Restrictive H-Colorings

algorithms and complexity results

Josep Díaz   Maria Serna   Dimitrios M. Thilikos

Departament de Llenguatges i Sistemes Informàtics

Universitat Politécnica de Catalunya

Barcelona, Spain

Summary

- Parameterized complexity
- Restrictive H-coloring
- A parameterization of restrictive H coloring
  - Hard cases
  - Fixed parameter tractable cases
    - Kernels and Compactors
    - Connected G
    - The case with no lists
- Open problems

Parameterization

Split the input I to a problem $\{I \mid P(I)\}$ in two components $I = (S, K)$ and fix the second part ahead of the input.

Independent set

Given a graph G and an integer k

Does G have an independent set of size k?

Parameterized Independent set

Given a graph G and an integer k

Parameter: k

Does G have an independent set of size k?

Which is the well known k-Independent set problem.

For each value k we have one problem $\rightarrow$ a *layer*

FPT-algorithms

For a parameterized problem $\{I = (S, K) \mid P(I)\}$ where K is the parameter.

Define an integer $k = f(K)$ that measures the size of K.

A fixed parameter algorithm (FPT-algorithm) is an algorithm that solves a parameterized problem in time $O(f(k)n^{O(1)})$ where n is the input size and the hidden constant is independent of both k and n.

- When K is fixed independently of the input an FPT-algorithm takes polynomial time.
- $f(k)$ can be any function.
- A parameterized problem with a layer that is NP-hard has no FPT-algorithm (unless $\mathsf{P} = \mathsf{NP}$).

Parameterized complexity

The goal of parameterized complexity is to study parameterizations of hard problems versus FPT-algorithms.

In parameterized complexity a hierarchy of parameterized problems is defined
The W-hierarchy

$$\mathsf{FPT} \subseteq \mathsf{W}[1] \subseteq \mathsf{W}[2] \subseteq \cdots \subseteq \mathsf{W}[SAT] \subseteq \mathsf{W}[P] \subseteq \mathsf{XP}.$$

$\mathsf{FPT} = \mathsf{W}[1]$ implies the existence of a $O(2^{o(n)})$ for the 3-SAT

Together with a reducibility that allows to prove hardness on those classes.

Parameterized Complexity [Downey, Fellows SIAM J. Computing and TCS 1995]

Parameterized Complexity of Counting problems [Flum, Grohe FOCS 2002]

Some examples of parameterized complexity

k-coloring is NP-hard ($k \geq 3$) unlikely to have a FPT-algorithm

k-independent set is W[1]-hard (but it has a $O(n^{k+1})$ algorithm)

k-vertex cover is in FPT

Restrictions to list H-coloring

List H-coloring allows to model an assignment problem from task to processors, preserving communication needs in which tasks have a list of prefered processors

Some processors might have limited load.

We can restrict the load of a vertex in H.

[DST WG 02 : Discrete Applied Mathematics 05]

Restrictive H-coloring problems

A partial weight assignment to H is a pair (C, K)
where $C \subseteq V(H)$ and $K : C \to \mathbb{N}$

A restrictive list H-coloring of (G, L) and (C, K), where G is a graph, L is a (H, G)-list, and (C, K) is a partial weight assignment, is a list H-coloring χ of (G, L) such that for all $c \in C$, $|\{v \mid \chi(v) = c\}| = K(c)$.

- Problems

  restrictive H-coloring, restrictive $\#H$-coloring.

  Input: G, C, K

  restrictive list H-coloring, restrictive list $\#H$-coloring.

  Input: G, C, K

- Notation

  $\mathcal{H}_H(G, L, C, K) =$ set of all restrictive list H-colorings of (G, L) and (C, K)

In a similar way we can think of having at most $K(c)$ pre-images of $c \in C$.

Restrictive H-coloring: Complexity

Restrictive H-coloring: Complexity

Problem	P	NP-complete/#P-complete	
restrictive list H-coloring	dichotomy$^{(3)}$		[DST DAM 05]
restrictive H-coloring	dichotomy$^{(3)}$		[DST DAM 05]
restrictive list #H-coloring	dichotomy$^{(3)}$		[DST DAM 05]
restrictive #H-coloring	dichotomy$^{(3)}$		[DST DAM 05]

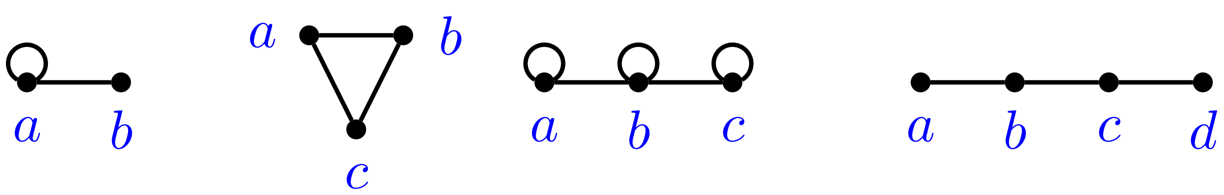
(3) All the connected components of H are either complete reflexive graphs or complete irreflexive bipartite graphs.

