# 29. Randomized Parametrized Algorithms (and Derandomization)

**Approach:** To solve a parametrized problem,
first design a <u>randomized</u> algorithm,
then <u>derandomize</u> it.
For the latter, apply standard techniques as a black-box.

**Domain:** Works best when we have to ensure
disjointness for a small number of things.

Illustrate on

**$k$-PATH:**

**Given:** Graph $G$, vertices $s, t \in V(G)$.
**Parameter:** $k \in \mathbb{N}$.
**Question:** Is there a simple path (does not repeat vertices)
from $s$ to $t$ with $k$ internal vertices.

**Note:** The problem can be used to encode HAMILTONIAN.

The approach works
in

**3 Steps:**

1.) <u>Color coding</u>: Use randomization to give additional structure to the problem.

2.) (Repeatedly) solve the enriched problem.

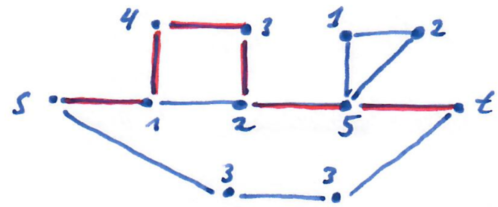
3.) Derandomize the algorithm.

## 29.1 Step 1: Color Coding

**Additional structure:** Assign colors from $[1, k]$
to the vertices $V(G) \setminus \{s, t\}$
uniformly and independently at random.

Example:

$k = 5$




The enriched problem is the following.

COLORFUL $k$-PATH:

Given: Graph $G$, $s, t \in V(G)$, and a coloring from $[1, k]$ to $V(G) \setminus \{s, t\}$.

Question: Is there a path from $s$ to $t$ where each color appears exactly once on the internal vertices.

Lemma (Relationship between the problems):

$k$-PATH holds for $(G, s, t)$ iff there is a coloring $c$ such that COLORFUL $k$-PATH holds for $(G, s, t, c)$.

Approach: Try out different colorings and look for a colorful $k$-path.

Key: $\boxed{\text{What is the probability that a coloring leads to success?}}$

Analysis:
- Consider the $k$-path between $s$ and $t$
- There are $k!$ good colorings of the internal vertices.
- There are $k^k$ colorings in total.

Hence, the probability that a coloring leads to a colorful $k$-path is

$$\frac{k!}{k^k} \underset{(*)}{>} \frac{\left(\frac{k}{e}\right)^k}{k^k} = \left(\frac{k}{e}\cdot\frac{1}{k}\right)^k = \frac{1}{e^k}.$$

(*) Here, we use $e\cdot\left(\frac{k}{e}\right)^k \leq k!$

Hence, the algorithm will output yes in case of a yes instance with probability at least $\frac{1}{e^k}$.

To estimate the probability of failure despite repeated tries, we use the following.

**Lemma (Basic bounds):**

For $x \geq 2$, $\frac{1}{4} \leq \left(1-\frac{1}{x}\right)^x \leq \frac{1}{e}$.

**Corollary:**

If the probability of success is $\geq p$ (assume wlog. $p \leq \frac{1}{2}$) then the probability that the algorithm does not say yes after $\frac{1}{p}$ repetitions is

$$(1-p)^{\frac{1}{p}} \underset{\text{(Basic bounds)}}{<} \frac{1}{e} \approx 0{,}38.$$

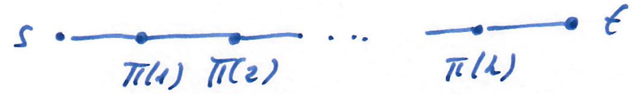
- We can thus repeat the search for a colorful $k$-path for a constant number of tries and by this make the error probability an arbitrary small constant.
  (The number of tries still depends on the required probability.)
- In the concrete case of $k$-path, we have $p = \frac{1}{e^k}$.
  Try, say, $100\cdot e^k$ random colorings.
  This gives a probability of a wrong answer of
  $$\left(1-\frac{1}{e^k}\right)^{100\cdot e^k} \underset{\text{(Corollary)}}{\leq} \left(\frac{1}{e}\right)^{100} = \frac{1}{e^{100}}.$$

## 29.2 Step 2: Solving COLORFUL k-PATH

Goal: Present two methods.

