输入：

1. 拓扑
2. 逻辑任务列表（两种类型的，符合约束的）
3. 参与人（权限、意愿、当前位置）

把逻辑任务转换为最优任务链的集合，依照优先级降序进行算法推演。

约束：

1. 逻辑阶段包括移动物体、operation两类
2. 不能有连续的两个移动物体阶段（可以允许发包人这么划分，我们自己做合并）
3. 拓扑图 点代表空间，线代表空间之间的关系，权限放在点上

场景：

笔记本电脑

仓库

资产处（登记）

电脑社团（安全软件）

实验室

发包人指定逻辑阶段（包括有起点有终点的阶段和只有终点的阶段），依照阶段顺序我们可以在拓扑图上得出数条覆盖所有相关地点的可达路径。

所有可达路径都已经按照优先级从高到低排序完毕（涉及的地理拓扑节点少的、总路程少的优先），先从优先级高的选起。其中每条路径都相当于一系列的带顺序的点的集合。

for each route R in AllPossibleRoutesSet{

myResult = findAssignmentDistribution(R, R.src, R.dest);

if(myResult == null) continue;

}

findAssignmentDistribution(route, src, dest){

//findDirectMatch()函数检测有没有worker能独立完成路径R的任务（考虑权限和意愿的限制）

if(findDirectMatch(route, src, dest).size() > 0){

choose worker wd with minimum extra moving distance;

result += (src, dest, wd);

return result;

}else{

if (SrcReachableSet.size()=0 || DestReachableSet.size()==0 )

return null;

for each worker W in SrcReachableSet{

choose worker w1 who can reach the furthest node along the route;

set the furthest node as n1;

}

for each worker W‘ in DestReachableSet{

choose worker w2 who can go furthest by walking backwards on the route;

set the furthest node as n2;

}

if subroute(src, n1) and subroute(n2, dest) have overlapping part{

choose node nj in the overlapping part which causes minimun extra moving distance for w2;

result += (src, nj, w1);

result += (nj, dest, w2);

return result;

}else{

result += (src, n1, w1);

result += (n2, dest, w2);

findAssignmentDistribution(route, n1, n2);

}

}

}