Restrictive H-coloring: Complexity

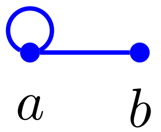
Problem	P	NP-complete/#P-complete	
restrictive list H-coloring	dichotomy$^{(3)}$		[DST DAM 05]
restrictive H-coloring	dichotomy$^{(3)}$		[DST DAM 05]
restrictive list $\#H$-coloring	dichotomy$^{(3)}$		[DST DAM 05]
restrictive $\#H$-coloring	dichotomy$^{(3)}$		[DST DAM 05]

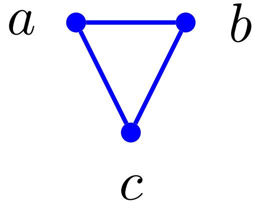
(3) All the connected components of H are either complete reflexive graphs or complete irreflexive bipartite graphs.

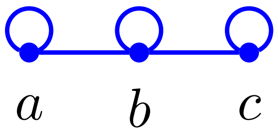
All the connected components of a graph H are either a complete reflexive graph or a complete irreflexive bipartite graph iff H does not contain as induced subgraphs any of the graphs

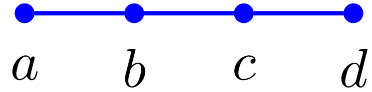


Hardness: Restrictive H-coloring

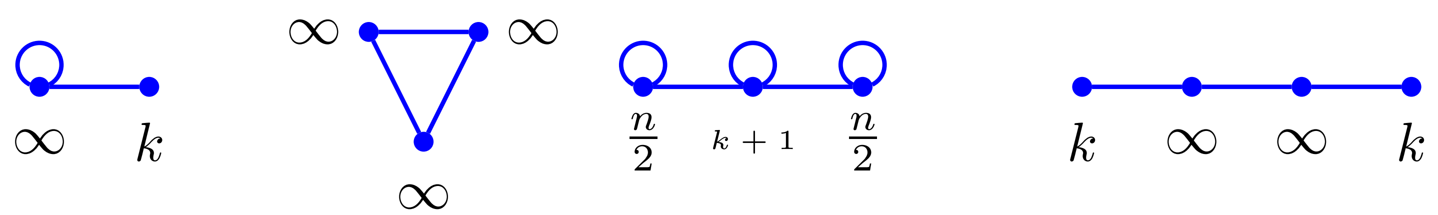








Hardness: restrictive H-coloring



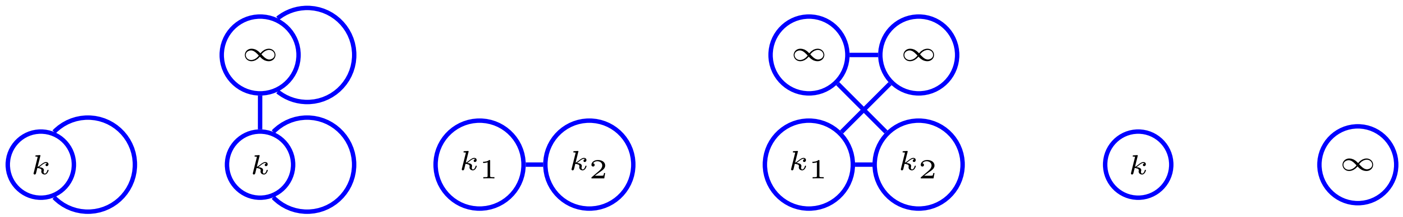
Which correspond to the NP-hard problems

- ☐ Independent set
- ☐ 3-coloring
- ☐ Balanced separator
- ☐ Balanced complete bipartite subgraph

As H is given, when it contains one of the above subgraphs, we put the weights in the adequate places and the remaining vertices of H get weight 0.

Easy cases: restrictive list $\#H$-coloring

Given a connected graph G, $|\mathcal{H}_H(G, \mathsf{w}_H)|$ can computed in polynomial time for



A, w_a B, w_b C, w_c D, w_d E, w_e F, w_f

Lemma: If all the connected components of H are either a complete irreflexive bipartite graph or a complete reflexive clique, then the restrictive list $\#H$-coloring problem can be solved in polynomial time for connected G.

Using additionally a dynamic programming algorithm

Theorem: If all the connected components of H are either a complete irreflexive bipartite graph or a complete reflexive clique, then the restrictive list $\#H$-coloring problem can be solved in polynomial time.

Parameterization: The (H, C, K)-coloring

[DST MFCS 01, DIMATIA-DIMACS 02, EUROCOMB 03, ESA 04, ...]

Parameterization: The (H, C, K)-coloring

We consider a bounded version of restrictive H-coloring.

Input to the restrictive list H-coloring problem: G, L, C ,K

We take as parameter (C, K) the partial weight assignment on H.

But we are parameterizing a parameterized problem!

Real parameter (H, C, K) a partially weighted graph

For a partially weighted graph (H, C, K) we set

$k = \sum_{c \in C} K(c) \quad h = |V(H)| \quad c = |C| \quad s = h - c$

Parameterized problems

For a partially weighted graph (H, C, K).

A list (H, C, K)-coloring of (G, L) is a
restrictive list H-coloring of (G, L) and (C, K).

