COL351: Analysis and Design of Algorithms

Ragesh Jaiswal, CSE, IITD

Computational Intractability: NP-complete problems

SUBSET-SUM

Given natural numbers $w_1, ..., w_n$ and a target number W, determine if there is a subset S of $\{1, ..., n\}$ such that $\sum_{i \in S} w_i = W$.

SCHEDULING

Given n jobs with start time s_i and duration t_i and deadline d_i, determine if all the jobs can be scheduled on a single machine such that no deadlines are missed.

- Claim 1: SUBSET-SUM $\in$ NP

SUBSET-SUM

Given natural numbers $w_1, ..., w_n$ and a target number W, determine if there is a subset S of $\{1, ..., n\}$ such that $\sum_{i \in S} w_i = W$.

SCHEDULING

Given n jobs with start time s_i and duration t_i and deadline d_i, determine if all the jobs can be scheduled on a single machine such that no deadlines are missed.

- Claim 1: SUBSET-SUM $\in$ NP
- Claim 2: SCHEDULING $\in$ NP

SUBSET-SUM

Given natural numbers $w_1, ..., w_n$ and a target number W, determine if there is a subset S of $\{1, ..., n\}$ such that $\sum_{i \in S} w_i = W$.

SCHEDULING

Given n jobs with start time s_i and duration t_i and deadline d_i, determine if all the jobs can be scheduled on a single machine such that no deadlines are missed.

- Claim 1: SUBSET-SUM $\in$ NP
- Claim 2: SCHEDULING $\in$ NP
- Claim 3: SUBSET-SUM $\leq_p$ SCHEDULING

SUBSET-SUM

Given natural numbers $w_1, ..., w_n$ and a target number W, determine if there is a subset S of $\{1, ..., n\}$ such that $\sum_{i \in S} w_i = W$.

SCHEDULING

Given n jobs with start time s_i and duration t_i and deadline d_i, determine if all the jobs can be scheduled on a single machine such that no deadlines are missed.

- Claim 1: SUBSET-SUM $\in$ NP
- Claim 2: SCHEDULING $\in$ NP
- Claim 3: SUBSET-SUM $\leq_p$ SCHEDULING

Proof sketch for Claim 3

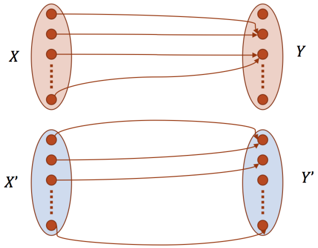
Given an instance of the subset sum problem $(\{w_1, ..., w_n\}, W)$, we construct the following instance of the Scheduling problem: $((0, w_1, S+1), ..., (0, w_n, S+1), (W, 1, W+1))$. We then argue that there is a subset that sums to W if and only if the $(n+1)$ jobs can be scheduled. Here $S = w_1 + ... + w_n$.

- Most of the polynomial-time reductions $X \leq_p Y$ that we have seen are of the following general nature: We give an efficient mapping from instances of X to instances of Y such that "yes" instances of X map to "yes" instances of Y and "no" instances of X map to "no" instances of Y.
- Such reductions have special name. They are called many-one reductions.

- Most of the polynomial-time reductions $X \leq_p Y$ that we have seen are of the following general nature: We give an efficient mapping from instances of X to instances of Y such that "yes" instances of X map to "yes" instances of Y and "no" instances of X map to "no" instances of Y.
- Such reductions have special name. They are called many-one reductions.

Many-one reduction

In order to show that $X \leq_p Y$ we design an efficient mapping f from the set of instances of X to set of instances of Y such that $s \in X$ iff $f(s) \in Y$.



3D-MATCHING

Given disjoint sets X, Y, and Z each of size n, and given a set T of triples (x, y, z), determine if there exist a subset of n triples in T such that each element of $X \cup Y \cup Z$ is contained in exactly one of these triples.

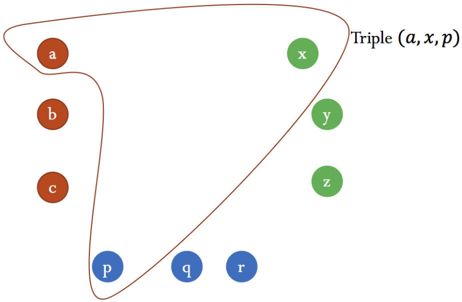


Figure: Let $T = \{(a, x, p), (a, y, p), (b, y, q), (c, z, r)\}$. Does there exist a 3D-Matching?

3D-MATCHING

Given disjoint sets X, Y, and Z each of size n, and given a set T of triples (x, y, z), determine if there exist a subset of n triples in T such that each element of $X \cup Y \cup Z$ is contained in exactly one of these triples.

- Claim 1: 3D-MATCHING $\in$ NP.

3D-MATCHING

Given disjoint sets X, Y, and Z each of size n, and given a set T of triples (x, y, z), determine if there exist a subset of n triples in T such that each element of $X \cup Y \cup Z$ is contained in exactly one of these triples.

- Claim 1: 3D-MATCHING $\in$ NP.
- Claim 2: 3D-MATCHING is NP-complete.

3D-MATCHING

Given disjoint sets X, Y, and Z each of size n, and given a set T of triples (x, y, z), determine if there exist a subset of n triples in T such that each element of $X \cup Y \cup Z$ is contained in exactly one of these triples.

