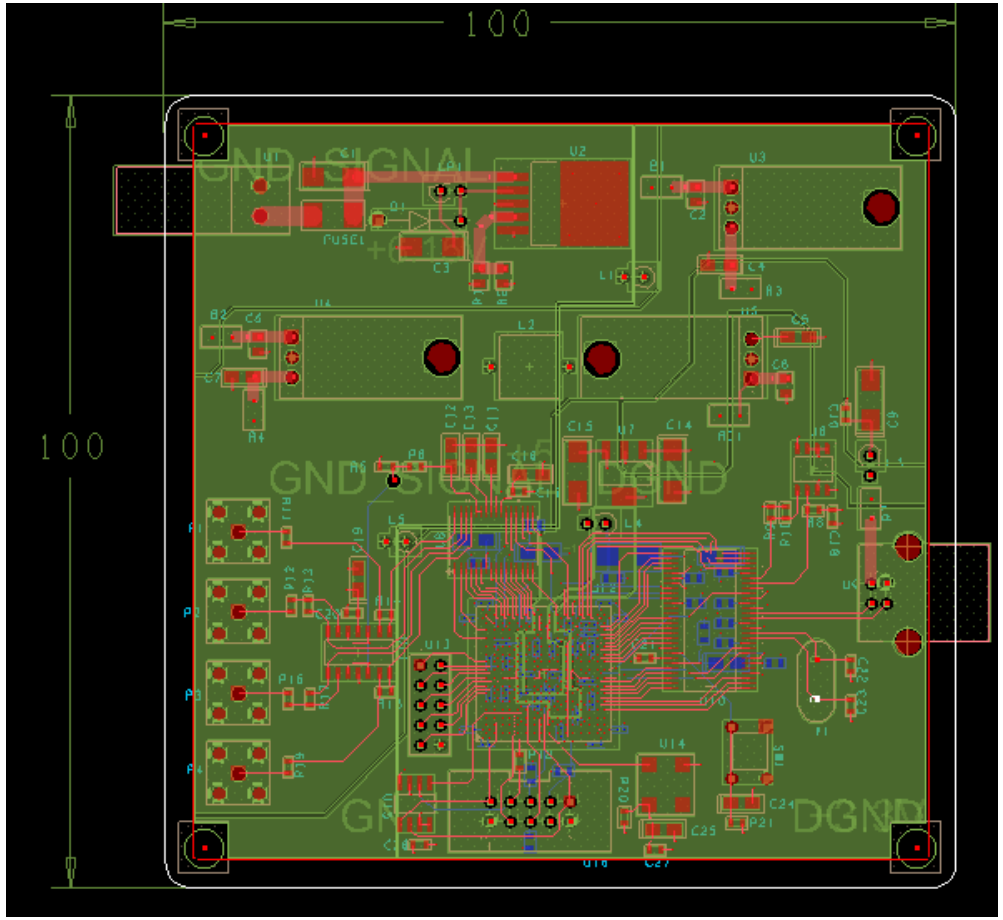
图作为一种数据结构

- **EDA背景下**
- **电路网表图：**
  - 对**电路中的元件（如电阻、电容、晶体管等）**及其连接关系的抽象描述
  - ➢节点：代表**电路中的元件**
  - ➢边：代表**元件之间的连接（如导线、布线）**



图作为一种数据结构

- **EDA背景下**
- **布局布线图：**
  - 电路设计的物理实现，包括每个元件的实际物理位置，以及连接它们的导线如何布置在芯片或者
  - ➢节点：代表**电路中的元件**
  - ➢边：代表**走线的路径**



