Iterated Type Partitions

Gennaro Cordasco, University of Campania “L.Vanvitelli”, Italy
Luisa Gargano, University of Salerno, Italy
Adele Rescigno, University of Salerno, Italy

Fixed Parameter Tractable (FPT)

A parameterized problem with input size n and parameter t is called *fixed parameter tractable (FPT)* if it can be solved in time $f(t) \cdot poly(n)$, where f is a computable function only depending on t.

Parameters

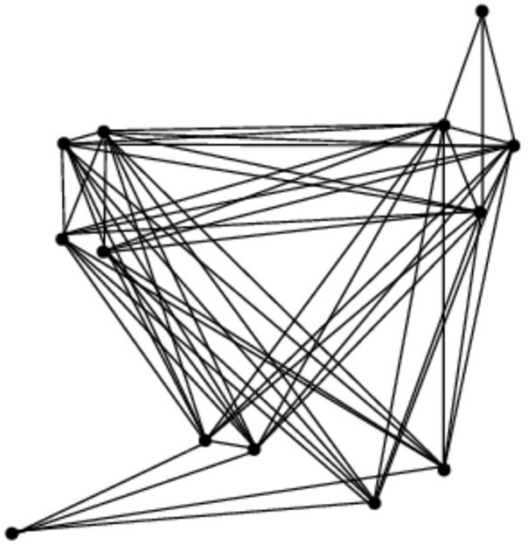
An important quality of a parameter is that it is easy to compute.

- Clique-width: the parameterized complexity of computing it is still an open problem
- Treewidth, rankwidth, and vertex cover: computing them are all NP-hard problems but they are computable in FPT time when their respective parameters are bounded
- Modular-width and neighborhood diversity: they are computable in polynomial time

Neighborhood diversity

$G = (V, E)$

$u, v \in V$ have the same *type* iff $N(v) \setminus \{u\} = N(u) \setminus \{v\}$

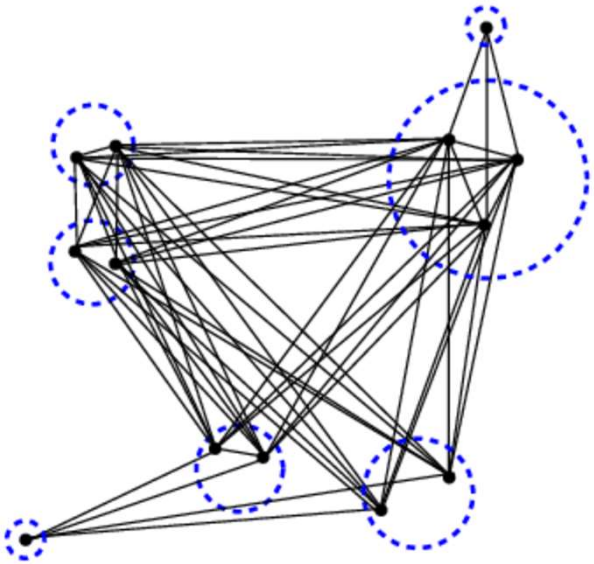


Neighborhood diversity

$G = (V, E)$

$u, v \in V$ have the same *type* iff $N(v) \setminus \{u\} = N(u) \setminus \{v\}$

A *type partition* of G is a partition $V_1, V_2, \ldots, V_t$ of the node set V such that all the nodes in V_i have the same type, for $i = 1,2, \ldots, t$



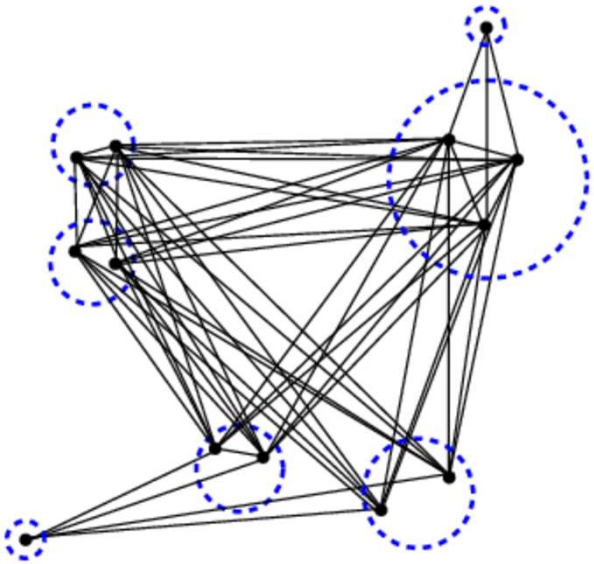
Neighborhood diversity

$G = (V, E)$

$u, v \in V$ have the same *type* iff $N(v) \setminus \{u\} = N(u) \setminus \{v\}$

A *type partition* of G is a partition $V_1, V_2, \ldots, V_t$ of the node set V such that all the nodes in V_i have the same type, for $i = 1,2, \ldots, t$

- Note that by definition, each V_i induces either a *clique* or an *independent set* in G



$nd(G) = 7$

The *neighborhood diversity* of $G, nd(G),$ introduced by Lampis, is the minimum number t of sets in a type partition $V_1, V_2, \ldots, V_t$ of G

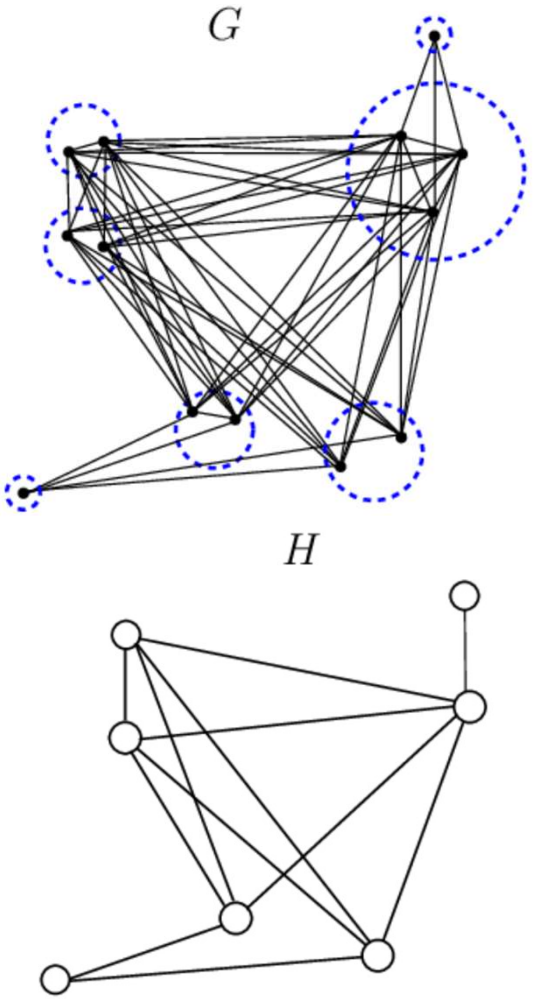
Iterated type partition

$G = (V, E)$

A type partition $V_1, V_2, \ldots, V_t$, of G

The *type graph* H of G is defined as

- $V(H) = \{1, 2, \ldots, t\}$
- $E(H) = \{(x, y) \mid x \neq y \text{ and } for\ each\ u \in V_x, v \in V_y$
  $it\ holds\ that\ (u, v) \in E(G)\}$

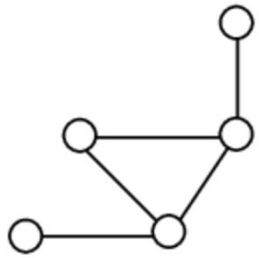


Iterated type partition

$G = (V, E)$

G is a *base graph* if it matches its type graph

- that is, the type partition of G consists of singletons, each representing a node in $V(G)$, and $nd(G) = |V(G)|$



Iterated type partition

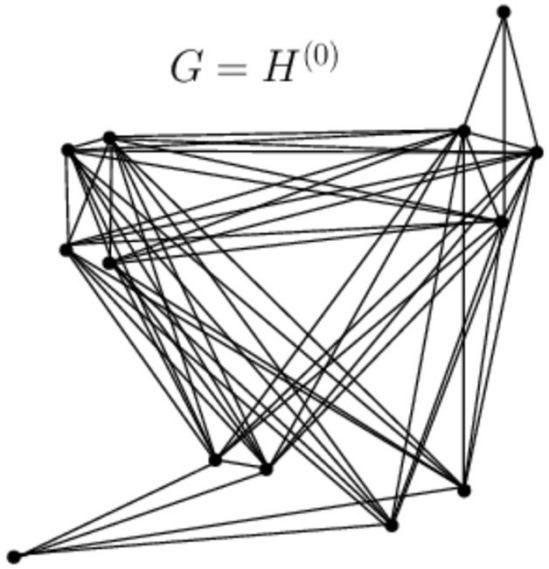
$H^{(0)} = G$

The *type graph sequence* of G is the graph sequence $H^{(0)}$ $H^{(1)}$ ... $H^{(d)}$

