Even delta-matroids and the complexity of planar Boolean CSPs

Alexandr Kazda, Vladimir Kolmogorov, Michal Rolínek
Our world: CSP\((\{0,1\}, \Gamma)\) where \(\Gamma\) contains constants \(\{0\}\) and \(\{1\}\).

We limit the instance shape – each variable appears at most \(k\) times. For which \(\Gamma\)s do we get easier CSP?


Only interesting case: \(k = 2\).
Our world: CSP(\{0, 1\}, \Gamma) where \Gamma contains constants \{0\} and \{1\}.

- We limit the instance shape – each variable appears at most \(k\) times. For which \Gamma s do we get easier CSP?
- Only interesting case: \(k = 2\).
Our world: CSP(\(\{0, 1\}, \Gamma\)) where \(\Gamma\) contains constants \(\{0\}\) and \(\{1\}\).

We limit the instance shape – each variable appears at most \(k\) times. For which \(\Gamma\)s do we get easier CSP?

- Only interesting case: \(k = 2\).
Our world: CSP(\{0, 1\}, \Gamma) where \Gamma contains constants \{0\} and \{1\}.

We limit the instance shape – each variable appears at most \(k\) times. For which \(\Gamma\)s do we get easier CSP?


Only interesting case: \(k = 2\).
Our world: CSP(\(\{0, 1\}, \Gamma\)) where \(\Gamma\) contains constants \(\{0\}\) and \(\{1\}\).

We limit the instance shape – each variable appears at most \(k\) times.
For which \(\Gamma\)s do we get easier CSP?


Only interesting case: \(k = 2\).
Our world: CSP(\{0, 1\}, \Gamma) where \Gamma contains constants \{0\} and \{1\}.

We limit the instance shape – each variable appears at most \(k\) times. For which \(\Gamma\)s do we get easier CSP?


Only interesting case: \(k = 2\).
Wlog each variable appears in exactly two constrains.

We can draw instances of this CSP as graphs with variables = edges.

Some people call this binary CSP, we prefer edge CSP.

Feder: The only new case is when all relations in $\Gamma$ are $\Delta$-matroids.
Edge CSP

- Wlog each variable appears in exactly two constrains.
- We can draw instances of this CSP as graphs with variables = edges.
- Some people call this binary CSP, we prefer edge CSP.
- Feder: The only new case is when all relations in \( \Gamma \) are \( \Delta \)-matroids.
Wlog each variable appears in exactly two constraints. We can draw instances of this CSP as graphs with variables = edges. Some people call this binary CSP, we prefer edge CSP. Feder: The only new case is when all relations in \( \Gamma \) are \( \Delta \)-matroids.
Wlog each variable appears in exactly two constrains.
We can draw instances of this CSP as graphs with variables = edges.
Some people call this binary CSP, we prefer edge CSP.
Feder: The only new case is when all relations in $\Gamma$ are $\Delta$-matroids.
Wlog each variable appears in exactly two constraints.

We can draw instances of this CSP as graphs with variables $\equiv$ edges.

Some people call this binary CSP, we prefer edge CSP.

Feder: The only new case is when all relations in $\Gamma$ are $\Delta$-matroids.
\(\Delta\)-matroids

- AKA “generalized matroids”
- \(R \neq \emptyset\) is an even \(\Delta\)-matroid if all tuples in \(R\) have the same parity and for all \(\alpha, \beta \in R\) and for all \(u\) variables such that \(\alpha(u) \neq \beta(u)\) there exists \(v \neq u\) such that \(\alpha(v) \neq \beta(v)\) and \(\alpha \oplus u \oplus v \in R\):

- \(\Delta\)-matroids: No parity restriction, enough to have \(\alpha \oplus u \in R\) instead of \(\alpha \oplus u \oplus v \in R\).
△-matroids

- AKA “generalized matroids”
- $R \neq \emptyset$ is an even △-matroid if all tuples in $R$ have the same parity and for all $\alpha, \beta \in R$ and for all $u$ variables such that $\alpha(u) \neq \beta(u)$ there exists $v \neq u$ such that $\alpha(v) \neq \beta(v)$ and $\alpha \oplus u \oplus v \in R$:

- △-matroids: No parity restriction, enough to have $\alpha \oplus u \in R$ instead of $\alpha \oplus u \oplus v \in R$. 
Δ-matroids

- AKA “generalized matroids”
- $R \neq \emptyset$ is an even Δ-matroid if all tuples in $R$ have the same parity and for all $\alpha, \beta \in R$ and for all $u$ variables such that $\alpha(u) \neq \beta(u)$ there exists $v \neq u$ such that $\alpha(v) \neq \beta(v)$ and $\alpha \oplus u \oplus v \in R$:

\[
\begin{align*}
\alpha &= 0 \ 0 \ 0 \ 1 \ 1 \\
\beta &= 1 \ 1 \ 1 \ 0 \ 1
\end{align*}
\]

- Δ-matroids: No parity restriction, enough to have $\alpha \oplus u \in R$ instead of $\alpha \oplus u \oplus v \in R$. 

