Introduction into Mathematics of Constraint Satisfaction

Andrei Krokhin
Durham University

Part III

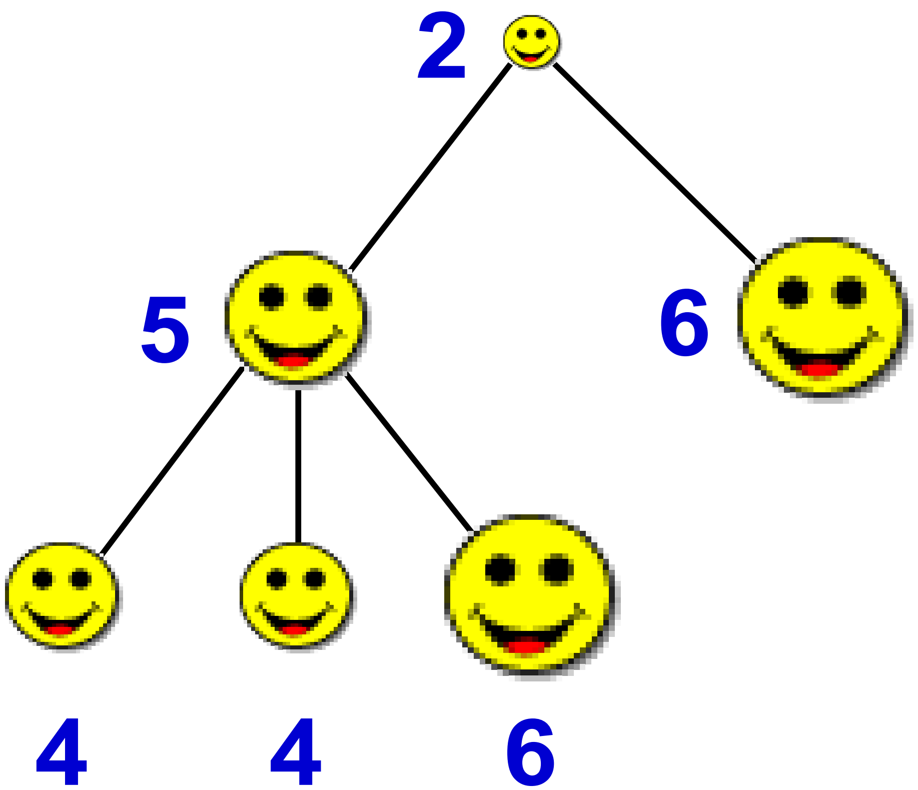
Outline of the Course

1. The CSP and its forms
   - Examples of CSPs
2. Complexity issues and computational questions
   - What questions do we ask about CSPs?
3. Mathematical techniques
   - Treewidth and its generalisations
   - Based on talk by Daniel Marx in Dagstuhl in 2009

The Party Problem

- Problem: Invite some colleagues for a party
- Constraint: Everyone should be having fun
  - Don't invite anyone and their direct boss at the same time
- Goal: Maximize the total fun factor of the invited people

The Party Problem

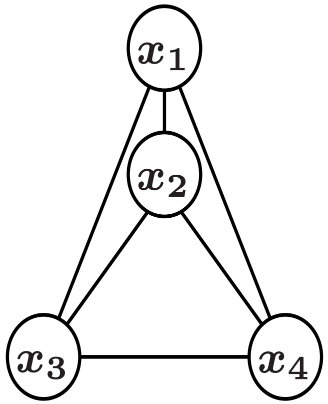


Solving the Party Problem

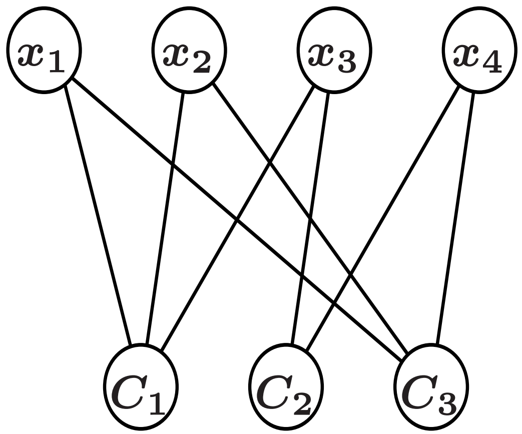
- Use dynamic programming
  - T_v – subtree rooted at v
  - $A[v]$ – weight of max independent set in T_v
  - $B[v]$ – same, but excluding v
- Assume $v_1, \ldots, v_k$ are children of v. Have recurrence relations
  - $B[v] = \sum_{i=1}^{k} A[v_i]$
  - $A[v] = \max\left(B[v], w(v) + \sum_{i=1}^{k} B[v_i]\right)$
- Compute $A[v]$ and $B[v]$ in the bottom-up order
- If r is the root, $A[r]$ is the optimum value

