Finding diverse solutions of combinatorial problems

Changki Yun

Dept. of Mathematical Science
Seoul National University

Sep 25, 2023

Table of Contents

Table of Contents

Beyond the "optimization"

Classic combinatorial optimization problem:

- Given a finite set $[n]$ and a **structure** $\Pi \subseteq 2^{[n]}$, and weight $w : [n] \to \mathbb{R}$, find $X \in \Pi$ such that $w(X) = \sum_{i \in X} w(i)$ is maximized.

Beyond the "optimization"

Classic combinatorial optimization problem:

- Given a finite set $[n]$ and a **structure** $\Pi \subseteq 2^{[n]}$, and weight $w : [n] \to \mathbb{R}$, find $X \in \Pi$ such that $w(X) = \sum_{i \in X} w(i)$ is maximized.
- How can we find r "multiple" solutions $X_1, \cdots, X_r$?

Beyond the "optimization"

Classic combinatorial optimization problem:

- Given a finite set $[n]$ and a **structure** $\Pi \subseteq 2^{[n]}$, and weight $w : [n] \to \mathbb{R}$, find $X \in \Pi$ such that $w(X) = \sum_{i \in X} w(i)$ is maximized.
- How can we find r "multiple" solutions $X_1, \cdots, X_r$?
- Easily we can find some "top r" solutions.
  - top r shortest paths, matchings, and spanning trees

Similarity from topmost solutions

- However, topmost solutions usually suffer from mutual similarities.
  - e.g. r best spanning trees are mostly spanned from a few edge swaps.

Similarity from topmost solutions

- However, topmost solutions usually suffer from mutual similarities.
  - e.g. r best spanning trees are mostly spanned from a few edge swaps.
- To cover many heuristics and real-world problems, we might need some "diverse" solutions.

Similarity from topmost solutions

- However, topmost solutions usually suffer from mutual similarities.
  - e.g. r best spanning trees are mostly spanned from a few edge swaps.
- To cover many heuristics and real-world problems, we might need some "diverse" solutions.
- We may think of "colorful" solutions – a number of mutually disjoint solutions.

Similarity from topmost solutions

- However, topmost solutions usually suffer from mutual similarities.
  - e.g. r best spanning trees are mostly spanned from a few edge swaps.
- To cover many heuristics and real-world problems, we might need some "diverse" solutions.
- We may think of "colorful" solutions – a number of mutually disjoint solutions.
- They are mostly NP-hard but **FPT** at the same time, and hard to extract out moderate number of solutions.

Similarity from topmost solutions

- However, topmost solutions usually suffer from mutual similarities.
  - e.g. r best spanning trees are mostly spanned from a few edge swaps.
- To cover many heuristics and real-world problems, we might need some "diverse" solutions.
- We may think of "colorful" solutions – a number of mutually disjoint solutions.
- They are mostly NP-hard but **FPT** at the same time, and hard to extract out moderate number of solutions.
- Combining those concepts, we devise a framework to find out moderate number of **diverse** solutions.

Diversity measure

We may think of a few diversity measure, universal for all set notations.

- $d_{\mathrm{sum}}(X_1, \cdots, X_r) := \sum_{1 \le i < j \le r} |X_i \setminus X_j| + |X_j \setminus X_i|$
- $d_{\min}(X_1, \cdots, X_r) := \min_{1 \le i < j \le r} |X_i \setminus X_j| + |X_j \setminus X_i|$
- $d_{\mathrm{cov}}(X_1, \cdots, X_r) := \left| \bigcup_{i=1}^{r} X_i \right|$

Mostly they are from the Hamming distance.

Diverse solutions

We review the papers named:

- *Hanaka et al, Finding Diverse Trees, Paths, and More (AAAI 2021)*
- *Hanaka et al, A Framework to Design Approximation Algorithms for Finding Diverse Solutions in Combinatorial Problems*

that is related to optimizing d_{sum} and d_{min} in various combinatorial problems.

Table of Contents

Diverse spanning trees

We face our first problem:

Diverse spanning tree

Given a graph $G = (V, E)$, find r (not necessarily distinct) spanning trees $T_1, \cdots, T_r$ maximizing $d_{\text{sum}}(E(T_1), \cdots, E(T_r))$.

We show that this is solvable in polynomial time.

Diverse spanning trees

As a subroutine, we employ that:

Colorful spanning trees

Given an edge-weighted graph $G = (V, E, w)$, we can find r edge-disjoint spanning trees $T_1, \cdots, T_r$ minimizing $w(T_1) + \cdots + w(T_r)$ if exists, or prove that there cannot be r edge-disjoint spanning trees.

