旅行商如何“破圈”——理解Bellman–Held–Karp算法

原创 高睿昊 LOA算法学习笔记 2021-02-04 18:24

“分解问题”是算法设计中常见的灵感源泉。从分治算法到动态规划算法再到贪心算法，其核心思想都是将原问题分解成子问题，缩小问题规模，然后从子问题的解中“组合”、“选择”出原问题的解。套用卜东波老师课堂经典语录即是：

这个问题太难了，不会做，先看看子问题怎么解。

当我们使用分解的方法去设计算法时，找到正确的分解方法是至关重要的。一般而言，我们可以通过观察算法输入的数据结构初步得到分解的思路。例如，当问题的输入是数组时，我们很容易想到将数组一分为二；当问题的输入是树状结构时，我们很容易想到从考察子树的角度入手。

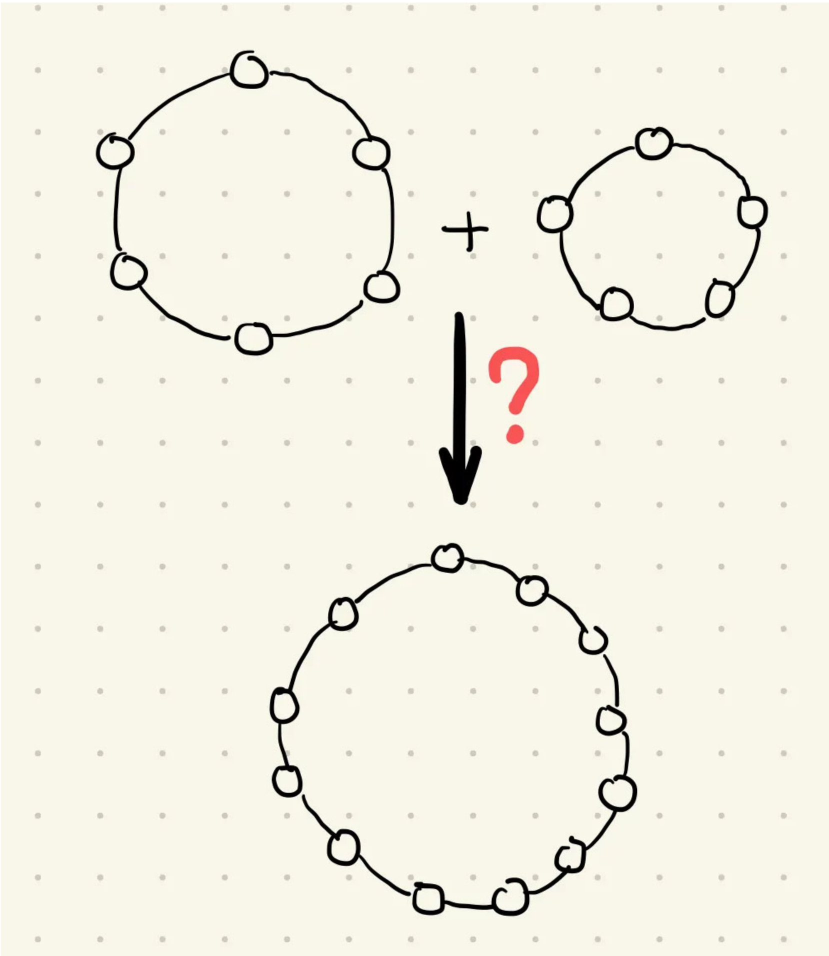
但事情也并非总是这么简单，有些时候，我们可能在“分解”这一环节就遇到困难。旅行商问题（TSP）即是一个很好的、“难以分解”的例子。而**Bellman–Held–Karp**算法则给出了一种巧妙的分解方法。

