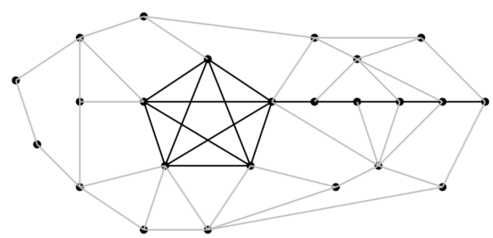
COMP 3804 — Solutions Tutorial April 5

Question 1: Let $K \geq 3$ be an integer. A K-*kite* is a graph consisting of a clique of size K and a path with K vertices that is connected to one vertex of the clique; thus, the number of vertices is equal to $2K$. In the figure below, the graph with the black edges forms a 5-kite.



The *kite problem* is defined as follows:

$$\text{Kite} = \{(G, K) : \text{ graph } G \text{ contains a } K\text{-kite}\}.$$

Prove that the language Kite is in **NP**.

Solution: The *verification algorithm* $\mathcal{V}$ does the following:

- It takes as input
  - a graph $G = (V, E)$ and an integer $K \geq 3$,
  - a set V' of vertices and an ordered sequence S of vertices.
- The verification algorithm does the following:
  - Check that $V' \subseteq V$ and V' has K vertices.
  - Check that[1] $S \subseteq V$ and S has K vertices.
  - Check that[2] $V' \cap S = \emptyset$.
  - Check that for each pair $u \neq v$ in V', $\{u, v\}$ is an edge in E.
  - Check that for each pair u, v of neighboring vertices in the sequence S, $\{u, v\}$ is an edge in E.
  - Let v be the first vertex in the sequence S. Check that there is a vertex u in V' such that $\{u, v\}$ is an edge in E.
  - If all of these are correct, then it returns YES. Otherwise, it returns NO.

[1]this is bad notation, because S is not a set
[2]again bad notation, because S is not a set

The *certificate* is of course the pair (V, S):

$$\begin{aligned}(G, K) \in \textsc{Kite} \;&\Leftrightarrow\; \text{there exists } (V', S) \\ &\;\; \text{such that } V' \text{ and } S \text{ form a kite in } G \\ &\Leftrightarrow\; \text{there exists a certificate } (V', S) \text{ such that} \\ &\;\; \mathcal{V}(G, K, V', S) \text{ returns YES.}\end{aligned}$$

Since $V' \cap S = \emptyset$, the length of the certificate (V', S) is at most $|V|$, which is at most the length of the graph G.

What is the running time of the verification algorithm:

- Checking that $V' \subseteq V$ and V' has K vertices can be done in $O(K|V|) = O(|V|^2)$ time.
- Checking that $S \subseteq V$ and S has K vertices can be done in $O(K|V|) = O(|V|^2)$ time.
- Checking that $V' \cap S = \emptyset$ can be done in $O(K^2) = O(|V|^2)$ time.
- Checking that for each pair $u \neq v$ in V', $\{u, v\}$ is an edge in E can be done in $O(K^2) = O(|V|^2)$ time (assuming that G is represented using an adjacency matrix).
- Checking that for each pair u, v of neighboring vertices in the sequence S, $\{u, v\}$ is an edge in E can be done in $O(K) = O(|V|)$ time.
- Let v be the first vertex in the sequence S. Checking that there is a vertex u in V' such that $\{u, v\}$ is an edge in E can be done in $O(K) = O(|V|)$ time.
- Thus, the total running time of the verification algorithm is $O(|V|^2)$, which is polynomial in the length of G.

This shows that $\textsc{Kite} \in \mathbf{NP}$.

Question 2: The *clique problem* is defined as follows:

$$\textsc{Clique} = \{(G, K) : \text{ graph } G \text{ contains a clique of size } K\}.$$

Prove that $\textsc{Clique} \leq_P \textsc{Kite}$, i.e., in polynomial time, $\textsc{Clique}$ can be reduced to $\textsc{Kite}$.

Solution: We need a function f such that

- f maps an input (G, K) to $\textsc{Clique}$ to an input (G', K') to $\textsc{Kite}$,
- $(G, K) \in \textsc{Clique} \Leftrightarrow (G', K') \in \textsc{Kite}$,
- the time to compute (G', K') is polynomial in the length of (G, K).

Here is the function f: Consider an input (G, K) to $\textsc{Clique}$. We set $K' = K$. The graph G' is obtained as follows:

- Make a copy of G.
- For every vertex v of G: create K new vertices, connect them into a path and connect the start vertex of this path to v.