Diverse spanning trees

Note that

$$d_{\mathrm{sum}}(E(T_1), \cdots, E(T_r)) = 2r(|V(G)| - 1) - 2 \sum_{1 \le i < j \le r} |E(T_i) \cap E(T_j)|$$

thus we're suppose to minimize

$$\sum_{1 \le i < j \le r} |E(T_i) \cap E(T_j)| = \sum_{e \in E(G)} \binom{\mathrm{cnt}(e)}{2}$$

where $\mathrm{cnt}(e) := |\{T_i \mid e \in T_i\}|$ is the occurrence of e among $T_1, \cdots, T_r$.

Diverse spanning trees

Now, we take r copies of every edges. An edge e will have r replicas $e_1, \cdots, e_r$, with each weight $w(e_i) = i - 1$. Renaming the multigraph as G', Now we can show that:

Diverse spanning trees

$$\min_{1 \le i < j \le r} |E(T_i) \cap E(T_j)| = \min_{G'} w\,(\text{edge-disjoint } r \text{ spanning trees})\,.$$

Thus, **diverse spanning trees** is solvable in polynomial time.

Diverse matroid bases

Now we can reveal the hidden language of matroids.

Matroid partition

Given a matroid $M = (E, \mathcal{I})$, with weight $w : E \to \mathbb{R}$, we can find r disjoint bases $B_1, \cdots, B_r$ minimizing $w(B_1) + \cdots + w(B_r)$ if exists, or prove there is no r disjoint bases in polynomial time.

Diverse matroid bases

Given a matroid $M = (E, \mathcal{I})$, we can find r bases $B_1, \cdots, B_r$ maximizing $d_{\text{sum}}(B_1, \cdots, B_r)$. In addition, we can solve the weight-minimizing solution if the original problem is weighted.

Finding k-path in a graph

- All we know that, finding a path of given length in a graph is NP-complete.

Finding k-path in a graph

- All we know that, finding a path of given length in a graph is NP-complete.
- However, what about if we can take the length k as a fixed parameter?

Finding k-path in a graph

- All we know that, finding a path of given length in a graph is NP-complete.
- However, what about if we can take the length k as a fixed parameter?
- There is a well-known $\mathcal{O}(k2^k(V+E))$ randomized algorithm with a certain success probability. (Alon & Zwick, 1995)

Color coding (Alon & Zwick, 1995)

- Randomly choose a "color" from $[k]$ and assign for each vertex.
- Run a dynamic programming with table definition
  - `bool` $\mathrm{dp}(C, v)$: If we can end on vertex v, going through vertices with color set $C \subseteq [k]$, keeping colors of visited vertices distinct.

If a k-path exists, we will assign distinct colors on the path with probability $\frac{k!}{k^k} \geq e^{-k}$.

Color coding (Alon & Zwick, 1995)

- Randomly choose a "color" from $[k]$ and assign for each vertex.
- Run a dynamic programming with table definition
  - `bool` $\mathrm{dp}(C, v)$: If we can end on vertex v, going through vertices with color set $C \subseteq [k]$, keeping colors of visited vertices distinct.

If a k-path exists, we will assign distinct colors on the path with probability $\frac{k!}{k^k} \geq e^{-k}$.
Thus we can devise an $\mathcal{O}((2e)^k(V+E))$ randomized algorithm to find a k-path if exists.

FPT

Note that the complexity $\mathcal{O}((2e)^k(V+E))$ is linear to $V+E$ once we assume k is a fixed constant.

FPT

Note that the complexity $\mathcal{O}((2e)^k(V+E))$ is linear to $V+E$ once we assume k is a fixed constant. The naive algorithm would use $\mathcal{O}(V^k)$ complexity in contrast.

FPT

Note that the complexity $\mathcal{O}((2e)^k(V+E))$ is linear to $V+E$ once we assume k is a fixed constant. The naive algorithm would use $\mathcal{O}(V^k)$ complexity in contrast.

Fixed-Parameter Tractibility

If a problem with input size n with parameter k, is solvable in $f(k)n^{\mathcal{O}(1)}$ time for any computable function f, then we call the problem is **FPT (Fixed-Parameter Tractible)**.

Color coding derandomized

- Back to our own problem, we note that we can derandomize this technique by introducing k-*perfect hash family*.

