

第一章 背景知识

1.1 图基本定义

定义 1.1 (标号图^[1]). 一个可以被四元组 $G = (V, E, \Sigma, \lambda)$ 表示的图称为标号图 (*labeled graph*), 其中 V 为有限的节点集合, E 为有限的边集合 $\in V \times V$, Σ 是标号集合, λ 是一个标号函数用于给各个节点与边分配标号 $\lambda: V \cup E \rightarrow \Sigma$.

如图1.1就是一个包含六个节点的标号图。需要注意的是标号与标识的区别, 标号是图的固有属性, 标识只是为了方便使用人为添加的记号。在图1.1中, v_i 就是标识, 而 A, B, C, D 则是标号。

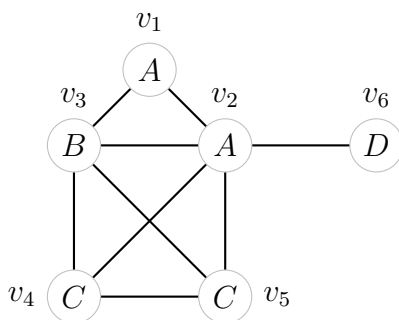


图 1.1 标号图

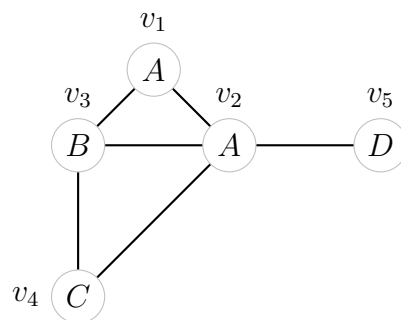


图 1.2 图1.1的子图

定义 1.2 (子图). 如果一幅图 $G = (V, E, \Sigma, \lambda)$ 和另一幅图 $G' = (V', E', \Sigma', \lambda')$ 有 1-1 映射的关系 $f: V \rightarrow V'$, 那么图 G 就是 G' 的子图 (*subgraph*), 用 $G \in G'$ 表示。 f 可以有这么几种

- 对于所有 $v \in V, \lambda(v) = \lambda'(f(v))$
- 对于所有 $(u, v) \in E, (f(u), f(v)) \in E'$
- 对于所有 $(u, v) \in E, \lambda(u, v) = \lambda'(f(u), f(v))$

换言之, 如果一幅图和另一幅图的节点标签, 边关系, 边标签能一一对应上, 那么这副图就是另一幅图的子图。如图1.2就是图一个1.1的子图, 节点标签, 边关系及边标签均可一一对应, 只有标识可以不同。

定义 1.3 (顶点的度^[2]). 一个顶点 u 的度数是与它相关联的边的数目, 记做 $degree(u)$ 。 $degree(v_i) = |E(v_i)|, v_i \in V (i = 1, 2, \dots, n)$, 图1.1中 v_2 的度 $degree(v_2) = 5$, 图1.2中 v_2 的度为 $degree(v_2) = 4$ 。

定义 1.4 (图的尺寸^[3]). 一般由图中节点数 $|V(G)|$ 定义。因此图1.1的尺寸 (*size*) 为 6, 图1.2为 5。

定义 1.5 (路径). 在图 $G(V, E)$ 中, 若从顶点 v_i 出发, 沿着一些边经过一些顶点 $v_{p1}, v_{p2}, \dots, v_{pm}$, 到达顶点 v_j , 则称顶点序列 $(v_i, v_{p1}, v_{p2}, \dots, v_{pm}, v_j)$ 为从顶点 v_i 到 v_j 的一条路径 (*path*)。如在图1.1中, (v_1, v_2, v_5) 就是一条 v_1 到 v_5 的路径。

1.2 图存储表示方式

图存储表示方法有很多种, 常用的有 2 种: 邻接矩阵 (*Adjacency Matrix*) 和邻接表 (*Adjacency List*)。本节还将介绍一种新颖的图表示方法简化包表示 (*Reduced Bag Representation*)^[1]。

1.2.1 邻接矩阵

在邻接矩阵存储方法中, 除了一个记录各个顶点信息的顶点数组外, 还有一个表示各个顶点之间关系的矩阵, 称为邻接矩阵。设 $G(V, E)$ 是一个具有 n 个顶点的图, 则图的邻接矩阵是一个 $n \times n$ 的二维数组, 用 $Edge[n][n]$ 表示, 定义为:

$$Edge[i][j] = \begin{cases} 1, & \text{if } (i, j) \in E \\ 0, & \text{else} \end{cases} \quad (1-1)$$

1.2.2 邻接表

邻接表就是将同一个顶点发出的边连接在同一个称为边链表的单链表中。边链表的每个节点代表一条边, 称为边节点。每个边节点有两个域: 该边终点的序号以及下一个边节点的指针。

1.2.3 包表示法

包表示法是王教授等人在 *G-Hash*^[1] 中提出的一种方法。这种方法的主要思想是将图中的各个节点特征提取出来形成一个列表, 然后将这些特征就作为每个节

点的标识。这样整幅图就变成了一个字符串的包，也就是一组字符串。通常，我们用这个节点的标号和其邻接节点的不同标号数目为特征。例1.1所示就是一个常用的包表示方法

例 1.1. 表1.1是图1.1的包表示。我们以节点标号和其邻接节点各个标号的个数作为特征，因此有五个特征：本身标号，A,B,C,D 分别的个数。所以加上第一列的标识，表一共有六列。而一共有六个不同节点，所以有六行。我们以 v_3 为例，详细说明下抽取特征的步骤。首先 v_3 的标号是 B, 所以 Label 为 B, 然后和 v_3 邻接的共有四个节点，分别是 v_1, v_2, v_4, v_5 ，其标号分别为 A, A, C, C 所以 $\#A = 2, \#C = 2$ 其余为 0。节点 v_3 的包表示就是“B,2,0,2,0”，如表1.1所示。需要注意一点的是，包表示不仅仅这一种表示方法，对于选取什么样的特征并没有限制。不过特征必须是离散的，这样才可以用字符串表示。

Nodes	Label	#A	#B	#C	#D
v_1	A	1	1	0	0
v_2	A	1	1	2	1
v_3	B	2	0	2	0
v_4	C	1	1	1	0
v_5	C	1	1	1	0
v_6	D	1	0	0	0

表 1.1 图1.1的包表示

参考文献

- [1] Xiaohong Wang, Aaron Smalter, and Jun Huan. G-hash:towards fast kernel-based similarity search in large graph databases. *ACM*, 2009.
- [2] 任嘉辰王桂平, 王衍. 图论算法理论, 实现及应用. 北京大学出版社, 2011.
- [3] 谭伟, 杨书新. 图数据精确查询与近似查询的研究. 2013.