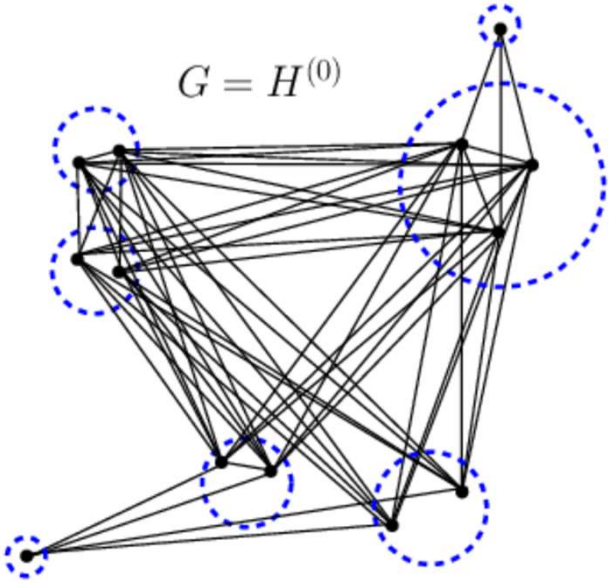
- $H^{(i)}$ denotes the type graph of $H^{(i-1)}$, for $i \geq 1$
- d is the smallest integer such that $H^{(d)}$ is a base graph

The iterated type partition of G, denoted by $itp(G)$, is the number of nodes of $H^{(d)}$

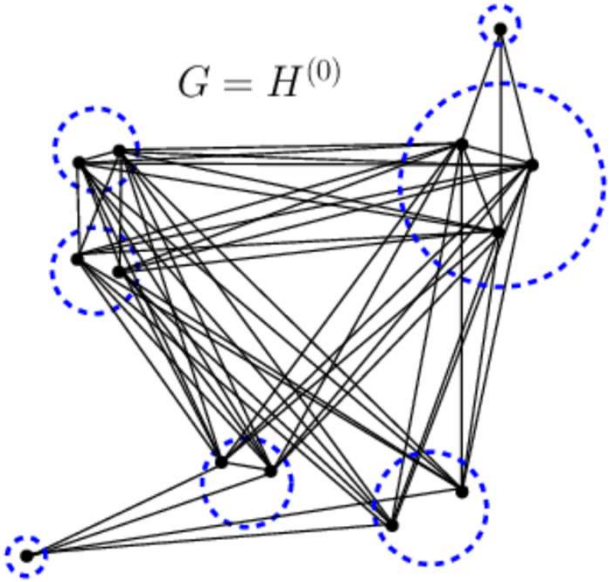
Iterated type partition

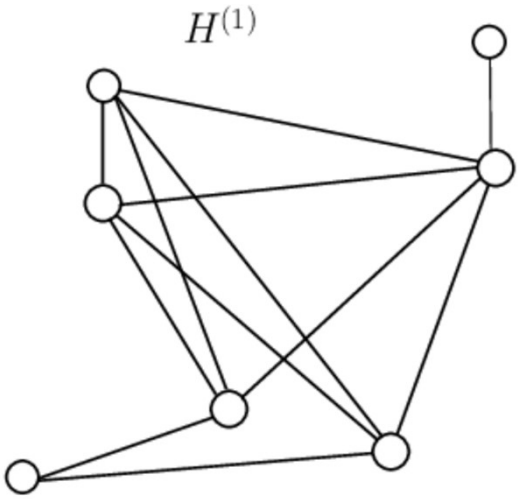


Iterated type partition

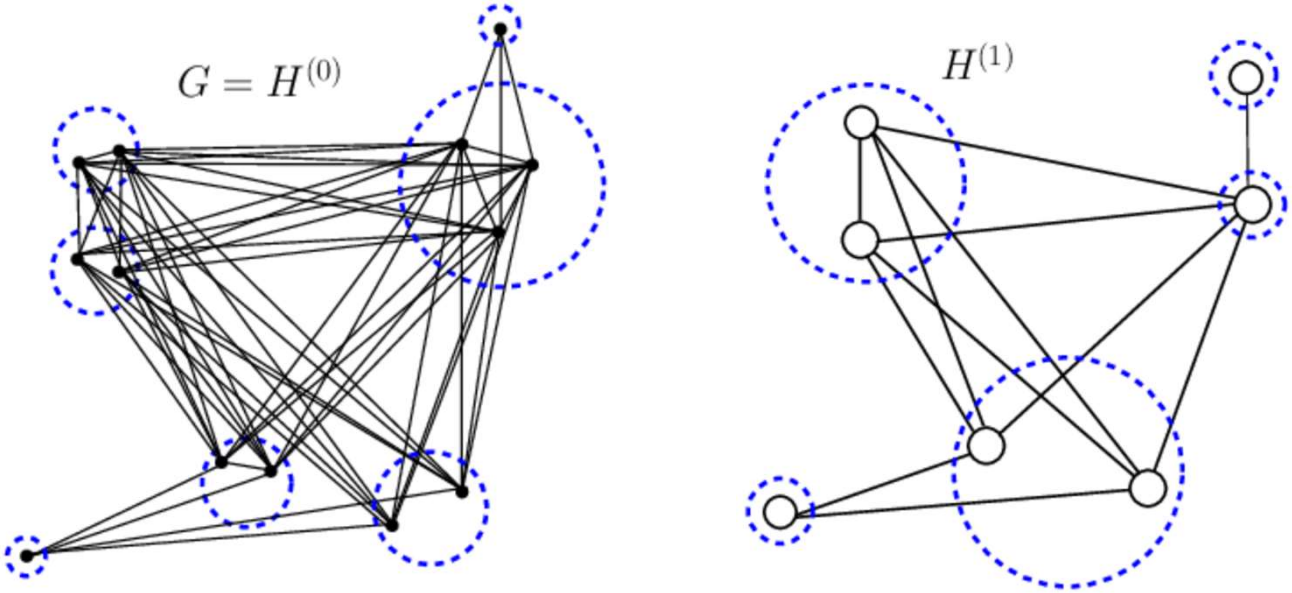


Iterated type partition

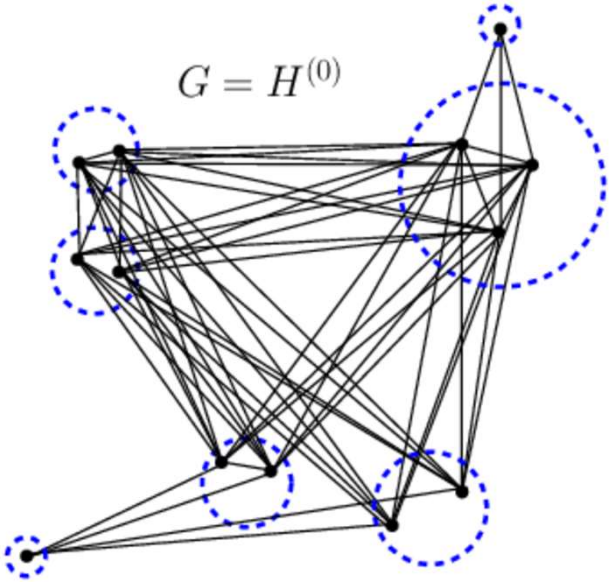


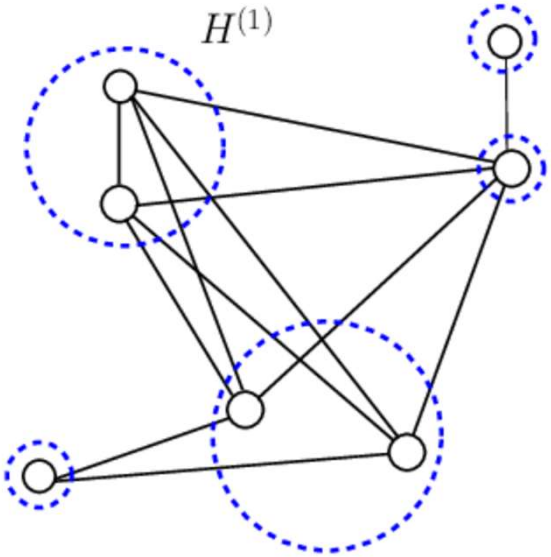


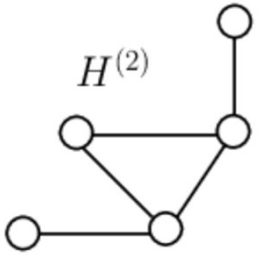
Iterated type partition



Iterated type partition







$itp(G) = 5$

Iterated type partition and the other parameteres

$$itp(G) \leq nd(G)$$

$itp(G)$ can be much smaller than $nd(G)$

Iterated type partition and the other parameteres

$$mw(G) \leq itp(G)$$

Iterated type partition is a “special case” of modular-width in which the modules can only be independent sets or cliques

Iterated type partition and the other parameteres

$$itp(G) \leq 2^{vc(G)} + vc(G)$$

$$cw(G) \leq itp(G) + 1$$

$itp(G)$ is incomparable to $tw(G)$

Iterated type partition
our results

Theorem 1. *There exists a polynomial time algorithm which, for any input graph* G *computes the type graphs sequence of* G *and, consequently, finds the value* $itp(G)$.

