BILL START RECORDING

HW 10 Solutions

Problem 1 Set Up

Def Let $G = (V, E)$ be a graph. A **vertex cover (VC) for G of size k** is a set $U \subseteq V$ such that

Problem 1 Set Up

Def Let $G = (V, E)$ be a graph. A **vertex cover (VC) for G of size k** is a set $U \subseteq V$ such that

1) $|U| = k$

Problem 1 Set Up

Def Let $G = (V, E)$ be a graph. A **vertex cover (VC) for G of size k** is a set $U \subseteq V$ such that

1) $|U| = k$

2) $(\forall (a, b) \in E)[(a \in U) \vee (b \in U)]$

Problem 1 Set Up

Def Let $G = (V, E)$ be a graph. A **vertex cover (VC) for G of size k** is a set $U \subseteq V$ such that

1) $|U| = k$

2) $(\forall (a, b) \in E)[(a \in U) \vee (b \in U)]$

$\mathrm{VC} = \{(G, k) \colon G \text{ has a VC of size} \le k\}$.

Problem 1 Set Up

Def Let $G = (V, E)$ be a graph. A **vertex cover (VC) for G of size k** is a set $U \subseteq V$ such that

1) $|U| = k$

2) $(\forall (a, b) \in E)[(a \in U) \vee (b \in U)]$

$\mathrm{VC} = \{(G, k) \colon G$ has a VC of size $\leq k\}$.

It is known that VC is NP-complete.

Problem 1a

Want: a connected graph on 1000 vertices that has a VC of size 1.

Problem 1a

Want: a connected graph on 1000 vertices that has a VC of size 1.

$V = \{1, \ldots, 1000\}$

Problem 1a

Want: a connected graph on 1000 vertices that has a VC of size 1.

$V = \{1, \ldots, 1000\}$
$E = \{(1, 2), (1, 3), \ldots, (1, 1000)\}$

Problem 1a

Want: a connected graph on 1000 vertices that has a VC of size 1.

$V = \{1, \ldots, 1000\}$
$E = \{(1, 2), (1, 3), \ldots, (1, 1000)\}$

$U = \{1\}$ is a VC of size 1.

Problem 1b

Want: a connected graph on 1000 vertices so that the smallest vertex cover for it has size 999.

Take the complete graph on 1000 vertices.
$V = \{1, \ldots, 1000\}$
$E = \{(i, j) \colon 1 \leq i < j \leq 1000\}$
$U = \{1, 2, \ldots, 999\}$ is a VC of size 999.
We leave it to the reader that there is not a smaller VC.

Problem 1c

Want: a connected graph G on 1000 vertices s.t.:

Problem 1c

Want: a connected graph G on 1000 vertices s.t.:

G has a VC of size 500.

Problem 1c

Want: a connected graph G on 1000 vertices s.t.:

G has a VC of size 500.
G does not have a VC of size 499.

Problem 1c

Want: a connected graph G on 1000 vertices s.t.:

G has a VC of size 500.
G does not have a VC of size 499.

We take the cycle on 1000 vertices. Formally:

Problem 1c

Want: a connected graph G on 1000 vertices s.t.:

G has a VC of size 500.
G does not have a VC of size 499.

We take the cycle on 1000 vertices. Formally:
$V = \{1, \dots, 1000\}$

Problem 1c

Want: a connected graph G on 1000 vertices s.t.:

G has a VC of size 500.
G does not have a VC of size 499.

We take the cycle on 1000 vertices. Formally:
$V = \{1, \ldots, 1000\}$
$E = \{(1, 2), (2, 3), \ldots, (999, 1000), (1000, 1)\}$

Problem 1c

Want: a connected graph G on 1000 vertices s.t.:

G has a VC of size 500.
G does not have a VC of size 499.

We take the cycle on 1000 vertices. Formally:
$V = \{1, \ldots, 1000\}$
$E = \{(1,2), (2,3), \ldots, (999, 1000), (1000, 1)\}$

$U = \{2, 4, \ldots, 1000\}$ is a VC of size 500.

Problem 1c

Want: a connected graph G on 1000 vertices s.t.:

G has a VC of size 500.
G does not have a VC of size 499.

We take the cycle on 1000 vertices. Formally:
$V = \{1, \ldots, 1000\}$
$E = \{(1, 2), (2, 3), \ldots, (999, 1000), (1000, 1)\}$

$U = \{2, 4, \ldots, 1000\}$ is a VC of size 500.

No set of size 499 works. Left to the reader.