Problems

- □ (H, C, K)-coloring $\#(H, C, K)$-coloring

  Input: G

  Parameter: k

- □ list (H, C, K)-coloring list $\#(H, C, K)$-coloring problem

  Input: G, L

  Parameter: k

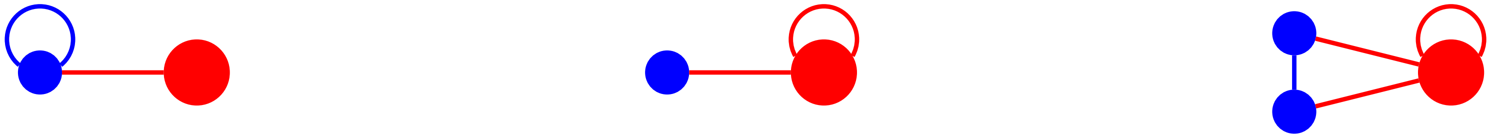
Notation for sets of (H, C, K)-colorings $\mathcal{H}_{(H,C,K)}(G, L)$ $\mathcal{H}_{(H,C,K)}(G)$

(H, C, K)-coloring

The problem captures some well known parameterized problems as particular cases?

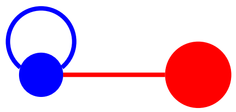
(H, C, K)-coloring

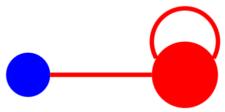
The problem captures some well known parameterized problems as particular cases?

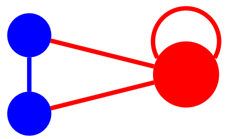


(H, C, K)-coloring

The problem captures some well known parameterized problems as particular cases?



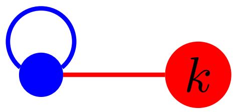




Any graph has a H-coloring

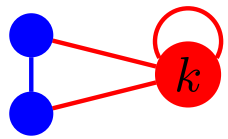
(H, C, K)-coloring

The problem captures some well known parameterized problems as particular cases?



k-Independent Set

k-Vertex Cover



k-remove for bipartite

Complexity (H, C, K)-coloring

Problem	P	NP-complete/#P-complete	
list (H, C, K)-coloring	dichotomy$^{(1)}$		[DST MFCS 01, DM ??]
(H, C, K)-coloring	list $(H - C)$-coloring in P $^{(1)}$	$(H - C)$-coloring NP-hard$^{(2)}$	[DST MFCS 01, DM ??]
list $\#(H, C, K)$-coloring	dichotomy$^{(3)}$		[DST EUROCOMB 03, DM ??]
$\#(H, C, K)$-coloring	list $\#(H - C)$-coloring in P $^{(3)}$	(H, C, K) irreducible	[DST EUROCOMB 03, DM ??]

(1) $H - C$ is a bi-arc graph.

(2) $H - C$ is bipartite or contains a loop.

(3) All the connected components of $H - C$ are either complete reflexive graphs or complete irreflexive bipartite graphs.

Let (H, C, K) be a partially weighted graph and let c be a vertex in C. We call (H, C, K) *c-reducible* if H has an $(H - \{c\})$-coloring χ such that $\chi(c) \in V(H) - C$. We say that (H, C, K) is *reducible* if it is c-reducible for some $c \in C$, otherwise (H, C, K) is said to be *irreducible.*

Complexity (H, C, K)-coloring

Problem	P	NP-complete/#P-complete	
list (H, C, K)-coloring	dichotomy$^{(1)}$		[DST MFCS 02, DM 06]
(H, C, K)-coloring	list $(H - C)$-coloring in P $^{(1)}$	$(H - C)$-coloring NP-hard$^{(2)}$	[DST MFCS 02, DM 06]
list $\#(H, C, K)$-coloring	dichotomy$^{(3)}$		[DST EUROCOMB 03, DM 06]
$\#(H, C, K)$-coloring	list $\#(H - C)$-coloring in P $^{(3)}$	(H, C, K) irreducible	[DST EUROCOMB 03, DM 06]

(1) $H - C$ is a bi-arc graph.

(2) $H - C$ is bipartite or contains a loop.

(3) All the connected components of $H - C$ are either complete reflexive graphs or complete irreflexive bipartite graphs.

There are partially weighted graphs (H, C, K) and (H', C', K') for which $H - C = H' - C'$, $H - C$ satisfies (2), but (H, C, K)-coloring belongs to P but (H', C', K') is NP-complete

Parameterized complexity of (H, C, K)-coloring

Some W[1]-hard cases

Theorem: The list (H, C, K)-coloring problem is W[1]-hard if there is a looped vertex in $H - C$ connected to a un-looped vertex in C.

Theorem: The (H, C, K)-coloring problem is W[1]-hard, in the case that $H = K_1^r \oplus H'$ and $C = V(H')$, for some graph H' which contains at least one un-looped vertex.

By a parameterized reduction from the W[1]-hard problem k-independent set.

Easy cases: with FPT-algorithm

Easy cases: with FPT-algorithm

We have to design FPT-algorithms for both decision and counting version.