电路网表图 vs 布局布线图

电路网表图：Netlist

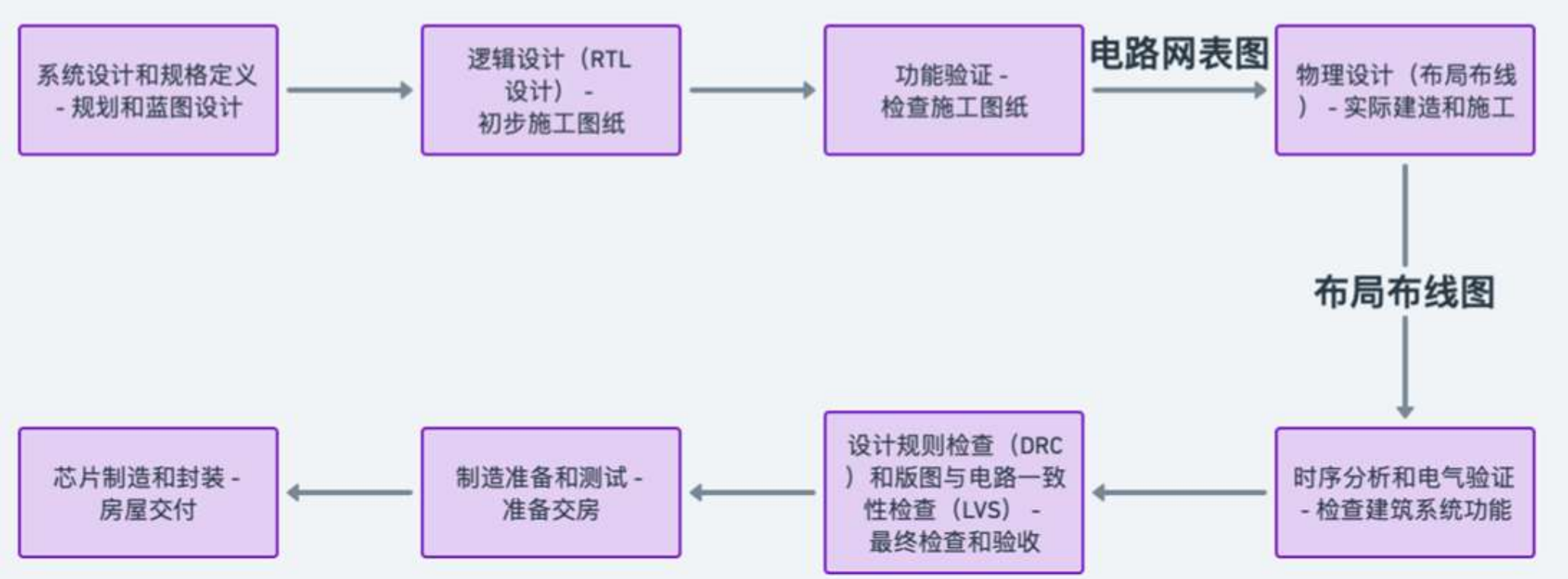
- 逻辑综合与功能验证的output，物理设计的input

布局布线图：Layout and Routing

- 物理设计的output

- 相同点：两个图的**点**都代表**电路元件**
- 区别：
  - Netlist只描述了电路的逻辑连接关系，**不涉及任何物理位置信息**
  - Layout and Routing是在Netlist上进行的物理实现，它定义了元件的**具体物理位置**和连接导线的布线方式，还需**考虑物理约束**（如信号延迟、寄生效应、功耗、面积等）

