Structural Parameterizations of Dominating Set Variants

Dishant Goyal Ashwin Jacob Kaushtubh Kumar
Diptapriyo Majumdar Venkatesh Raman

June 6, 2018, CSR, Moscow, Russia

Outline

Dominating Set Variants

Dominating Set Variants

- DOMINATING SET (DS) :
  A subset $S \subseteq V(G)$ such that every vertex not in S is adjacent to at least one member of S.

Dominating Set Variants

- Dominating Set (DS) :
  A subset $S \subseteq V(G)$ such that every vertex not in S is adjacent to at least one member of S.
- Independent Dominating Set (IDS) :
  S is also independent.

Dominating Set Variants

- Dominating Set (DS) :
  A subset $S \subseteq V(G)$ such that every vertex not in S is adjacent to at least one member of S.
- Independent Dominating Set (IDS) :
  S is also independent.
- Efficient Dominating Set (EDS) :
  For every vertex $v \in V, |N[v] \cap S| = 1$.

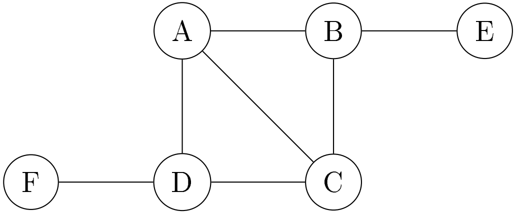
Dominating Set Variants

- DOMINATING SET (DS) :
  A subset $S \subseteq V(G)$ such that every vertex not in S is adjacent to at least one member of S.
- INDEPENDENT DOMINATING SET (IDS) :
  S is also independent.
- EFFICIENT DOMINATING SET (EDS) :
  For every vertex $v \in V, |N[v] \cap S| = 1$.
- THRESHOLD DOMINATING SET (ThDS) with threshold r :
  For every vertex $v, |N(v) \cap S| \geq r$.

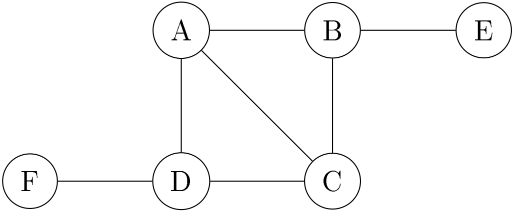
Dominating Set Variants

- Dominating Set (DS) :
  A subset $S \subseteq V(G)$ such that every vertex not in S is adjacent to at least one member of S.
- Independent Dominating Set (IDS) :
  S is also independent.
- Efficient Dominating Set (EDS) :
  For every vertex $v \in V, |N[v] \cap S| = 1$.
- Threshold Dominating Set (ThDS) with threshold r :
  For every vertex $v, |N(v) \cap S| \geq r$.
- Total Dominating Set (TDS) : ThDS with $r = 1$.

Example

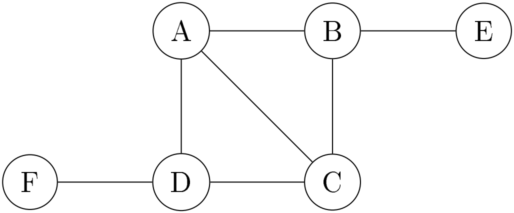


Example



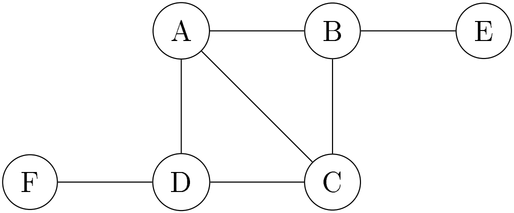
$\{B, D\}$ is a Dominating Set. Also an IDS.

Example



$\{B, D\}$ is a Dominating Set. Also an IDS.
$\{B, F\}$ is an EDS.

Example



$\{B, D\}$ is a Dominating Set. Also an IDS.
$\{B, F\}$ is an EDS.
$\{A, B, D\}$ is a Total Dominating Set.

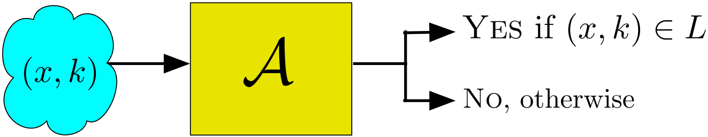
Parameterized Problem

- A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$. Input instance of L is (x, k) where $x \in \Sigma^*, k \in \mathbb{N}$. k is called *parameter*.

