Tutorial 4

BFS 基础

BFS skeleton

BFS 的推进过程像是地平线一样层层向外扩展

遍历过程可以看作基于一个“调度器”，遍历当前节点时每发现新节点就将其加入调度器，对下个节点遍历时只需从调度器中取出一个节点。

对于 BFS，调度器就是队列，MST 算法中还会以优先级队列做调度器

BFS-WRAPPER(G)

```
foreach node v in G do
    v.color := WHITE
    v.parent := NULL
    v.dis := INF
foreach node v in G do
    if v.color == WHITE do
        BFS(v)
```

BFS(v)

```
Initialize an empty queue queNode
v.color := GRAY
v.dis := 0
queNode.ENQUE(v)
while queNode != empty do
    w := queNode.DEQUE()
    foreach neighbor x of w do
        if x.color == WHITE do
            x.color := GRAY
            x.parent := w
            x.dis := w.dis + 1
            queNode.ENQUE(x)
```

```
    <processing of node w>
    w.color := BLACK
```

parent, dis 与最短路径，以及围绕 queue 的归纳证明

在 BFS 的过程中，维护了每个节点的 parent 的信息，即根据 BFS 推进过程产生的祖先后继关系，而且维护了从遍历起始节点到每个节点的最短路径信息，对所有节点依据到顶点距离的不同进行了等价类划分

记源点 s 到节点 v 的最短路径长度为 $\delta(s,v)$ ，首先可看出如下性质

> **引理 5.1** 对有向或无向图 G ，对任意边 uv ，有 $\delta(s,v) \leqslant \delta(s,u)+1$
>
> 证明：若源点可达 u ，由于边 uv 的存在，源点可达 v ，由于 s 到 v 的最短路径必然不超过 s 到 v 的路径长，故 $\delta(s,v) \leqslant \delta(s,u)+1$，若源点不可达 u ，则 $\delta(s,u)=\infty$ ，不等式仍然成立

对于 BFS 中的 v.dis 与 $\delta(s,v)$ ，有

> **引理 5.2** 从节点 s 开始 BFS，遍历结束时对任意节点 v 有 $v.dis \geqslant \delta(s,v)$
>
> 证明：对队列上的**操作**归纳，归纳不变量：对任意 v ，有 $v.dis \geqslant \delta(s,v)$
>
> Basis. 队列执行第一个操作，源点入队，此时有 $s.dis=\delta(s,s)=0$ ，对于其他任意节点 v ，有 $v.dis=\infty \geqslant \delta(s,v)$
>
> I.H. 队列进行任意操作后，对任意 v ，有 $v.dis \geqslant \delta(s,v)$
>
> Ind.Step. 队列只有入队和出队操作，节点的 dis 值在入队前决定，且只被赋值一次，故出队对结果没有影响，只考虑入队操作，若在处理节点 u 时发现白色邻居 v ，则根据 BFS 框架
>
> $$v.dis = u.dis + 1$$
>
> 又根据 I.H.，
>
> $$\begin{gathered} u.dis \geqslant \delta(s,u) \\ v.dis \geqslant \delta(s,u)+1 \end{gathered}$$
>
> 根据引理 5.1
>
> $$v.dis \geqslant \delta(s,v)$$
>
> 得证

围绕 BFS queue 的归纳证明还有下面的引理

> **引理 5.3** 在 BFS 过程中队列元素为 $< v_1, v_2, \ldots, v_r >$ ，则有
>
> $$v_i.dis \leqslant v_{i+1}.dis(1 \leqslant i \leqslant r-1), v_r.dis \leqslant v_1.dis + 1$$
>
> 证明：对队列上的操作归纳
>
> Basis. 队列执行第一个操作，源点入队，显然成立
>
> I.H. 执行任意操作后上述结论成立
>
> Ind.Step. 假设 v_1 出队，根据 I.H.，有
>
> $$v_r.dis \leqslant v_1.dis + 1 \leqslant v_2.dis + 1$$
>
> 且队列中元素的序关系仍成立，故出队操作后结论仍成立
>
> 入队时需结合 BFS 框架的运行过程，一个节点入队时一定是之前从队列中取出一个节点进行处理，设取出的节点为 u ，处理 u 时遇到白色邻居 v_{r+1} 将其入队，根据 BFS，$v_{r+1}.dis = u.dis + 1$ ，设 u 出队后队头为 v_1 ，则根据 I.H.，有
>
> $$u.dis \leqslant v_1.dis$$
> $$v_{r+1}.dis = u.dis + 1 \leqslant v_1.dis + 1$$
>
> 设 u 出队之前队头为 u 队尾为 v_r ，同样根据 I.H. 有
>
> $$v_r.dis \leqslant u.dis + 1 = v_{r+1}.dis$$
>
> 队列中其余元素的序关系未受影响，得证

根据上述引理可得

> **定理 5.1** 假设从源点 s 开始 BFS，则对任意节点 v ，有 $v.dis = \delta(s, v)$ ，且从 s 到 v 的由 TE 组成的路径就是 s 到 v 的最短路径