Iterated type partition
our results

	EQC	DS, VC	Coloring
cw	W[1]-hard	FPT	W[1]-hard
mw	W[1]-hard	FPT	FPT
itp	W[1]-hard	FPT $O(2^{itp(G)} + poly(n))$	FPT $O(itp(G)^{2,5\, itp(G)+o(itp(G))} \log n + poly(n))$
nd	FPT	FPT	FPT
vc	FPT	FPT	FPT

Equitable Coloring: W[1]-hardness

Equitable Coloring

Instance: A graph G = (V,E) and an integer k.

Question: Is it possible to color the nodes of G with exactly k colors in such a way that nodes connected by an edge receive different colors and each color class has either size ⌊|V |/k⌋ or ⌈|V |/k⌉?

We present a reduction from the following Bin Packing problem

Bin Packing

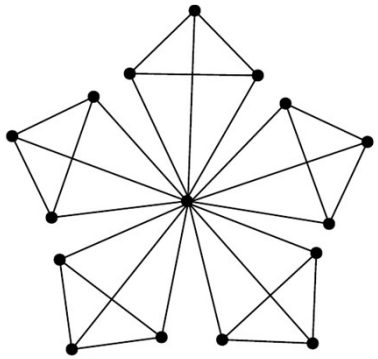
Instance: A collection of ℓ items having sizes $a_1, a_2, \ldots, a_\ell$, a number k of bins, and a bin capacity B.

Question: ∃ a k-partition $P_1, \ldots, P_k$ of $A = \{a_1, a_2, \ldots, a_\ell\}$ such that $\sum_{a_j \in P_i} a_j$ = B, $\forall\, i = 1,2, \ldots, k$

Equitable Coloring: W[1]-hardness

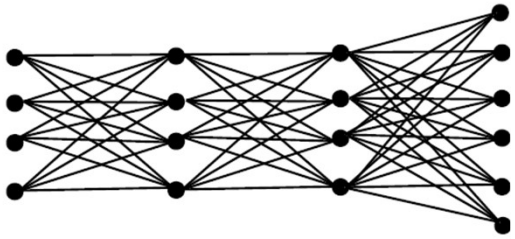
Gadgets:

$F_{a,k}$-*flower* is a graph obtained by joining $a+1$ cliques of size k to a central node



$F_{4,3}$–*flower*

(k,ℓ,B)–*chain* is a sequence of $k+1$ independent sets $S_1,\ldots,S_k,S_{k+1}$ with $|S_i|=B$, for $i=1,..,k$, and $|S_{k+1}|=\ell+1$ where between each pair of consecutive sets in the sequence S_i,S_{i+1} there is a complete bipartite graph



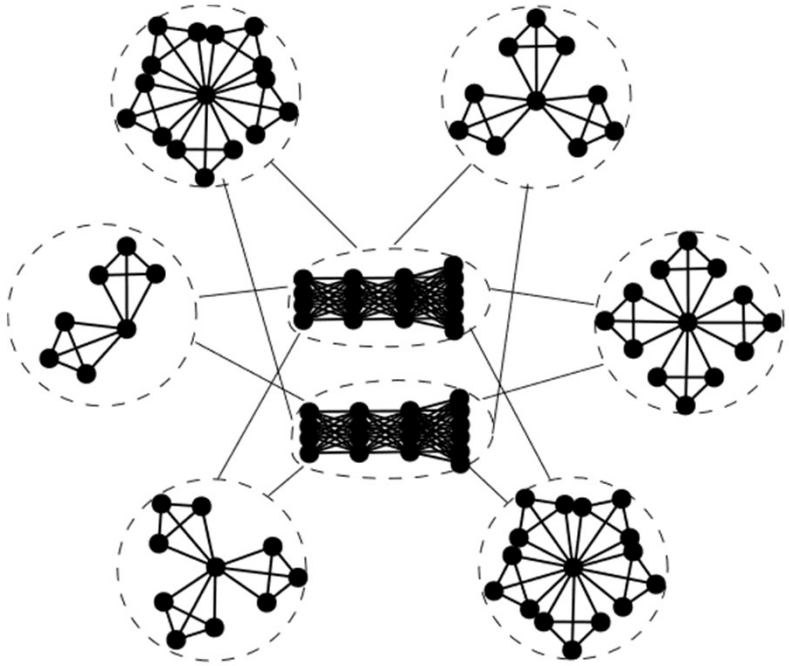
$(3,5,4)$–*chain*

Equitable Coloring: W[1]-hardness

Let $< A = \{a_1, a_2, \ldots, a_\ell\}, k, B >$ be an instance of Bin-Packing

A graph G is defined as follows:

- consider the disjoint union of two (k, ℓ, B)–*chains* plus the flowers $F_{a1,k}, \ldots, F_{a\ell,k}, F_{B,k}$
- join each node in the flowers to each node in the chains.



The graph G obtained by
$< A = \{2,1,4,2,3\}, \mathrm{k} = 3, \ B = 4 >$

Equitable Coloring: W[1]-hardness

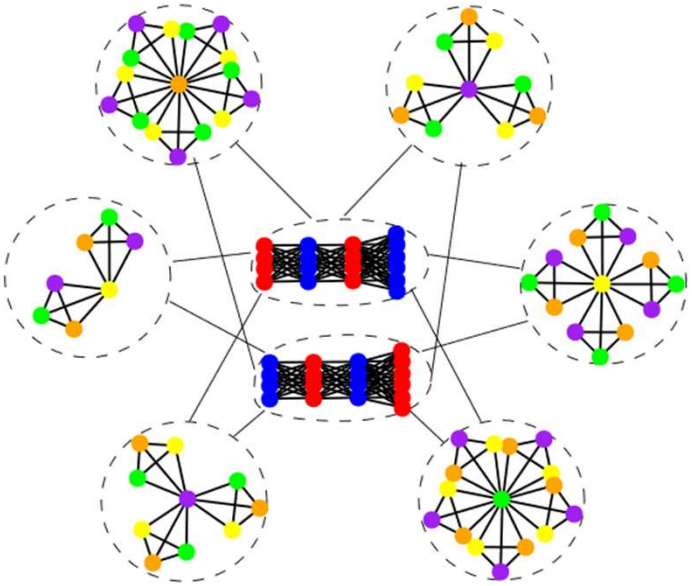
Lemma $< A = \{a_1, a_2, \dots, a_\ell\}, k, B >$ *is a YES instance of Bin-Packing iff* G *is equitably* $(k+3)$*–colorable.*

Proof.

Given a k-partition $P_1, \dots, P_k$ of A that solves the Bin-Packing instance, i.e. $\sum_{a_j \in P_i} a_j$ = B, we can construct an equitable $(k+3)$ –coloring c of the nodes of G.

- We use colors $k+2$ and $k+3$ to properly color the nodes in the two (k, ℓ, B)*–chains*
- Use colors $1, \dots, k+1$ to properly color the nodes in the flowers $F_{a1,k}, \dots, F_{a\ell,k}, F_{B,k}$
- We can prove that each class of colors contains exactly $Bk + \ell + 1 = \frac{|V(G)|}{k+3}$ nodes.

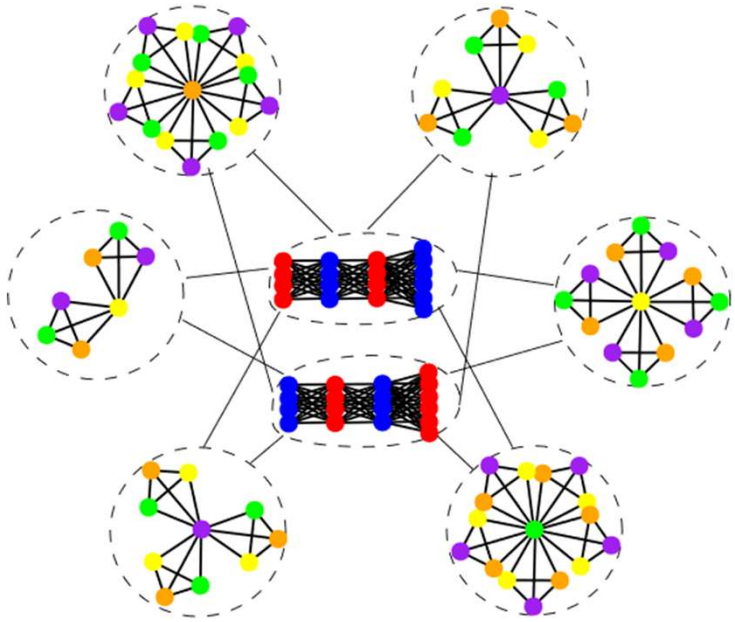
$A = \{2,1,4,2,3\}$ $B = 4$ $k = 3$
$P_1 = \{1,3\}$ $P_2 = \{2,2\}$ $P_3 = \{4\}$



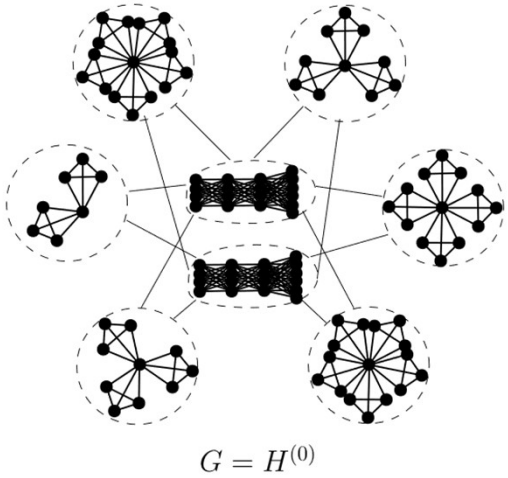
Equitable Coloring: W[1]-hardness

Now, let c be an equitable $(k+3)$ −coloring of G.