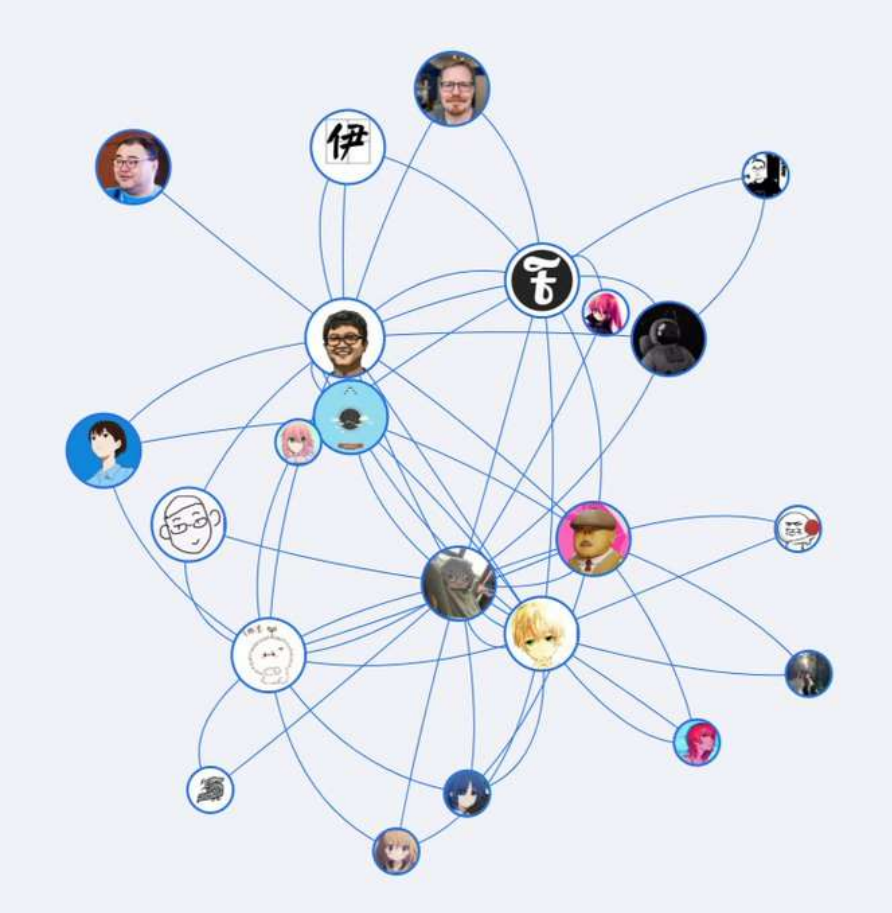
电路网表图 vs 布局布线图



有了「图」后，我们想问？

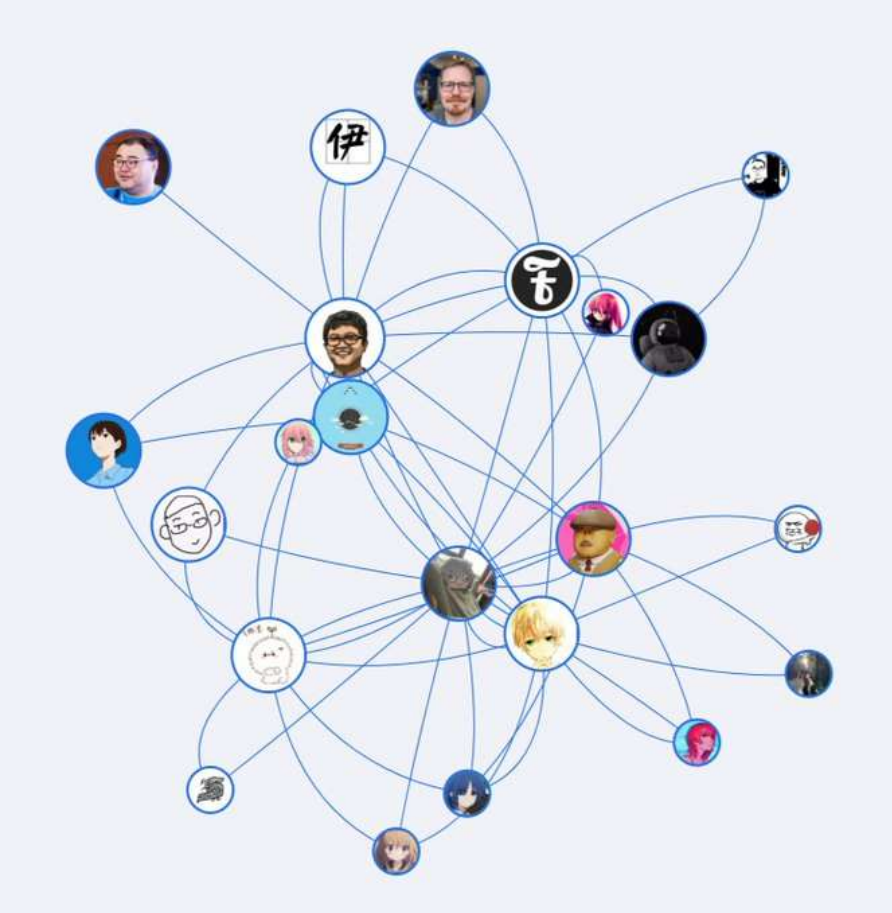
1. **为什么需要图结构？（Why）**
2. **可以拿它来做什么事？怎么做？（How）**