Problem 1c

$$\mathrm{VC}_{1000} = \{G \colon G \text{ has a VC of size } 1000\}.$$

Problem 1c

$\mathrm{VC}_{1000} = \{G\colon G \text{ has a VC of size } 1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.

Problem 1c

$\mathrm{VC}_{1000} = \{G\colon G \text{ has a VC of size } 1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.
ALGORITHM

Problem 1c

$\mathrm{VC}_{1000} = \{G\colon G$ has a VC of size $1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.

Problem 1c

$\mathrm{VC}_{1000} = \{G : G \text{ has a VC of size } 1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.
ALGORITHM
1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $U \subseteq V$ of size 1000 test if U is a vertex cover.

Problem 1c

$\mathrm{VC}_{1000} = \{G \colon G \text{ has a VC of size } 1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $U \subseteq V$ of size 1000 test if U is a vertex cover.
(Test: Visit each edge. Need that one of its ends is in U.)

Problem 1c

$\mathrm{VC}_{1000} = \{G : G$ has a VC of size $1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $U \subseteq V$ of size 1000 test if U is a vertex cover.
(Test: Visit each edge. Need that one of its ends is in U.)
If YES then jump out of the loop and output YES.

Problem 1c

$\mathrm{VC}_{1000} = \{G : G$ has a VC of size $1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $U \subseteq V$ of size 1000 test if U is a vertex cover.
(Test: Visit each edge. Need that one of its ends is in U.)
If YES then jump out of the loop and output YES.
3) (If got here then no U worked.) Output NO.

Problem 1c

$\mathrm{VC}_{1000} = \{G : G \text{ has a VC of size } 1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $U \subseteq V$ of size 1000 test if U is a vertex cover.
(Test: Visit each edge. Need that one of its ends is in U.)
If YES then jump out of the loop and output YES.
3) (If got here then no U worked.) Output NO.

END OF ALGORITHM

Problem 1c

$\mathrm{VC}_{1000} = \{G : G$ has a VC of size $1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $U \subseteq V$ of size 1000 test if U is a vertex cover.
(Test: Visit each edge. Need that one of its ends is in U.)
If YES then jump out of the loop and output YES.
3) (If got here then no U worked.) Output NO.

END OF ALGORITHM

Number of U's tested is $\binom{n}{1000} \le n^{1000}$.

Problem 1c

$\mathrm{VC}_{1000} = \{G : G$ has a VC of size $1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $U \subseteq V$ of size 1000 test if U is a vertex cover.
(Test: Visit each edge. Need that one of its ends is in U.)
If YES then jump out of the loop and output YES.
3) (If got here then no U worked.) Output NO.

END OF ALGORITHM

Number of U's tested is $\binom{n}{1000} \le n^{1000}$.
Each test took $O(|E|) = O(n^2)$.

Problem 1c

$\mathrm{VC}_{1000} = \{G : G \text{ has a VC of size } 1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $U \subseteq V$ of size 1000 test if U is a vertex cover.
(Test: Visit each edge. Need that one of its ends is in U.)
If YES then jump out of the loop and output YES.
3) (If got here then no U worked.) Output NO.

END OF ALGORITHM

Number of U's tested is $\binom{n}{1000} \le n^{1000}$.
Each test took $O(|E|) = O(n^2)$.
So time is $O(n^{1002})$.

Problem 1c

$\mathrm{VC}_{1000} = \{G : G$ has a VC of size $1000\}$.
Show that $\mathrm{VC}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $U \subseteq V$ of size 1000 test if U is a vertex cover.
(Test: Visit each edge. Need that one of its ends is in U.)
If YES then jump out of the loop and output YES.
3) (If got here then no U worked.) Output NO.

END OF ALGORITHM

Number of U's tested is $\binom{n}{1000} \le n^{1000}$.
Each test took $O(|E|) = O(n^2)$.
So time is $O(n^{1002})$. Thats a polynomial!

Problem 1d: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

Problem 1d: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

The algorithm was in time $O(n^{1002})$.

Problem 1d: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

The algorithm was in time $O(n^{1002})$.

VOTE:

Problem 1d: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

The algorithm was in time $O(n^{1002})$.

VOTE:

$\exists$ an algorithm that is substantially better than $O(n^{1002})$.

Problem 1d: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

The algorithm was in time $O(n^{1002})$.

VOTE:
$\exists$ an algorithm that is substantially better than $O(n^{1002})$.
Does not $\exists$ an algorithm that is substantially better than $O(n^{1002})$.

Problem 1d: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

The algorithm was in time $O(n^{1002})$.

