补充定义
辨认安全边
Kruskal算法
并查集---判断连接连通分量
Prim算法
优先队列---维持存储最近权
Boruvka---升级Prim

MIN Spanning Tree

补充定义

- 循环不变式：在每遍循环之前，A是某个MST的一个子集。
- 安全边safe：这条边加入A，不违反循环不变式
- 割cut

> 把点集分成两份
> 也会把连接两部分的边叫做cut边

- respect边集：不包含cut边
- light edge轻量级边：cut边中权重最小的

辨认安全边

- light edge is safe for A

Kruskal算法

> 贪心，每次加入light edge
> 按权重升序判断，连接了不同分量就采用，不维持连通性

```
KruskalMST(G,w):
```

```
A = ∅
Sort edges into weight increasing order
for (each edge (u,v) taken in weight increasing order)
  if (adding edge (u,v) does not form cycle in A)
    A = A ∪ {(u,v)}
return A
```

并查集版本---判断连接连通分量

- FIND-SET（a）==FIND-SET（b）

> 判断结点是否属于同一棵树

- UNION合并树

```
KruskalMST(G,w):
A = ∅
Sort edges into weight increasing order
for (each node u in V(G))
  MakeSet(u)
for (each edge (u,v) taken in weight increasing order)
  if (Find(u) != Find(v))
    A = A ∪ {(u,v)}
    Union(u,v)
return A
```

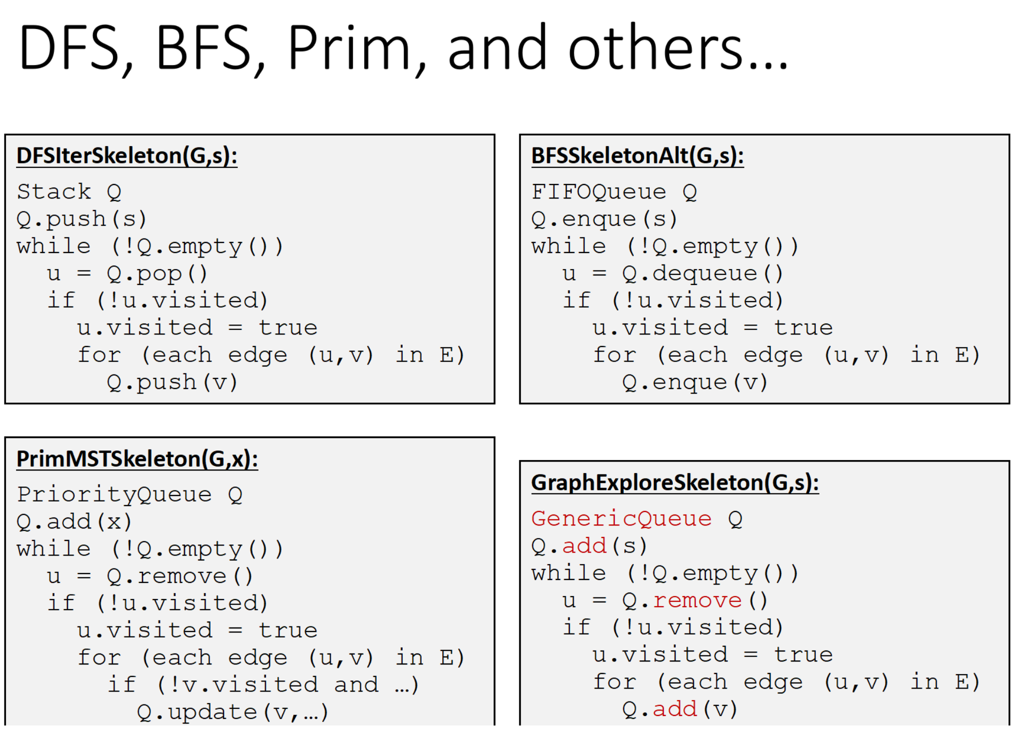
Prim算法

> 贪心，维持连通性，不断在现有所有成员的邻居里扩张
> 起始点随意

```
PrimMST(G,w):
A = ∅
Cx = {x}
while (Cx is not a spanning tree)
  Find MWOE (u,v) of Cx
  A = A ∪ {(u,v)}
  Cx = Cx ∪ {v}
return A
```

优先队列---维持存储最近权

```
PrimMST(G,w):
Pick an arbitrary node x
for (each node u)
  u.dist = INF, u.parent = NIL, u.in = false
x.dist = 0
Build a priority queue Q based on “dist” values
while (Q is not empty)
  u = Q.ExtractMin()
  u.in = true
  for (each edge (u,v))
    if (v.in==false and w(u,v)<v.dist)
      v.parent = u, v.dist = w(u,v)
      Q.Update(v,w(u,v))
```



DFS, BFS, Prim, and others…

DFSIterSkeleton(G,s):
```
Stack Q
Q.push(s)
while (!Q.empty())
  u = Q.pop()
  if (!u.visited)
    u.visited = true
    for (each edge (u,v) in E)
      Q.push(v)
```

BFSSkeletonAlt(G,s):
```
FIFOQueue Q
Q.enque(s)
while (!Q.empty())
  u = Q.dequeue()
  if (!u.visited)
    u.visited = true
    for (each edge (u,v) in E)
      Q.enque(v)
```

PrimMSTSkeleton(G,x):
```
PriorityQueue Q
Q.add(x)
while (!Q.empty())
  u = Q.remove()
  if (!u.visited)
    u.visited = true
    for (each edge (u,v) in E)
      if (!v.visited and …)
        Q.update(v,…)
```

GraphExploreSkeleton(G,s):
```
GenericQueue Q
Q.add(s)
while (!Q.empty())
  u = Q.remove()
  if (!u.visited)
    u.visited = true
    for (each edge (u,v) in E)
      Q.add(v)
```

Boruvka---升级Prim

> prim是对一个连通分支不断扩张
>
> Boruvka对所有联通分支同时扩张

```
BoruvkaMST(G,w):
G’ = (V,Ø)
do
  ccCount = CountCCAndLabel(G’)
  for (i=1 to ccCount)
    safeEdge[i] = NIL
  for (each edge (u,v) in E(G))
    if (u.ccNum != v.ccNum)
      if (safeEdge[u.ccNum]==NIL or w(u,v)<w(safeEdge[u.ccNum]))
        safeEdge[u.ccNum] = (u,v)
      if (safeEdge[v.ccNum]==NIL or w(u,v)<w(safeEdge[v.ccNum]))
        safeEdge[v.ccNum] = (u,v)
  for (i=1 to ccCount)
    Add safeEdge[i] to E(G’)
while (ccCount > 1)
return E(G’)
```