Parameterized Problem

- A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$. Input instance of L is (x, k) where $x \in \Sigma^*, k \in \mathbb{N}$. k is called *parameter*.
- Example: Feedback Vertex Set parameterized by Solution Size.
  $L = \{(G, k) | \exists S \subseteq V(G)$ such that $|S| \leq k$ and $G \setminus S$ is acyclic$\}$.

Fixed-Parameter Tractability (FPT)



- Algorithm $\mathcal{A}$ runs in $f(k) \cdot |x|^c$ time.
- $\mathcal{A}$ is called FIXED PARAMETER ALGORITHM.

Examples

- Feedback Vertex Set parameterized by solution size k admits $\mathcal{O}(3.618^k \cdot n^{\mathcal{O}(1)})$ time algorithm [KP'14].
- Vertex Cover parameterized by solution size k admits $\mathcal{O}(1.27^k \cdot n^{\mathcal{O}(1)})$ time algorithm [CKJ'01].

Hardness

Hardness

- $W[1]$-hard : Problems unlikely to be FPT.

Hardness

- $W[1]$-hard : Problems unlikely to be FPT.
  Examples : INDEPENDENT SET, CLIQUE parameterized by solution size k

Hardness

- $W[1]$-hard : Problems unlikely to be FPT.
  Examples : INDEPENDENT SET, CLIQUE parameterized by solution size k
- Para NP-hard : Problems that are NP-hard for a constant value for the parameter.

Hardness

- $W[1]$-hard : Problems unlikely to be FPT.
  Examples : Independent Set, Clique parameterized by solution size k
- Para NP-hard : Problems that are NP-hard for a constant value for the parameter.
  Example: k-coloring

Dominating Set Variants

- All dominating set variants parameterized by solution size are $W[1]$-hard. Most of them are actually $W[2]$-hard.
- Other parameters?

Structural Parameterizations

- Parameters based on the structural properties of the input.
- Example : Maximum degree, treewidth, Minimum Vertex Cover, deletion distance to an easy instance

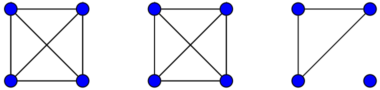
Cluster graph and Split graph

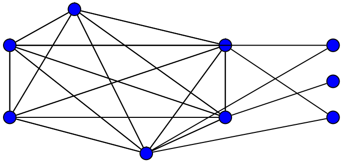
Cluster graph and Split graph

- CLUSTER GRAPH : Every connected component of the graph is a clique.

Cluster graph and Split graph

- Cluster graph : Every connected component of the graph is a clique.
- Split graph : The vertex set of the graph can be partitioned into a clique and an independent set.





- All dominating set variants are solvable in polynomial time on cluster graphs.
- Dominating Set, TDS and ThDS are NP-hard on split graphs.
- EDS and IDS are solvable in polynomial time on split graphs.

Outline

Results for deletion distance to cluster graph

	Algorithms	Lower Bounds
DS, TDS	$\mathcal{O}^*(3^k)$	$\mathcal{O}^*((2-\varepsilon)^k)$
IDS	$\mathcal{O}^*(3^k)$	$\mathcal{O}^*((2-\varepsilon)^k)$
EDS	$\mathcal{O}^*(3^k)$	$\mathcal{O}^*(2^{o(k)})$
ThDS	$\mathcal{O}^*((r+2)^k)$	

Table: Results for deletion distance to cluster graph

Results for deletion distance to split graph

	Algorithms	Lower Bounds
DS, TDS		para-NP-hard
IDS	$\mathcal{O}^*(2^k)$	$\mathcal{O}^*((2-\varepsilon)^k)$
EDS	$\mathcal{O}^*(3^{k/2})$	$\mathcal{O}^*(2^{o(k)})$
ThDS		para-NP-hard

Table: Results for deletion distance to split graph

Outline

Problem Definition

DOMINATING SET-CLUSTER VD
Input: An undirected graph $G = (V, E), S \subseteq V(G)$ such that every component of $G \setminus S$ is a clique and an integer ℓ.
Parameter: $|S|$
Question: Is there a dominating set in G of size ℓ?