VOTE:
$\exists$ an algorithm that is substantially better than $O(n^{1002})$.
Does not $\exists$ an algorithm that is substantially better than $O(n^{1002})$.
The question is UNKNOWN TO SCIENCE!

Problem 1d

Problem 1d

$\exists$ an algorithm that is substantially better than $O(n^{1002})$.

Problem 1d

$\exists$ an algorithm that is substantially better than $O(n^{1002})$.
We sketch two algorithms.

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

1) Root is G.

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

1) Root is G.

2) Pick an edge (a, b).

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

1) Root is G.

2) Pick an edge (a, b). a or b **must be in VC**.

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

1) Root is G.

2) Pick an edge (a, b). a or b **must be in VC**.

3)

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

1) Root is G.

2) Pick an edge (a, b). a or b **must be in VC**.

3)
Left side is $G - \{a\}$. Think of a as being put into a VC.

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

1) Root is G.

2) Pick an edge (a, b). a or b **must be in VC**.

3)
Left side is $G - \{a\}$. Think of a as being put into a VC.
Right side is $G - \{b\}$. Think of b as being put into a VC.

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

1) Root is G.

2) Pick an edge (a, b). a or b **must be in VC**.

3)
Left side is $G - \{a\}$. Think of a as being put into a VC.
Right side is $G - \{b\}$. Think of b as being put into a VC.
4) Repeat on each side until depth 1000.

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

1) Root is G.

2) Pick an edge (a, b). a or b **must be in VC**.

3)
Left side is $G - \{a\}$. Think of a as being put into a VC.
Right side is $G - \{b\}$. Think of b as being put into a VC.
4) Repeat on each side until depth 1000.
5) If some leaf node is empty then have VC of size ≤ 1000.

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

1) Root is G.

2) Pick an edge (a, b). a or b **must be in VC**.

3)
Left side is $G - \{a\}$. Think of a as being put into a VC.
Right side is $G - \{b\}$. Think of b as being put into a VC.
4) Repeat on each side until depth 1000.
5) If some leaf node is empty then have VC of size ≤ 1000.
6) If all leaf nodes have some edge then NO VC of size ≤ 1000.

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

1) Root is G.

2) Pick an edge (a, b). a or b **must be in VC**.

3)
Left side is $G - \{a\}$. Think of a as being put into a VC.
Right side is $G - \{b\}$. Think of b as being put into a VC.
4) Repeat on each side until depth 1000.
5) If some leaf node is empty then have VC of size ≤ 1000.
6) If all leaf nodes have some edge then NO VC of size ≤ 1000.
Algorithm takes time $O(n)$ but the mult constant is 2^{1000}.

Problem 1d-Algorithm 1

Input G. Form a tree of depth ≤ 1000 as follows

1) Root is G.

2) Pick an edge (a, b). a or b **must be in VC**.

3)
Left side is $G - \{a\}$. Think of a as being put into a VC.
Right side is $G - \{b\}$. Think of b as being put into a VC.
4) Repeat on each side until depth 1000.
5) If some leaf node is empty then have VC of size ≤ 1000.
6) If all leaf nodes have some edge then NO VC of size ≤ 1000.
Algorithm takes time $O(n)$ but the mult constant is 2^{1000}.
Much better in practice, and has been improved.

Problem 1d-Algorithm 2

Input $G = (V, E)$. $|V| = n$. $|E| = m$.

Problem 1d-Algorithm 2

Input $G = (V, E)$. $|V| = n$. $|E| = m$.

1) If $\exists v$ of deg ≥ 1001 then put v in the VC. (easy to prove that v must be in the VC) and $G \leftarrow G - \{v\}$. Want 999 sized VC for G.

Problem 1d-Algorithm 2

Input $G = (V, E)$. $|V| = n$. $|E| = m$.

1) If $\exists v$ of deg ≥ 1001 then put v in the VC. (easy to prove that v must be in the VC) and $G \leftarrow G - \{v\}$. Want 999 sized VC for G.

If $\exists v$ of deg ≥ 1000 then put v in the VC. (easy to prove that v must be in the VC) and $G \leftarrow G - \{v\}$. Want 998 sized VC for G.

Problem 1d-Algorithm 2

Input $G = (V, E)$. $|V| = n$. $|E| = m$.

1) If $\exists v$ of deg ≥ 1001 then put v in the VC. (easy to prove that v must be in the VC) and $G \leftarrow G - \{v\}$. Want 999 sized VC for G.

If $\exists v$ of deg ≥ 1000 then put v in the VC. (easy to prove that v must be in the VC) and $G \leftarrow G - \{v\}$. Want 998 sized VC for G.