Let $G = (V, E)$. We can compute (G', K') in time $O(|V| + |E| + K|V|) = O(|V|^2)$, which is polynomial in the length of G.

Assume that $(G, K) \in$ Clique. Let $V' \subseteq V$ be a clique in G of size K. Take an arbitrary vertex v in this clique. In G', this vertex v has a path with K vertices attached to it. This path does not share vertices with the clique. Thus, G' contains a K-kite, i.e., $(G', K') \in$ Kite.

Assume that $(G', K') \in$ Kite. Let (V', S) be a K-kite in G', where V' represents the clique of size K and S represents the path with K vertices that is attached to the clique. Observe that V' must be a subset of the vertex set of the graph G: If V' contains a new vertex in G', then this vertex has degree two and, thus, cannot be part of the clique (we assume here that $K \geq 4$, the other cases can be handled as well). Therefore, V' is a clique in G, i.e., $(G, K) \in$ Clique.

Question 3: The *subset sum problem* is defined as follows:

$$\text{SubsetSum} = \{(S,t) : \begin{array}{l} S \text{ is a set of integers, } t \text{ is an integer,} \\ \exists S' \subseteq S \text{ such that } \sum_{x \in S'} x = t \ \}. \end{array}$$

The *partition problem* is defined as follows:

$$\text{Partition} = \{S : \begin{array}{l} S \text{ is a set of integers,} \\ \exists S' \subseteq S \text{ such that } \sum_{x \in S'} x = \sum_{y \in S \setminus S'} y \ \}. \end{array}$$

- Prove that SubsetSum $\leq_P$ Partition, i.e., in polynomial time, SubsetSum can be reduced to Partition.
- Prove that Partition $\leq_P$ SubsetSum, i.e., in polynomial time, Partition can be reduced to SubsetSum.

Solution: We start with

$$\text{SubsetSum} \leq_P \text{Partition}.$$

We need a function f such that

- f maps an input (S, t) to SubsetSum to an input T to Partition,
- $(S, t) \in$ SubsetSum $\Leftrightarrow T \in$ Partition,
- the time to compute T is polynomial in the length of (S, t).

Here is the function f: Consider an input (S,t) to SubsetSum, where $S = \{a_1, a_2, \ldots, a_n\}$. The input to Partition is the set

$$T = \{a_1, a_2, \ldots, a_n, s - 2t\},$$

where

$$s = a_1 + a_2 + \cdots + a_n.$$

The time to compute T is $O(n)$, which is polynomial in the length of S.

Assume that $(S,t) \in$ SubsetSum. Let $S' \subseteq S$ be such that

$$\sum_{a_i \in S'} a_i = t.$$

Note that

$$\sum_{a_i \in S \setminus S'} a_i = s - t$$

and

$$\sum_{x \in T} x = s + (s - 2t) = 2s - 2t.$$

Let $T' = S' \cup \{s - 2t\}$. Then

$$\sum_{x \in T'} x = \left(\sum_{a_i \in S'} a_i \right) + (s - 2t) = t + (s - 2t) = s - t$$

and

$$\sum_{x \in T \setminus T'} x = \left(\sum_{a_i \in S \setminus S'} a_i \right) = s - t.$$

Thus, $T \in$ Partition.

For the other direction, we assume that $T \in$ Partition. Let $T' \subseteq T$ be such that

$$\sum_{x \in T'} x = \sum_{x \in T \setminus T'} x.$$

Since $\sum_{x \in T} x = 2s - 2t$, we have

$$\sum_{x \in T'} x = \sum_{x \in T \setminus T'} x = s - t.$$

Assume first that $s - 2t \in T'$. Let $S' = T' \setminus \{s - 2t\}$. Then

$$\sum_{x \in S'} x = \left(\sum_{x \in T'} x \right) - (s - 2t) = (s - t) - (s - 2t) = t$$

and, therefore, $(S,t) \in$ SubsetSum.

Now assume that $s - 2t \in T \setminus T'$. Let $S' = (T \setminus T') \setminus \{s - 2t\}$. Then

$$\sum_{x \in S'} x = \left(\sum_{x \in T \setminus T'} x \right) - (s - 2t) = (s - t) - (s - 2t) = t$$

and, therefore, $(S, t) \in$ SubsetSum.