Problem Definition

DOMINATING SET-CLUSTER VD
Input: An undirected graph $G = (V, E), S \subseteq V(G)$ such that every component of $G \setminus S$ is a clique and an integer ℓ.
Parameter: $|S|$
Question: Is there a dominating set in G of size ℓ?

- Assume S is given as part of input. If not use the algorithm by [BCKP'16] to get S in time $\mathcal{O}^*(1.9102^k)$.

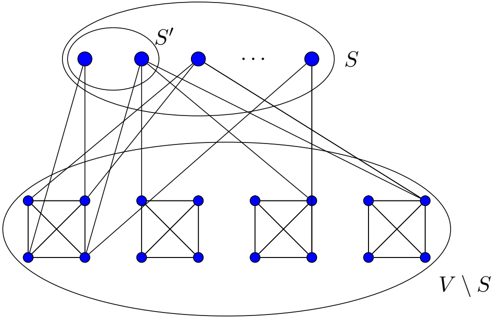
Algorithm

Algorithm

- Guess S', the part of the solution intersecting with S.
- Delete $N[S'] \cap S$.

Algorithm

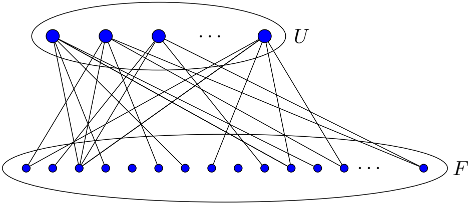
- Guess S', the part of the solution intersecting with S.
- Delete $N[S'] \cap S$.



Disjoint Problem Definition

DS-DISJOINTCLUSTER
Input: An undirected graph $G = (V, E)$, $S \subseteq V$ such that every connected component of $G \setminus S$ is a clique, a $(0, 1)$ vector $(f_1, f_2, \ldots, f_q)$ corresponding for the cliques $(C_1, C_2, \ldots, C_q)$ and an integer ℓ.
Parameter: $|S|$
Question: Does there exist a subset $T \subseteq V \setminus S$ of size ℓ, that dominates all vertices of S and all vertices of all cliques C_i with flags $f_i = 1$?

Set Cover
Input: A universe U, a family of sets $\mathcal{F} = \{S_1, \ldots, S_m\}$ of subsets of U and an integer ℓ.
Parameter: $|U| = k$
Question: Does there exist a subset $\mathcal{F}' \subseteq \mathcal{F}$ of size ℓ covering U?



Equivalent problem

SET-COVER WITH PARTITION

Input: A universe U, a family of sets $\mathcal{F} = \{S_1, \ldots, S_m\}$, a partition $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_q)$ of $\mathcal{F}$, a $(0,1)$ vector $(f_1, f_2, \ldots, f_q)$ corresponding to each block in the partition $(\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_q)$ and an integer ℓ.

Parameter: $|U| = k$

Question: Does there exist a subset $\mathcal{F}' \subseteq \mathcal{F}$ of size ℓ covering U and from each block $\mathcal{B}_i$ with flags $f_i = 1$ at least one set is picked?

Universe $U = S$.

For each vertex $v \in V \setminus S$, we define a set $S_v = N(v) \cap S$.

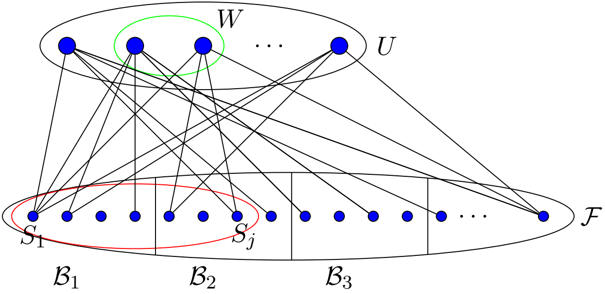
Family of sets $\mathcal{F} = \{S_v : v \in V \setminus S\}$.