证明可见课本 P.69

BFS edge

对于BFS，类似在 DFS 中讨论的一样，根据遍历的进行，将图中的边分为 TE, BE, DE, CE

TE

处理节点 u 时发现白色邻居 v ，则 uv 为 TE，有向图和无向图的 TE 相似，表示了发现新节点的过程，也表示了遍历的推进。在某个连通片中遍历时，所有 TE 组成的子图是连通的且包含连通片中的所有点，忽略边的方向，这些 TE 构成连通片的一个生成树，即 **广度优先遍历树**

BE

对于有向图，遍历节点 u 时发现其黑色邻居 v ，且 v 是 u 在遍历树中的祖先，则边 uv 为 BE，且有 $0 \leqslant v.dis < u.dis$

对于无向图，不可能存在 BE

DE

BFS 特性决定不可能出现 DE，考虑 uv 为 DE，当 u 离开队列处理其邻居 v 时

- v 不可能是白色，否则 uv 为 TE
- v 不可能是灰色，否则 u 离开队列时 v 在队列中，与 u 是 v 在遍历树中的祖先矛盾
- v 不可能是黑色，u 刚刚开始处理时若 v 已处理结束，同样与 u 是 v 在遍历树中的祖先矛盾

无论是有向图还是无向图的 BFS 均不会产生 DE

CE

处理节点 u 时，发现灰色或黑色节点 v ，且 v 不是 u 的祖先，则边 uv 为 CE（对于无向图，v 只能为灰色）

对于有向图，$v.dis$ 可以小于 $u.dis$ ，可以等于 $u.dis$ ，也可以大于 $u.dis$ ，但最多不超过 $u.dis+1$。

对于无向图，$v.dis$ 可以等于 $u.dis$ ，可以大于 $u.dis$ ，但最多不超过 $u.dis+1$ ，但不同于有向图，$v.dis$ 不能小于 $u.dis$

> **推论 5.1** 对于 BFS 中的 CE
>
> - 有向图中 CE uv 满足 $v.dis \leqslant u.dis+1$
> - 无向图中 CE uv 满足 $u.dis \leqslant v.dis \leqslant u.dis+1$
>
> 对于有向图，考虑 $v.dis$ 最大的可能取值，显然节点发现越晚，dis 越大，由于边 uv 的存在，最迟在遍历 u 时 v 必被访问到 ，此时节点 v 在队列中，根据引理 5.3 ，$v.dis$ 最大为 $u.dis+1$ ，其余情况 v 被发现得更早，$v.dis$ 只会更小
>
> 对于无向图，由于 v 只能为灰色，在队列中，根据引理 5.3 即可得出结论

BFS 应用

二部图

二部图是解决计算机领域的 **匹配问题** 的重要工具

> **定义 5.1** 二部图 给定无向图 $G=(V,E)$ ，称之为二部图，如果存在顶点 V 的划分 V_1,V_2 满足
>
> $$V_1\cap V_2=\varnothing,V_1\cup V_2=V$$
>
> 且使得图中任意边满足其一个端点在 V_1 ，一个端点在 V_2 ，（即 V_1,V_2 内部任意顶点间没有边相连）

容易验证一个图是二部图 $\iff$ 它是可以二着色的，即可以为任意顶点染上两种颜色之一，使得每条边端点的颜色不同

二部图可通过图遍历验证，即在图遍历的过程中对图二着色，当发现矛盾时即说明 G 不是二部图，否则若能无矛盾二着色则说明 G 是二部图。从顶点 v 开始二着色，不失一般性将其染为红色，则其邻居必须染为蓝色，其邻居的邻居必须染为红色……不难看出这一过程与 BFS 的相似，可以将检测二部图的算法嵌入 BFS skeleton

处理每个顶点的邻居时，若未染色，则根据二着色规则为其染色，否则根据二着色规则检测其染色是否矛盾

寻找 K 度子图

给定无向图 G ，则 G 的 k 度子图 H 满足 H 每个顶点的入度都不小于 k

现在需要找出一个给定图的 k 度子图，显然，一个顶点若其在 G 中的度小于 k ，则其不可能属于任何 k 度子图，且与其相连的边也不可能属于任何 k 度子图。则寻找 k 度子图的策略即为扫描图中所有度小于 k 的点，若没有则 G 即为 k 度子图，否则删去这些度小于 k 的点，且这一过程要迭代地向外扩展，若删除某节点导致其邻居的度小于 k ，则其邻居也需加入需要删除的节点

这一过程同样可基于 BFS 实现，删除一个节点时检查其邻居，若其度小于 k 则将其加入调度器

DFS 应用

SCC: DFS or BFS?

性质决定 SCC 算法的遍历过程不能用 DFS 代替 BFS

hint：考虑保证 SCC 算法正确的引理，当 DFS 替换为 BFS 时能否仍保证正确

S-T graph

是否存在一个点，满足 one to all/all to one

解决其中一个问题即可通过图转置解决另一个问题

考虑 one to all 问题

hint：使用 SCC 得到收缩图，收缩图一定是 DAG，考虑其中入度为 0 的点即可

影响力问题

一个点的影响力定义为从其可达的点的数量（不包括自身）

寻找图中影响力最大/最小的点

hint：使用 SCC 得到收缩图，考虑其中入度/出度为 0 的点

小孩排队问题，选课问题

题 4.20，4.22

hint：问题转化为拓扑排序/关键路径