- Claim 1: 3D-MATCHING $\in$ NP.
- Claim 2: 3D-MATCHING is NP-complete.
  - Claim 2.1: 3-SAT $\leq_p$ 3D-MATCHING.
  - Proof sketch of Claim 2.1: We will show an efficient many-one reduction.

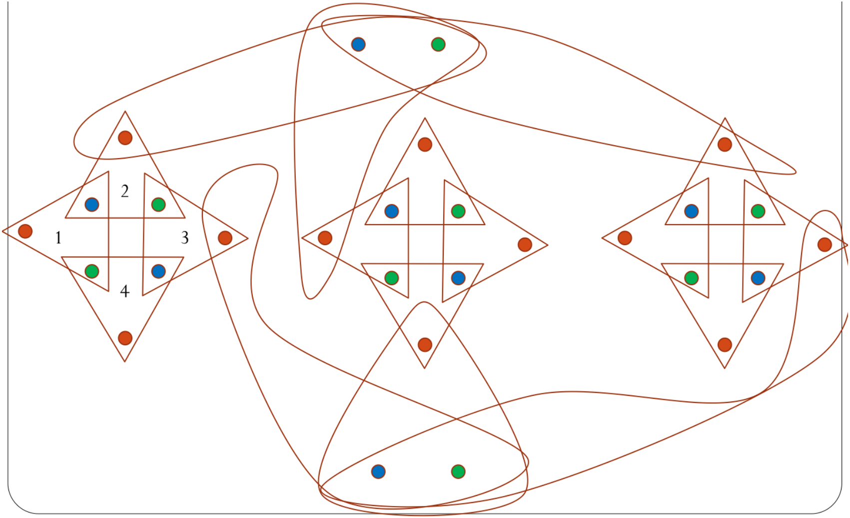


Figure: Example construction for $(x_1 \vee \bar{x}_2 \vee x_3), (\bar{x}_1 \vee x_2 \vee x_3)$

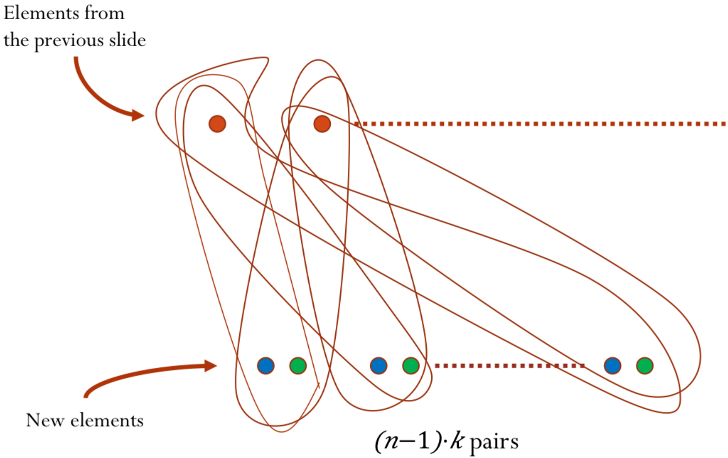


Figure: Example construction for $(x_1 \vee \bar{x}_2 \vee x_3), (\bar{x}_1 \vee x_2 \vee x_3)$. k denotes the number of clauses.

SUBSET-SUM

Given natural numbers $w_1, ..., w_n$ and a target number W, determine if there is a subset S of $\{1, ..., n\}$ such that $\sum_{i \in S} w_i = W$.

- Claim 1: SUBSET-SUM $\in$ NP.

SUBSET-SUM

Given natural numbers $w_1, ..., w_n$ and a target number W, determine if there is a subset S of $\{1, ..., n\}$ such that $\sum_{i \in S} w_i = W$.

- Claim 1: SUBSET-SUM $\in$ NP.
- Claim 2: SUBSET-SUM is NP-complete.

SUBSET-SUM

Given natural numbers $w_1, ..., w_n$ and a target number W, determine if there is a subset S of $\{1, ..., n\}$ such that $\sum_{i \in S} w_i = W$.

- Claim 1: SUBSET-SUM $\in$ NP.
- Claim 2: SUBSET-SUM is NP-complete.
  - Claim 2.1: 3D-MATCHING $\leq_p$ SUBSET-SUM.
  - Proof sketch: We will show an efficient many-one reduction. Given an instance (X, Y, Z, T) of the 3D-MATCHING problem, we construct an instance of the SUBSET-SET problem.
    - We first construct a $3n$-bit vector. Given a triple $t_i = (x_1, y_3, z_5)$, we construct the following vector v_i:

1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0

SUBSET-SUM

Given natural numbers $w_1, ..., w_n$ and a target number W, determine if there is a subset S of $\{1, ..., n\}$ such that $\sum_{i \in S} w_i = W$.

- Claim 1: SUBSET-SUM $\in$ NP.
- Claim 2: SUBSET-SUM is NP-complete.
  - Claim 2.1: 3D-MATCHING $\leq_p$ SUBSET-SUM.
  - Proof sketch: We will show an efficient many-one reduction. Given an instance (X, Y, Z, T) of the 3D-MATCHING problem, we construct an instance of the SUBSET-SET problem.
    - We first construct a $3n$-bit vector. Given a triple $t_i = (x_1, y_3, z_5)$, we construct the following vector v_i:

1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0

    - Let w_i be the value of v_i in base $(|T| + 1)$ and $W = \sum_{i=0}^{3n-1} (|T| + 1)^i$.
    - Claim 2.1.1: There is a 3D-Matching iff there is a subset $\{w_1, ..., w_{|T|}\}$ that sums to W.

End