2) Repeat until seek a VC for G of size k and G has all vertices of degree $\leq k$ where $k \leq 1000$.

Problem 1d-Algorithm 2

Input $G = (V, E)$. $|V| = n$. $|E| = m$.

1) If $\exists v$ of deg ≥ 1001 then put v in the VC. (easy to prove that v must be in the VC) and $G \leftarrow G - \{v\}$. Want 999 sized VC for G.

If $\exists v$ of deg ≥ 1000 then put v in the VC. (easy to prove that v must be in the VC) and $G \leftarrow G - \{v\}$. Want 998 sized VC for G.

2) Repeat until seek a VC for G of size k and G has all vertices of degree $\leq k$ where $k \leq 1000$.

3) (comment) If $G = (V, E)$ has a VC of size $\leq k$ then note that each element of the VC covers $\leq k$ edges, so $|E| \leq k^2$, so
$|V| \leq |E| \leq k^2$.

Problem 1d-Algorithm 2

Input $G = (V, E)$. $|V| = n$. $|E| = m$.

1) If $\exists v$ of deg ≥ 1001 then put v in the VC. (easy to prove that v must be in the VC) and $G \leftarrow G - \{v\}$. Want 999 sized VC for G.

If $\exists v$ of deg ≥ 1000 then put v in the VC. (easy to prove that v must be in the VC) and $G \leftarrow G - \{v\}$. Want 998 sized VC for G.

2) Repeat until seek a VC for G of size k and G has all vertices of degree $\leq k$ where $k \leq 1000$.

3) (comment) If $G = (V, E)$ has a VC of size $\leq k$ then note that each element of the VC covers $\leq k$ edges, so $|E| \leq k^2$, so $|V| \leq |E| \leq k^2$.

4) Look at all k-sized subsets of the V to see if any form a VC.

Problem 1d-Algorithm 2

Input $G = (V, E)$. $|V| = n$. $|E| = m$.

1) If $\exists v$ of deg ≥ 1001 then put v in the VC. (easy to prove that v must be in the VC) and $G \leftarrow G - \{v\}$. Want 999 sized VC for G.

If $\exists v$ of deg ≥ 1000 then put v in the VC. (easy to prove that v must be in the VC) and $G \leftarrow G - \{v\}$. Want 998 sized VC for G.

2) Repeat until seek a VC for G of size k and G has all vertices of degree $\leq k$ where $k \leq 1000$.

3) (comment) If $G = (V, E)$ has a VC of size $\leq k$ then note that each element of the VC covers $\leq k$ edges, so $|E| \leq k^2$, so
$|V| \leq |E| \leq k^2$.

4) Look at all k-sized subsets of the V to see if any form a VC.

Takes time $O(n + m) + k^{k^2} \leq O(n) + 1000^{1000^2}$.

Best Known VC Algorithm for fixed k

Best Known VC Algorithm for fixed k

$\mathrm{VC}_k = \{G\colon G$ has a VC of size $k\}$.

Best Known VC Algorithm for fixed k

$\mathrm{VC}_k = \{G \colon G$ has a VC of size $k\}$.

Our Alg 1 can be generalized to solve VC_k in time $O(2^k n)$.

Best Known VC Algorithm for fixed k

$\mathrm{VC}_k = \{G \colon G$ has a VC of size $k\}$.

Our Alg 1 can be generalized to solve VC_k in time $O(2^k n)$.

Our Alg 2 can be generalized to solve VC_k in time $O(n + k^{k^2})$.

Best Known VC Algorithm for fixed k

$\mathrm{VC}_k = \{G : G \text{ has a VC of size } k\}$.

Our Alg 1 can be generalized to solve VC_k in time $O(2^k n)$.

Our Alg 2 can be generalized to solve VC_k in time $O(n + k^{k^2})$.

The Best Known Algorithm takes time $O(kn + 1.2738^k)$.

Best Known VC Algorithm for fixed k

$\mathrm{VC}_k = \{G\colon G \text{ has a VC of size } k\}$.

Our Alg 1 can be generalized to solve VC_k in time $O(2^k n)$.

Our Alg 2 can be generalized to solve VC_k in time $O(n + k^{k^2})$.

The Best Known Algorithm takes time $O(kn + 1.2738^k)$.

It works very well in practice.

Respect Lower Bounds!

You probably thought that VC_k required roughly n^k time.

Respect Lower Bounds!

You probably thought that VC_k required roughly n^k time.

A clever trick got the run time so that the the degree of the poly does not depend on k.