Algorithmic techniques

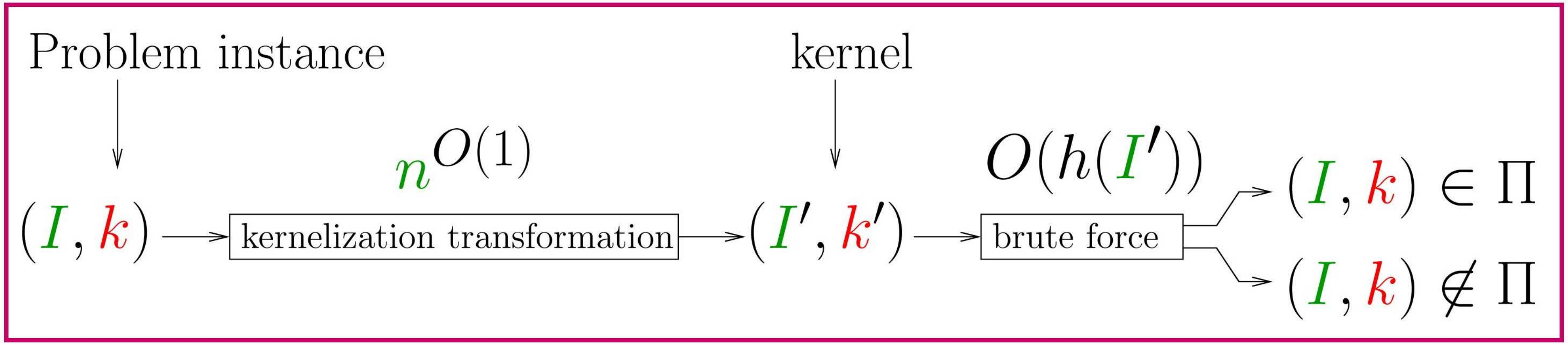
Decision problems

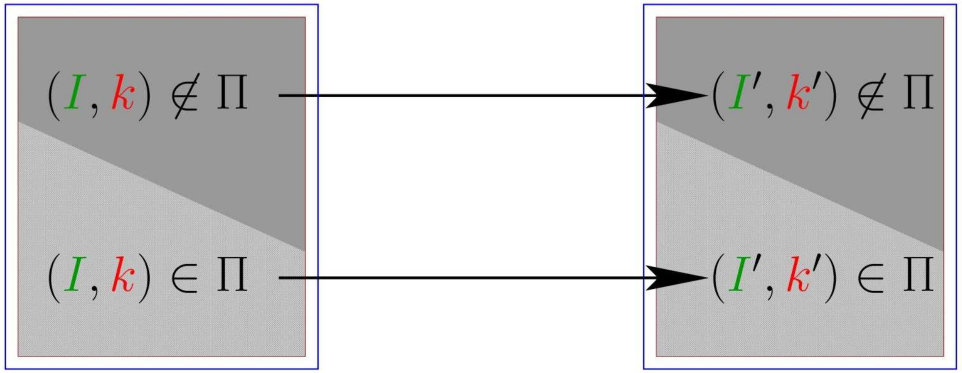
Reduction to a problem Kernel

Counting problems

Compactor enumeration

Technique: kernelization





$|I'| \leq g(k)$ and $k' \leq k$

$(I,k) \in \Pi \Leftrightarrow (I',k) \in \Pi$

Overall time: $O(h(g(k))) + n^{O(1)} = \boxed{O(f(k)) + n^{O(1)}} \in \mathsf{FPT}$

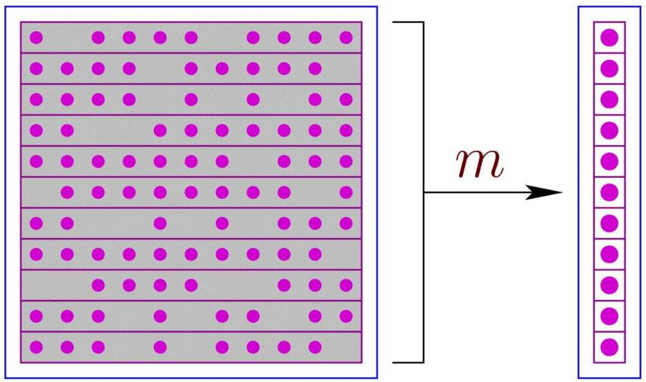
Technique: Compactor construction

Problem instance → (I, k) → [compactor generation] → $\mathsf{cmp}(I, k)$ ← compactor (a set) → output $\sum_{C \in \mathsf{cmp}(I,k)} |m^{-1}(C)|$

(i) $\mathsf{cmp}(I, k)$ can be *enumerated* in FPT

(ii) there is a *surjective* function $m : \mathsf{solutions}(I, k) \rightarrow \mathsf{cmp}(I, k)$ and
for any $C \in \mathsf{cmp}(I, k)$, $|m^{-1}(C)|$ can be *computed* in FPT

$\mathsf{solutions}(I, k)$ $\mathsf{cmp}(I, k)$



Conclusion:

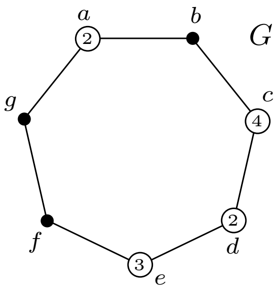
If there exists a $\mathsf{cmp}(I, k)$ satisfying (i) and (ii) then $\phi(I, k)$ can be computed in FPT

Tribal graphs

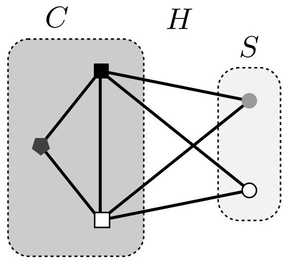
A tribal graph $\widetilde{G}$ is a graph G together with a vertex weight assignment p.

