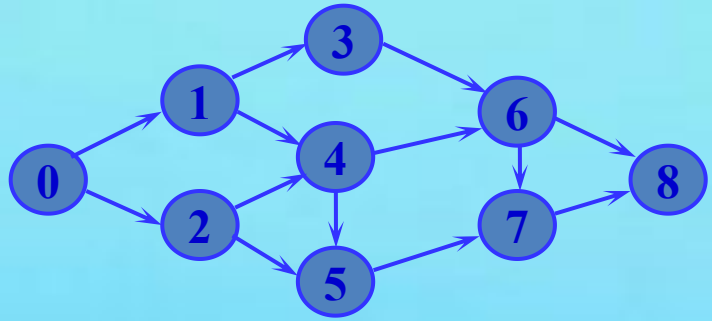
本节主题：

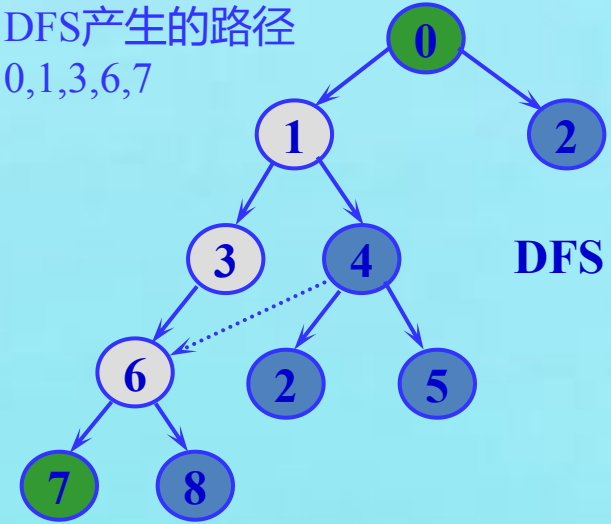
BFS的应用

求最短路径

- 问题
  - 求不带权连通图G中从顶点u到顶点v的一条最短路径。
- 基础
  - 求顶点u和顶点v的最短路径，即求之间边数最少的顶点序列。
- 例如
  - 从顶点0出发找顶点7