Respect Lower Bounds!

You probably thought that VC_k required roughly n^k time.

A clever trick got the run time so that the the degree of the poly does not depend on k.

Any proof of a lower bound has to show that there is no clever trick.

Respect Lower Bounds!

You probably thought that VC_k required roughly n^k time.

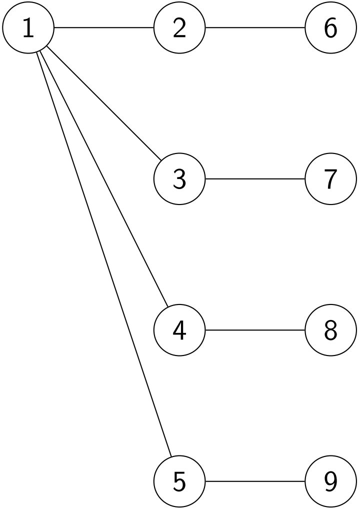
A clever trick got the run time so that the the degree of the poly does not depend on k.

Any proof of a lower bound has to show that there is no clever trick.

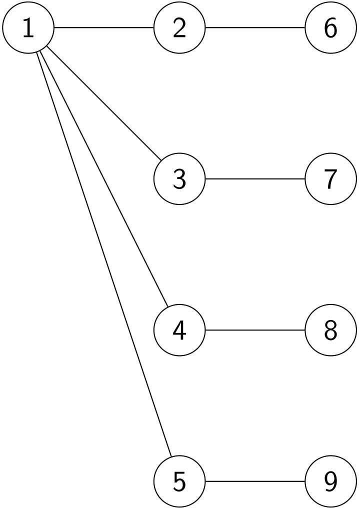
Respect lower bounds!

1e-Graph Where Greedy Alg Is Not Opt

1e-Graph Where Greedy Alg Is Not Opt

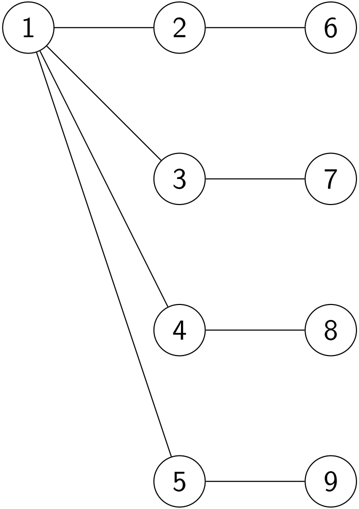


1e-Graph Where Greedy Alg Is Not Opt



Greedy algorithm produces $\{1, 2, 3, 4, 5\}$, 5 vertices.

1e-Graph Where Greedy Alg Is Not Opt



Greedy algorithm produces $\{1, 2, 3, 4, 5\}$, 5 vertices.
Optimal is $\{2, 3, 4, 5\}$, 4 vertices.

Problem 2 Set Up

Def Let $G = (V, E)$ be a graph. A **Dom Set (DS) for G of size k** is a set $D \subseteq V$ such that

Problem 2 Set Up

Def Let $G = (V, E)$ be a graph. A **Dom Set (DS) for G of size k** is a set $D \subseteq V$ such that

1) $|D| = k$

Problem 2 Set Up

Def Let $G = (V, E)$ be a graph. A **Dom Set (DS) for G of size k** is a set $D \subseteq V$ such that

1) $|D| = k$

2) $(\forall v \in V)[(v \in D) \vee ((\exists w \in D)[(v, w) \in E]]$

Problem 2 Set Up

Def Let $G = (V, E)$ be a graph. A **Dom Set (DS) for G of size k** is a set $D \subseteq V$ such that

1) $|D| = k$

2) $(\forall v \in V)[(v \in D) \vee ((\exists w \in D)[(v, w) \in E]]$

$\mathrm{DS} = \{(G, k)\colon G \text{ has a DS of size } \leq k\}$.

Problem 2 Set Up

Def Let $G = (V, E)$ be a graph. A **Dom Set (DS) for G of size k** is a set $D \subseteq V$ such that

1) $|D| = k$

2) $(\forall v \in V)[(v \in D) \vee ((\exists w \in D)[(v, w) \in E]]$

$\mathrm{DS} = \{(G, k) \colon G$ has a DS of size $\leq k\}$.

It is known that DS is NP-complete.

Problem 2a

Want: connected graph on 1000 vertices that has a DS of size 1.

Problem 2a

Want: connected graph on 1000 vertices that has a DS of size 1.

$V = \{1, \ldots, 1000\}$

Problem 2a

Want: connected graph on 1000 vertices that has a DS of size 1.

