FPT ALGORITHMS, PART V:

COLOR CODING

• Recall: a *path* in a graph G is a sequence of *distinct* vertices $v_1, \ldots, v_k$ such that $v_i v_{i+1} \in E(G)$ for all $1 \leq i \leq k-1$.
We also view paths as subgraphs.

k-Path:
INSTANCE: A graph G and integer k
PARAMETER: k
QUESTION: Does G contain a path on k vertices? (a k-path)

• Recall: We had an extremely unpractical FPT algorithm based on the Excluded Grid Theorem. (Either G contains a $k \times k$-grid minor, or has tree width at most 20^{2k^5}.)

(Q) Is there a faster FPT algorithm for k-Path?

- Kernelization?
- Branching?
- A good tree width based algorithm?
- ...?

Color coding

Main idea: 'Guess' a partition of the vertices into k sets (colors), and assume there is a k-path in which all vertices receive different colors.

- A *k-color assignment* for a graph $G = (V, E)$ is a function $f : V \to \{1, \dots, k\}$.
  (Note: this does not have to be a proper *vertex coloring.)*

- Let f be a k-color assignment of G. A subgraph H of G is *colorful* if all vertices of H are colored differently. The *color set* of H is $\{f(v) : v \in V(H)\}$.

Colorful k-Path:
INSTANCE: A graph G with k-color assignment f, and integer k.
PARAMETER: k
QUESTION: Does G contain a *colorful* path of length k?

- Determining whether a *colorful* k-path exists turns out to be much easier!

- We do not lose too much by making this assumption: if we solve this problem for *enough* different k-color assignments, we can solve the original problem.

- Essential: 'enough' is a function of k!

Finding colorful k-paths

Colorful Rooted k-Path
INSTANCE: A graph G with k-color assignment f, a vertex $v \in V(G)$, and integer k.
PARAMETER: k
QUESTION: Does G contain a *colorful* path of length k, that starts in v?

Proposition

An FPT algorithm for Colorful Rooted k-Path gives an FPT algorithm for Colorful k-Path.

Proof: Let G, f, k be an instance for Colorful k-Path.
Add a new vertex v and assign it a new color $k+1$. Connect it to all other vertices by an edge. This gives the graph G'.
G' has a colorful $(k+1)$-path starting in v if and only if G has a colorful k-path. $\square$

A dynamic programming algorithm

Let G be a graph with k-color assignment f and given start vertex v.

• For $C \subseteq \{1, \ldots, k\}$ and a vertex $w \in V(G)$, $P(w, C) = 1$ if there exists a colorful path from v to w with color set C, and $P(w, C) = 0)$ otherwise.
(So such a path has $|C|$ vertices.)

Algorithm 1

For all $w \in V$ and $C \subseteq \{1, \ldots, k\}$:
    $P(w, C) = 0$
$P(v, f(v)) = 1$
For $i := 1, \ldots, k$:
    For all pairs (w, C) with $P(w, C) = 1$ and $|C| = i$:
        If w has a neighbor u with $f(u) \notin C$:
            $P(u, C \cup \{f(u)\}) = 1$
If there is a vertex u with $P(u, \{1, \ldots, k\}) = 1$ then
    return 'YES'
else
    return 'NO'

- By induction over $|C|$: Algorithm 1 sets $P(u, C) = 1$ if and only if G contains a colorful path from v to u with color set C. (The different colors ensure that the vertices are distinct.)
- Therefore Algorithm 1 returns 'YES' if and only if G contains a colorful k-path starting in v.

• Complexity: every combination of a color set C and vertex w is considered once for initialization, and once for recursion.
For every such combination the computations take polynomial time.
So the total complexity is $2^k n^{O(1)}$, with $n = |V(G)|$.
Summarizing:

Theorem

Deciding whether a graph G with k-color assignment f contains a colorful k-path starting at $v \in V(G)$ can be done in time $2^k n^{O(1)}$, with $n = |V(G)|$.

Randomized algorithms

(Q) What is the probability that by choosing an arbitrary k-color assignment and solving Colorful k-path, we also find the correct answer to k-Path on G?

Proposition

Let graph G contain a k-path P, and let f be a random *k-color assignment for G. With probability at least e^{-k}, P is colorful.*

Proof: P is colorful with probability $k!/k^k$. Using Stirling's formula $k! \approx \sqrt{2\pi k}(\frac{k}{e})^k$ we can write:

$$\frac{k!}{k^k} \approx \frac{\sqrt{2\pi k}}{e^k} \geq \frac{1}{e^k}.$$

□

Algorithm 2

Repeat the following $\lceil e^k \rceil$ times:

  choose a random k-color assignment f for G

  If G contains a colorful k-Path for color assignment f then

    return 'YES'

return 'NO'

Proposition

If G contains no k-path, Algorithm 2 returns 'NO'. If G contains a k-path, Algorithm 2 returns 'YES' with probability at least $1/2$.