Graphs and Hypergraphs of an Instance

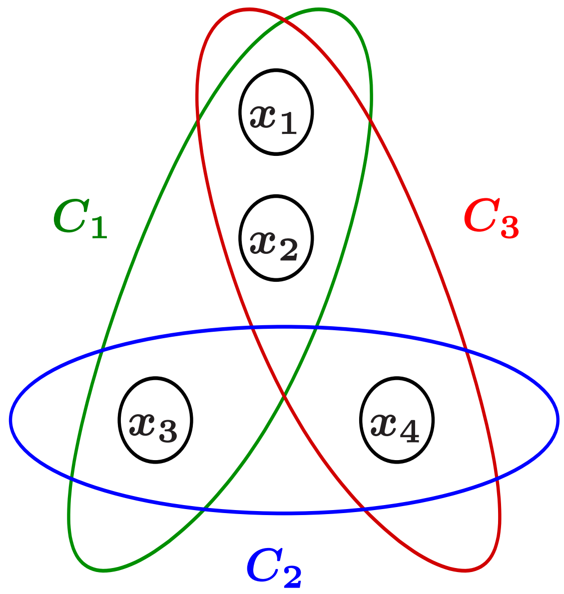
$$I = C_1(x_2, x_1, x_3) \wedge C_2(x_4, x_3) \wedge C_3(x_1, x_4, x_2)$$



Primal graph



Incidence graph



Hypergraph

Tree-Shaped CSP

- Binary CSP is tractable on instances whose primal graph is a tree is easy.
- Proof 1 — Dynamic programming algorithm
  - For each $v \in V$, $d \in D$, let $x[v, d] = \mathit{true}$ if the subtree rooted at v has a solution
  - Leaves are trivial
  - Compute $x[v, d]$ in bottom-up order

Tree-Shaped CSP

- Binary CSP is tractable on instances whose primal graph is a tree is easy.
- Proof 2 — Arc-consistency algorithm
  - For each $v \in V$, $d \in D$,
  - If some constraint $R(v, u)$ does not support $(d, *)$
  - Delete tuples $(d, *)$ from each constraint on $(v, *)$
  - Repeat
  - When stabilized, either all constraints are empty, or there is a solution.
- How to generalise these two ideas?