A K, V K, M R (IST Austria)  Edge CSP for even Δ-matroids 4 / 12
Δ-matroids

- AKA "generalized matroids"

- $R \neq \emptyset$ is an even Δ-matroid if all tuples in $R$ have the same parity and for all $\alpha, \beta \in R$ and for all $u$ variables such that $\alpha(u) \neq \beta(u)$ there exists $v \neq u$ such that $\alpha(v) \neq \beta(v)$ and $\alpha \oplus u \oplus v \in R$:

  \[
  \begin{align*}
  \alpha &= 0 \ 0 \ 0 \ 1 \ 1 \\
  \beta &= 1 \ 1 \ 1 \ 0 \ 1
  \end{align*}
  \]

- Δ-matroids: No parity restriction, enough to have $\alpha \oplus u \in R$ instead of $\alpha \oplus u \oplus v \in R$. 

A K, V K, M R (IST Austria)
\(\Delta\text{-matroids}\)

- AKA “generalized matroids”

- \(R \neq \emptyset\) is an even \(\Delta\text{-matroid}\) if all tuples in \(R\) have the same parity and for all \(\alpha, \beta \in R\) and for all \(u\) variables such that \(\alpha(u) \neq \beta(u)\) there exists \(v \neq u\) such that \(\alpha(v) \neq \beta(v)\) and \(\alpha \oplus u \oplus v \in R:\)

\[
\begin{align*}
\alpha &= 0 0 0 1 1 \\
\beta &= 1 1 1 0 1
\end{align*}
\]

- \(\Delta\text{-matroids}\): No parity restriction, enough to have \(\alpha \oplus u \in R\) instead of \(\alpha \oplus u \oplus v \in R\).
-matroids

- AKA “generalized matroids”
- \( R \neq \emptyset \) is an even \( \Delta \)-matroid if all tuples in \( R \) have the same parity and for all \( \alpha, \beta \in R \) and for all \( u \) variables such that \( \alpha(u) \neq \beta(u) \) there exists \( v \neq u \) such that \( \alpha(v) \neq \beta(v) \) and \( \alpha \oplus u \oplus v \in R \):

\[
\begin{align*}
\alpha &= 0 \ 0 \ 0 \ 1 \ 1 \\
\beta &= 1 \ 1 \ 1 \ 0 \ 1 \\
\alpha \oplus u \oplus v &= 1 \ 1 \ 0 \ 1 \ 1
\end{align*}
\]

- \( \Delta \)-matroids: No parity restriction, enough to have \( \alpha \oplus u \in R \) instead of \( \alpha \oplus u \oplus v \in R \).
\(\Delta\)-matroids

- AKA “generalized matroids”
- \(R \neq \emptyset\) is an even \(\Delta\)-matroid if all tuples in \(R\) have the same parity and for all \(\alpha, \beta \in R\) and for all \(u\) variables such that \(\alpha(u) \neq \beta(u)\) there exists \(v \neq u\) such that \(\alpha(v) \neq \beta(v)\) and \(\alpha \oplus u \oplus v \in R\):

\[
\begin{align*}
\alpha &= 0 \ 0 \ 0 \ 1 \ 1 \\
\beta &= 1 \ 1 \ 1 \ 0 \ 1 \\
\alpha \oplus u \oplus v &= 1 \ 1 \ 1 \ 0 \ 1 \\
\end{align*}
\]

- \(\Delta\)-matroids: No parity restriction, enough to have \(\alpha \oplus u \in R\) instead of \(\alpha \oplus u \oplus v \in R\).
Good and bad news about $\Delta$-matroids

- Intersection of two even $\Delta$-matroids need not be a $\Delta$-matroid:

$$\begin{align*}
\begin{bmatrix}
0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 \\
1 & 1 & 1 & 1
\end{bmatrix}
\cap
\begin{bmatrix}
0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
1 & 1 & 1 & 1
\end{bmatrix}
= \begin{bmatrix}
0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1
\end{bmatrix}
\end{align*}$$

- If there is any way to use polymorphisms here, we did not find it.