$V = \{1, \ldots, 1000\}$
$E = \{(1, 2), (1, 3), \ldots, (1, 1000)\}$

Problem 2a

Want: connected graph on 1000 vertices that has a DS of size 1.

$V = \{1, \ldots, 1000\}$
$E = \{(1,2), (1,3), \ldots, (1,1000)\}$

$D = \{1\}$ is a DS of size 1.

Problem 2b

Want: graph on 1000 vertices, smallest DS has size 1000.

Problem 2b

Want: graph on 1000 vertices, smallest DS has size 1000.

$V = \{1, \ldots, 1000\}$

Problem 2b

Want: graph on 1000 vertices, smallest DS has size 1000.

$V = \{1, \ldots, 1000\}$
$E = \emptyset$

Problem 2b

Want: graph on 1000 vertices, smallest DS has size 1000.

$V = \{1, \dots, 1000\}$
$E = \emptyset$

For all vertices there are no neighbors, so every vertex is in the DS.

Problem 2c

Want a graph on 1000 vertices s.t.:

Problem 2c

Want a graph on 1000 vertices s.t.:

G has a DS of size 500.

Problem 2c

Want a graph on 1000 vertices s.t.:

G has a DS of size 500.
G does not have a DS of size 499.

Problem 2c

Want a graph on 1000 vertices s.t.:

G has a DS of size 500.
G does not have a DS of size 499.

We take the set of 500 pairs of disjoint edges.

Problem 2c

Want a graph on 1000 vertices s.t.:

G has a DS of size 500.
G does not have a DS of size 499.

We take the set of 500 pairs of disjoint edges.
$V = \{1, \ldots, 1000\}$

Problem 2c

Want a graph on 1000 vertices s.t.:

G has a DS of size 500.
G does not have a DS of size 499.

We take the set of 500 pairs of disjoint edges.
$V = \{1, \ldots, 1000\}$
$E = \{(1, 2), (3, 4), \ldots, (999, 1000)\}$

Problem 2c

Want a graph on 1000 vertices s.t.:

G has a DS of size 500.
G does not have a DS of size 499.

We take the set of 500 pairs of disjoint edges.
$V = \{1, \ldots, 1000\}$
$E = \{(1,2), (3,4), \ldots, (999,1000)\}$
Dom Set: $\{1, 3, \ldots, 999\}$.

Problem 2c: Making it Connected

I originally asked for a **connected** graph on 1000 vertices that has a DS of size 500 but not 499.

Problem 2c: Making it Connected

I originally asked for a **connected** graph on 1000 vertices that has a DS of size 500 but not 499.

I thought C_{1000} would work but Nolawe-Isaac-Felix told me it does not.

Problem 2c: Making it Connected

I originally asked for a **connected** graph on 1000 vertices that has a DS of size 500 but not 499.

I thought C_{1000} would work but Nolawe-Isaac-Felix told me it does not.

I didn't intend for this to be a hard problem so I removed **connected** but also asked the class to, if they DID find a connected graph, email it to me for mine and the classes enlightenment.

Problem 2c: Making it Connected

I originally asked for a **connected** graph on 1000 vertices that has a DS of size 500 but not 499.

I thought C_{1000} would work but Nolawe-Isaac-Felix told me it does not.

I didn't intend for this to be a hard problem so I removed **connected** but also asked the class to, if they DID find a connected graph, email it to me for mine and the classes enlightenment.

The following people emailed me a solution:
Aiden Paul
Alex Mendelson (TA)
Anish Bhupalam
Ethan Price
Roc Yu

Problem 2c: Making it Connected

I originally asked for a **connected** graph on 1000 vertices that has a DS of size 500 but not 499.

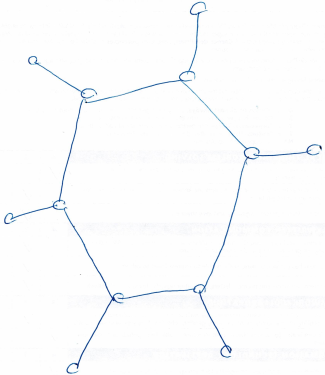
I thought C_{1000} would work but Nolawe-Isaac-Felix told me it does not.

I didn't intend for this to be a hard problem so I removed **connected** but also asked the class to, if they DID find a connected graph, email it to me for mine and the classes enlightenment.