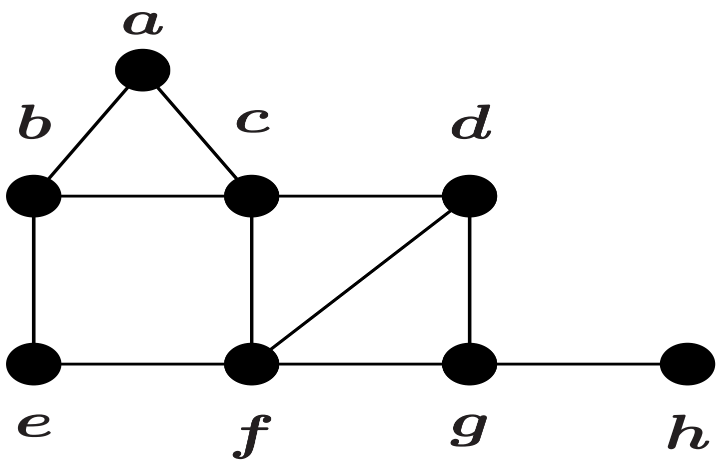
Treewidth

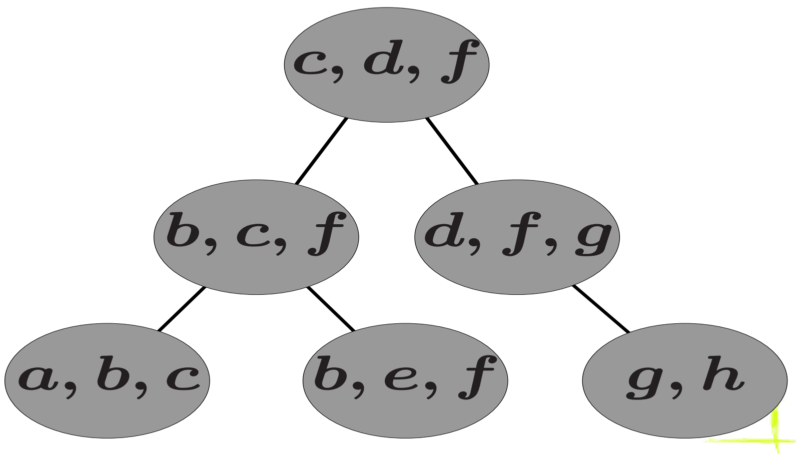
- Treewidth measures how tree-like a graph is
- introduced by Robertson and Seymour

A (hyper)graph G has treewidth $\leq k$ if there is a tree T, called tree decomposition, such that

- the nodes of T are subsets of V of size $\leq k+1$
- nodes containing any given element form a subtree
- for any (hyper)edge e in G , there is a node in T containing e.

Example





CSPs of Bounded Treewidth

- For each fixed k, binary CSP is solvable in polynomial time on instances of treewidth $\leq k$.
- for each fixed k, CSP is solvable in polynomial time on instances whose primal graph has treewidth $\leq k$.
- Two algorithms
  - k-consistency algorithm
  - using tree decomposition

k-Consistency

For a set $S \subseteq V$, a partial solution on S is a function $S \to D$ that satisfies all constraints with scopes in S.

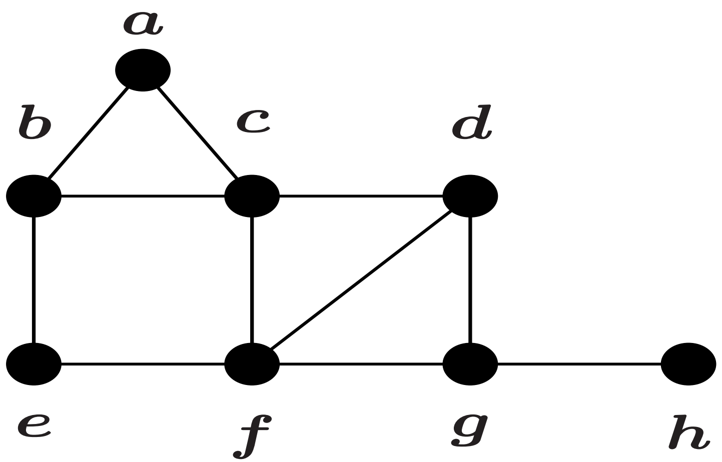
An instance is k-consistent if, for all $X \subseteq Y \subseteq V$ with $|Y| \leq k+1$, any partial solution on X extends to a partial solution on Y.

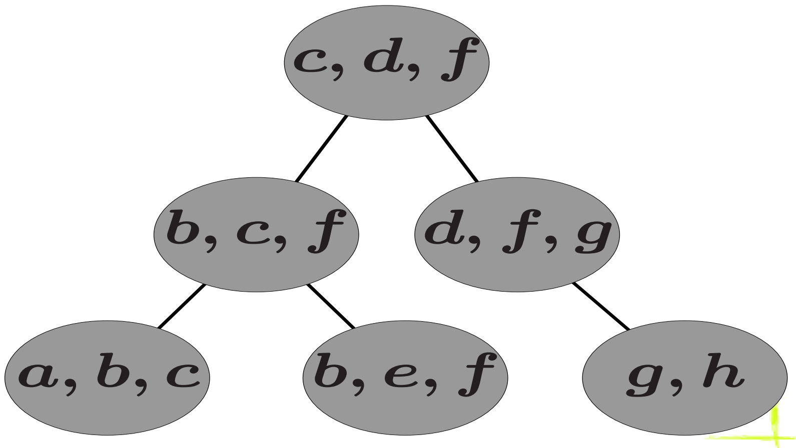
Can transform any instance into equivalent k-consistent:

- For each S with $|S| \leq k+1$ generate the list L_S of all partial solutions on S.
- Repeatedly remove violations of k-consistency.

Polynomial-time for each fixed k.