旅行商问题（Travelling salesman problem）

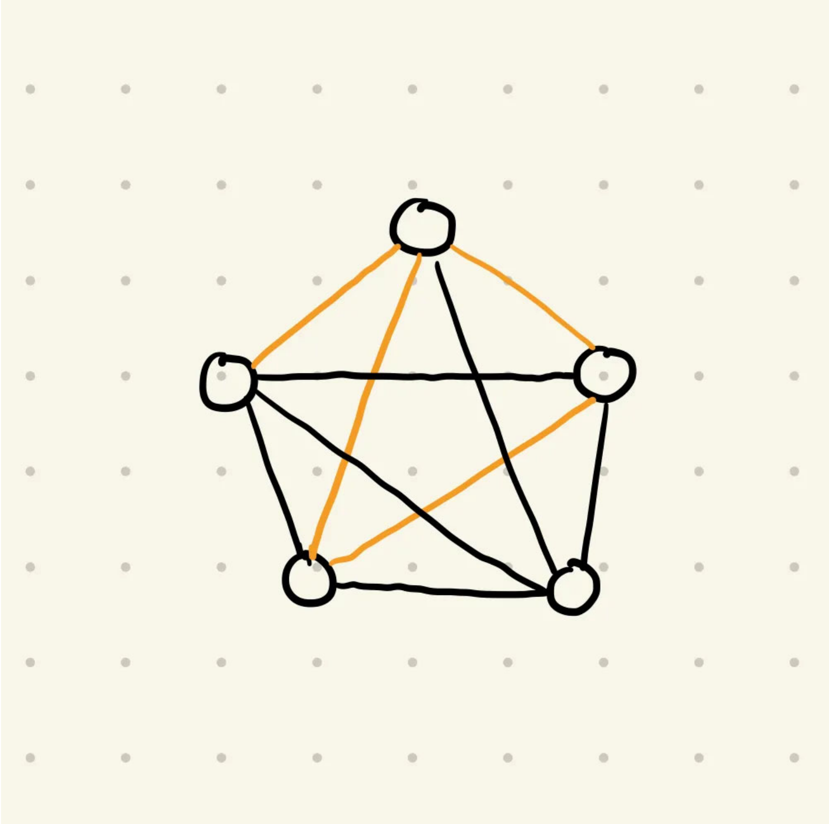
给定一系列城市（C0, C1, ..., Ci, ..., Cn）和城市之间的距离，要求给出在这些城市之间的一个“最短环游”。

我们很自然地想到使用一个带权无向全联通图对其进行建模——以城市作为顶点，城市之间的距离作为边的权重。这样，旅行商问题抽象成了在上述的图中，寻找一个权值最小的**环**。