$(\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_q) = (C_1, C_2, \ldots, C_q)$

Dynamic Programming Algorithm for Set-Cover with Partition

Dynamic Programming Algorithm for Set-Cover with Partition

For $W \subseteq U$, index j of set S_j and flag $f \in \{0, 1\}$



$OPT[W, j, f]$: cardinality of the minimum subset X of $\{S_1, \ldots, S_j\}$ covering W such that

$OPT[W, j, f]$: cardinality of the minimum subset X of $\{S_1, \ldots, S_j\}$ covering W such that

- from each block $\mathcal{B}_i$ with $f_i = 1$, there is at least one set in X

$OPT[W, j, f]$: cardinality of the minimum subset X of $\{S_1, \ldots, S_j\}$ covering W such that

- from each block $\mathcal{B}_i$ with $f_i = 1$, there is at least one set in X
- except the block $\mathcal{B}_x$ containing the set S_j where we reset the flag to f to indicate that at least f sets are required in that block.

Dynamic Programming Recurrence

Dynamic Programming Recurrence

- Case 1 : S_{j+1} is not the first set in its block $\mathcal{B}_x$.

Dynamic Programming Recurrence

- Case 1 : S_{j+1} is not the first set in its block $\mathcal{B}_x$.

$$OPT[W, j{+}1, f] = min\Big\{OPT[W, j, f], 1{+}OPT[W \setminus S_{j+1}, j, 0]\Big\}$$

Dynamic Programming Recurrence

- Case 1 : S_{j+1} is not the first set in its block $\mathcal{B}_x$.

$$OPT[W, j+1, f] = min\Big\{OPT[W, j, f], 1+OPT[W \setminus S_{j+1}, j, 0]\Big\}$$

- Case 2 : S_{j+1} is the first set in its block $\mathcal{B}_x$.

Dynamic Programming Recurrence

- Case 1 : S_{j+1} is not the first set in its block $\mathcal{B}_x$.

$$OPT[W, j{+}1, f] = min\Big\{OPT[W, j, f], 1{+}OPT[W \backslash S_{j+1}, j, 0]\Big\}$$

- Case 2 : S_{j+1} is the first set in its block $\mathcal{B}_x$.

$$OPT[W, j{+}1, f] = \begin{cases} 1 + OPT[W \setminus S_{j+1}, j, f_{x-1}] & \text{if } f = 1 \\ min\Big\{OPT[W, j, f_{x-1}], \\ 1 + OPT[W \setminus S_{j+1}, j, f_{x-1}]\Big\} \\ \qquad\qquad\qquad\qquad\qquad \text{if } f = 0 \end{cases}$$

Running Time

Running Time

- Number of subproblems : $2^{|U|+1} \cdot m$

Running Time

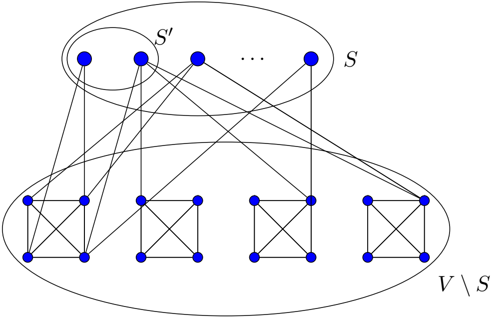
- Number of subproblems : $2^{|U|+1} \cdot m$
- Running time for SET-COVER WITH PARTITION: $\mathcal{O}(2^{|U|} \cdot m^2)$.

Running Time

- Number of subproblems : $2^{|U|+1} \cdot m$
- Running time for Set-Cover with Partition: $\mathcal{O}(2^{|U|} \cdot m^2)$.
- Running time for Dominating Set-Cluster VD : $\sum_{i=1}^{k} \binom{k}{i} \mathcal{O}^*(2^{k-i}) = \mathcal{O}^*(3^k)$

Other Variants

- EDS ,IDS, TDS : $\mathcal{O}^*(3^k)$
- ThDS : $\mathcal{O}^*((r+2)^k)$



Lower Bound Conjectures

Lower Bound Conjectures

- EXPONENTIAL TIME HYPOTHESIS (ETH) ([IPZ01,IP01])
  3-CNF-SAT cannot be solved in $\mathcal{O}^*(2^{o(n)})$ time where the input formula has n variables and m clauses.

Lower Bound Conjectures

- EXPONENTIAL TIME HYPOTHESIS (ETH) ([IPZ01,IP01])
  3-CNF-SAT cannot be solved in $\mathcal{O}^*(2^{o(n)})$ time where the input formula has n variables and m clauses.