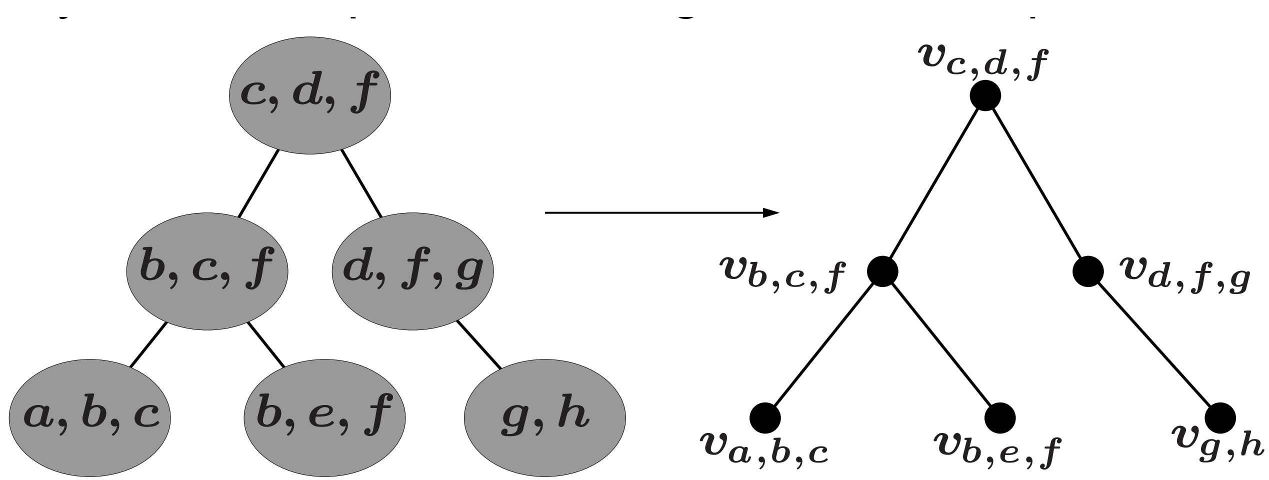
2-Consistency by Example





Using Tree Decomposition

Take an instance I and a tree decomposition T of the primal graph of I (of width $\leq k$). Form a binary CSP instance.



Transformation is polynomial-time for fixed k. Solve the tree instance.

Finding Tree Decompositions

- It is **NP**-hard to compute treewidth of graph, but tractable to decide for each fixed k.
- For each fixed k, there is a linear time algorithm that computes a tree decomposition of width k (if exists).
- There is a polynomial-time algorithm that finds a tree decomposition of width $O(k\sqrt{k})$ if there is one of width k.

Gaifman Graph of a Structure

- The Gaifman graph $G(\mathcal{A})$ of a structure $\mathcal{A}$ has
  - set of nodes A
  - edges (a, a') where $a \neq a'$ appear in the same tuple in some relation
- Treewidth of $\mathcal{A}$ = treewidth of $G(\mathcal{A})$
- Each tuple in $\mathcal{A}$ gives a clique in $G(\mathcal{A})$, hence must be contained in one "bag" of any tree decomposition.

Power of k-Consistency

A core of a structure $\mathcal{A}$ is its minimal substructure $\mathcal{A}'$ with $\mathcal{A} \to \mathcal{A}'$.

Fact. All cores of a structure are isomorphic.

Example: The core of any bipartite graph is K_2.

Theorem 1 (Dalmau,Kolaitis,Vardi'02)
If the core of $\mathcal{A}$ has treewidth $\leq k$ then the k-consistency algorithm correctly solves $\mathrm{CSP}(\{\mathcal{A}\}, -)$.

Theorem 2 (Atserias,Bulatov,Dalmau'07)
If the k-consistency algorithm correctly solves $\mathrm{CSP}(\{\mathcal{A}\}, -)$ *then the core of $\mathcal{A}$ has treewidth $\leq k$.*

Bounded Arity Case

Theorem 3 (Grohe'07) *Let $\mathfrak{A}$ be a recursively enumerable class of structures of bounded arity. Assuming* $\mathbf{FPT} \neq \mathbf{W}[\mathbf{1}]$, *TFAE*

- $\mathrm{CSP}(\mathfrak{A}, -)$ *is tractable.*
- *the cores of structures in $\mathcal{A}$ have bounded treewidth.*

Theorem 4 (Dalmau,Jonsson'04) *Let $\mathfrak{A}$ be a recursively enumerable class of structures of bounded arity. Assuming* $\mathbf{FPT} \neq \mathbf{W}[\mathbf{1}]$, *TFAE*

- *counting* $\mathrm{CSP}(\mathfrak{A}, -)$ *is tractable.*
- *the structures in $\mathcal{A}$ have bounded treewidth.*

Unbounded Arities

- For a class $\mathfrak{H}$ of hypergraphs, let CSP$(\mathfrak{H}, -)$ denote all CSP instances with hypergraphs in $\mathfrak{H}$.
- Know: bounded treewidth $\Rightarrow$ tractable
- Any instance with a single constraint of arity n has a hypergraph of large treewidth, but is trivial to solve.
- So there are more tractable cases out there.