- Exactly two colors among the $k+3$ are used to color the nodes in the (k,ℓ,B)–*chains*
- The color used to color the central node of the flower $F_{B,k}$ is not used to color the central nodes of any other flower
- By using these results and counting arguments, we can prove that the k classes of colors involving the central nodes of the flowers $F_{a1,k}$,..., $F_{a\ell,k}$ induce a k-partition of A

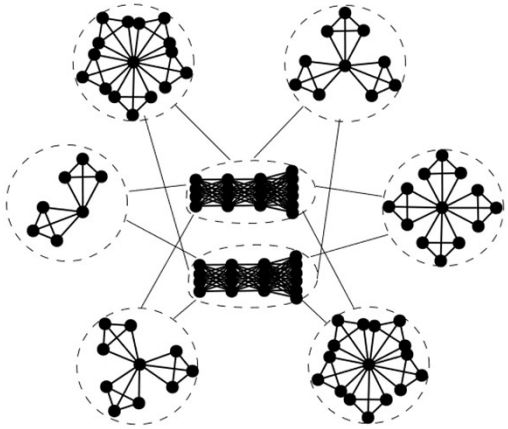


Equitable Coloring: W[1]-hardness

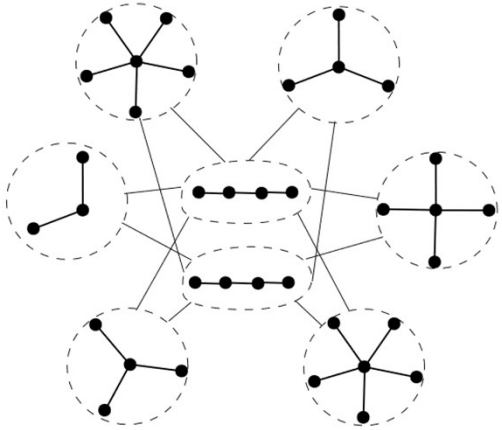


$G = H^{(0)}$

Equitable Coloring: W[1]-hardness

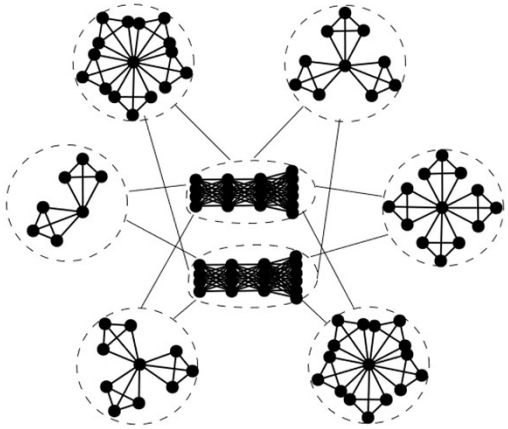


$G = H^{(0)}$

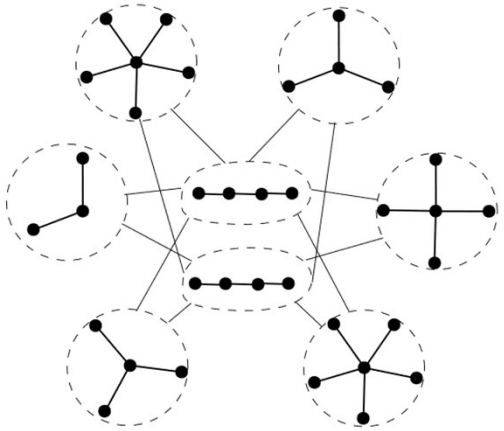


$H^{(1)}$

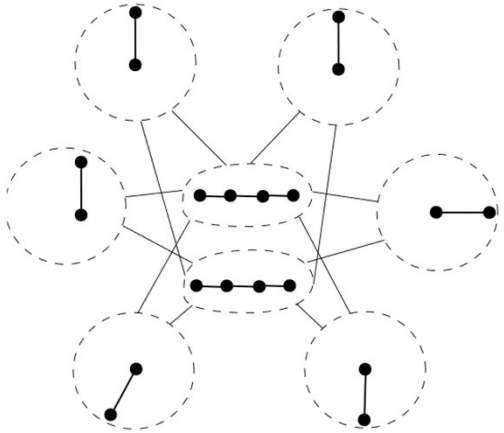
Equitable Coloring: W[1]-hardness



$G = H^{(0)}$

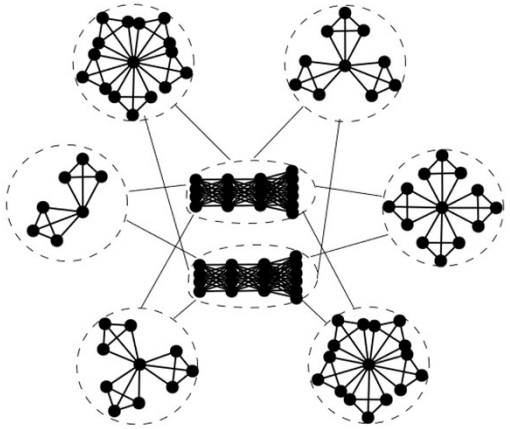


$H^{(1)}$

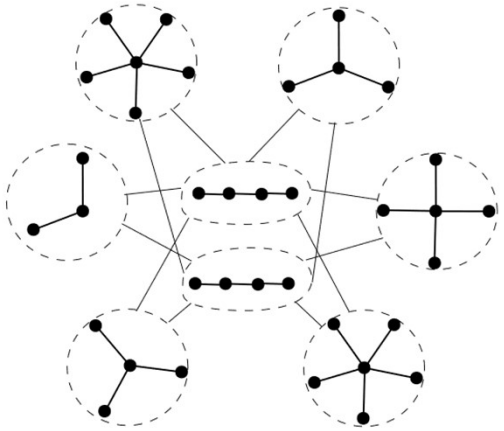


$H^{(2)}$

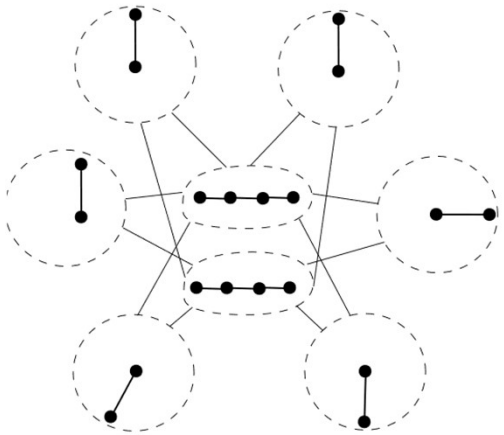
Equitable Coloring: W[1]-hardness



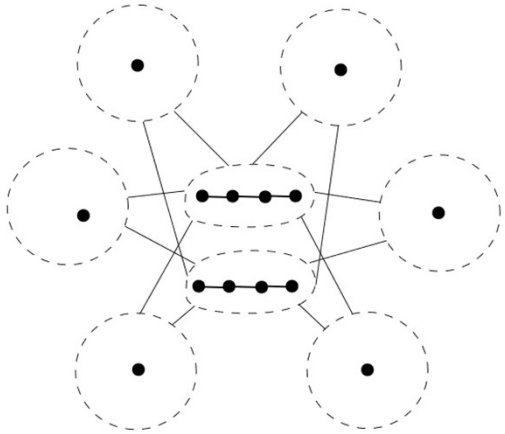
$G = H^{(0)}$



$H^{(1)}$

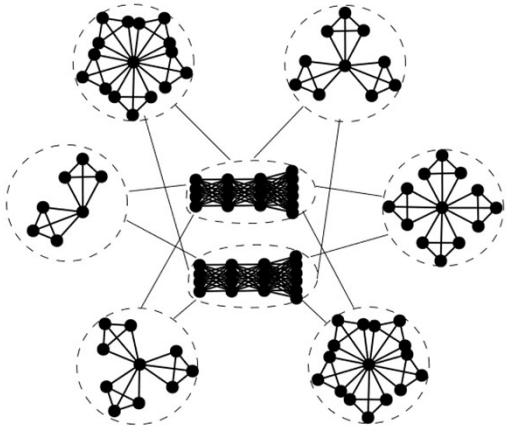


$H^{(2)}$

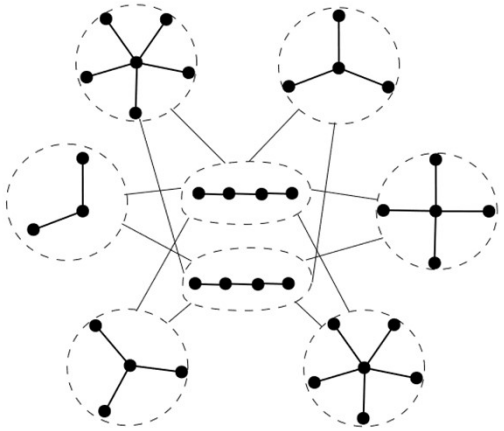


$H^{(3)}$

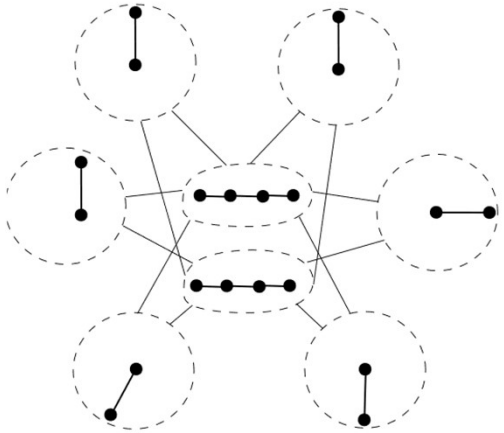
Equitable Coloring: W[1]-hardness



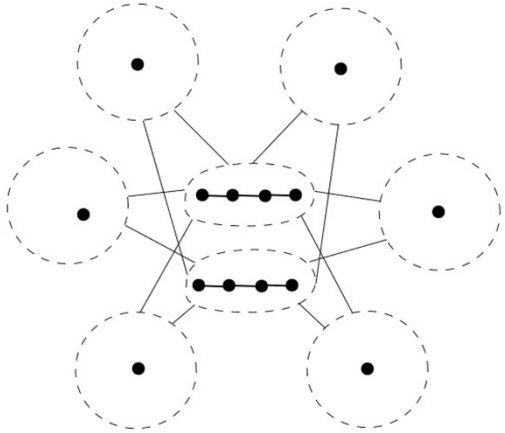
$G = H^{(0)}$



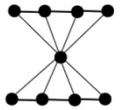
$H^{(1)}$



$H^{(2)}$



$H^{(3)}$



$H^{(4)}$

$itp(G) = 2k + 3$

Equitable Coloring: W[1]-hardness

Theorem 2. *The Equitable Coloring problem is* W[1]*-hard parametrized by* $itp(G)$