- However, (even) $\Delta$-matroids are closed under primitive positive definitions where each bound variable appears exactly twice and each free variable exactly once.
Good and bad news about Δ-matroids

- Intersection of two even Δ-matroids need not be a Δ-matroid:
  \[
  \begin{align*}
  \left\{ (0, 0, 0, 0), (1, 1, 0, 0), (0, 0, 1, 1), (1, 1, 1, 1) \right\} \cap \\
  \left\{ (0, 0, 0, 0), (1, 0, 1, 0), (0, 1, 0, 1), (1, 1, 1, 1) \right\} &= \left\{ (0, 0, 0, 0), (1, 1, 1, 1) \right\}
  \end{align*}
  \]

- If there is any way to use polymorphisms here, we did not find it.
- However, (even) Δ-matroids are closed under primitive positive definitions where each bound variable appears exactly twice and each free variable exactly once.
Good and bad news about Δ-matroids

- Intersection of two even Δ-matroids need not be a Δ-matroid:

\[
\begin{align*}
\{(0, 0, 0, 0), (1, 1, 0, 0), (0, 0, 1, 1), (1, 1, 1, 1)\} \\
\cap \\
\{(0, 0, 0, 0), (1, 0, 1, 0), (0, 1, 0, 1), (1, 1, 1, 1)\} = \\
\{(0, 0, 0, 0), (1, 1, 1, 1)\}
\end{align*}
\]

- If there is any way to use polymorphisms here, we did not find it.

- However, (even) Δ-matroids are closed under primitive positive definitions where each bound variable appears exactly twice and each free variable exactly once.
Good and bad news about $\Delta$-matroids

- Intersection of two even $\Delta$-matroids need not be a $\Delta$-matroid:

$$\begin{pmatrix}
(0 & 0 & 0 & 0) \\
(1 & 1 & 0 & 0) \\
(0 & 0 & 1 & 1) \\
(1 & 1 & 1 & 1)
\end{pmatrix} \cap
\begin{pmatrix}
(0 & 0 & 0 & 0) \\
(1 & 0 & 1 & 0) \\
(0 & 1 & 0 & 1) \\
(1 & 1 & 1 & 1)
\end{pmatrix} = \begin{pmatrix}
(0 & 0 & 0 & 0) \\
(1 & 1 & 1 & 1)
\end{pmatrix}$$

- If there is any way to use polymorphisms here, we did not find it.
- However, (even) $\Delta$-matroids are closed under primitive positive definitions where each bound variable appears exactly twice and each free variable exactly once.
Perfect matchings in graphs: Given $G$, assign 0 or 1 to each edge so that each vertex of $G$ is incident to exactly one edge labelled by 1.

Known to be polynomial (J. Edmonds, 1965).

Perfect matchings correspond to edge CSP with constraints of the form

$$\{(1 \ 0 \ 0 \ \ldots \ \ 0)
\ (0 \ 1 \ 0 \ \ldots \ \ 0)
\ (0 \ 0 \ 1 \ \ldots \ \ 0)
\ \ldots
\ (0 \ 0 \ 0 \ \ldots \ \ 1)\}\$$

These are even $\Delta$-matroids!
Perfect matchings in graphs: Given $G$, assign 0 or 1 to each edge so that each vertex of $G$ is incident to exactly one edge labelled by 1.

- Known to be polynomial (J. Edmonds, 1965).

- Perfect matchings correspond to edge CSP with constraints of the form

$$\{(1 \ 0 \ 0 \ \ldots \ 0)$$

$$(0 \ 1 \ 0 \ \ldots \ 0)$$

$$(0 \ 0 \ 1 \ \ldots \ 0)$$

$$\ldots$$

$$(0 \ 0 \ 0 \ \ldots \ 1)\}$$

- These are even $\Delta$-matroids!
Perfect matchings in graphs: Given $G$, assign 0 or 1 to each edge so that each vertex of $G$ is incident to exactly one edge labelled by 1.

Known to be polynomial (J. Edmonds, 1965).