### Method 1: Try All Permutations

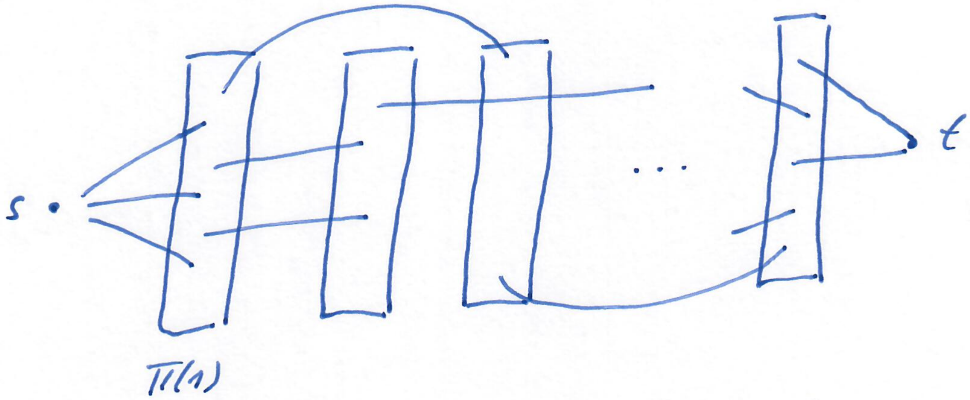
Idea: • The colors encountered on a colorful k-path form a permutation $\pi$ of $\{1, \ldots, k\}$.




• Try all $k!$ permutations.

Algorithm: • For a given permutation $\pi$, check whether there is a path with this order of colors.

Setting




↳ Remove edges connecting non-adjacent colors.

↳ Interprete the remaining edges as being directed.

↳ Check for a directed s-t-path.

↳ Can be done in $O(|E(G)|)$.

The running time is $\boxed{O(k! \cdot |E(G)|)}$.

### Method 2: Dynamic Programming

Introduce $2^k \cdot |V(G)|$ Boolean variables $T[C, v]$ with $C \subseteq [1, k]$ and $v \in V(G)$ so that

$T[C,v] = \text{true}$ iff $\exists$ an $s$-$v$-path where
each color in $C$ appears exactly once
and no other color appears.

<u>Base case:</u> $T[\emptyset, s] = \text{true}$

$T[\emptyset, v] = \text{false}$. for all $v \neq s$.

<u>Recurrence:</u> $T[C,v] = \bigvee_{\{u,v\} \in E(G)} T[C \setminus \{r\}, u]$, where $r = \text{color}(v)$.

- If we know every $T[C,v]$ with $|C| = i$,
then we can determine every $T[C,v]$ with $|C| = i+1$.
All values can be computed in $\boxed{O(2^k \cdot |V(G)| \cdot |E(G)|)}$.

<u>Lemma:</u>
There is a colorful $k$-path between $s$ and $t$
iff $T[[1,k], v] = \text{true}$ for some neighbor $v$ of $t$.

<u>Theorem:</u>
Using Method 2, we obtain an $O^*((2e)^k)$-time algorithm
with constant error probability.

$e^k$ tries
$2^k$-time per try

## <u>29.3 Step 3: Derandomization</u>

<u>Idea:</u> Instead of trying $O(e^k)$ random colorings,
only try particularly good ones.
The particularly good colorings need to have the property
that they contain a solution, provided one exists.

<u>Hope:</u> There are less of the particularly good colorings
and we can determine them efficiently.

**Definition:**

A family of functions $\mathcal{H}$ from $[1,k]^{[1,n]}$

is a <u>k-perfect</u> family of hash functions,

if for every $S \subseteq [1,n]$ with $|S| = k$

there is an $h \in \mathcal{H}$ such that $h(x) \neq h(y)$ for all $x, y \in S$.

// Every k-element set has a function $h \in \mathcal{H}$
that makes it colorful.

**Approach:** • Do not go through random colorings.

go through a k-perfect family $\mathcal{H}$ of functions from $[1,k]^{V(G)}$.

If there is a solution $S$ (these are the internal vertices),

then there is $h \in \mathcal{H}$ making the vertices in $S$ colorful.