Corollary 1. *The Equitable Coloring problem is* W[1]*-hard parametrized by* $mw(G)$

Algorithms: a general scheme

Let P be a problem to be solved on an input graph G.

1. Iterate by generating the whole type graph sequence of G.
2. On each graph G′ in the type graph sequence, a generalized version P′ of the original problem is defined
   - with P′ in G′ being equivalent to P in G
3. Optimally solve P′ on the base graph and reconstruct the solution on the reverse type graph sequence (hence solving P in G).

If the time to solve P′ on the base graph is f and the construction of the solution for P′ can be done in $poly(n)$ time, then the whole algorithm needs $O(f + poly(n))$ time.

Algorithms: minimum dominating set

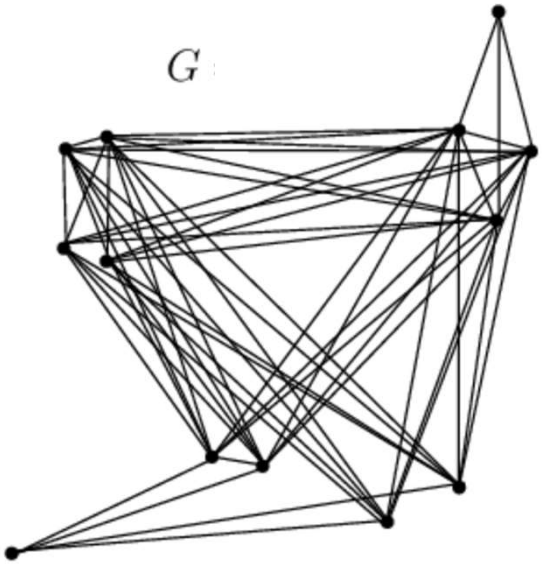
$G = (V, E)$ $Q \subseteq V$

A semi-total Dominating Set *of* G *with respect to* Q, *called* Q-stds *of* G, *is a set* $D \subseteq V$ *such that every node in* Q *is adjacent to a node in* D, *and every other node is either a node in* D *or is adjacent to a node in* D.

The set D *is called an* optimal Q-stds *of* G, *if its size is minimum among all the* Q-stds *of* G.

Note that $\emptyset$-stds *of* G becomes the Dominating Set *of* G

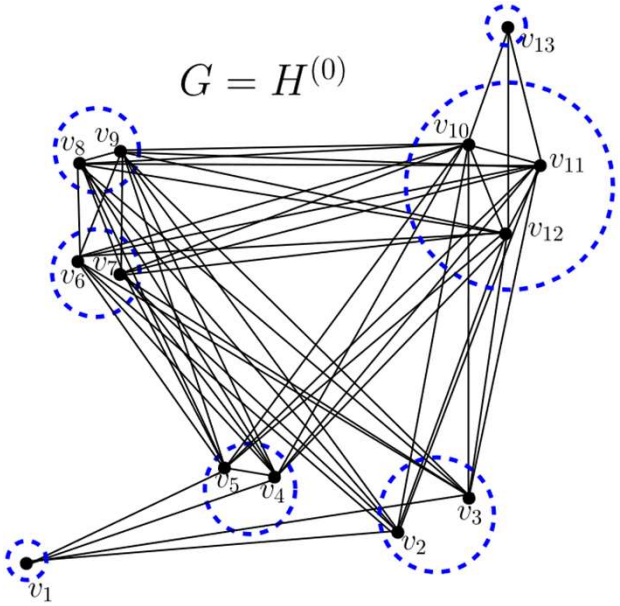
Algorithms: minimum dominating set

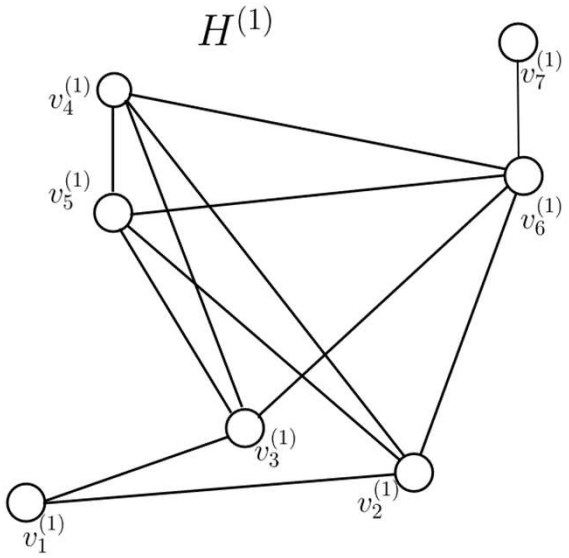


(r_1)

$H = G$
$Q = \emptyset$

Algorithms: minimum dominating set



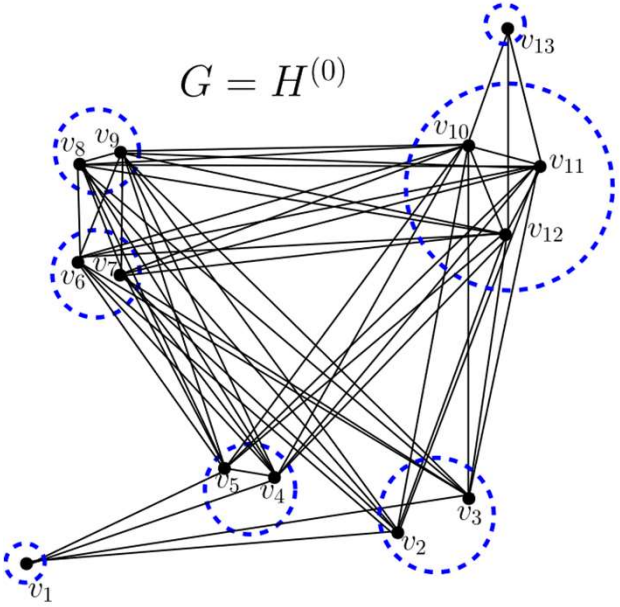


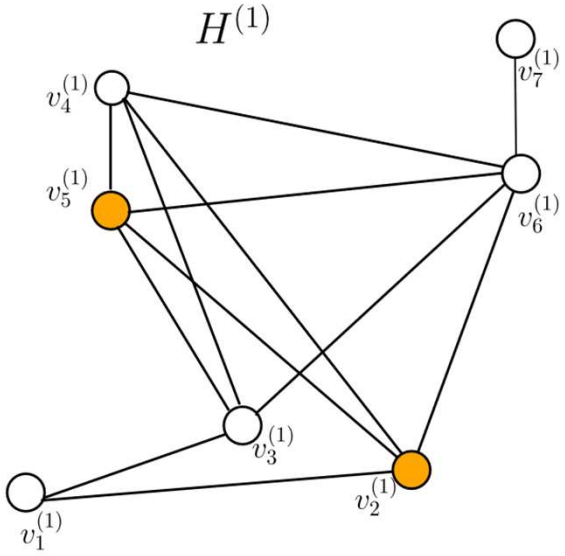
(r_1)