A list (H, C, K)-coloring of a tribal graph $\widetilde{G}$ and an $(H, \widetilde{G})$ list L is a mapping $w : V(\widetilde{G}) \times V(H) \rightarrow \{0, \ldots, k\}$ where:

1. $\forall v \in V(\widetilde{G})$ and $a \in V(H) - C$, $w(v, a) \leq 1$.
2. $\forall v \in V(\widetilde{G})$ and $a \in C$, $w(v, a) \leq K(a)$.
3. $\forall v \in V(\widetilde{G})$, $1 \leq \sum_{a \in H} w(v, a) \leq p(v)$.
4. $\{v, u\} \in E(\widetilde{G}) \implies \forall a, b \in H$ with $w(v, a) > 0$ and $w(u, b) > 0$, $\{a, b\} \in E(H)$.
5. $\forall a \in C$, $\sum_{v \in V(\widetilde{G})} w(v, a) = K(a)$,
6. $\forall v \in V(\widetilde{G})$ and $a \in V(H)$ with $w(v, a) > 0$, $a \in L(v)$.



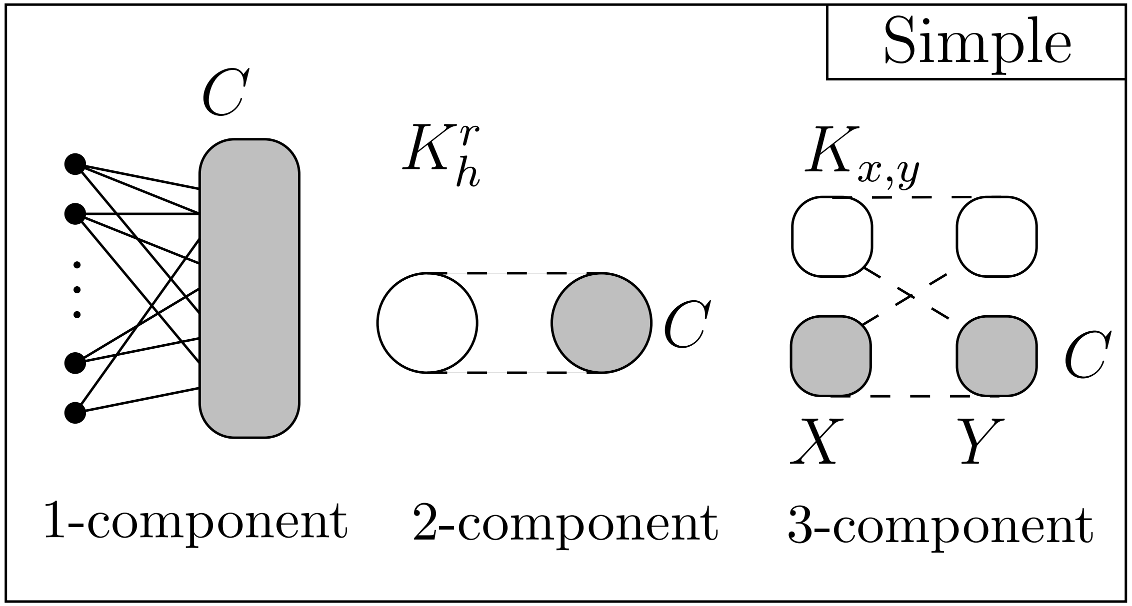
$L(a) = \{$ ● ⬟ $\}$
$L(b) = \{$ □ ○ $\}$
$L(c) = \{$ ⬟ ● ○ $\}$
$L(d) = \{$ ○ ■ $\}$
$L(e) = \{$ ○ ⬟ $\}$
$L(f) = \{$ □ $\}$
$L(g) = \{$ ○ ● ■ $\}$



	a	b	c	d	e	f	g
■				2			1
□		1				1	
⬟			2		2		
○			1		1		
●	1		1				

List (H, C, K)-coloring: Connected G

Partially weighted graphs



Generic Kernel/Compactor construction

Define $\mathcal{P}$ to be the partition of $V(G)$ induced by the equivalence relation,

$$v \sim u \text{ iff } [N_G(v) = N_G(u) \ \wedge \ L(v) = L(u)].$$

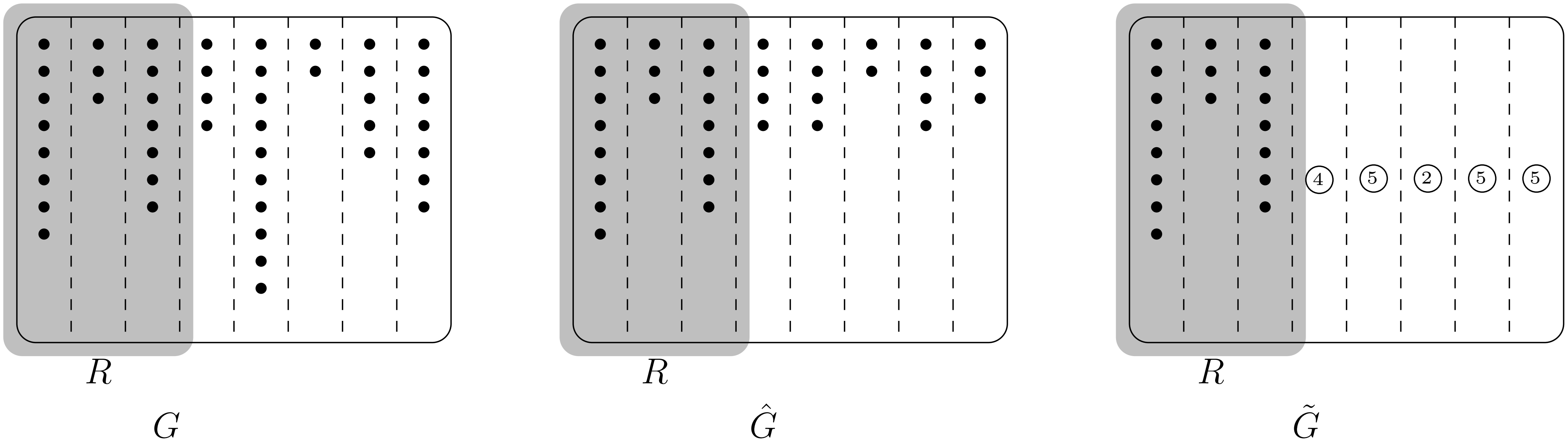
For $v \in V(G)$, $P_v = \{u \mid u \sim v\}$ and for any $Q \in \mathcal{P}$, we select a representative vertex $v_Q \in Q$.

