ch9_解答

T9.1

9.1 利用图的匹配算法，给出定点覆盖问题的一个近似比为 2 的近似算法。

定点覆盖问题是一个NP难问题，所以我们需要找到一个近似算法来解决它。这里我们可以利用图的最大匹配算法来给出一个近似比为2的近似算法。

算法步骤如下：

1. 给定一个无向图G(V,E)。
2. 使用匈牙利算法（Hungarian Algorithm）或其他最大匹配算法找到G的一个最大匹配M。
3. 对于M中的每一条边（u,v），将顶点u和v都加入定点覆盖集合C。这样每条边都至少有一个端点在C中，且C是一个定点覆盖。
4. 返回定点覆盖集合C。

这个近似算法的近似比为2，证明如下：

- 由于每次我们都选择一对匹配的顶点加入定点覆盖集合C，所以|C| = 2 * |M|。
- 设OPT为定点覆盖问题的最优解，那么在最优解中，每个顶点至多覆盖一条匹配边。所以OPT >= |M|。
- 结合上述两个结论，我们得到：|C| = 2 * |M| <= 2 * OPT。因此，该算法的近似比为2。

T9.2

9.2 将最短并行算法中任务的任意顺序修改为处理时间的递增顺序，证明修改后的算法的近似比为 4/3。

我们来考虑最短并行算法（List Scheduling Algorithm）的一个变种，即将任务按照处理时间的递增顺序进行调度。我们将证明这个修改后的算法的近似比为4/3。

假设我们有m个处理器和n个任务，任务按照处理时间递增顺序进行排序。我们使用LPT（Longest Processing Time）表示这个修改后的算法。

证明如下：

1. 设Cmax表示所有处理器上最长的完成时间。
2. 设T表示所有任务的处理时间之和，即T = Σ ti（i从1到n）。
3. 设Tmax表示最长处理时间的任务，即Tmax = max{t1, t2, ..., tn}。
4. 设L表示LPT调度的完成时间。

我们需要证明：L <= 4/3 * OPT，其中OPT是最优解。

根据平均负载原理，任何调度算法的Cmax至少需要满足以下限制：

- OPT >= T/m，因为每个处理器至少需要处理总处理时间的1/m。
- OPT >= Tmax，因为最长的任务至少需要处理它自己的处理时间。

结合以上两个限制，我们得到OPT >= max{T/m, Tmax}。

根据Graham证明的一个结论，LPT算法的调度时间满足：L <= 4/3 * Tmax。

现在，我们可以证明L <= 4/3 * OPT。分两种情况：

情况1：Tmax >= T/m，那么OPT >= Tmax。由L <= 4/3 * Tmax，我们得到L <= 4/3 * OPT。

情况2：Tmax < T/m，那么OPT >= T/m。又因为L <= 4/3 * Tmax < 4/3 * T/m，所以L < 4/3 * OPT。

综上，我们证明了LPT算法的近似比为4/3。

T9.3

> 9.3 利用最小生成树算法和最大权值匹配算法给出满足三角不等式的旅行商问题的一个 3/2-近似算法。（提示：利用最大匹配改造最小生成树使其存在哈密顿环，分析近似比时注意使用三角不等式）。

对于满足三角不等式的旅行商问题（TSP），我们可以利用最小生成树算法（如Kruskal或Prim算法）和最大权值匹配算法（如Blossom算法）来设计一个3/2–近似算法。

算法步骤如下：

1. 给定一个满足三角不等式的完全图G(V, E)，其权重函数为w。

2. 在G中找到一个最小生成树T。
3. 找到T中的顶点集V中的所有奇度顶点，记为O。
4. 在图G的子图G[O]中找到一个最小权重完美匹配M。
5. 将T中的边和M中的边进行并集，得到一个新的子图H。
6. 对子图H进行欧拉回路的访问，得到一个欧拉回路EulerTour。
7. 对EulerTour进行shortcutting操作，确保每个顶点仅访问一次，得到一个哈密顿回路HamiltonianCycle。