$H = G$
$Q = \emptyset$
$H' = H^{(1)}$
$Q' = \{v_2^{(1)}, v_5^{(1)}\}$

$Q' = \{x \in V(H') \mid (V_x \cap Q \neq \emptyset \text{ or } V_x \text{ is an independent set})\}$

Algorithms: minimum dominating set

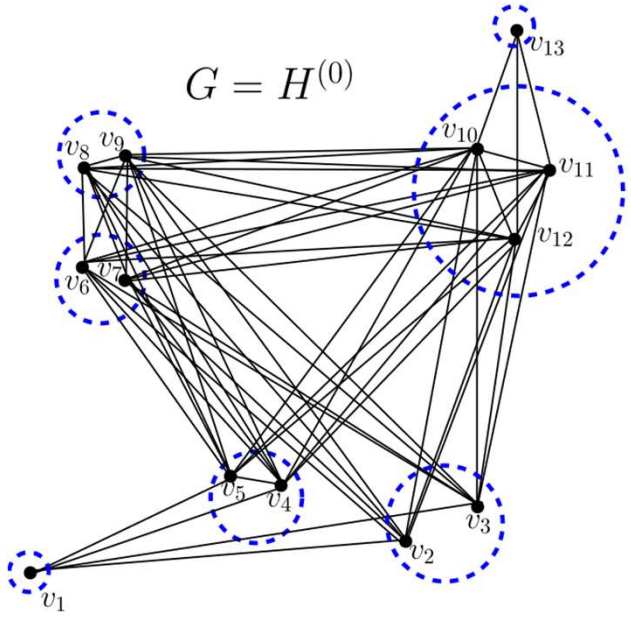


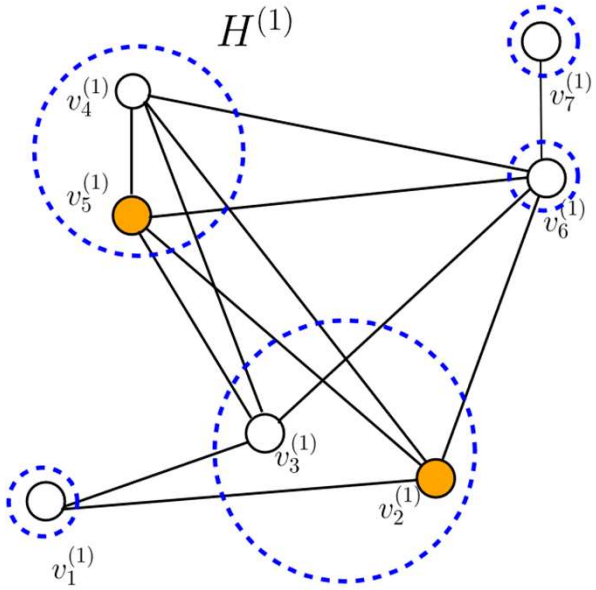


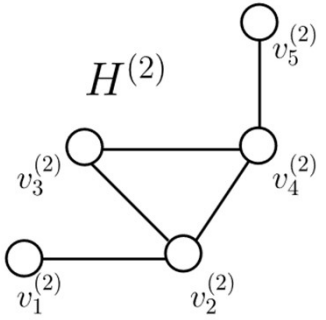
(r_1)

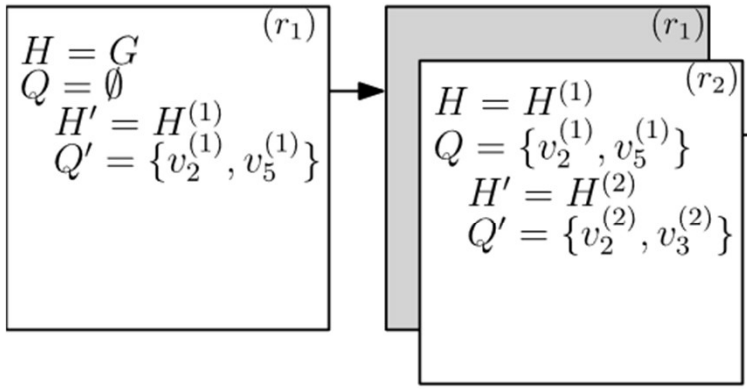
$H = G$
$Q = \emptyset$
$H' = H^{(1)}$
$Q' = \{v_2^{(1)}, v_5^{(1)}\}$