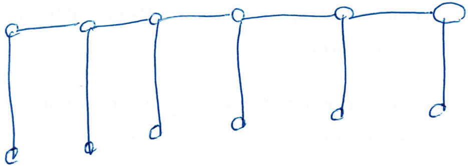
The following people emailed me a solution:
Aiden Paul
Alex Mendelson (TA)
Anish Bhupalam
Ethan Price
Roc Yu

The next two slides show graphs on 12 vertices that have a DS of size 6 but not 5. They convey the ideas.

First Graph on 12 Vertics, DS Size 6, Not 5



Second Graph on 12 Vertics, DS Size 6, Not 5



Problem 2d

$$\mathrm{DS}_{1000} = \{G\colon G \text{ has a DS of size } 1000\}.$$

Problem 2d

$\mathrm{DS}_{1000} = \{G \colon G$ has a DS of size $1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.

Problem 2d

$\mathrm{DS}_{1000} = \{G\colon G$ has a DS of size $1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.
ALGORITHM

Problem 2d

$\mathrm{DS}_{1000} = \{G : G \text{ has a DS of size } 1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.
ALGORITHM
1) Input $G = (V, E)$. Let $n = |V|$.

Problem 2d

$\mathrm{DS}_{1000} = \{G \colon G \text{ has a DS of size } 1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $D \subseteq V$ of size 1000 test if D is a DS.

Problem 2d

$\mathrm{DS}_{1000} = \{G : G$ has a DS of size $1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $D \subseteq V$ of size 1000 test if D is a DS.
(Test: see if each vertex is in D or has a neighbor in D.)

Problem 2d

$\mathrm{DS}_{1000} = \{G : G \text{ has a DS of size } 1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.
ALGORITHM
1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $D \subseteq V$ of size 1000 test if D is a DS.
(Test: see if each vertex is in D or has a neighbor in D.)
If YES then jump out of the loop and output YES.

Problem 2d

$\mathrm{DS}_{1000} = \{G \colon G$ has a DS of size $1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.
ALGORITHM
1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $D \subseteq V$ of size 1000 test if D is a DS.
(Test: see if each vertex is in D or has a neighbor in D.)
If YES then jump out of the loop and output YES.
3) (If got here then no D worked.) Output NO.

Problem 2d

$\mathrm{DS}_{1000} = \{G : G \text{ has a DS of size } 1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $D \subseteq V$ of size 1000 test if D is a DS.
(Test: see if each vertex is in D or has a neighbor in D.)
If YES then jump out of the loop and output YES.
3) (If got here then no D worked.) Output NO.

END OF ALGORITHM

Problem 2d

$\mathrm{DS}_{1000} = \{G\colon G \text{ has a DS of size } 1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $D \subseteq V$ of size 1000 test if D is a DS.
(Test: see if each vertex is in D or has a neighbor in D.)
If YES then jump out of the loop and output YES.
3) (If got here then no D worked.) Output NO.

END OF ALGORITHM

Number of tests: $\binom{n}{1000} \le n^{1000}$.

Problem 2d

$\mathrm{DS}_{1000} = \{G : G$ has a DS of size $1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $D \subseteq V$ of size 1000 test if D is a DS.
(Test: see if each vertex is in D or has a neighbor in D.)
If YES then jump out of the loop and output YES.
3) (If got here then no D worked.) Output NO.

END OF ALGORITHM

Number of tests: $\binom{n}{1000} \leq n^{1000}$.
Each test took $O(|E|) = O(n^2)$.

Problem 2d

$\mathrm{DS}_{1000} = \{G\colon G$ has a DS of size $1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $D \subseteq V$ of size 1000 test if D is a DS.
(Test: see if each vertex is in D or has a neighbor in D.)
If YES then jump out of the loop and output YES.
3) (If got here then no D worked.) Output NO.

END OF ALGORITHM

Number of tests: $\binom{n}{1000} \le n^{1000}$.

Each test took $O(|E|) = O(n^2)$.

So time is $O(n^{1002})$.

Problem 2d

$\mathrm{DS}_{1000} = \{G \colon G$ has a DS of size $1000\}$.
Show that $\mathrm{DS}_{1000} \in \mathrm{P}$.

ALGORITHM

1) Input $G = (V, E)$. Let $n = |V|$.
2) For all $D \subseteq V$ of size 1000 test if D is a DS.
(Test: see if each vertex is in D or has a neighbor in D.)
If YES then jump out of the loop and output YES.
3) (If got here then no D worked.) Output NO.

END OF ALGORITHM

Number of tests: $\binom{n}{1000} \le n^{1000}$.

Each test took $O(|E|) = O(n^2)$.

So time is $O(n^{1002})$. Thats a polynomial!