Perfect matchings correspond to edge CSP with constraints of the form

$$\left\{ \begin{array}{ccc}
(1 & 0 & 0 & \ldots & 0) \\
(0 & 1 & 0 & \ldots & 0) \\
(0 & 0 & 1 & \ldots & 0) \\
& \vdots & & & \\
(0 & 0 & 0 & \ldots & 1) \\
\end{array} \right\}$$

These are even $\Delta$-matroids!
Perfect matchings in graphs: Given $G$, assign 0 or 1 to each edge so that each vertex of $G$ is incident to exactly one edge labelled by 1.

Known to be polynomial (J. Edmonds, 1965).

Perfect matchings correspond to edge CSP with constraints of the form

$$\{(1 \ 0 \ 0 \ \ldots \ \ 0)$$

$$\ (0 \ 1 \ 0 \ \ldots \ \ 0)$$

$$\ (0 \ 0 \ 1 \ \ldots \ \ 0)$$

$$\ \ldots$$

$$\ (0 \ 0 \ 0 \ \ldots \ \ 1)\}$$

These are even $\Delta$-matroids!
Perfect matchings in graphs: Given $G$, assign 0 or 1 to each edge so that each vertex of $G$ is incident to exactly one edge labelled by 1.

Known to be polynomial (J. Edmonds, 1965).

Perfect matchings correspond to edge CSP with constraints of the form

$$\{(1 \ 0 \ 0 \ \ldots \ 0) \\
(0 \ 1 \ 0 \ \ldots \ 0) \\
(0 \ 0 \ 1 \ \ldots \ 0) \\
\vdots \\
(0 \ 0 \ 0 \ \ldots \ 1)\}$$

These are even $\Delta$-matroids!
Planar CSPs

- CSP(\{0,1\}, \Gamma) with incidence graphs of instances \textit{planar}.
- Dvořák and Kupec show that all interesting cases of planar CSP can be reduced to edge CSP with \Delta-matroid constraints.
- If there is a polynomial algorithm for edge CSP with even \Delta-matroid constraints, we have a dichotomy for planar CSP.
Planar CSPs

- CSP(\{0, 1\}, \Gamma) with incidence graphs of instances planar.
- Dvořák and Kupec show that all interesting cases of planar CSP can be reduced to edge CSP with \(\Delta\)-matroid constraints.
- If there is a polynomial algorithm for edge CSP with even \(\Delta\)-matroid constraints, we have a dichotomy for planar CSP.

CSP($\{0, 1\}, \Gamma$) with incidence graphs of instances planar.

- Dvořák and Kupec show that all interesting cases of planar CSP can be reduced to edge CSP with $\Delta$-matroid constraints.
- If there is a polynomial algorithm for edge CSP with even $\Delta$-matroid constraints, we have a dichotomy for planar CSP.

CSP($\{0, 1\}, \Gamma$) with incidence graphs of instances planar.

Constraints – faces of a planar graph, variables – vertices.

Dvořák and Kupec show that all interesting cases of planar CSP can be reduced to edge CSP with $\Delta$-matroid constraints.

If there is a polynomial algorithm for edge CSP with even $\Delta$-matroid constraints, we have a dichotomy for planar CSP.
Planar CSPs

- CSP({0, 1}, Γ) with incidence graphs of instances planar.
- Dvořák and Kupec show that all interesting cases of planar CSP can be reduced to edge CSP with Δ-matroid constraints.
- If there is a polynomial algorithm for edge CSP with even Δ-matroid constraints, we have a dichotomy for planar CSP.

CSP(\{0, 1\}, \Gamma) with incidence graphs of instances planar.

Constraints – faces of a planar graph, variables – vertices.

Dvořák and Kupec show that all interesting cases of planar CSP can be reduced to edge CSP with \(\Delta\)-matroid constraints.

If there is a polynomial algorithm for edge CSP with even \(\Delta\)-matroid constraints, we have a dichotomy for planar CSP.
Our strategy

- We generalize Edmond’s blossom algorithm for perfect matchings.
- Edge labeling $f$ assigns 0 or 1 to each half-edge: Pair $\{v, C\}$ where $v$ lies in constraint $C$ so that all constraints are satisfied.
- Variable is consistent in $f$ if both half edges corresponding to $v$ have the same labels.
- Edge labeling with all variables consistent = a solution of the instance.
- We want to augment a given labeling $f$: Find $g$ labeling with fewer inconsistencies.
- If $f$ is an edge labeling that can be improved, there is an augmenting $f$-walk $p$ from one inconsistent variable to another.
Our strategy