- STRONG EXPONENTIAL TIME HYPOTHESIS (SETH)([IPZ01])

  There is no $\varepsilon > 0$ such that $\forall q \geq 3$, q-CNFSAT can be solved in $\mathcal{O}^*((2-\varepsilon)^n)$ time where n is the number of variables in input formula.

Lower Bound Conjectures

- Exponential Time Hypothesis (ETH) ([IPZ01,IP01])
3-CNF-SAT cannot be solved in $\mathcal{O}^*(2^{o(n)})$ time where the input formula has n variables and m clauses.

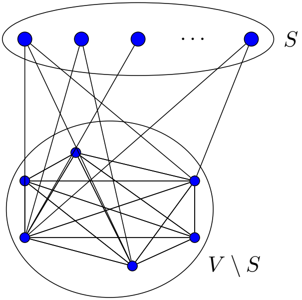
- Strong Exponential Time Hypothesis (SETH)([IPZ01])

  There is no $\varepsilon > 0$ such that $\forall q \geq 3$, q-CNFSAT can be solved in $\mathcal{O}^*((2-\varepsilon)^n)$ time where n is the number of variables in input formula.

- Set Cover Conjecture (SCC)

  There is no $\varepsilon > 0$ such that SET COVER can be solved in $\mathcal{O}^*((2-\varepsilon)^n)$ time where n is the size of the universe.

- Dominating Set-Cluster VD and cannot be solved in $O^*((2-\varepsilon)^k)$ running time for any $\varepsilon > 0$ unless SCC fails.



Lower Bounds cont'd

- IDS-ClusterVD cannot be solved in time $\mathcal{O}^*((2-\varepsilon)^k)$ for any $\varepsilon > 0$ unless SETH fails.
- EDS-Vertex Cover cannot be solved in $2^{o(|S|)}$ time unless ETH fails.

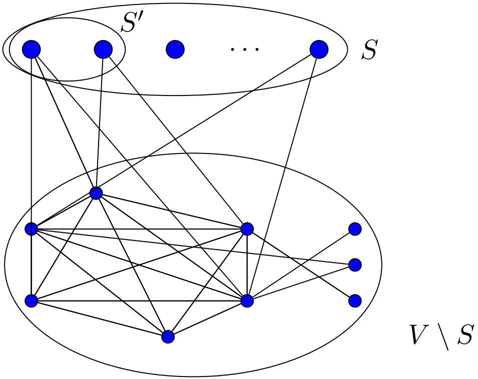
Outline

Para-NP-hardness

- Dominating Set and Total Dominating Set are NP-hard on Split graphs.
- Hence para-NP-hard for deletion Distance to Split Graphs.

Algorithm for EDS and IDS

- EDS and IDS can be solved in $\mathcal{O}^*(2^k)$ time.



Lower Bounds

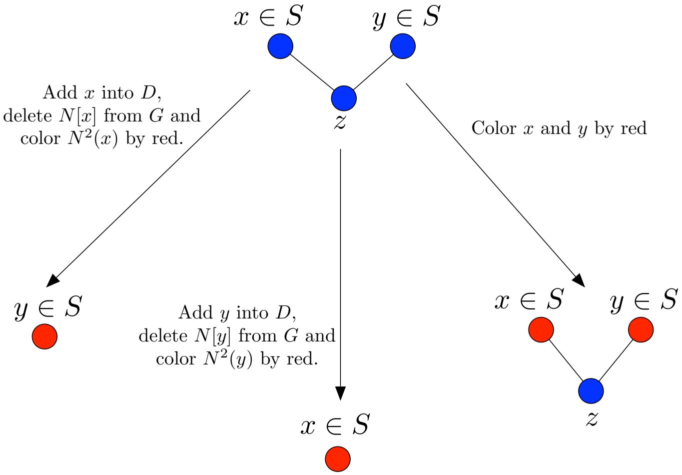
- IDS-SplitVD cannot be solved in $\mathcal{O}^*((2-\varepsilon)^k)$ time unless SETH fails.
- EDS-SplitVD cannot be solved in $2^{o(k)}$ time unless ETH fails.