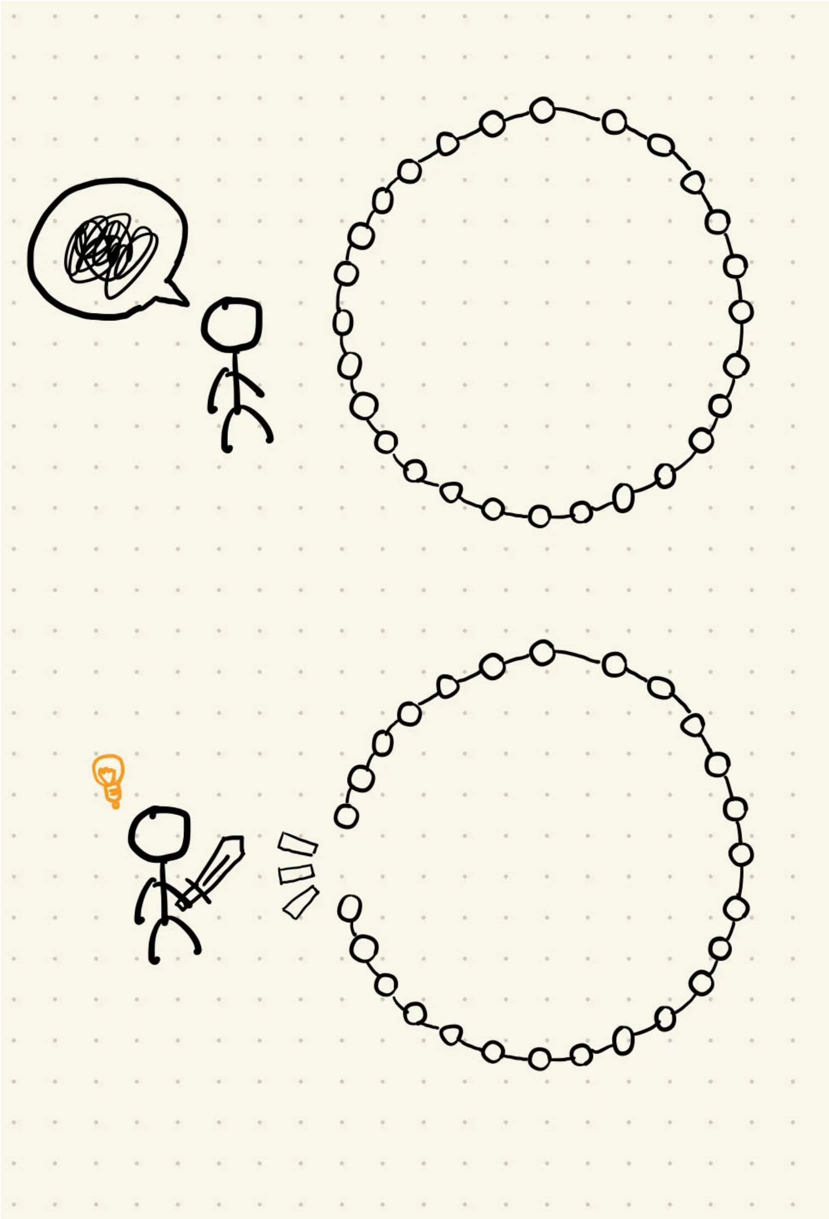
而正是这个环，给问题的分解带来了困难。一方面，环要求覆盖所有顶点。这首先否定了我们**分解顶点集合**的想法——虽然划分顶点集合可以缩小问题规模，但在子问题上构造的“小环”没有办法合并成“大环”。



另一方面，环首位相接的结构特点使得**按边集合分解**的思路也行不通——选出的n-1条边大概率无法构成环。



本着“哪里有压迫，哪里就有反抗”的思想，既然环（圈）带来了困难，那我们就来想办法“破圈”！



Bellman-Held-Karp算法（下文简称BHK算法）的终极奥义即是“破圈”，将寻找环的问题变得可以分解。

从哪里出发?

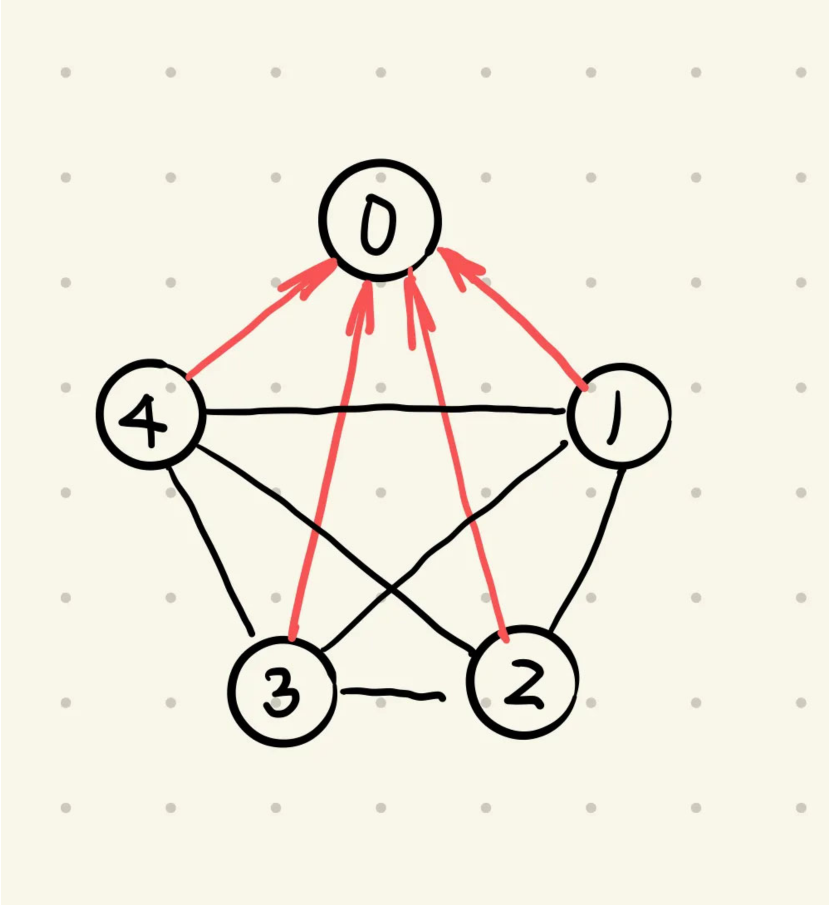
旅行商问题并没有指定出发的城市。在思考算法时，我们应该敏锐的察觉到**问题本身给我们的暗示。**“没有指定出发的城市”很强烈地暗示了我们，从哪里出发其实并不重要！