Hence, the above algorithm returns yes as required.

**Analysis:** Let $t$ be the time it takes

to construct a k-perfect family $\mathcal{H}$

of functions from $[1,k]^{V(G)}$.

Then

k-path can be solved in $O^*(t + |\mathcal{H}| \cdot 2^k)$.

**Theorem (Schmidt & Siegel 1990, Alon, Yuster, Zwick 1995, Black-box for us):**

There is a k-perfect family of hashing functions from $[1,k]^{[1,n]}$

having size $2^{O(k)} \cdot \log n$ that can be constructed

in time polynomial in the size of the family.

**Theorem:**

There is a deterministic $2^{O(k)} \cdot \mathrm{poly}(\text{input})$-time algorithm

solving k-PATH.

## 29.4 Induced Subgraph Isomorphism

Goal: · Practice color coding once more

· Introduce another hashing/coloring lemma.

**ISGI**

Given: Graphs $G, H$.

Parameter: Degree of $G$, $\deg(G)$,

number of vertices of $H$, $|V(H)|$.

Question: Does $G$ contain $H$ as an induced subgraph?

(edge in $G$ iff edge in $H$)

Note: · Encodes $k$-CLIQUE, so FPT by parameterizing only by $k$ is unlikely (more on FPT-hardness later on).

· Naive algorithm: $|V(G)|^{O(|V(H)|)}$.

Goal: $\deg(G)^{O(|V(H)|)}$.

In what follows, we assume $H$ is connected.

Step 1: Color Coding

· Color the vertices of $G$: ↳ red with probability $p$

↳ blue with probability $1-p$.

· Delete all blue vertices.

· Determine whether any red connected component is equal to $H$. (isomorphic)

// This is the enriched problem.

Success probability:

· If $G$ does not contain $H$, the algorithm always says no.

· If $G$ contains $H$,

↳ all vertices of $H$ are colored red

with probability $p^{|V(H)|}$

$\hookrightarrow$ all neighboring vertices of $H$ in $G$ are colored blue with probability $(1-p)^{\deg(G)\cdot|V(H)|}$.

- The success probability is thus

$$p^{|V(H)|}\cdot(1-p)^{\deg(G)\cdot|V(H)|}$$

$$\left(\text{Set } p=\frac{1}{\deg(G)}\right) = \left(\frac{1}{\deg(G)}\right)^{|V(H)|}\cdot\left(1-\frac{1}{\deg(G)}\right)^{\deg(G)\cdot|V(H)|}$$

$$\left(\begin{array}{c}\text{Lemma on}\\ \text{basic bounds}\end{array}\right) \geq \frac{1}{\deg(G)^{|V(H)|}}\cdot\left(\frac{1}{4}\right)^{|V(H)|}$$

$$= \frac{1}{(4\cdot\deg(G))^{|V(H)|}}$$

<u>Step 2: Solving Graph Isomorphism</u>

- Can be done (this is a recent breakthrough by Babai '2015) in time $2^{\text{polylog}(k)}$.
- For a constant success rate, we repeat the test $(4\cdot\deg(G))^{|V(H)|}$ - times.
- Yields a total runtime of $(4\cdot\deg(G))^{|V(H)|+o(|V(H)|)}$.

<u>Step 3: Derandomization</u>

Let $\mathcal{F}=\{c_1,\dots,c_t\}$ be a family of colorings from $\{0,1\}^{V(G)}$.

Then $\mathcal{F}$ is a <u>$k$-universal coloring family</u>, if

for every set of at most $k$ vertices

and for every way of coloring $S$

there is $c_i\in\mathcal{F}$ that colors $S$ like that.

**Lemma (Siegel & Schmidt 1990):**

One can construct a $h$-universal coloring family $\mathcal{F}$ of size $2^{O(h)} \cdot \log n$ in time polynomial in the size of the family.

To use this for induced subgraph isomorphism:

$h = (\deg(G)+1) \cdot |V(H)|$.

**Theorem:**

ISGI can be solved in time $2^{O(\deg(G) \cdot |V(H)|)}$.

This can be improved to $\deg(G)^{|V(H)|}$.