利用B+树，提高QuCloud中提出初始映射算法的搜索效率

相关主要论文："QuCloud: A New Qubit Mapping Mechanism for Multi-programming Quantum Computing in Cloud Environment"

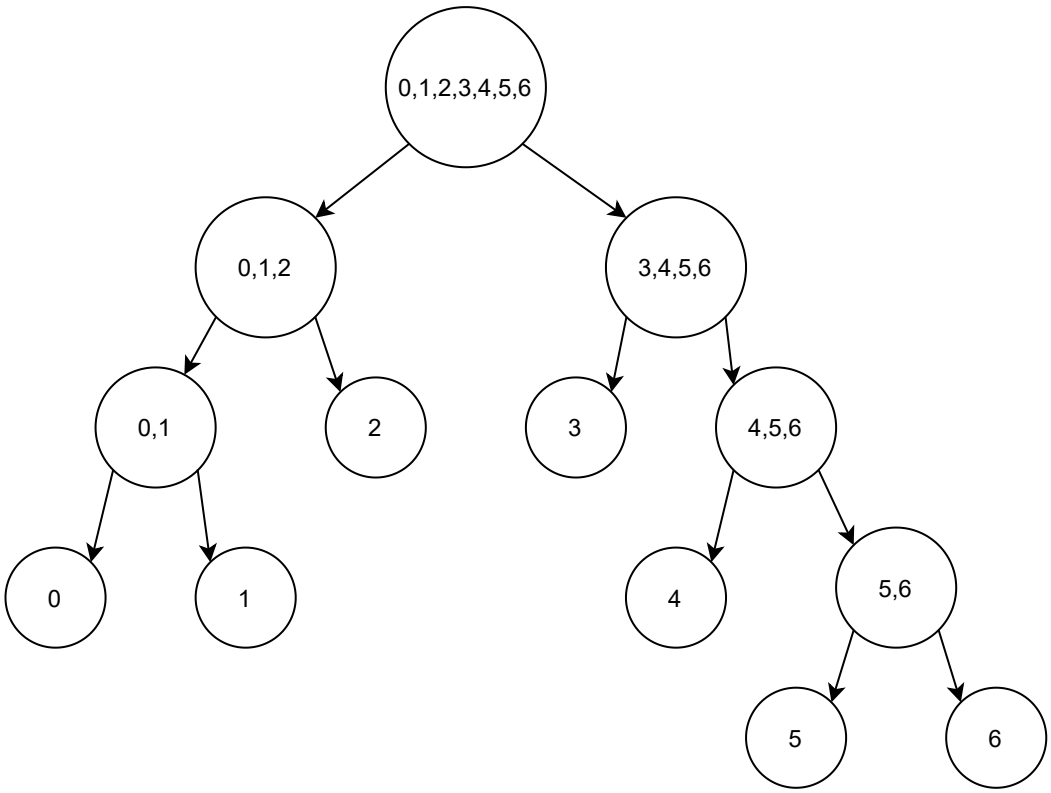
实验代码以及测试结果数据：https://github.com/ex193170/Quantum-Compilation-Efficiency

Motivation

近几年来，有大量的研究关于量子比特如何进行初始化映射，以及Swap Mapping的研究。目前的主要研究方向是**如何得到一个可靠性高的初始化映射，目前许多方法没有考虑到算法性能的需求。**在实际应用中，**随着量子计算机的量子比特数的增加、需要处理的量子程序增多**，如果算法复杂度较高，将会导致**整个映射时间较长而无法满足实际需求。**

