# 计算理论作业 10

张配天-2018202180

2021 年 6 月 6 日

## 1

证明 $IS \leq_p SAFEMARRIAGE(SM)$。设 $E$ 为 IS 的一个实例, $G(V', E')$ 是 $E$ 的图, 共 $n$ 个节点。

构造 $G$, 有 $2n$ 个节点, 对于 $G$ 中节点 $u_i, v_i$, 其由 $G'$ 中的 $p_i$ 得到, 且若 $p_i \in S$, 则只需 $E$ 满足 $IS$ 当且仅当 $G$ 有大小为 $k$ 的 SM。

若 $E$ 满足 IS, 由于 $p_i \in V'$ 在 $V$ 中被分裂为 $u_i, v_i$, 对 $p_i \in S$, $u_i, v_i$ 相连, 由于 $|S| = k$, 那么 $G$ 中任意连通分量的节点数为 2 或 1, 因此 $G$ 有大小为 $k$ 的 SM。将 $u_i, v_i$ 合并为 $p_i$, 之后将 $p_*$ 连伟一个分量, 就能得到 $E'$。

得证。

## 2

### 2.1

先证 DS 是 NP 的, 对于一个验证 $V$, 对于输入 $\langle\langle G, k\rangle, S\rangle$, 检查 $S$ 是否为 $G$ 节点的子集且 $|S| < k$, 检查 $G$ 中任意节点是否在 $S$ 中有一个邻居, 如果通过则接受; 因此 DS 为 NP。

### 2.2

再整 DS 是 NP-hard, 证明 3SAT$\leq_p$DS。设 $E$ 是 3AT 的一个实例, 有 n 个变量和 m 个项, 构造图 G, 有 3n+m 个节点, 其中标记为 $x_i, \bar{x_i}, y_i$, 他们互相连结构成三角形, $c_i$ 与其组成组相连, 这个构造在线性时间内完成。

假设 E 是满足 3SAT 的, 则存在一组赋值为 $c_i$ 均为 true, 构造 S 集, 若 $x_i$ 为 true, 则 $x_i \in S$, 有 $|S| = n$。因此 tS 满足 DS 集的条件, 则 G 有一个最大为 $n$ 的 DS 集。

## 3

### 3.1

先证明 subgraph isomorphism 属于 NP

给定 G1 的子图 G2' 以及 G2 ‘和 G2 之间结点的同态映射关系，可以在多项式时间内验证 G2 ‘于 G2 是否同构，首先判断 G2 ‘和 G2 的映射关系是否是双射，再看 G2 ‘中每一个 (u,v)，是否有对应的 (f(u),f(v)) 属于 G2

### 3.2

证明 CLIQUE 问题可以规约到 subgraph isomorphism

f(<G,k>) = <G1,G2>

假设 CLIQUE 问题的输入是 <G, k>

令 G2 是一个有 k 个结点的完全图，G1=G，那么问题 “G1 是否包含一个于 G2 同构的子图” 其实就是 G 是否包含一个大小为 k 的 clique

显然，如果 <G,k> 属于 CLIQUE，那么 <G1,G2> 属于 subgraph isomorphism

显然，如果 <G,k> 不属于 CLIQUE，那么 <G1,G2> 不属于 subgraph isomorphism

## 4

### 4.1

TRUE-SAT 问题属于 NP

使用 NP 问题的另一个定义。

考虑问题 $R = \{(F, X)|\text{F is a boolean formula, X are variables, F(X)=true}\}$

令 y 为 variable 的一组赋值，y 满足 $|y| \leq |x|^k$，可以在多项式时间内判定 y 能否使得该 boolean formula F 为 true(将赋值代入，计算一下结果)，则 <x, y> 属于 R 可在多项式时间内判定

### 4.2

下面证明 $3 - SAT \leq_m TRUE - SAT$

“YES to YES”

如果存在一组 variable 可以满足该 3-SAT 问题

- 令 Y=false，由该组 variable 满足 SAT 可知，G 中含有 4 个 variable 的 clause 都是 true，并且由于 Y==false，$(\neg Y \cup X_i)$ 均为 true，所以至少有 Y 为 false 的一组 variable 使得 G=true，所以构造出的 G 属于 TRUE-SAT

“NO to NO “

如果存在一组 variable 满足构造出的 TRUE-SAT

那么把满足该 TRUE-SAT 问题的 variable 中 $X_i$ 的赋值都赋给对应的 SAT 问题，这组 variable 可以使得该 SAT 问题为 true

# 5

## 5.1 a.

给定一个 deterministic single-tape TM M0 在多项式时间内判定问题 SPATH
Run M0 on input <G,a,b,k>:

- 用 M0 模拟 Dijkstra 算法，起点设置为 a
- 运行完 Dijkstra 算法以后，看 a 和 b 之间的最短路径的长度是否 k，如果小于等于 k，就 accept，否则 reject

## 5.2 b.

### 5.2.1

首先证明 LPATH 属于 NP 给定一条从 a 到 b 到简单路径，去计数该路径到长度（显然多项式时间复杂度），如果长度大于等于 k，那么 <G,a,b,k> 属于 LPATH

### 5.2.2

下面证明 LPATH 属于 NP-HARD
将哈密顿路径问题规约到 LPATH 问题
记哈密顿问题的输入为 <G, s, t>
f(<G, s, t>) = <G, a, b, k>