社交网络图中的问题



- easy
  - ① XXX的朋友有谁?
  - ② A和B是不是朋友?
  - ③ 哪些人的A和B的共同朋友?
- medium：
  - ① 谁是最"中心"的人
  - ② 谁是"桥梁"节点（连接两个群体）
  - ③ 哪些人形成一个紧密的朋友群体

社交网络图中的问题



- Hard:
  - ① 网络中是否存在社区结构，如何划分这些社区？
  - ② 如果存在社区，这些社区的人倾向一起去干什么？
  - ③ 信息是如何在这个社交网络中传播的？
  - ④ 是否存在两个子群体，彼此之间几乎没有联系？

图算法：回答社交网络问题

1. XXX的朋友有谁？
   - 节点的neighbor是哪些
2. A和B是不是朋友？
   - 检查A和B之间是否存在边
3. 谁是最“中心”的人？
   - 涉及图的中心性（centrality）的概念，可以通过度中心性、介数中心性、接近中心性等方法来衡量。
4. 是否存在两个子群体，彼此之间几乎没有联系？
   - 涉及分割（graph partitioning）或社群划分问题

轨道交通网络图中的问题

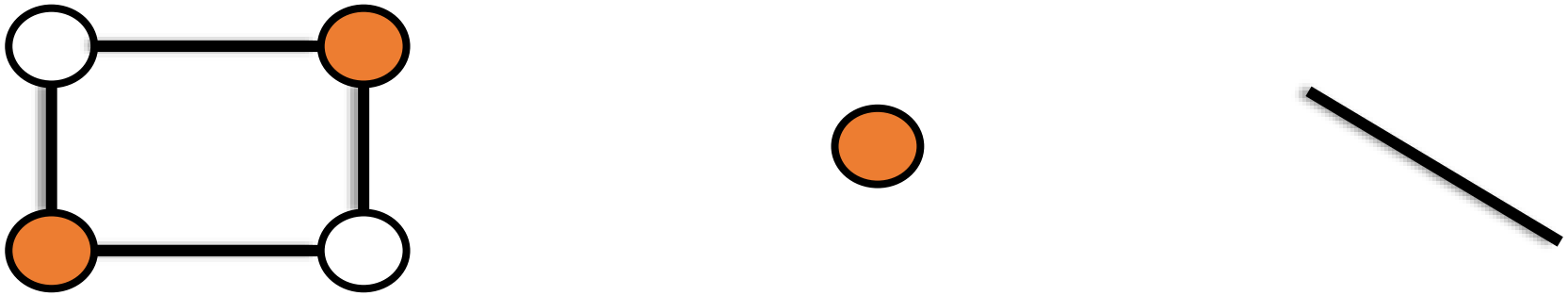
- Easy:
  - ① 某个地铁站有多少条线路经过?
  - ② 两个地铁站之间是否有直接的轨道连接?
- Medium:
  - ① 从一个地铁站到另一个站的最短路径是什么?
- Hard:
  - ① 在地铁网络中，如果增加一个新站点，如何选择位置以最大程度优化整个网络?