$R \subseteq V(G)$ is a closed set for $\mathcal{P}$, if for any $v \in R$ we have $P_v \subseteq R$.

Generic Kernel/Compactor construction

For $v \in V(G)$, let P_v^k be P_v if $|P_v| \leq k$, otherwise it is a subset of P_v with $k+1$ vertices. Define $\widehat{G} = G\left[R \cup \left(\cup_{v \notin R} P_v^k\right)\right]$.

Define $\widetilde{G} = (G[R \cup \{v_Q \mid Q \in \mathcal{P} \text{ and } v_Q \notin R\}], p)$, where $p(v) = 1$ when $v \in R$, and $p(v_Q) = min\{|Q|, k+s\}$, for $s = |V(H) - C|$.



$k = 3, s = 2$

Generic Kernel/Compactor construction

Lemma: Let (H, C, K) be a partially weighted graph and R a closed set of $\mathcal{P}$. Given a graph G together with a (H, G)-list L, $\mathcal{H}(G, L) \neq \emptyset$ iff $\mathcal{H}(\widehat{G}, L) \neq \emptyset$.

Fundamental property for $(\widehat{G}, L)$ to be a Kernel

Lemma: Let (H, C, K) be a partially weighted graph and R a closed set of $\mathcal{P}$. Given a graph G together with a (H, G)-list L. Then, there is a surjective function from $\mathcal{H}(G, L)$ into $\mathcal{H}(\widetilde{G}, L)$.

Fundamental property for $\mathcal{H}(\widetilde{G}, L)$ to be a Compactor

The remaining properties follow from an adequate selection of the closed set R.

Case 1: G is connected and $H - C$ is edgeless

The k-splitting of G is the partition (R_1, R_2, R_3) of $V(G)$ where

R_1 is the set of vertices with degree more than k
R_2 is formed by the non isolated vertices in $G' = G[V(G) - R_1]$,
and R_3 contains the isolated vertices in G'.

Lemma: Let (H, C, K) be a partially weighted graph, where $H - C$ is edge-less. Given a graph G, let (R_1, R_2, R_3) be the k-splitting of G. Then $R_1 \cup R_2$ is a closed set. Furthermore if $|R_1| > k$ or $|R_2| > k^2 + k$, then $\mathcal{H}_{(H,C,K)}(G, L) = \emptyset$.

k-splitting is a well know partition to obtain a kernelization for k-Independent set [Buss, Goldsmith 93]

Case 1: G is connected and $H - C$ is edgeless

R_1 is the set of vertices with degree at least k
R_2 is formed by the non isolated vertices in $G' = G[V(G) - R_1]$,
and R_3 contains the isolated vertices in G'.

Now we assume $|R_1| \leq k$ and $|R_2| \leq k^2 + k$

Furthermore, all the vertices in R_3 have degree at most k and

For any $v \in R_3$, $N_G(v) \subseteq R_1$.

Therefore:

$\widehat{G}$ has size $\leq 2k + k^2 + (k+1)2^{k+h}$ and
can be obtained in time $O((k+h)n + 2^{k+h})$ Kernel

We also show that

- □ $|\mathcal{H}(\widetilde{G}, L)| = f(k, h)$
- □ The information provided by $w \in \mathcal{H}(\widetilde{G}, L)$ is enough to compute in FPT the size of the subset of colorings w represents.

Compactor

Case 1: The last piece

For the decision version we have to solve an instance of list (H, C, K)-coloring when $H - C$ has no edges.

So we have to devise a fast exact algorithm for this particular case:

So we have to devise a fast exact algorithm for this particular case:

Theorem: Let (H, C, K) be a partially weighted graph, where $H - C$ is edge-less. Given an input graph G, there is an algorithm that decides whether there is a list (H, C, K)-coloring of (G, L) in $O\left(2^k c^k \left((k+h)n + nk\sqrt{n+k}\log k\right)\right)$ steps.