Algorithms: minimum dominating set

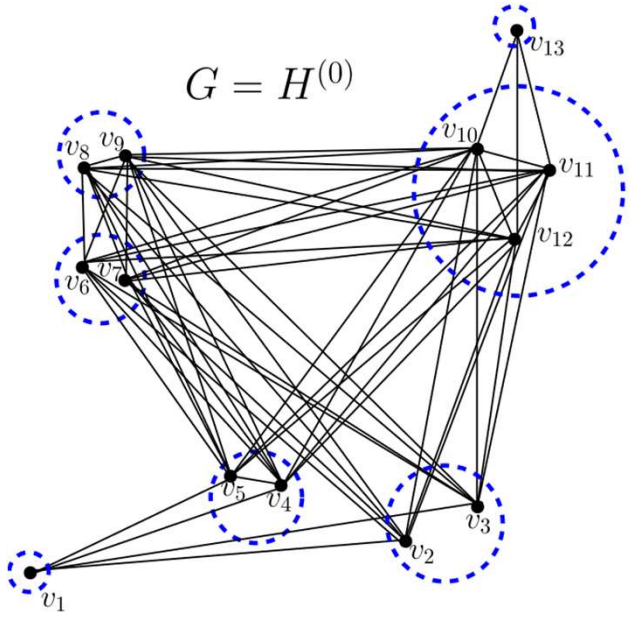


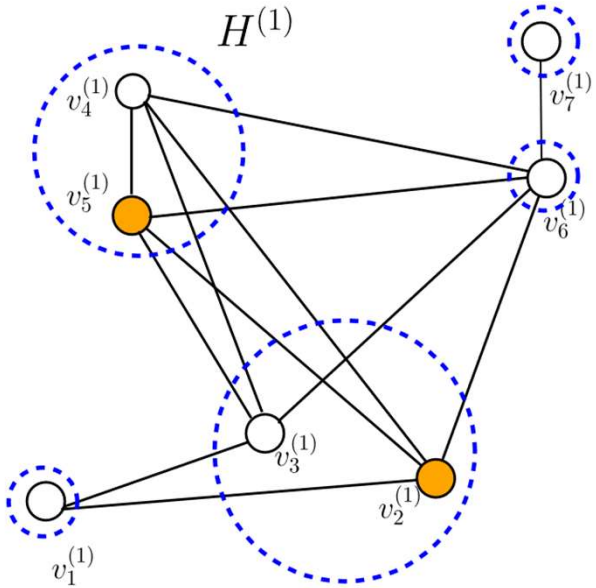


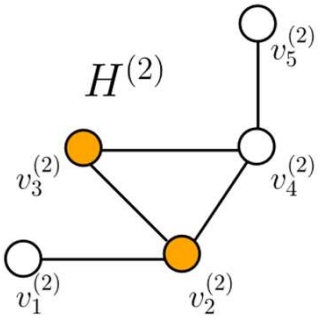


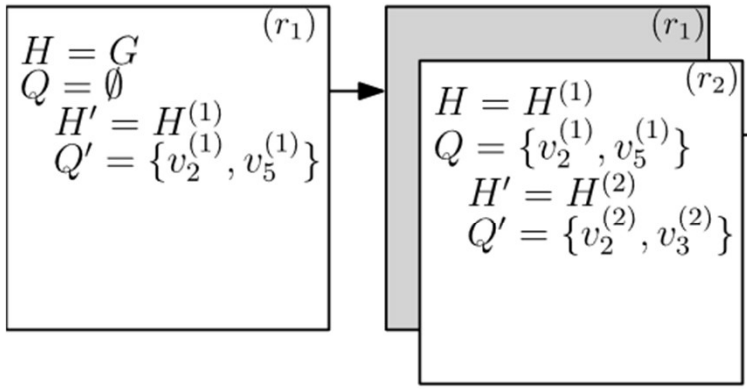


Algorithms: minimum dominating set

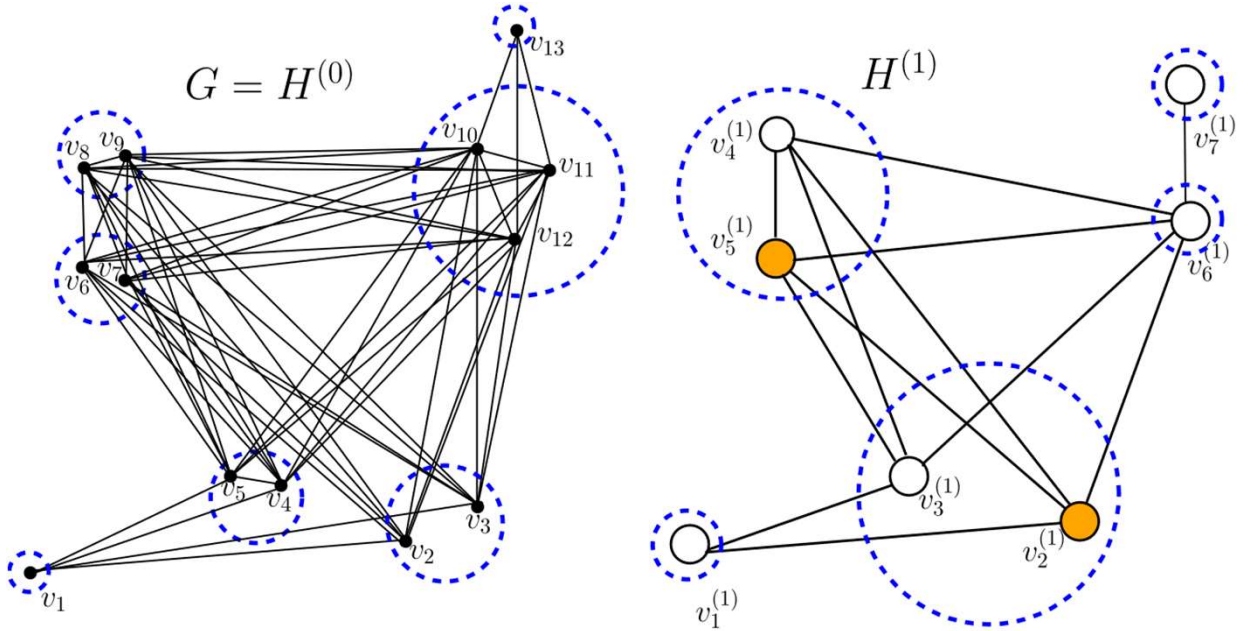


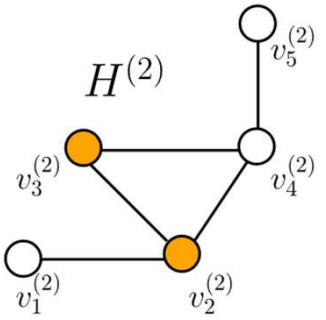


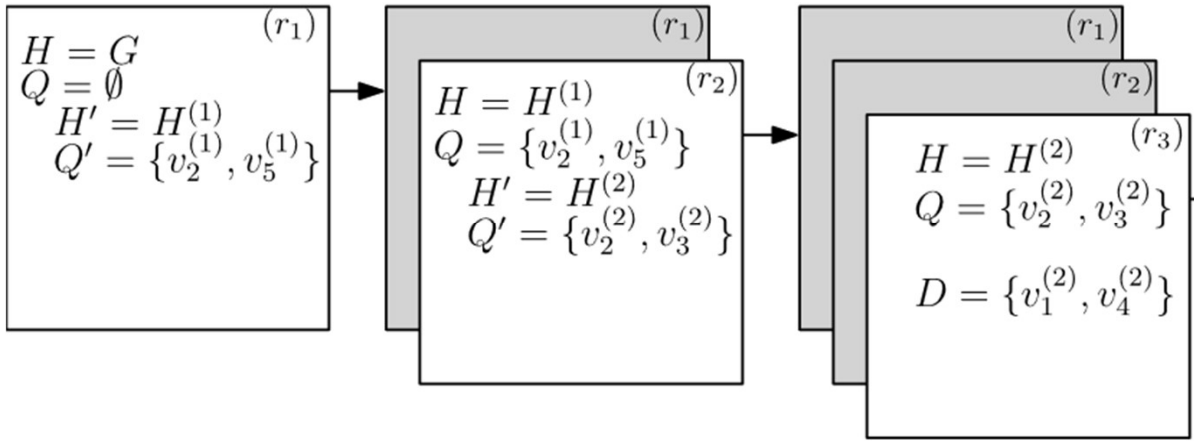




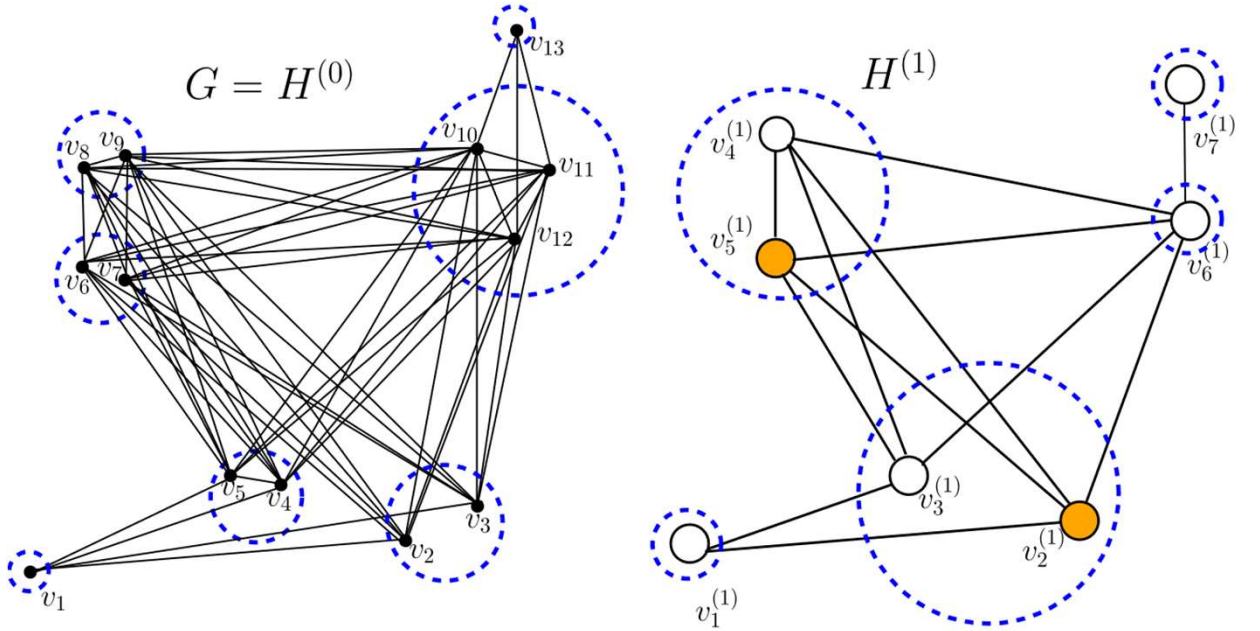
Algorithms: minimum dominating set

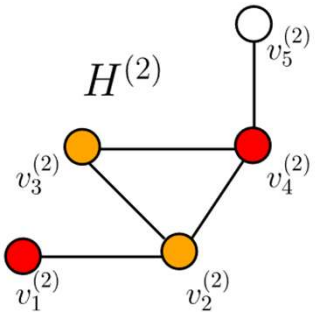


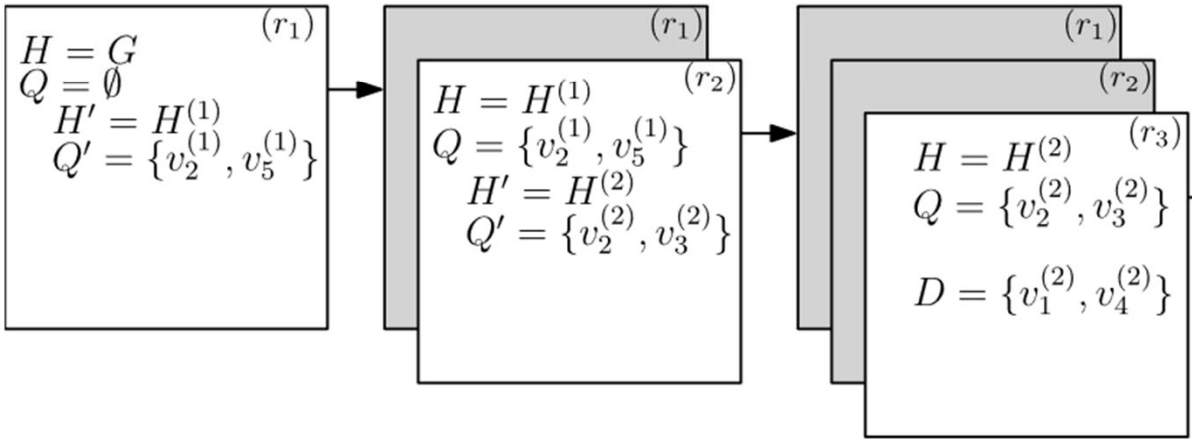




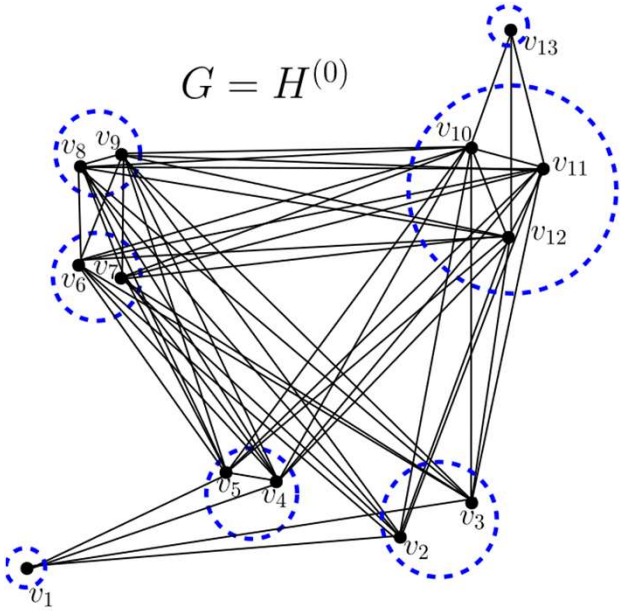
Algorithms: minimum dominating set

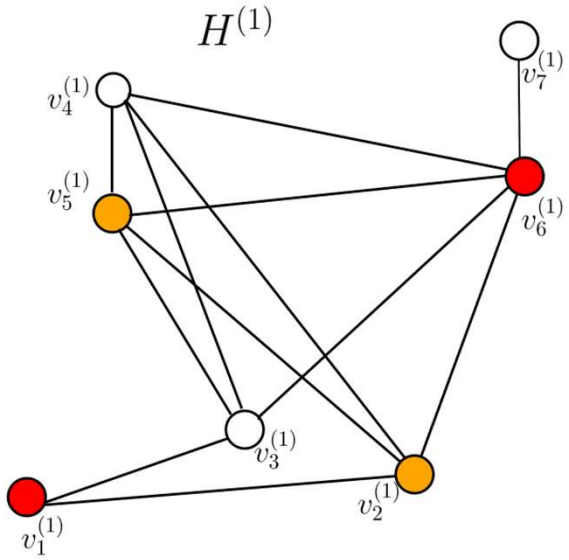


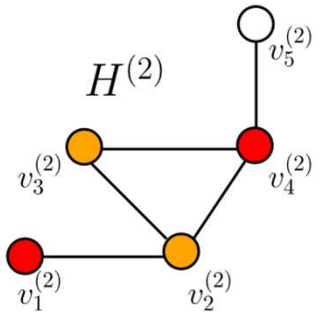


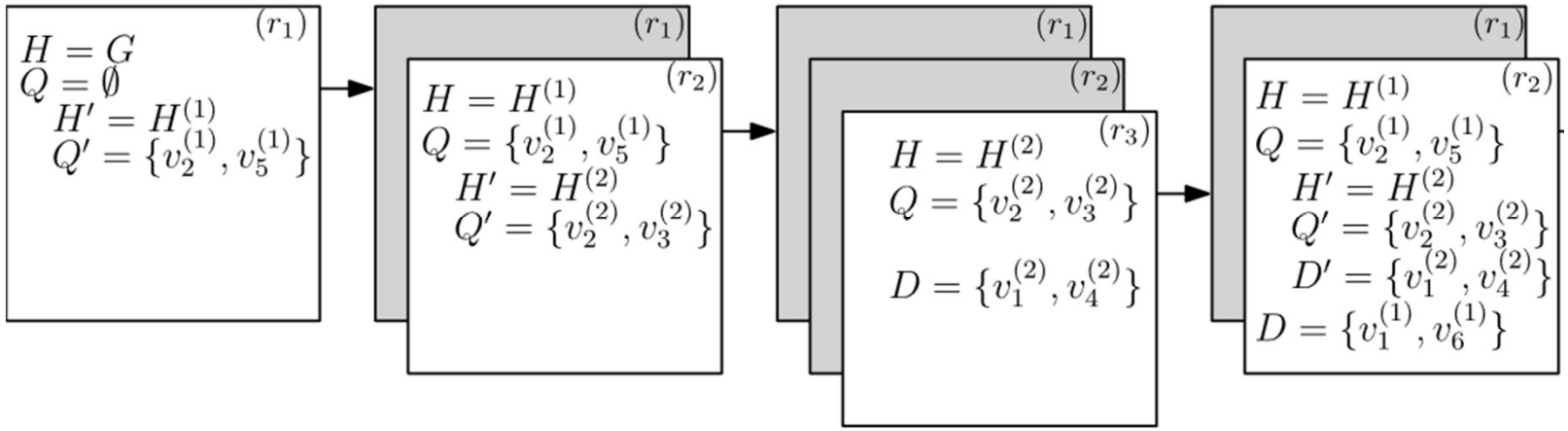