图算法：回答轨道交通网络问题

1. 某个地铁站有多少条线路经过？
   - 这对应于地铁站在图中的度数（degree），表示有多少条轨道与该地铁站相连
2. 两个地铁站之间是否有直接的轨道连接？
   - 这是一个关于边的问题，检查两个地铁站（节点）之间是否有直接的轨道（边）连接
3. 从一个地铁站到另一个站的最短路径是什么？
   - 涉及最短路径问题和赋权问题
4. 在地铁网络中，如果增加一个新站点，如何选择位置以最大程度优化整个网络？
   - 这是一个网络扩展问题，要求分析新的节点加入如何影响整体网络的效率和连通性。

回顾：图的基础概念

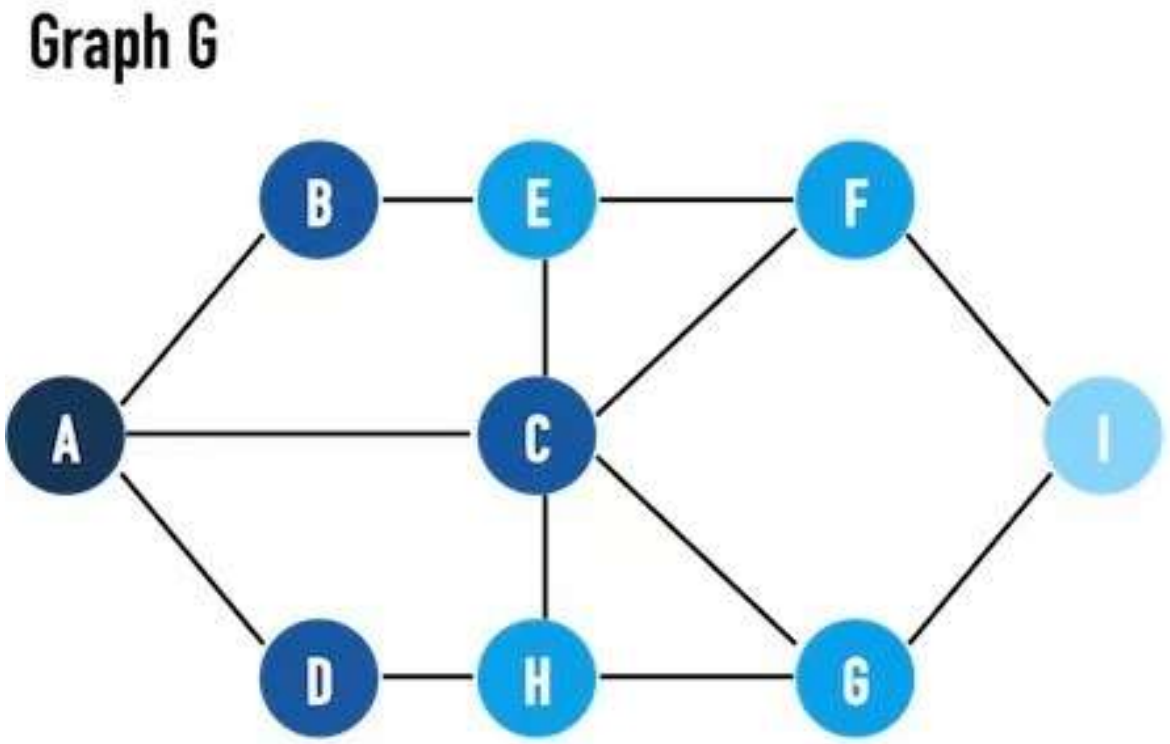
- **无向图**：
  - 无向图是由一个节点集合 V 和一个边集合 E 组成的二元组 $G=(V,E)$，其中：
    - V 是一组节点（顶点），即 $V=\{v\_1,v\_2,\ldots,v\_n\}$，**且V不能为空**
    - E 是一组**无序对**，表示节点之间的连接，即 $E=\{\{u,v\}\mid u,v\in V\}$
- **有向图**：
  - 有向图是由一个节点集合 V 和一个有序边集合 E 组成的二元组 $G=(V,E)$，其中：
    - V 是一组节点（顶点），即 $V=\{v\_1,v\_2,\ldots,v\_n\}$
    - E 是一组**有序对**，表示节点之间的有向连接，即 $E=\{(u,v)\mid u,v\in V\}$