- We generalize Edmond’s blossom algorithm for perfect matchings.
- Edge labeling $f$ assigns 0 or 1 to each half-edge: Pair $\{v, C\}$ where $v$ lies in constraint $C$ so that all constraints are satisfied.
- Variable is consistent in $f$ if both half edges corresponding to $v$ have the same labels.
- Edge labeling with all variables consistent = a solution of the instance.
- We want to augment a given labeling $f$: Find $g$ labeling with fewer inconsistencies.
- If $f$ is an edge labeling that can be improved, there is an augmenting $f$-walk $p$ from one inconsistent variable to another.
Our strategy

- We generalize Edmond’s blossom algorithm for perfect matchings.
- Edge labeling $f$ assigns 0 or 1 to each half-edge: Pair $\{v, C\}$ where $v$ lies in constraint $C$ so that all constraints are satisfied.
- Variable is consistent in $f$ if both half edges corresponding to $v$ have the same labels.
- Edge labeling with all variables consistent = a solution of the instance.
- We want to augment a given labeling $f$: Find $g$ labeling with fewer inconsistencies.
- If $f$ is an edge labeling that can be improved, there is an augmenting $f$-walk $p$ from one inconsistent variable to another.
Our strategy

- We generalize Edmond’s blossom algorithm for perfect matchings.
- Edge labeling $f$ assigns 0 or 1 to each half-edge: Pair $\{v, C\}$ where $v$ lies in constraint $C$ so that all constraints are satisfied.
- Variable is consistent in $f$ if both half edges corresponding to $v$ have the same labels.
- Edge labeling with all variables consistent = a solution of the instance.
- We want to augment a given labeling $f$: Find $g$ labeling with fewer inconsistencies.
- If $f$ is an edge labeling that can be improved, there is an augmenting $f$-walk $p$ from one inconsistent variable to another.
Our strategy

- We generalize Edmond’s blossom algorithm for perfect matchings.
- Edge labeling $f$ assigns 0 or 1 to each half-edge: Pair $\{v, C\}$ where $v$ lies in constraint $C$ so that all constraints are satisfied.
- Variable is consistent in $f$ if both half edges corresponding to $v$ have the same labels.
- Edge labeling with all variables consistent = a solution of the instance.
- We want to augment a given labeling $f$: Find $g$ labeling with fewer inconsistencies.
- If $f$ is an edge labeling that can be improved, there is an augmenting $f$-walk $p$ from one inconsistent variable to another.
Our strategy

- We generalize Edmond’s blossom algorithm for perfect matchings.
- Edge labeling $f$ assigns 0 or 1 to each half-edge: Pair $\{v, C\}$ where $v$ lies in constraint $C$ so that all constraints are satisfied.
- Variable is consistent in $f$ if both half edges corresponding to $v$ have the same labels.
- Edge labeling with all variables consistent = a solution of the instance.
- We want to augment a given labeling $f$: Find $g$ labeling with fewer inconsistencies.
- If $f$ is an edge labeling that can be improved, there is an augmenting $f$-walk $p$ from one inconsistent variable to another.
Our strategy

- We generalize Edmond’s blossom algorithm for perfect matchings.
- Edge labeling $f$ assigns 0 or 1 to each half-edge: Pair $\{v, C\}$ where $v$ lies in constraint $C$ so that all constraints are satisfied.
- Variable is consistent in $f$ if both half edges corresponding to $v$ have the same labels.
- Edge labeling with all variables consistent $\Rightarrow$ a solution of the instance.
- We want to augment a given labeling $f$: Find $g$ labeling with fewer inconsistencies.
- If $f$ is an edge labeling that can be improved, there is an augmenting $f$-walk $p$ from one inconsistent variable to another.
Example
Sketch of the algorithm

- Take $f$, search from all inconsistent variables, building a forest of visited variables and constraints.
- If we can find $f$-walks $u \ldots Cv$ and $u' \ldots Dv$ for $u, u'$ inconsistent, we can augment and make $u, u'$ consistent.
- If we find $f$-walks $u \ldots Cv$ and $u \ldots Dv$, we have found a blossom. This we contract and re-run the algorithm on a smaller instance.
- If we don’t find any of the above, then $f$ can not be augmented.
Sketch of the algorithm