Algorithms: minimum dominating set

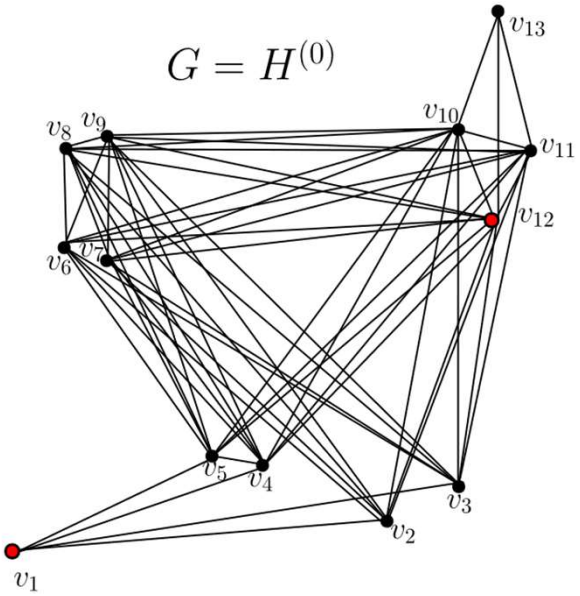


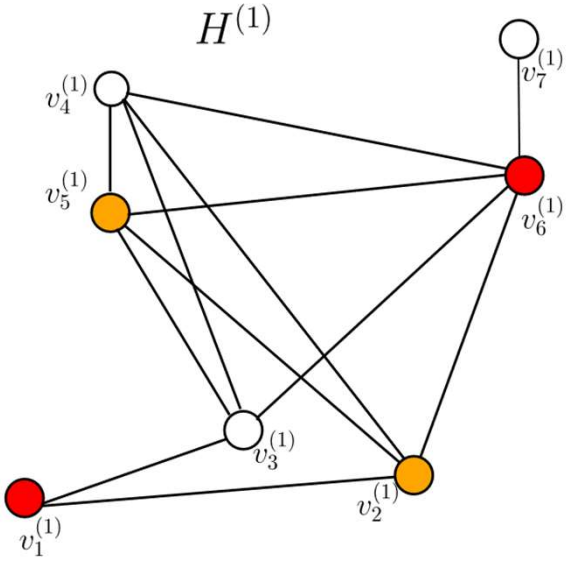


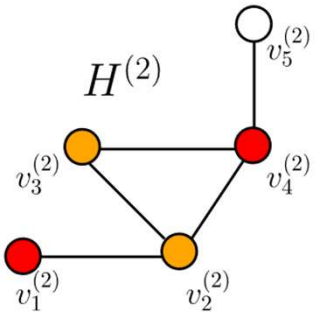


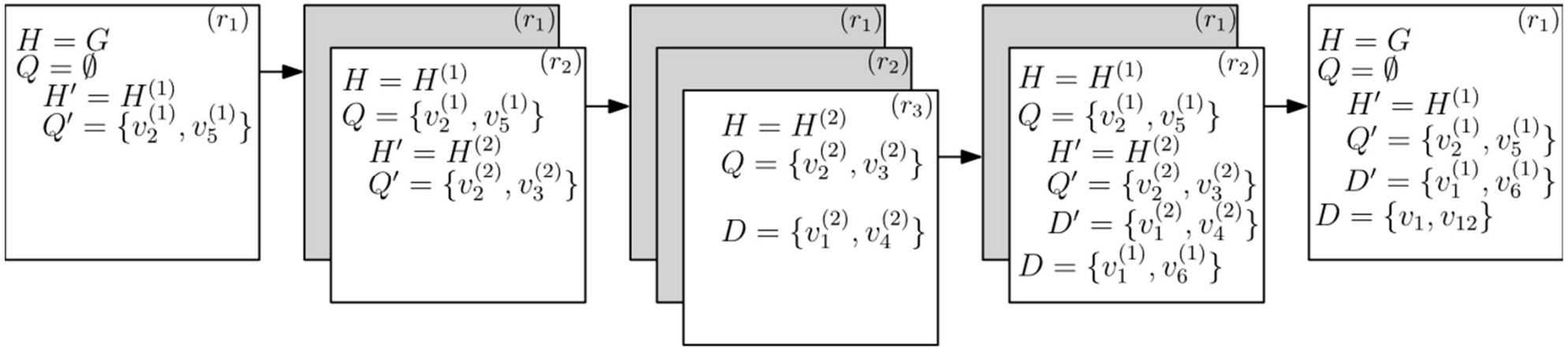


Algorithms: minimum dominating set









Algorithms: minimum dominating set

Theorem 4. *There is an algorithm that returns a* minimum dominating set *of a graph* G *in time* $O(2^{itp(G)} + poly(n))$.

Conclusion and future work

Iterated type partition

- EQC is W[1]-hard w.r.t. $itp(G)$
  - EQC is W[1]-hard w.r.t. $mv(G)$
  - Hardness drops for $nd(G)$
- FPT algorithms for DS, VC, Coloring w.r.t. $itp(G)$

Thank you!

Future work

- General algorithmic scheme → meta–algorithm
- FPT algorithm for Edge Dominating Set w.r.t. $itp(G)$