回顾：图的基础概念

- **邻接**：如果边 (u,v) 存在，则节点 u 和节点 v 是邻接的。
- **度**（Degree）：
  - **无向图**：节点的度是与该节点相连的边的数量。
  - **有向图**：节点的**入度**是指向该节点的边的数量，**出度**是从该节点出发的边的数量。
- **路径**（Path）：从一个节点到另一个节点的一个**节点序列**，其中每一对相邻节点之间都有边连接。
- **连通图**（Connected Graph）：在无向图中，任意两个节点之间都有路径相连。
- **强连通图**（Strongly Connected Graph）：在有向图中，任意两个节点之间都有双向路径相连。

Rethink：图的存储（表示方法）



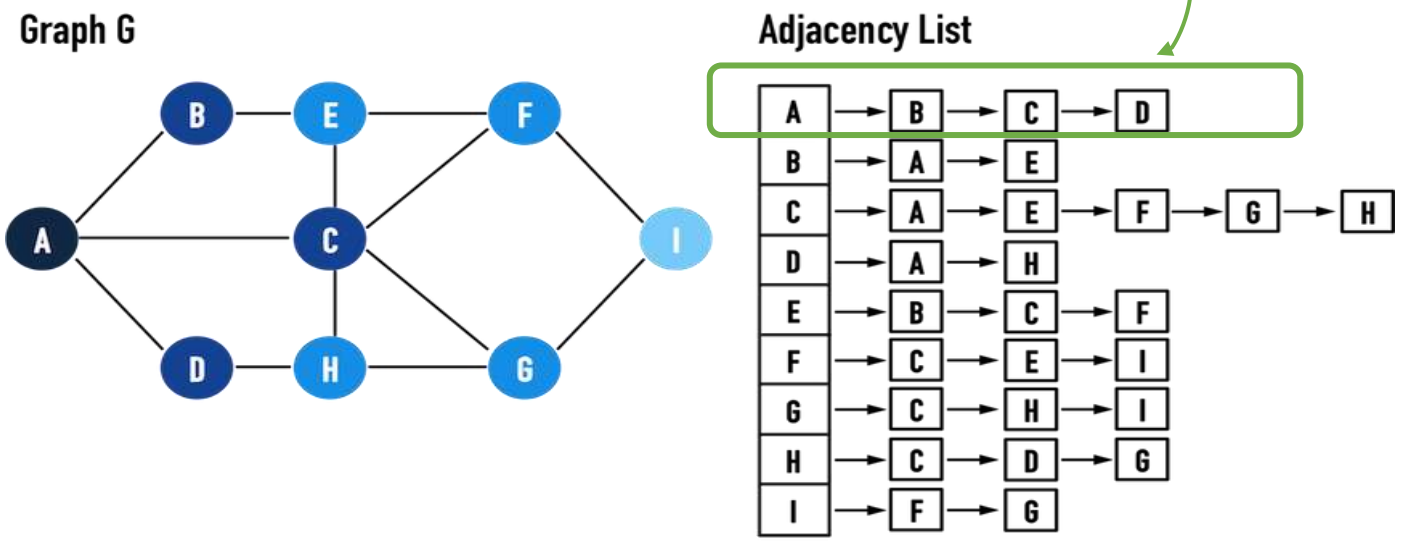
人直观看见的图是一个"一目了然"的整体结构（除非是规模太大、看不清所有顶点和边的图）