- 将图 G 中的 s 点记作 a 点，将 t 点记作 b 点
- 令 k=n-1（n 为图 G 中点的个数）

下面说明 “YES to YES “

如果 G 中 s 到 t 之间存在哈密顿路径，那么该路径的长度一定为 n-1（因为只能经过每个点一次），那么该路径就是 s 到 t 之间长度为 n-1 的简单路径

“NO to NO “

如果 G 中不存在 s 到 t 之间到哈密顿路径，即 a 和 b 之间不存在一条长度为 n-1 的简单路径。同时这也说明了 a 和 b 之间不存在一条长度大于 n 的简单路径，因为如果长度大于 n，而图中只有 n 个点，势必会有至少一个点在路径中出现了两次，则该路径不是简单路径。综合上述，如果 G 中不存在 s 到 t 之间到哈密顿路径，即 a 和 b 之间不存在一条长度大于等于 n-1 的简单路径。

# 6

## 6.1

首先证明这是一个 NP 的问题

对于 PUZZLE 问题，如果给定一个卡牌的排列方式，我们显然可以在多项式时间内判定该解是否属于 PUZZLE，因此 PUZZLE 是 NP

### 6.2

下面将 3SAT 规约到 PUZZLE
对于一个给定的 boolean formula $\phi$, 将它转化为下面这些卡牌：
令 $x_1, x_2...x_m$ 是 $\phi$ 的 variable，令 $c_1, c_2...c_l$ 是 clause
如果 z 是 $c_j$ 的 literal，记作 $z \in c_j$
将每一张卡牌的两列洞记作 column1,column2，每一列有 l 个可能的洞
将 column1 在左边 column2 在右边的情况，记作卡牌正面向上，反之为背面向上
下面规约形式化的表示为
F = "On input $\phi$:

1. Create one card for each $x_i$ as follows: in column 1 punch out all holes except any hole j such that $x_i \in c_j$; in column 2 punch out all holes except hole j' such that $\overline{x_i} \in c_j$
2. Create an extra card with all holes in column 1 punched out and no hole in column 2 punched out
3. Output the description of cards created."

"YES to YES"
给定一个满足 $\phi$ 的赋值方式，用下面的方式构造 PUZZLE 的解。
For each $x_i$ assigned True(False), put its card face up(down), and put the extra card face up. Then, every hole in column 1 is covered because the associated clause has a literal assigned True, and every hole in column 2 is covered by the extra card.
"NO to NO"
给定一个 PUZZLE 的解，用下面的方式构造满足 $\phi$ 的赋值方式。
Flip the deck so that the extra card covers column 2. Assign the variables according to the corresponding cards, True for up and False for down. Because every hole in column 1 is covered, every clause must contain at least one literal assigned True in this assignment. Hence it satisfies $\phi$.

## 7

### 7.1

证明 3-color 是 NP
给定一种给图中每一个点的染色方式，很容易在多项式时间内验证这是否是满足 3color 问题的染色方式（只需要遍历每一个结点，并比较其和邻居结点的染色方式是否相同）

## 7.2

下面证明 $3-SAT \leq_m 3-COLOR$

对于一个给定的 3cnf boolean formula $\phi = c_1 \wedge c_2 \wedge c_3 \wedge ... \wedge c_l$,

令 $x_1, x_2 ... x_m$ 是 $\phi$ 的 variable, $c_i$ 是 clauses

建立一个图 $G_\phi$ containing 2m+ 6l+ 3 nodes:

- 2 nodes for each variable
- 6 nodes for each clause
- and 3 extra nodes

用 hint 中的 subgraph gadget 来描述该图

$G_\phi$ contains a variable gadget for each variable xi, two OR-gadgets for each clause, and one palette gadget.

The four bottom nodes of the OR-gadgets will be merged with other nodes in the graph, as follows.

Label the nodes of the palette gadget T, F, and R. Label the nodes in each variable gadget + and —and connect each to the R node in the palette gadget. In each clause, connect the top of one of the OR-gadgets to the F node in the palette. Merge the bottom node of that OR-gadget with the top node of the other OR-gadget. Merge the three remaining bottom nodes of the two OR-gadgets with corresponding nodes in the variable gadgets so that if a clause contains the literal xi, one of its bottom nodes is merged with the + node of xi whereas if the clause contains the literal $\overline{x_i}$, one of its bottom nodes is merged with the —node of xi.

"YES to YES"

The three colors are called T, F, and R. Color the palette with its labels. For each variable, color the + node T and the —node F if the variable is True in a satisfying assignment; otherwise reverse the colors. Because each clause has one True literal in the assignment, we can color the nodes of the OR-gadgets of that clause so that the node connected to the F node in the palette is not colored F. Hence we have a proper 3-coloring.

"NO to NO"

If we start out with a 3-coloring, we can obtain a satisfying assignment by taking the colors assigned to the + nodes of each variable. Observe that neither node of the variable gadget can be colored R, because all variable nodes are connected to the R node in the palette. Furthermore, if both bottom nodes of an 0R-gadget are colored F, the top node must be colored F, and hence, each clause contain a true literal. Otherwise, the three bottom nodes that were merged with variable nodes would be colored F, and then both top nodes would be colored F, but one of the top nodes is connected to the F node in the palette.