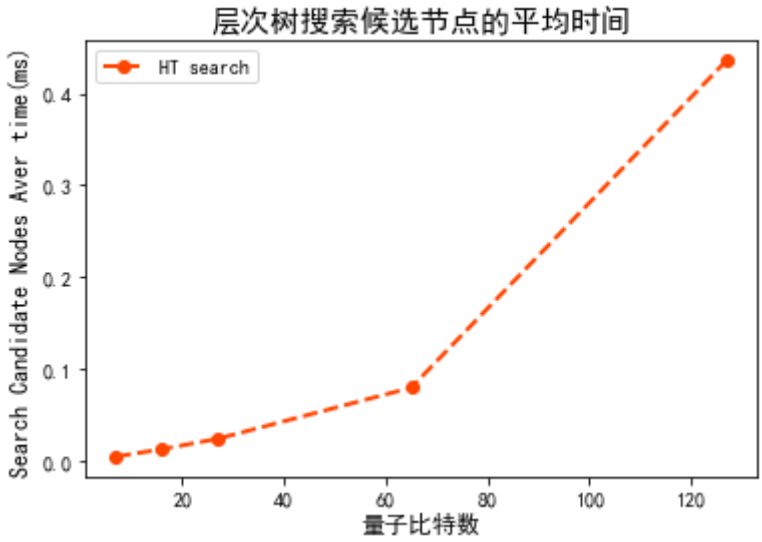
接下来的工作主要是对**QuCloud [9] 中的初始映射算法CDAP性能进行分析与优化。**

我们发现在*QuCloud*的初始映射算法CDAP是基于社区检测算法，通过启发式搜索，最终得到一些可靠性强的映射候选区域，并使用层次树-*HT*（本质就是一个二叉树）。随着量子计算机的量子比特数的增加，*HT*的深度不断的增大。如图是*IBM Q7*通过*CDAP*算法，在权重w=1的情况下，构建得到的HT，其中树的深度为5。对于IBM Q16在w=1的情况下，能够构造出深度为6的HT。**其中构建的HT的深度处于在log2(n)到(n-1)之间。**



CDAP算法中，在采用HT树选择候选节点的时候，遍历叶节点，从叶节点不断向根节点搜索，从而找到量子比特数符合要求的节点，即为所有候选映射区域。比如，在上述IBM Q7的HT上，需要得到对三量子比特的程序的映射区域，我们首先从{0}出发，搜索{0,1},{0,1,2},{0,1,2,3,4,5,6}，然后将{0,1,2},{0,1,2,3,4,5,6}添加进候选映射区域中，然后选择节点{1},{2},{3},{4},{5},{6}，并搜索其父节点，最后得到全部符合量子程序要求的候选映射区域。

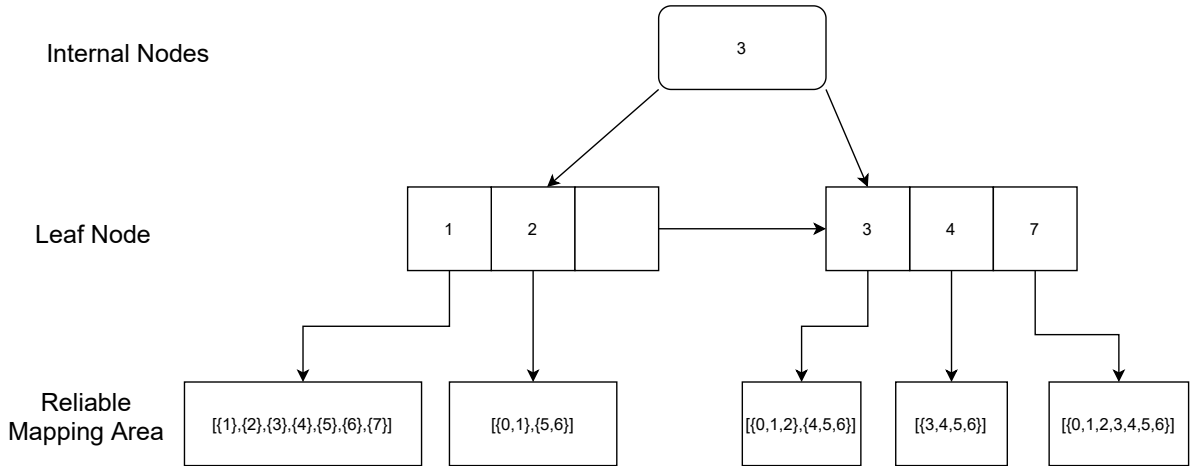
我们采用IBM云平台（https://quantum-computing.ibm.com/services?services=systems）中提供的实际量子计算机的物理量子比特分布图以及相关数据进行实验，其中量子比特数分别为7、16、27、65以及127。如下图，我们可以发现，**随着量子计算机的规模增大，采用HT搜索消耗的时间呈非线性增大。**