现在，我们来分析这个算法的近似比：

设TSP_OPT是TSP问题的最优解。由于最小生成树T是TSP_OPT的一个下界，我们有：

T <= TSP_OPT

因为我们在步骤4中找到了G[O]的一个最小权重完美匹配M，又由于三角不等式，可以得到M的权重和至多是TSP_OPT的1/2。所以：

M <= 1/2 * TSP_OPT

所以，我们得到子图H的权重和满足：

H = T + M <= TSP_OPT + 1/2 * TSP_OPT = 3/2 * TSP_OPT

在步骤7中，由于三角不等式，shortcutting操作不会增加哈密顿回路的权重和。因此，得到的哈密顿回路HamiltonianCycle的权重和仍然满足：

HamiltonianCycle <= 3/2 * TSP_OPT

所以，我们证明了这个算法是一个3/2-近似算法。

T9.4

> 9.4 Steiner tree 问题的输入是加权连通图 $G=(V,E)$和一个顶点子集 $T\subseteq V$，输出是总权值最小的一个边集子集 $E'\subseteq E$ 使得在 T 中的任意两个顶点在 $G'=(V,E')$上是连通的。试利用最短路径算法和最小生成树算法为 Steiner tree 问题设计一个近似比为 2 的近似算法。（提示：分析近似比时需要利用课堂讲授的结论）

为了设计一个Steiner树问题的近似比为2的近似算法，我们可以使用最短路径算法和最小生成树算法。以下是算法的步骤：

1. 给定一个加权连通图G=(V,E)和一个顶点子集T⊆V。
2. 计算T中所有顶点对之间的最短路径。可以使用Floyd-Warshall算法、Dijkstra算法或者其他最短路径算法。

3. 构造一个新图G'=(T, E')，其中E'是T中所有顶点对之间的最短路径边的集合。这样，G'是一个完全图。
4. 在G'中计算最小生成树MST(G')。可以使用Kruskal算法或者Prim算法。
5. 将MST(G')的边转换回原图G中的边。我们可以通过在原图G中找到对应的最短路径边来实现这一步。
6. 返回转换后的边集E''，即得到的Steiner树。

下面我们分析这个算法的近似比：

1. 设OPT表示Steiner树问题的最优解。
2. 设MST(T)表示在G'中的最小生成树的权重和。
3. 设E''表示最后得到的Steiner树的边集。

我们已知：

MST(T) <= 2 * OPT

这个结论可以从以下事实推导出：对于Steiner树的最优解，可以找到一个生成树，使得生成树的权重和不超过最优解的两倍。这个结论可以通过将Steiner树中的每个Steiner点替换为从该点到连接的终端点的边来证明。

在步骤5中，我们将MST(G')的边转换回原图G中的边，由于我们使用了最短路径来构造G'，转换后的边权重和不会增加。因此，我们得到：

E'' <= MST(T)

结合上述两个不等式，我们得到：

E'' <= MST(T) <= 2 * OPT

所以，我们证明了这个算法是一个近似比为2的近似算法。

T9.5

10.5试修改 10.3.1 的集合覆盖算法求解加权集合覆盖问题，并分析它的近似比。

加权集合覆盖问题可以描述为：给定一个有限集合U，一个包含U的子集的集合S，以及每个子集的权重。目标是找到一个子集的子集，它们的并集等于U，并且具有最小的总权重。

我们可以修改贪心的集合覆盖算法来解决加权集合覆盖问题，并分析它的近似比。

以下是加权集合覆盖问题的贪心算法：

1. 初始化一个空集合C，表示选择的子集。
2. 当U不为空时，重复以下步骤：

a. 选择一个子集S_i，使得S_i与U的交集除以S_i的权重（即效用函数）具有最大值。

b. 将S_i加入C。

c. 从U中移除S_i中的所有元素。

现在，我们分析这个算法的近似比：

1. 设OPT表示加权集合覆盖问题的最优解。
2. 设C表示贪心算法找到的解。

为了分析近似比，我们需要找到一个比较C和OPT的上界。我们可以观察到，贪心算法在每一步都选择了具有最大效用函数值的子集。这意味着在每一步，贪心算法选择的子集至少和OPT中任何子集一样好。因此，我们可以得出以下结论：

C <= H * OPT

其中H是|U|的哈尔摩尼数，即H = Σ(1/i)（i从1到|U|）。哈尔摩尼数的上界为ln(|U|) + 1。

因此，我们证明了加权集合覆盖问题的贪心算法的近似比为O(log |U|)。