Theorem: Let (H, C, K) be a partially weighted graph, where $H - C$ is edge-less. Given an input graph G and a (H, G)-list L, there is an algorithm that decides whether there is a list (H, C, K)-coloring of (G, L) in time

$$O\left((h+k)n + 2^{k+h} + 2^{5k/2}c^k p(k,h)\right),$$

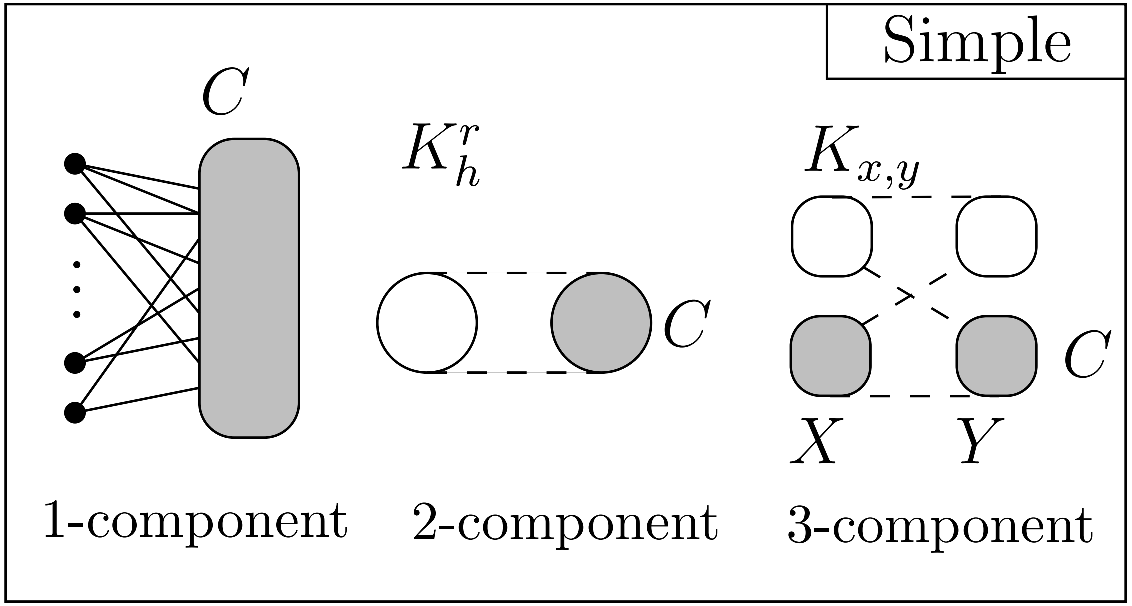
for some polynomial p.

Case 1: The last piece

For the counting version we apply directly the proposed schema of enumerate-and-count taking care of the cost of constructing the compactor.

Theorem: Let (H, C, K) be a partially weight assignment, where $H - C$ is edge-less. Given a graph G and a (H, G)-list L then, there is an algorithm that outputs the number of list (H, C, K)-colorings of (G, L) within time $O(f_1(k)n + f_2(k)\log n + f_3(k))$.

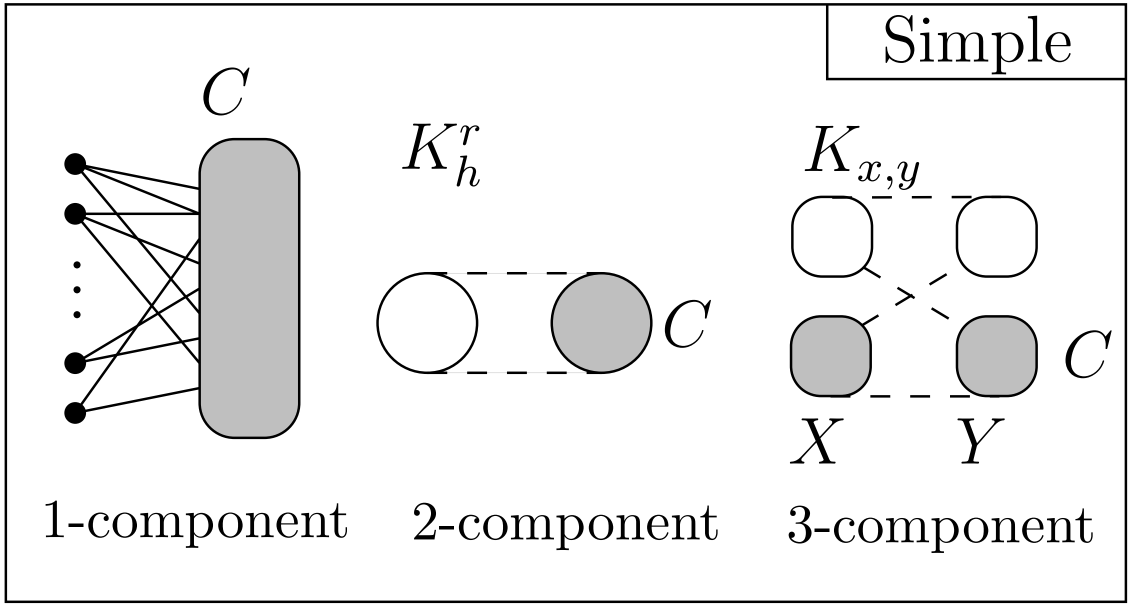
list (H, C, K)-coloring: Connected G



Using similar ideas but with different spplittings and ad-hoc algorithms we find FPT algorithms for the counting and decision versions of the list (H, C, K)-coloring problem for each one of the 1,2 or 3-components for connected G.