Improved Algorithm for EDS

- EDS-SplitVD can be solved in $\mathcal{O}^*(3^{k/2})$ time.
  - Color all the vertices of G blue initially.
  - Whenever a vertex gets undeletable, make it red.
  - Measure $=$ Number of blue vertices in S, initial value k.

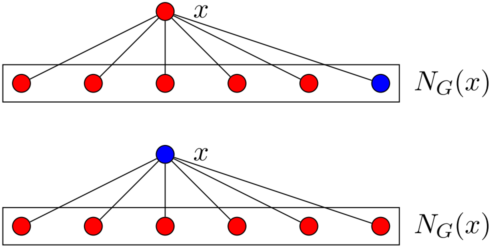
Branching Rule 1

- $x, y \in S$ blue vertices with distance at most 2.



Reduction Rule 1

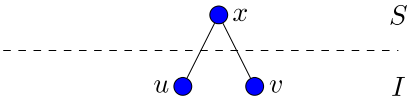
- Blue vertices are forced below.



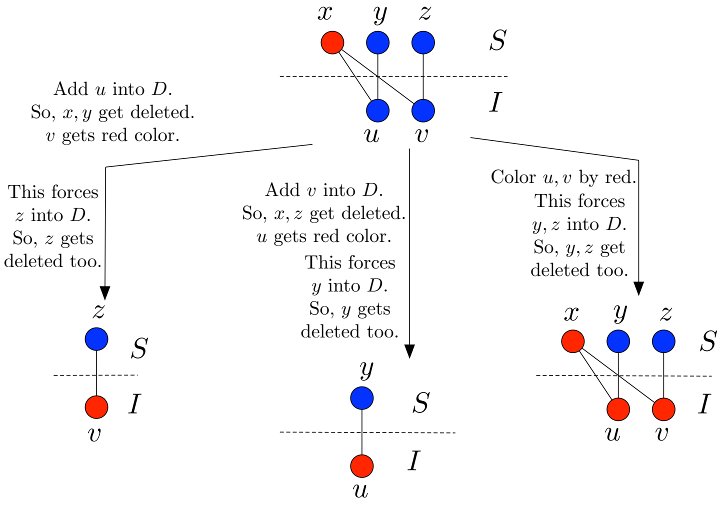
- Guess the intersection of the clique part of the split graph C in the solution.
- One-time branch on at most $|C| + 1$ cases.
- Now only vertices of independent set part I left below.
- Note that after Branching Rule 1, any vertex in I have exactly one blue vertex in S as its neighbour .

Reduction Rule 2

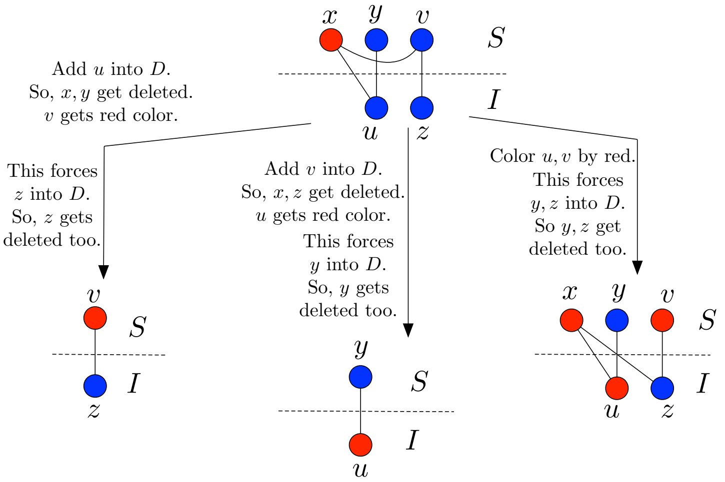
- x is forced in the solution.



Branching Rule 2.1



Branching Rule 2.2



- After applying every above rules, look at blue vertices $u \in S, v \in I$ and $(u, v) \in E(G)$.
- Either u or v in solution as $N(u) \setminus \{v\} = N(v) \setminus \{u\}$.

Future Work

- Close the $3^k - 2^k$ upper-lower bound gap for (DS/IDS/TDS)-Cluster VD.
- Deletion distance to other easy instances.
- Other dominating set variants.

THANK YOU