稍加思考我们就会发现，由于环游覆盖所有城市，并且城市之间都是互相可达的（不存在由于疫情防控带来的封城），选择从哪里出发并不会影响最终的结果。

那么，为了算法实现的便捷性，我们可以选择一个确定的城市作为起点。例如，选择顶点集合中的第一个点作为出发点。

从哪里回来?

当确定从哪里出发后，BHK算法十分巧妙的将原问题分解成了n-1个子问题。让我们通过一个具体的例子来理解这个极其精妙的分解方法。



以包含5个城市的问题为例。当我们确定从0号城市**出发**后，可以将所有的环游分成4类，分别是：从1返回0的环游、从2返回0的环游、从3返回0的环游以及从4返回0的环游。

如果用

$$\mathrm{cd}(0,i)$$

表示从城市0出发，**并从城市i返回城市0**的最短路径，那么TSP问题就可以转化成：

$$min\{\mathrm{cd}(0,1),\mathrm{cd}(0,2),\mathrm{cd}(0,3),\mathrm{cd}(0,4)\}$$

是不是感觉豁然开朗？熟悉的最优子结构回来了！更重要的是，在子问题中，环被消除了，没有了环的掣肘，后续的分解就变得简单了！

继续分解

接下来的工作就是求解cd(0,i)。

此处千万不可犯糊涂，cd(0,i)并不是从城市0到城市i的最短路径。这个路径是要求**经过所有除了0和i的其他节点**的。

看起来还是有点难，但不必担心，在刚刚破圈的思路基础上稍加转换即可找到方法。

设计一个子问题:

$$\mathrm{m}(k,S,j)$$

k和j是两个城市，S是一个城市的集合，上述式子**表示从k出发，经过S中所有城市到达城市j的最短路径。**
以上述表达为基础，cd(0,1)可以用 m(0,{2,3,4},1) 表示。
m(k,S,j) 是易于分解的，可以继续使用确定“从哪里到达j”的思路确定子问题。如果一条路径是从 k出发，最终经过v 到达 j，那么这条路径的长度可以表示为：

$$\mathrm{m}(k, S-\{v\}, v)+d(v, j)$$

d(v,j) 表示从城市v到城市j的距离。即，先求解从k出发，经过不包括的v所有城市(S-{v})，到达v的最短距离，再加上从v到j的距离。
　有了这样的子问题分解，想要求出m(k,S,j) 就十分容易了。

“破圈”的本质

　BHK算法通过确定“从哪里回来”实现破圈，本质上是**选择边的多步决策。**
　当我们确定环游cd最终由城市1返回城市0时，即是选择了从1到0的边；当我们确定最短路径m最终由 v 到达 j 时，即是选择了从v到j的边。
　每一步选择都使得子问题的规模缩小，使得我们可以应用分解的思路来求解问题。

总结

　我认为**Bellman–Held–Karp**算法中最值得我们学习和借鉴的是“破圈”的思想，将不可分解组合的“环”转变成可以分解组合的“路径”。日后，当我们遇到类似难以分解的算法问题时，也可以尝试先寻找转换的突破口。