Next we show that

$$\text{Partition} \leq_P \text{SubsetSum}.$$

We need a function f such that

- f maps an input S to Partition to an input (T, t) to SubsetSum,
- $S \in$ Partition $\Leftrightarrow (T, t) \in$ SubsetSum,
- the time to compute (T, t) is polynomial in the length of S.

Here is the function f: Consider an input S to Partition, where $S = \{a_1, a_2, \ldots, a_n\}$. The input to SubsetSum is the set

$$T = \{2a_1, 2a_2, \ldots, 2a_n\},$$

and the integer

$$t = a_1 + a_2 + \cdots + a_n.$$

The time to compute (T, t) is $O(n)$, which is polynomial in the length of S.

Assume that $S \in$ Partition. Let $S' \subseteq S$ be such that

$$\sum_{a_i \in S'} a_i = \sum_{a_i \in S \setminus S'} a_i.$$

Note that each of these two sums is equal to $t/2$ (which must be an integer, because $S \in$ Partition). Let

$$T' = \{2a_i : a_i \in S'\}.$$

Then

$$\sum_{x \in T'} x = 2 \cdot \sum_{a_i \in S'} a_i = 2 \cdot t/2 = t.$$

Thus, $(T, t) \in$ SubsetSum.

For the other direction, we assume that $(T, t) \in$ SubsetSum. Let $T' \subseteq T$ be such that

$$\sum_{x \in T'} x = t.$$

Let

$$S' = \{a_i \in S : 2a_i \in T'\}.$$

Then

$$\sum_{x \in S'} x = \frac{1}{2} \cdot \sum_{x \in T'} x = t/2$$

and

$$\sum_{x \in S \setminus S'} x = \sum_{x \in S} x - \sum_{x \in S'} x = t - t/2 = t/2.$$

Thus, $S \in$ PARTITION.

Question 4: The *clique and independent set problem* is defined as follows:

CLIQUEINDEPSET = $\{(G, K)$: graph G contains a clique of size K and G contains an independent set of size K $\}$.

Prove that CLIQUE $\leq_P$ CLIQUEINDEPSET, i.e., in polynomial time, CLIQUE can be reduced to CLIQUEINDEPSET.

Solution: We need a function f such that

- f maps an input (G, K) to CLIQUE to an input (G', K') to CLIQUEINDEPSET,
- $(G, K) \in$ CLIQUE $\Leftrightarrow (G', K') \in$ CLIQUEINDEPSET,
- the time to compute (G', K') is polynomial in the length of (G, K).

Here is the function f: Consider an input (G, K) to CLIQUE. We set $K' = K$. The graph G' is obtained as follows:

- Make a copy of G.
- Add K new vertices, each of them having degree zero.

Let $G = (V, E)$. We can compute (G', K') in time $O(|V| + |E| + K) = O(|V| + |E|)$, which is polynomial in the length of G.

Assume that $(G, K) \in$ CLIQUE. Let $V' \subseteq V$ be a clique in G of size K. Let V'' be the set of K new vertices. Then V' is a clique of size K in G' and V'' is an independent set of size K in G'. Thus, $(G', K) \in$ CLIQUEINDEPSET.

Assume that $(G', K) \in$ CLIQUEINDEPSET. Let V' be a clique of size K in G' and let V'' be an independent set of size K in G'. Then V' cannot contain any of the new vertices. Thus, V' is a clique of size K in G, i.e., $(G, K) \in$ CLIQUE.

Question 5: Let φ be a Boolean formula in the variables $x_1, x_2, \ldots, x_n$. We say that φ is in *conjunctive normal form* (CNF) if it is of the form

$$\varphi = C_1 \wedge C_2 \wedge \ldots \wedge C_m,$$

where each C_i, $1 \leq i \leq m$, is of the following form:

$$C_i = l_1^i \vee l_2^i \vee \ldots \vee l_{k_i}^i.$$

Each l_j^i is a *literal*, which is either a variable or the negation of a variable.

The *satisfiability problem* is defined as follows:

$$\text{Sat} = \{\varphi : \varphi \text{ is in CNF-form and is satisfiable}\}.$$

Prove that Clique $\leq_P$ Sat, i.e., in polynomial time, Clique can be reduced to Sat.

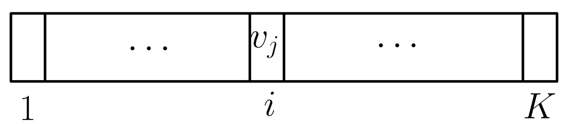
Solution: We need a function f such that

- f maps an input (G, K) to Clique to a Boolean formula φ in CNF-form,
- G has a clique of size $K \Leftrightarrow \varphi$ is satisfiable,
- the time to compute φ is polynomial in the length of G.