Improved Implementation

对于上述使用HT的搜索方法，其搜索方法效率十分的低下，并且搜索的算法时间复杂度相对来说较高。为了能够提高候选映射区域的搜索效率，我们从数据库的搜索中获取灵感，**采取B+树的结构来存储可靠映射区域。**

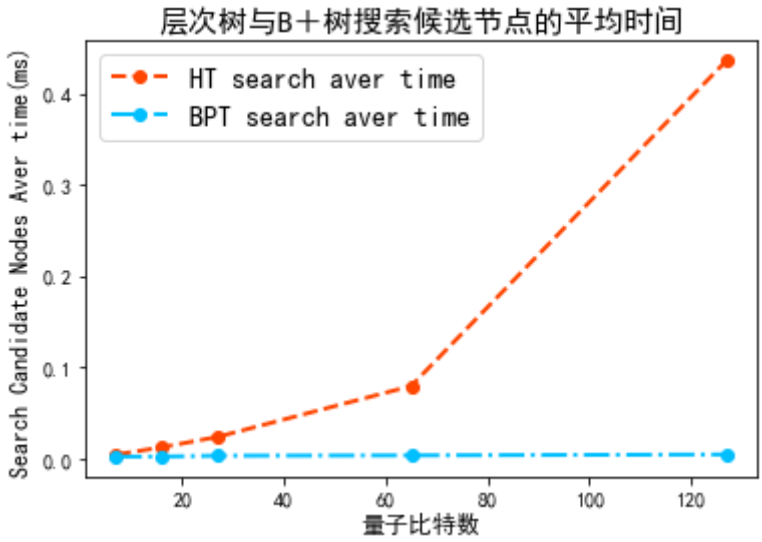
如下图，我们对IBM Q7量子计算机，使用HT树来构建了一个B+树来存储可靠候选区域节点。其中**叶节点代表可靠区域的量子比特数并能指向对应可靠映射区域的具体数据**，内部节点仅仅用来索引信息，可靠映射区域数据通过叶节点来进行直接索引。



使用B+树进行存储可靠映射区域，**如果有量子比特数为3的量子程序，需要寻找符合其量子比特数要求的所有候选映射区域。**

首先从根节点{3}出发，进入叶节点{3,4,7}，找到第一个量子比特数大于等于3的节点，然后取出所有叶节点{3}右边的所有数据，最终得到所有候选区域为[{0,1,2},{4,5,6},{3,4,5,6},{01,2,3,4,5,6}]。在这个过程中，我们只需要遍历一次，访问两次节点即可，一次是根节点，然后一次是叶节点{3,4,7}，**因此，相应是搜索时间将会大大降低。**

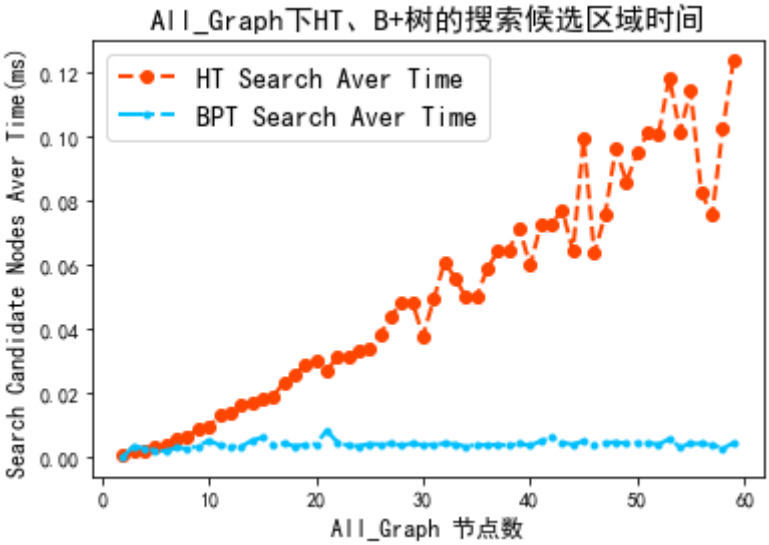
我们采用上述使用过的IBM Q7、IBM Q16、IBM Q27、IBM Q65、IBM Q127的实际量子计算系统。**我们分别使用HT以及B+树的进行候选映射区域搜索，并统计所消耗的平均时间**（搜索量子比特数[1,n]的量子程序消耗总时间/n），如下图：