G must be mapped to only one of the components of H

Simple (H, C, K)

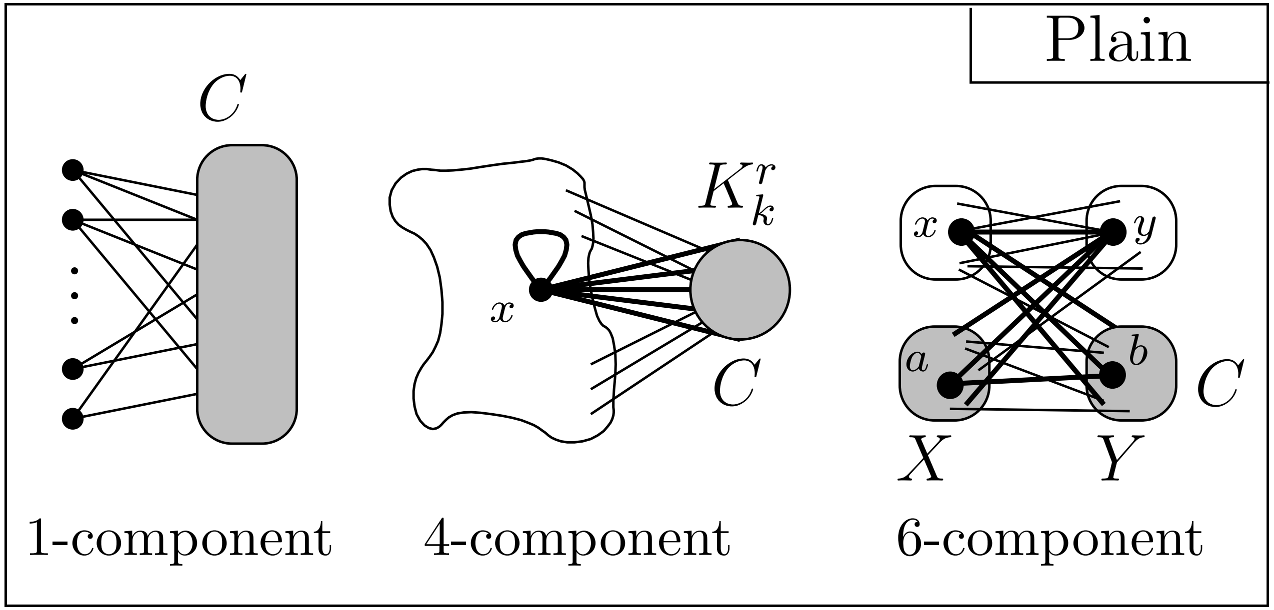


(H, C, K) is simple whenever all the connected components of H

are 1, 2 or 3-components.

We can prove that List $\#(H, C, K)$-coloring and $\#(H, C, K)$-coloring have FPT-algorithms for simple (H, C, K)

Plain (H, C, K)



We present a parameterized reduction from the (H, C, K)-coloring problem for plain (H, C, K) to the list (H, C, K)-coloring for simple (H, C, K).

The reduction is done through a series of sub-reductions that gradually transforms a plain (H, C, K) into a particular case of simple (H, C, K).

Summary of FPT results

The following problems admit FPT-algorithms:

- List #(H,C,K)-coloring and #(H,C,K)-coloring for simple (H,C,K).
- List (H,C,K)-coloring, for simple (H,C,K).
- (H,C,K)-coloring for plain (H,C,K).

The complexity of our FPT-algorithms is *linear* in n (thus efficient)

for counting problems time bound is $O(f_1(k)n + f_2(k)g + f_3(k)\log n + f_4(k))$

for decision problems time bound is $O(f_1'(k)n + f_2'(k)g + f_4'(k))$.

assuming the connected components of G are given as part of the input.

Other results

- An algorthm to enumerate list (H, C, K)-colorings for simple (H, C, K). The algorithm, after a preprocesing phase (FPT), requires linear additional time per element.

- In the case of list (H, C, K)-coloring, the hardness results can be extended to further restrictions on the list L:
  - ⋆ for any vertex v, $H[L(v)]$ is connected,
    connected list (H, C, K)-coloring
  - ⋆ $L(v)$ has either one vertex or all $V(H)$,
    one-all list (H, C, K)-coloring.

- All the results for (H, C, K)-colorings are also true for $(H, C, \leq K)$-colorings.

Open problems

- Close the complexity gap for (H, C, K)-coloring and $\#(H, C, K)$-coloring

  Requires to study properties of C and K for decision.

  Requires to understand the role of c-reducibility for counting.

Open problems

- Find a tight characterization for the (H, C, K) giving raise to FPT-algorithms.

The class of simple (H, C, K) verify that $H - C$ is a complete reflexive or a complete irreflexive bipartite. Are there any other nice properties of $H - C$?

Our hardness results show that loops in C play a especial role, as by removing them or by adding them we get cases of list (H, C, K)-coloring that are W[1]-hard. What is the exact role?

[Reed, Smith, Vetta Operation Research Letters 04] show that the k-remove for bipartite problem, the (H, C, K)-coloring problem where $H = K_{1,1} \oplus K_1^r$ and $C = V(K_1^r)$, belongs to FPT.

We conjecture that when $H = K_{x,y} \oplus K_c^r$ with $C = V(K_c^r)$ is the (H, C, K)-coloring problem belongs to FPT.