Consider an input (G, K) to Clique, where $G = (V, E)$ and $V = \{v_1, v_2, \ldots, v_n\}$. A clique of size K, if it exists, will be represented by an ordered sequence of K vertices.

We will use Kn Boolean variables x_{ij}, where $1 \leq i \leq K$ and $1 \leq j \leq n$. The meaning of these variables is as follows:

$$x_{ij} = \textit{true} \Leftrightarrow \text{ the vertex at position } i \text{ in the clique is } v_j.$$



A clique of size K exists if and only if all of the following are true:

1. For each $i = 1, 2, \ldots, K$: There is at least one vertex at position i.
2. For each $i = 1, 2, \ldots, K$: There is at most one vertex at position i.
3. For each $1 \leq i < i' \leq K$: The vertices at positions i and i' are distinct.
4. For each $1 \leq i < i' \leq K$: The vertices at positions i and i' form an edge in G.

We are going to describe each of these four conditions by clauses.

Item 1: For position i, we get the clause

$$x_{i1} \vee x_{i2} \vee \cdots \vee x_{in} = \bigvee_{j=1}^{n} x_{ij}.$$

For all positions i, we get K clauses

$$\bigwedge_{i=1}^{K} \bigvee_{j=1}^{n} x_{ij}.$$

The total size of all these clauses is Kn, which is at most n^2.

Item 2: Consider one position i and two distinct vertices v_j and $v_{j'}$. If $x_{ij} \wedge x_{ij'}$ is true, then both v_i and $v_{j'}$ are at position i. Thus, $x_{ij} \wedge x_{ij'}$ must be false, i.e., $\neg(x_{ij} \wedge x_{ij'})$ must be true, which is the same as the clause

$$\neg x_{ij} \vee \neg x_{ij'}.$$

For all positions i and all distinct vertices v_j and $v_{j'}$, we get $K \cdot \binom{n}{2}$ clauses

$$\bigwedge_{i=1}^{K} \bigwedge_{1 \leq j < j' \leq n} \left(\neg x_{ij} \vee \neg x_{ij'}\right).$$

The total size of all these clauses is

$$K \cdot \binom{n}{2} \cdot 2 = O(n^3).$$

Item 3: Consider two distinct positions i and i', and one vertex v_j. If $x_{ij} \wedge x_{i'j}$ is true, then vertex v_j is at both positions i and i'. Thus, $x_{ij} \wedge x_{i'j}$ must be false, i.e., $\neg(x_{ij} \wedge x_{i'j})$ must be true, which is the same as the clause

$$\neg x_{ij} \vee \neg x_{i'j}.$$

For all distinct positions i and i', and all vertices v_j, we get $\binom{K}{2} \cdot n$ clauses

$$\bigwedge_{1 \leq i < i' \leq K} \bigwedge_{j=1}^{n} \left(\neg x_{ij} \vee \neg x_{i'j}\right).$$

The total size of all these clauses is

$$\binom{K}{2} \cdot n \cdot 2 = O(n^3).$$

Item 4: Consider two distinct positions i and i', and an non-edge $\{v_j, v_{j'}\}$. If $x_{ij} \wedge x_{i'j'}$ is true, then the vertices v_j and $v_{j'}$ at positions i and i' do not form an edge. Thus, $x_{ij} \wedge x_{i'j'}$ must be false, i.e., $\neg(x_{ij} \wedge x_{i'j'})$ must be true, which is the same as the clause

$$\neg x_{ij} \vee \neg x_{i'j'}.$$

For all distinct positions i and i', and all non-edges $\{v_j, v_{j'}\}$, we get $\binom{K}{2} \cdot \left(\binom{n}{2} - |E|\right)$ clauses

$$\bigwedge_{1 \leq i < i' \leq K} \bigwedge_{\{v_j, v_{j'}\} \notin E} \left(\neg x_{ij} \vee \neg x_{i'j'}\right).$$

The total size of all these clauses is

$$\binom{K}{2} \cdot \left(\binom{n}{2} - |E|\right) \cdot 2 \leq \binom{K}{2} \cdot \binom{n}{2} \cdot 2 = O(n^4).$$

The final Boolean formula φ that we are looking for is the conjunction (logical AND) of all clauses in Items 1—4. The total size of φ is $O(n^4)$, which is polynomial in the length of the graph G.