Another Example

- Let $\mathfrak{H}_d$ be the class of hypergraphs in which all vertices can be covered by d hyperedges.
- For fixed d, need to look only at assignments satisfying those d constraints.
- So can solve corresponding CSP in poly-time.
- Idea: enough to cover bags in tree decomposition with a fixed number of constraints.

Hypertree Width

- Invented by Gottlob, looking for tractable database query languages.
- The generalized hypertree width of a tree decomposition of a hypergraph is the min number of hyperedges needed to cover each bag.
- The generalized hypertree width $ghw(H)$ of a hypergraph H is the min number over all decompositions.
- The original hypertree width had an additional technical condition, but $ghw \leq hw \leq 3 \cdot ghw$.

Using Hypertree Width

- if a bag B is covered by k constraints, can compute the partial solutions on B in time $|I|^k$
- this gives a poly-time algorithm for fixed k if can find good tree decompositions.
- it's **NP**-hard to decide if $ghw(H) \leq 3$.
- For each fixed k, there is a poly-time algorithm that either finds a tree decomposition with $ghw(H) \leq 3k$ or correctly concludes that $ghw(H) > k$.
- So, if $\mathfrak{H}$ has bounded hypertree width, then CSP$(\mathfrak{H}, -)$ is tractable.

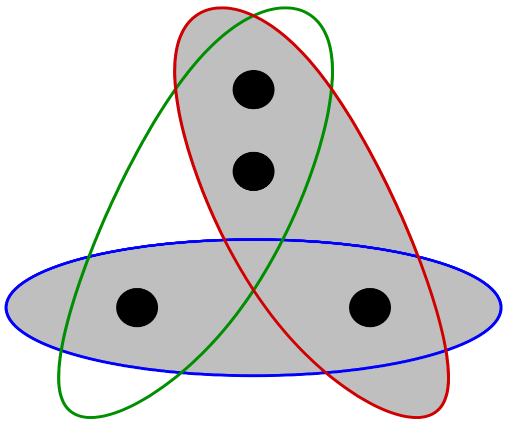
Beyond Hypertree Width

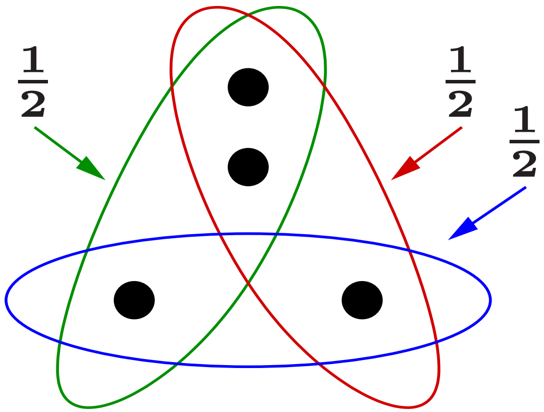
- Is there a more general property for tractability?
- Key feature: small number of solutions in a bag
- When can we ensure this property?

Fractional Edge Cover

Edge cover of a hypergraph H is a family of hyperedges that covers all nodes.
$\rho(H)$ = size of the smallest cover.

Fractional edge cover is a weight assignment to hyperegdes so that each vertex is covered by total weight ≥ 1.
$\rho^*(H)$ = smallest total weight.





Using Fractional Edge Covers

- If instance I has hypergraph H with $\rho^*(H) = w$ then I has at most $|I|^w$ solutions, and they can be efficiently enumerated [Grohe,Marx'07]
- Hence CSP$(\mathfrak{H}, -)$ is tractable if $\mathfrak{H}$ has bounded fractional edge cover number.