Rethink：图的存储（表示方法）

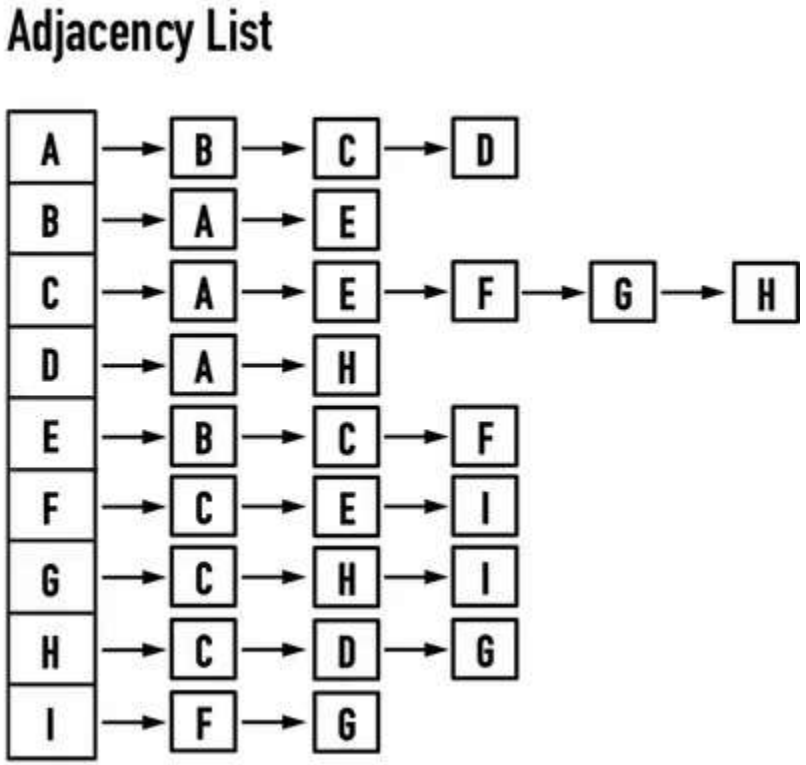
- 计算机，**无法“自然地看见”一个非线性的整体结构**
- 而是**线性地存储**图的“局部信息”
- “局部信息”组成图的“全局信息”

重新思考：图的存储（表示方法）

- 计算机“眼里的图”，局部地存储每个点的**局部线性信息：**
  - 邻接表
  - 邻接矩阵



计算机“眼里的图”



- 直观能“看到的”（**局部线性信息**）：
  ① 有多少个节点
  ② 每个节点有多少个邻接点

- 不能直观“看到的”？（**非线性信息**）：
  ① 点A和点I是否是相连的?
  ② 整个图是否是连通的?

不能“直观”得到部分：
需要图算法的帮助

图算法

- 输入： **给定的图数据结构** + A + B +...
- 经过： 一系列的规则和操作
- 输出： what we want

BFS: Breadth First Search

- BFS本身是一种**【机制、方式】**：
  ①机制：breadth-first，也就是**广度优先**
  ②当把这种机制用到输出具体的东西才是一种算法，比如用breadth-first search的方式来遍历一个图，对应的算法：Breadth-First Search Transversal 广度优先遍历
  ③直观理解：广度优先 = “水平式”

BFS可以做的事情

- 搜索
  - 图中有没有目标节点
  - 从起始节点出发，能否到达目标节点
  - 目标节点与起始节点的位置关系是怎样
- 遍历
  - **遍历的定义**：遍历是指通过某种系统的方法访问数据结构中的所有节点或元素，**每个节点或元素至少访问一次**
  - BFS 遍历：从某点开始"水平式"遍历一个图
    - 输入：图，起始节点；输出：节点被访问的顺序
    - **学BFS遍历就是如何"水平式"遍历：这一系列的规则和操作是什么?**