Proof: The first statement is obvious.
Since the color assignments are chosen independently, the probability that a given k-path P is not colorful in any color assignment is

$$(1 - e^{-k})^{\lceil e^k \rceil} \leq e^{-1} < \frac{1}{2}.$$

• A randomized algorithm A for a decision problem is a *Monte Carlo* algorithm if

- A always returns 'NO' on NO-instances, and
- A returns 'YES' with probability at least $\frac{1}{2}$ on YES-instances.

Theorem

A Monte Carlo algorithm with complexity $(2e)^k \cdot n^{O(1)}$ exists for deciding whether a graph on n vertices contains a k-path.

Derandomization

- Let V and P be sets and $k = |P|$. A *k-perfect family of hash functions* from V to P is a family F of functions $f : V \to P$ such that for every $K \subseteq V$ with $|K| = k$, there exists an $f \in F$ for which the restriction to K is bijective.

- So if $V = V(G)$ and $P = \{1, \dots, k\}$, this is a family of color assignments of G, such that every subset $K \subseteq V$ with $|K| = k$ is colorful in at least one color assignment.

Theorem

For all $n, k \in \mathbb{N}$ there is a k-perfect family of hash functions from $\{1, \dots, n\}$ to $\{1, \dots, k\}$ of cardinality $2^{O(k)} \cdot \log^2 n$, which can be computed in time $2^{O(k)} \cdot n \cdot \log^2 n$.

Corollary

An FPT algorithm with complexity $2^{O(k)} \cdot n^{O(1)}$ exists for deciding whether a graph on n vertices contains a k-path.

Generalization: finding binary tree subgraphs

- A rooted tree T is *binary* if every non-leaf has two children.

k-Colorful Binary Tree Subgraph:
INSTANCE: A graph G with k-color assignment f, and a rooted binary tree T on k vertices.
PARAMETER: k
QUESTION: Does G contain a colorful subgraph H isomorphic to T?

- Let T be a (rooted) binary tree.
For $t \in V(T)$, $T(t)$ denotes the subtree of T induced by t and all descendants of t, with root t.
The *height* of a rooted tree is the length of a longest path starting at the root. The *height* of $t \in V(T)$ is the height of $T(t)$.

- Let T be a tree with root r, and G be a graph with $v \in V(G)$.
We say that *G contains a copy H of T rooted at v* if G contains a subgraph H such that there is an isomorphism $\phi : V(T) \to V(H)$ with $\phi(r) = v$.

- Let G be a graph with k-color assignment f, and let T be a tree on k vertices.
For every $t \in V(T)$, $u \in V(G)$ and $C \subseteq \{1, \ldots, k\}$, define $T(u, t, C) = 1$ if G contains a copy H of $T(t)$ rooted at u, such that H is colorful and has color set C, and $T(u, t, C) = 0$ otherwise.
(So if $T(u, t, C) = 1$, then $|C| = |V(T(t))|$.)

Recursion formula

Let $t \in V(T)$, $u \in V(G)$ and $C \subseteq \{1, \ldots, k\}$ with $|C| = |V(T(t))|$ and $f(u) \in C$.
Let t_1, t_2 be the children of t in T.

Proposition

$T(u, t, C) = 1$ if and only if $C \setminus \{f(u)\}$ can be partitioned into sets C_1, C_2, and neighbors u_1, u_2 of u can be identified, such that $T(u_i, t_i, C_i) = 1$ for $i = 1, 2$.

- The above recursion formula computes $T(u, t, C)$ using values (T, u_i, t_i, C_i) where the height of $T(t_i)$ is less than the height of $T(t)$.

Algorithm 3

INPUT: a graph G with k-color assignment f, and a binary tree T on k vertices with root r.

For all $v \in V(G)$, $t \in V(T)$ and $C \subseteq \{1, \ldots, k\}$:
  $T(v, t, C) = 0$
For all $v \in V(G)$ and leaves t of T:
  $T(v, t, f(v)) = 1$.
For $h = 1, \ldots, k$:
  For all $t \in V(T)$ of height h:
    For all $C \subseteq \{1, \ldots, k\}$ with $|C| = |V(T(t))|$:
      For all $v \in V(G)$ with $f(v) \in C$:
        If we can identify neighbors v_1 and v_2 of v, and partition $C \setminus \{f(v)\}$ into C_1 and C_2 as stated above,
          then: $T(v, t, C) = 1$.
If there is a vertex $v \in V(G)$ with $T(v, r, \{1, \ldots, k\}) = 1$ then
  return 'YES'
else
  return 'NO'

- Algorithm 3 correctly answers whether G contains a colorful copy of T.