Theorem 5 (Atserias,Grohe,Marx'09)

For a class $\mathfrak{H}$ of hypergraphs, TFAE

- *Each* CSP$(\mathfrak{H}, -)$ *instance has poly many solutions.*
- *Solutions can be enumerated in poly time.*
- *$\mathfrak{H}$ has bounded fractional edge cover number.*

Fractional Hypertree Width

- Can take this further and require that each bag in a decomposition has bounded fractional edge cover number.
- Get the notion of bounded fractional hypertree width.
- For each w, there is a poly-time algorithm that either finds a tree decomposition with $fhw(H) = O(w^3)$ or correctly concludes that $fhw(H) > w$. [Marx'09]
- So, if $\mathfrak{H}$ has bounded fractional hypertree width, then CSP($\mathfrak{H}, -$) is tractable.

FPT and ETH

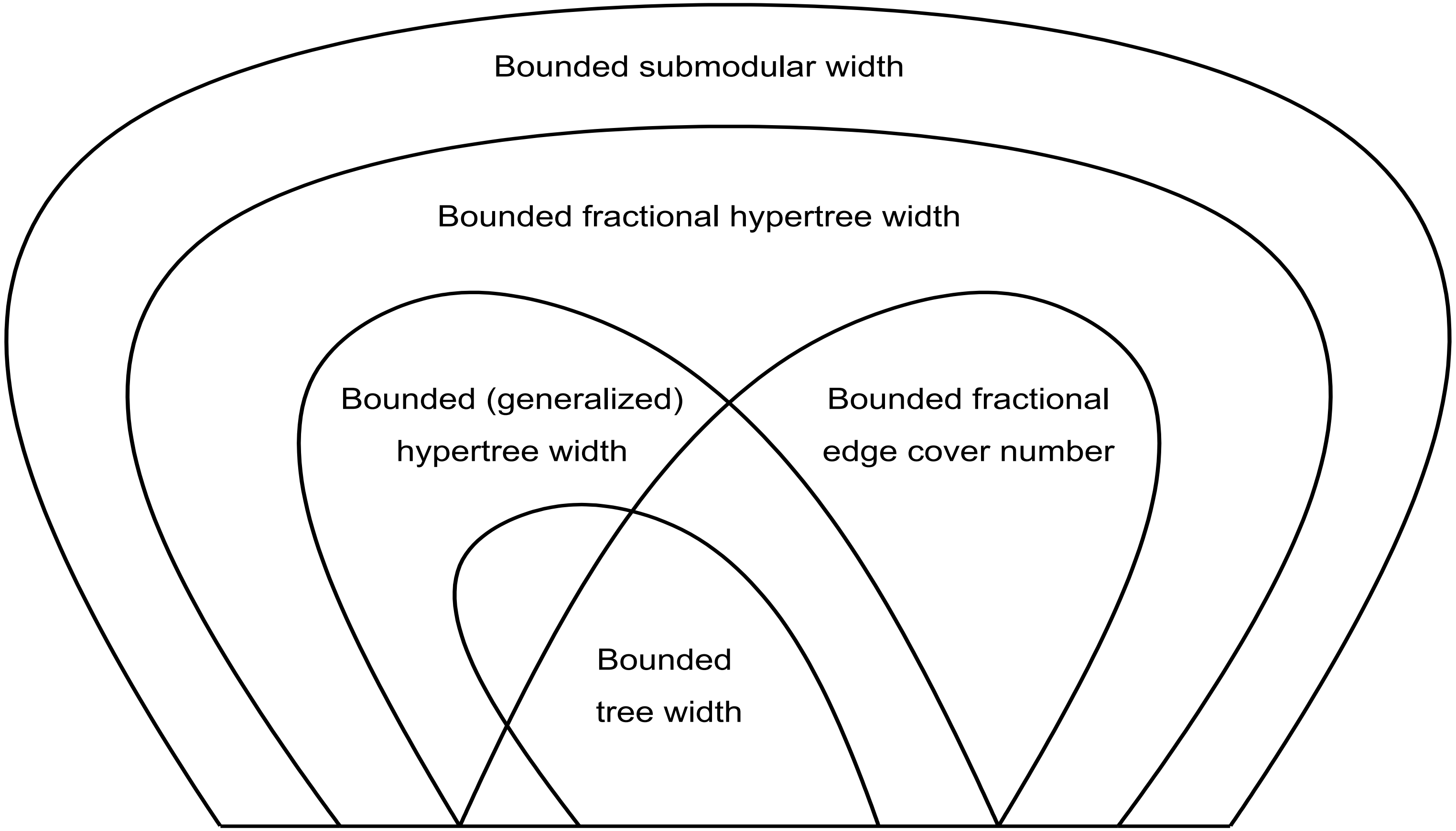
- Say that $\mathrm{CSP}(\mathfrak{H}, -)$ is FPT if can be solved in time $g(k) \cdot |I|^c$ where k is the number of variables.
- ETH (Exponential Time Hypothesis) — there is no $2^{o(n)}$ algorithm for n-variable 3-SAT.