- Take $f$, search from all inconsistent variables, building a forest of visited variables and constraints.
  - If we can find $f$-walks $u \ldots Cv$ and $u' \ldots Dv$ for $u, u'$ inconsistent, we can augment and make $u, u'$ consistent.
  - If we find $f$-walks $u \ldots Cv$ and $u \ldots Dv$, we have found a blossom. This we contract and re-run the algorithm on a smaller instance.
  - If we don’t find any of the above, then $f$ can not be augmented.
Sketch of the algorithm

- Take $f$, search from all inconsistent variables, building a forest of visited variables and constraints.
- If we can find $f$-walks $u \ldots Cv$ and $u' \ldots Dv$ for $u, u'$ inconsistent, we can augment and make $u, u'$ consistent.
- If we find $f$-walks $u \ldots Cv$ and $u \ldots Dv$, we have found a blossom. This we contract and re-run the algorithm on a smaller instance.
- If we don’t find any of the above, then $f$ can not be augmented.
Sketch of the algorithm

- Take $f$, search from all inconsistent variables, building a forest of visited variables and constraints.
- If we can find $f$-walks $u \ldots Cv$ and $u' \ldots Dv$ for $u, u'$ inconsistent, we can augment and make $u, u'$ consistent.
- If we find $f$-walks $u \ldots Cv$ and $u \ldots Dv$, we have found a blossom. This we contract and re-run the algorithm on a smaller instance.
- If we don’t find any of the above, then $f$ can not be augmented.
Sketch of the algorithm

- Take $f$, search from all inconsistent variables, building a forest of visited variables and constraints.
- If we can find $f$-walks $u \ldots Cv$ and $u' \ldots Dv$ for $u, u'$ inconsistent, we can augment and make $u, u'$ consistent.
- If we find $f$-walks $u \ldots Cv$ and $u \ldots Dv$, we have found a blossom. This we contract and re-run the algorithm on a smaller instance.
- If we don’t find any of the above, then $f$ can not be augmented.
Consequences and future work

- We have finished the classification started by Dvořák and Kupec.
- To get dichotomy for edge CSPs, all that is needed is to generalize our argument from even $\Delta$-matroids to all $\Delta$-matroids.
- We can go beyond even $\Delta$-matroids and cover many previously known polynomial classes, but there still remains a large gap.
- We are now beginning to look at valued version of edge CSP for even $\Delta$-matroids.
- Generalization to value sets larger than 2 is going to be hard.
Consequences and future work

- We have finished the classification started by Dvořák and Kupec.
- To get dichotomy for edge CSPs, all that is needed is to generalize our argument from even $\Delta$-matroids to all $\Delta$-matroids.
- We can go beyond even $\Delta$-matroids and cover many previously known polynomial classes, but there still remains a large gap.
- We are now beginning to look at valued version of edge CSP for even $\Delta$-matroids.
- Generalization to value sets larger than 2 is going to be hard.
We have finished the classification started by Dvořák and Kupec.

To get dichotomy for edge CSPs, all that is needed is to generalize our argument from even $\Delta$-matroids to all $\Delta$-matroids.

We can go beyond even $\Delta$-matroids and cover many previously known polynomial classes, but there still remains a large gap.

We are now beginning to look at valued version of edge CSP for even $\Delta$-matroids.

Generalization to value sets larger than 2 is going to be hard.
We have finished the classification started by Dvořák and Kupec.

To get dichotomy for edge CSPs, all that is needed is to generalize our argument from even $\Delta$-matroids to all $\Delta$-matroids.

We can go beyond even $\Delta$-matroids and cover many previously known polynomial classes, but there still remains a large gap.

We are now beginning to look at valued version of edge CSP for even $\Delta$-matroids.

Generalization to value sets larger than 2 is going to be hard.
Consequences and future work

- We have finished the classification started by Dvořák and Kupec.
- To get dichotomy for edge CSPs, all that is needed is to generalize our argument from even $\Delta$-matroids to all $\Delta$-matroids.
- We can go beyond even $\Delta$-matroids and cover many previously known polynomial classes, but there still remains a large gap.
- We are now begining to look at valued version of edge CSP for even $\Delta$-matroids.
- Generalization to value sets larger than 2 is going to be hard.
We have finished the classification started by Dvořák and Kupec.

To get dichotomy for edge CSPs, all that is needed is to generalize our argument from even $\Delta$-matroids to all $\Delta$-matroids.

We can go beyond even $\Delta$-matroids and cover many previously known polynomial classes, but there still remains a large gap.

We are now beginning to look at valued version of edge CSP for even $\Delta$-matroids.

Generalization to value sets larger than 2 is going to be hard.
Thank you for your attention.