Problem 2e: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

Problem 2e: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

The algorithm was in time $O(n^{1002})$.

Problem 2e: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

The algorithm was in time $O(n^{1002})$.

VOTE:

Problem 2e: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

The algorithm was in time $O(n^{1002})$.

VOTE:

$\exists$ an algorithm that is substantially better than $O(n^{1002})$.

Problem 2e: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

The algorithm was in time $O(n^{1002})$.

VOTE:

$\exists$ an algorithm that is substantially better than $O(n^{1002})$.

Does not $\exists$ an algorithm that is substantially better than $O(n^{1002})$.

Problem 2e: Think About

Your algorithm in Part d ran in time $O(n^d)$ for some d.

The algorithm was in time $O(n^{1002})$.

VOTE:
$\exists$ an algorithm that is substantially better than $O(n^{1002})$.
Does not $\exists$ an algorithm that is substantially better than $O(n^{1002})$.
The question is UNKNOWN TO SCIENCE!

Problem 2f

UNKNOWN TO SCIENCE.

Problem 2f

UNKNOWN TO SCIENCE.

Def A problem of the form

$$\{(G,k) \colon G \text{ does the hokey pokey} \le k \text{ times }\}$$

is **Fixed Parameter Tractable (FPT)** if,

Problem 2f

UNKNOWN TO SCIENCE.

Def A problem of the form

$$\{(G,k) \colon G \text{ does the hokey pokey} \le k \text{ times }\}$$

is **Fixed Parameter Tractable (FPT)** if,

for all k, there is an algorithm for

$$\{G \colon G \text{ does the hokey pokey} \le k \text{ times }\}$$

with run time $f(k)n^{O(1)}$ where the $O(1)$ is ind. of k.

Problem 2f

UNKNOWN TO SCIENCE.

Def A problem of the form

$$\{(G, k) \colon G \text{ does the hokey pokey} \le k \text{ times }\}$$

is **Fixed Parameter Tractable (FPT)** if,

for all k, there is an algorithm for

$$\{G \colon G \text{ does the hokey pokey} \le k \text{ times }\}$$

with run time $f(k)n^{O(1)}$ where the $O(1)$ is ind. of k.

We showed that VC is FPT.

Problem 2f

UNKNOWN TO SCIENCE.

Def A problem of the form

$$\{(G,k)\colon G \text{ does the hokey pokey} \le k \text{ times }\}$$

is **Fixed Parameter Tractable (FPT)** if,

for all k, there is an algorithm for

$$\{G\colon G \text{ does the hokey pokey} \le k \text{ times }\}$$

with run time $f(k)n^{O(1)}$ where the $O(1)$ is ind. of k.

We showed that VC is FPT.

There is a complexity theory of FPT.

Problem 2f

UNKNOWN TO SCIENCE.

Def A problem of the form

$$\{(G,k) \colon G \text{ does the hokey pokey} \le k \text{ times } \}$$

is **Fixed Parameter Tractable (FPT)** if,

for all k, there is an algorithm for

$$\{G \colon G \text{ does the hokey pokey} \le k \text{ times } \}$$

with run time $f(k)n^{O(1)}$ where the $O(1)$ is ind. of k.

We showed that VC is FPT.

There is a complexity theory of FPT.
Theory says DS is prob not FPT.

Problem 2f

UNKNOWN TO SCIENCE.

Def A problem of the form

$$\{(G,k)\colon G \text{ does the hokey pokey} \le k \text{ times }\}$$

is **Fixed Parameter Tractable (FPT)** if,

for all k, there is an algorithm for

$$\{G\colon G \text{ does the hokey pokey} \le k \text{ times }\}$$

with run time $f(k)n^{O(1)}$ where the $O(1)$ is ind. of k.

We showed that VC is FPT.

There is a complexity theory of FPT.
Theory says DS is prob not FPT.
Similar to NP-completeness saying SAT is prob not in P.

Problem 2f

UNKNOWN TO SCIENCE.

Def A problem of the form

$$\{(G,k)\colon G \text{ does the hokey pokey} \le k \text{ times }\}$$

is **Fixed Parameter Tractable (FPT)** if,

for all k, there is an algorithm for

$$\{G\colon G \text{ does the hokey pokey} \le k \text{ times }\}$$

with run time $f(k)n^{O(1)}$ where the $O(1)$ is ind. of k.

We showed that VC is FPT.

There is a complexity theory of FPT.
Theory says DS is prob not FPT.
Similar to NP-completeness saying SAT is prob not in P.
Still UNKNOWN TO SCIENCE!