Structural Restrictions in FPT

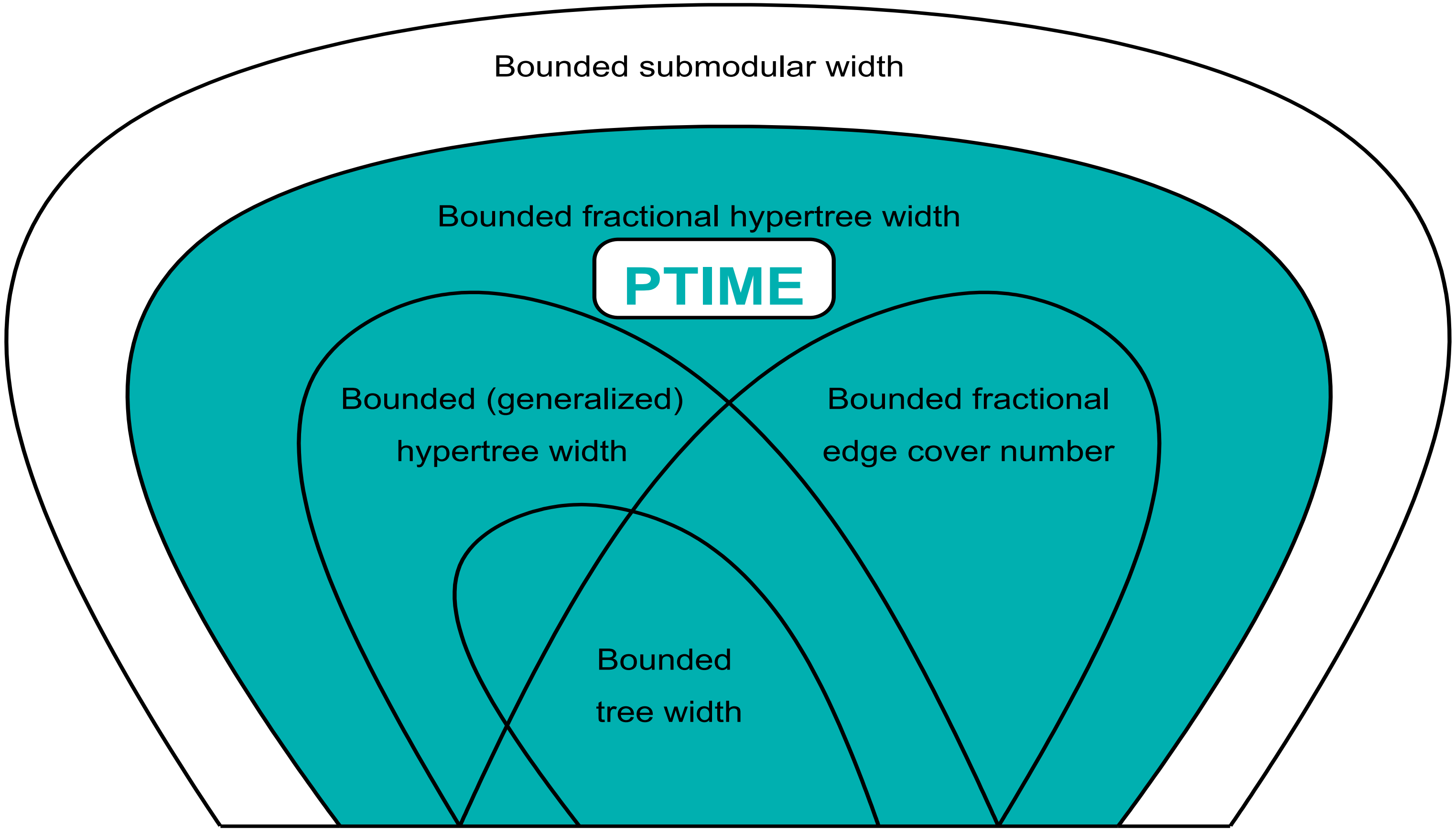
- Submodular width, invented by Marx, always $\leq fhw$.
- "optimises" over several different types of decompositions.