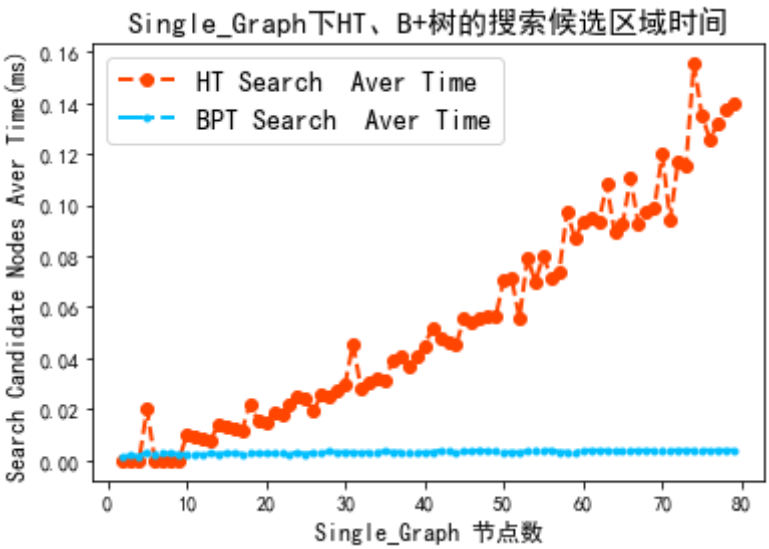


通过IBM的实际系统数据进行实验，表明使用B+树进行存储可靠映射区域，的确能够提升可靠候选映射区域搜索的效率。

为了更好的验证结果的准确性，我们通过随机生成大量规模连续的量子计算机数据。主要分为两类，All_Graph 代表完全图，Single_Graph代表环形图（每个节点有且仅有两条边，比如0节点连接1节点以及n-1节点，n为节点总个数）。

我们分别统计Single_Graph和All_Graph使用HT和B+树进行搜索所需时间，如下图：





实验结果与实际系统的实验结果相似。随着量子比特数的增加，使用HT树的搜索效率要慢与使用BPT树的搜索速度。

Other

1、关于**权重w**，权重w是在CDAP社区检测算法中使用的一个参数，用于调节拓扑结构与错误率之前的权重比例。w会一定程度上影响HT的深度，理论上，w越大，HT树的深度会越大，冗余量子比特数（比如你映射量子比特为3的量子程序，但是可靠区域是只有2或者4，则3为冗余量子比特数）会较少，可能会退化为贪心算法。这样随着w增大，理论上，HT的搜索消耗时间应该会增大，B+树的搜索消耗时间也可能会增大。

2、关于**社区检测算法的时间复杂度较高**（构造HT树使用的时间）：通过实验发现，随着量子比特树的增加，通过计算F值（奖惩值）来构建HT的时间呈多项式增加，大概是在O(n^2)到O(n^4)之间，其中使用Single_Graph大概呈现O(n^2)的时间复杂度，使用All_Graph大概呈现O(n^3)的时间复杂度。大概在实际系统中，构造IBM Q127大概需要花费24s，而IBM Q7只需要花费0.0011s。如下图：