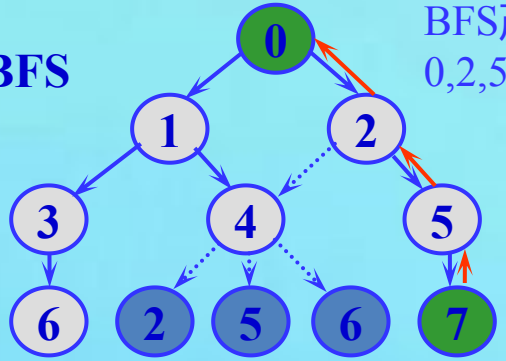


DFS产生的路径
0,1,3,6,7



BFS

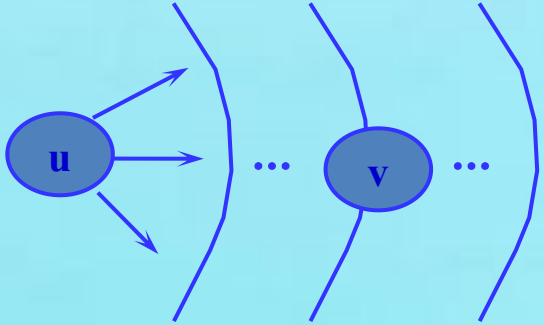
BFS产生的路径
0,2,5,7

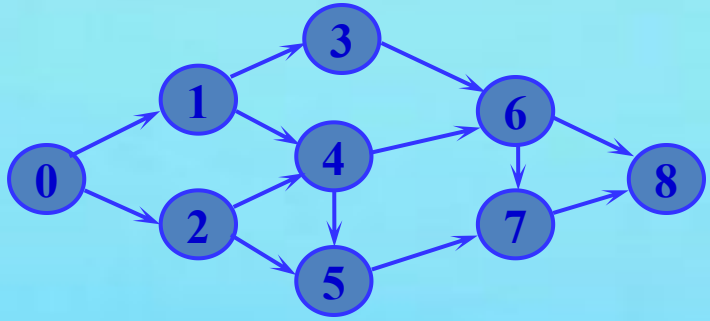


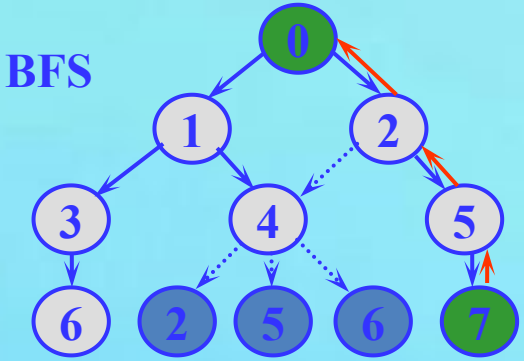
广度优先遍历算法思路

- 从顶点u出发一层一层地向外扩展
- 利用队列记录访问的顺序
- 当第一次找到顶点v时队列中便包含了最短路径
- 利用队列输出最短路径（逆路径）
- 关键的辅助数据结构

  ——只进不出的非环形队列







求最短逆路径算法

```
void ShortPath(ALGraph *G,int u,int v)
{
   ArcNode *p;
   int w,i;
   //设置visited数组
   //设置队列
   while (front!=rear)   //队不空循环
   {
     //出队顶点w
     if (w==v)  //找到终点
     {
       //找到v时输出路径之逆并退出
     }
     //访问w的所有邻接点
   }
}
```

```
int visited[MAXV];
for (i=0; i<G->n; i++)
   visited[i]=0;
visited[u]=1;
```

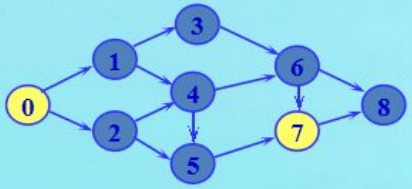
```
QUERE qu[MAXV];
int front=-1,rear=-1;
rear++;
qu[rear].data=u;
qu[rear].parent=-1;
```

```
front++;
w=qu[front].data;
```

```
i=front;
while (qu[i].parent!=-1)
{
   printf("%2d ",qu[i].data);
   i=qu[i].parent;
}
printf("%2d\n",qu[i].data);
break;
```

```
p=G->adjlist[w].firstarc;
while (p!=NULL)
{
   if (visited[p->adjvex]==0)
   {
      visited[p->adjvex]=1;
      rear++;
      qu[rear].data=p->adjvex;
      qu[rear].parent=front;
   }
   p=p->nextarc;
}
```

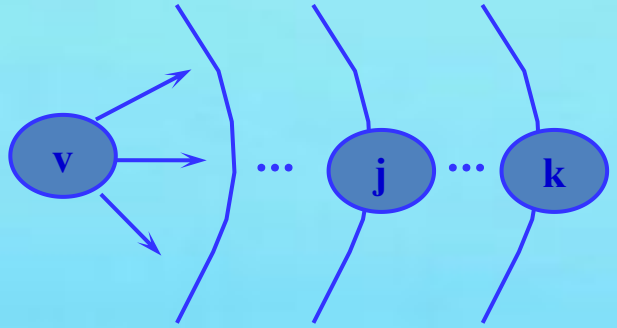
```
typedef struct
{
   int data;
   int parent;
} QUERE;
```



	da	par	
0	0	-1	[0]
1	1	0	[1]
2	2	0	[2]
3	3	1	[3]
4	4	1	[4]
5	5	2	[5]
6	6	3	[6]
7	7	5	[7]
8	8	6	[8]

求最远的顶点

- 问题
  - 求不带权连通图G中，距离顶点v最远的顶点k
- 策略
  - 一层一层向外扩，找到最后一个到达的顶点 —— BFS
- 关键的辅助数据结构
  - 队列（可以是环形）




```
int Maxdist(ALGraph *G,int v)
{
   ArcNode *p;
   int i,j,k;
   //初始化队列和visited数组
   while (rear!=front)
   {
      //顶点k出队
      //访问顶点k的邻接点
   }
   return k;
}
```

```
int Qu[MAXV];
int visited[MAXV];
int front=0,rear=0;
for (i=0; i<G->n; i++)
   visited[i]=0;
rear++;
Qu[rear]=v;
visited[v]=1;
```

```
front=(front+1)%MAXV;
k=Qu[front];
```

```
p=G->adjlist[k].firstarc;
while (p!=NULL)
{
   j=p->adjvex;
   if (visited[j]==0)
   {
      visited[j]=1;
      rear=(rear+1)%MAXV;
      Qu[rear]=j; //进队
   }
   p=p->nextarc;
}
```