- Complexity: $k \cdot k \cdot 2^k \cdot n \cdot n^2 \cdot 2^k \cdot n^{O(1)} \in 4^k \cdot n^{O(1)}$, where $n = |V(G)|$.

(At most:
k choices of h,
k choices of t,
2^k choices of C,
n choices of v,
n^2 choices of v_1 and v_2, and
2^k choices of C_1 and C_2.)

By combining this with a k-perfect family of k-color assignments:

Theorem

In time $2^{O(k)} \cdot n^{O(1)}$ we can decide whether a graph G on n vertices contains a given binary tree T on k vertices as subgraph.

Finding subgraphs of small tree width

k-Colorful Subgraph Isomorphism:
INSTANCE: A graph G with k-color assignment f, and graph H on k vertices.
PARAMETER: k
QUESTION: Does G have a colorful subgraph G' isomorphic to H?

Goal: an algorithm with complexity $n^{w+O(1)} \cdot 2^{O(k)}$, where $n = |V(G)|$ and w is the tree width of H.

- The problem of deciding whether a graph G contains a complete graph on k-vertices is unlikely to admit an FPT algorithm for parameter k (it is $W[1]$-hard), so we cannot hope to remove w from the exponent.

Let (T, X) be a nice tree decomposition of a graph H.
• For $v \in V(T)$, define $H(v)$ as before (the subgraph of H induced by the union of X_v and the sets X_w for all descendants w of v.)

Let $v \in V(T)$, let $\psi : X_v \to V(G)$, and let $C \subseteq \{1, \ldots, k\}$.
• We define $S(v, \psi, C) = 1$ if there exists an isomorphism ϕ from $H(v)$ to a subgraph G' of G such that
- G' is colorful with color set C, and
- for all $x \in X_v$, $\phi(x) = \psi(x)$,
and $S(v, \psi, C) = 0$ otherwise.

Proposition (Introduce)

Let u be an introduce node of T with child v, with $X_u \setminus X_v = \{x\}$.
Then $S(u, \psi, C) = 1$ iff
- $S(v, \psi', C') = 1$, where ψ' is the restriction of ψ to X_v and $C' = C \setminus \{f(\psi(x))\}$, and
- for all $y \in X_u$, if $xy \in E(H)$ then $\psi(x)\psi(y) \in E(G)$.

Proposition (Forget)

Let u be a forget node of T with child v, with $X_v \setminus X_u = \{x\}$. Then $S(u, \psi, C) = 1$ iff there exists a $\psi' : X_v \to V(G)$ such that ψ is its restriction to X_u, and $S(v, \psi', C) = 1$.

Proposition (Join)

Let u be a join node of T with children v and w.
Then $S(u, \psi, C) = 1$ iff there exist C_v and C_w with $C_v \cup C_w = C$, $C_v \cap C_w = X_u$ such that $S(v, \psi, C_v) = 1$ and $S(w, \psi, C_w) = 1$.

- With the above recursion formulas, we can compute $S(r, \psi, \{1, \dots, k\})$ for all ψ.
- G contains a colorful copy of H if $S(r, \psi, \{1, \dots, k\}) = 1$ for some ψ.

Theorem

In time $n^{w+O(1)} \cdot 2^{O(k)}$ we can decide whether a graph G on n vertices with k-color assignment f contains a colorful copy of a graph H on k vertices for which a tree decomposition of width w is given.

- Exercise: work out the details of the complexity bound.

Theorem

In time $n^{w+O(1)} \cdot 2^{O(k)}$ we can decide whether a graph G on n vertices contains a copy of a graph H on k vertices for which a tree decomposition of width w is given.

Color coding - Summary

- Color coding is a useful technique for solving various subgraph problems; fixing a coloring makes dynamic programming possible.

- This method can be generalized to finding homomorphisms between general relational structures.

- Similar to iterative compression, color coding shows that it may be useful to 'guess' properties of a solution.
This gives an FPT algorithm for parameter k provided that we only need to consider $f(k)$ different guesses to solve the original problem.

- Giving a randomized (Monte Carlo) algorithm may be easier than giving an algorithm that is always correct.

Techniques for FPT algorithms

- Kernelization

Vertex Cover, Max Sat, d-Hitting Set, Max Leaves Spanning Tree

- Branching / bounded search trees

Vertex Cover, Vertex Coloring, Skew Separator

- Iterative compression

Vertex Cover, (Directed) Feedback Vertex Set

- Tree width

Dynamic Programming: Vertex Coloring, Vertex Cover
Tree width based algorithms: Vertex Cover, FVS,
$(n-k)$-Dominating Set, Max Leaves Spanning Tree, k-Path,
Planar Vertex Cover/Independent Set/Dominating Set

- Color coding

k-Path, Binary Tree Subgraph, Subgraph Isomorphism