Theorem 6 (Marx'10) *Assuming ETH, if $\mathfrak{H}$ is an recursively enumerable class of hypergraphs, then* CSP$(\mathfrak{H}, -)$ *is FPT iff $\mathfrak{H}$ has bounded submodular width.*

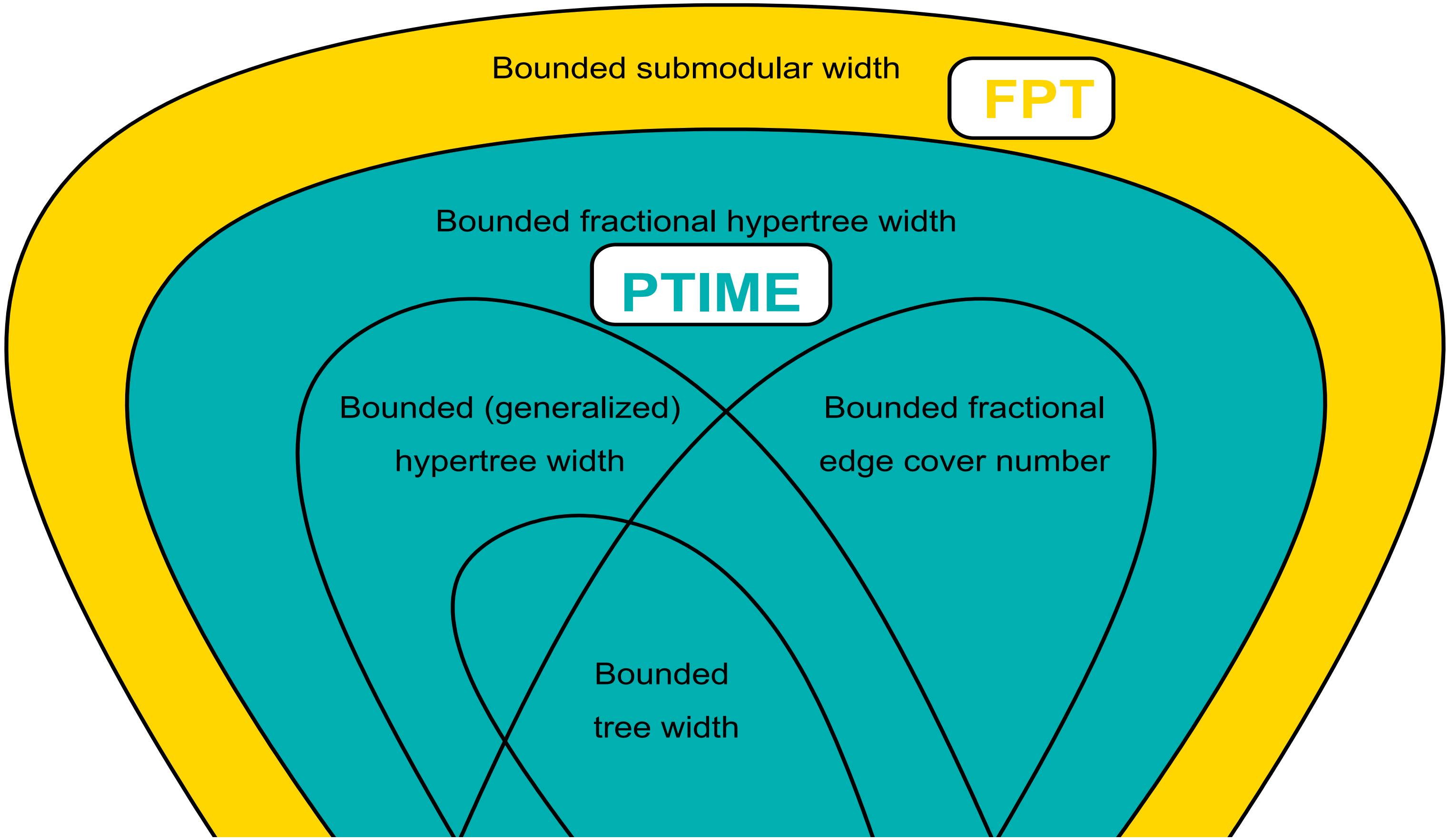
Structural Restrictions: Overview



Structural Restrictions: Overview



Structural Restrictions: Overview



Structural Restrictions: Overview