Color coding derandomized

- Back to our own problem, we note that we can derandomize this technique by introducing k-*perfect hash family*.
- Those are the family $\mathcal{H}$ of hash function $h : U \to [k]$, where U is the given domain.
- For every subset $X \subseteq U$ with $|X| = k$, there exists a function $h \in \mathcal{H}$ such that $|h(X)| = k$.

Color coding derandomized

- Back to our own problem, we note that we can derandomize this technique by introducing k-*perfect hash family*.
- Those are the family $\mathcal{H}$ of hash function $h : U \to [k]$, where U is the given domain.
- For every subset $X \subseteq U$ with $|X| = k$, there exists a function $h \in \mathcal{H}$ such that $|h(X)| = k$.
- It is known that there is a k-perfect hash family of size $\mathcal{T}(k, |U|) := e^k k^{\mathcal{O}(\log k)} \log |U|$ constructible in time $|U| \cdot \mathcal{T}(k, |U|)$.

Color coding derandomized

- Back to our own problem, we note that we can derandomize this technique by introducing k-*perfect hash family*.
- Those are the family $\mathcal{H}$ of hash function $h : U \to [k]$, where U is the given domain.
- For every subset $X \subseteq U$ with $|X| = k$, there exists a function $h \in \mathcal{H}$ such that $|h(X)| = k$.
- It is known that there is a k-perfect hash family of size $\mathcal{T}(k, |U|) := e^k k^{\mathcal{O}(\log k)} \log |U|$ constructible in time $|U| \cdot \mathcal{T}(k, |U|)$.
- Now we can confirm there's a deterministic algorithm that solving k-path problem in $\mathcal{T}(k, V) \cdot k2^k(V + E)$ time.

Diverse k-paths from color coding

- Now we can extend this technique to find r diverse k-paths.

Diverse k-paths from color coding

- Now we can extend this technique to find r diverse k-paths.
- We increase the number of colors to kr.

Diverse k-paths from color coding

- Now we can extend this technique to find r diverse k-paths.
- We increase the number of colors to kr.
- Suppose there are r different color sets $C_1, \cdots, C_r \subseteq [kr]$ with $|C_i| = k$, corresponding to k-paths $P_1, \cdots, P_r$ where vertices of P_i has all the colors of C_i.

Diverse k-paths from color coding

- Now we can extend this technique to find r diverse k-paths.
- We increase the number of colors to kr.
- Suppose there are r different color sets $C_1, \cdots, C_r \subseteq [kr]$ with $|C_i| = k$, corresponding to k-paths $P_1, \cdots, P_r$ where vertices of P_i has all the colors of C_i.
- Obviously, $d_*(V(P_1), \cdots, V(P_r)) \geq d_*(C_1, \cdots, C_r)$ no matter which d_* we choose.

Diverse k-paths from color coding

- Now we can extend this technique to find r diverse k-paths.
- We increase the number of colors to kr.
- Suppose there are r different color sets $C_1, \cdots, C_r \subseteq [kr]$ with $|C_i| = k$, corresponding to k-paths $P_1, \cdots, P_r$ where vertices of P_i has all the colors of C_i.
- Obviously, $d_*(V(P_1), \cdots, V(P_r)) \geq d_*(C_1, \cdots, C_r)$ no matter which d_* we choose.
- Both sides reaches to the same optimum once the desired answer $\mathcal{P}$ receives all distinct colors.

Diverse k-paths from color coding: analysis

We can achieve the equality condition with

- Coloring from a kr-perfect hash family
- For each coloring, iterating through all r-tuples of feasible colors

Thus consuming about $\mathcal{T}(kr, V) \cdot \left(\binom{kr}{k} \cdot k(V+E) + \binom{kr}{k}^r \cdot kr^2 \right)$

Diverse k-paths from color coding: analysis

We can achieve the equality condition with

- Coloring from a kr-perfect hash family
- For each coloring, iterating through all r-tuples of feasible colors

Thus consuming about $\mathcal{T}(kr, V) \cdot \left(\binom{kr}{k} \cdot k(V+E) + \binom{kr}{k}^r \cdot kr^2 \right)$
Which definitely achieves FPT.

The Framework

If we are capable to find a k-colorful variation of a structure Π in $f(k) \cdot n^{\mathcal{O}(1)}$ time, we can extend this into $g(k, r) \cdot n^{\mathcal{O}(1)}$ algorithm to find r diverse solutions.

The Framework

If we are capable to find a k-colorful variation of a structure Π in $f(k) \cdot n^{\mathcal{O}(1)}$ time, we can extend this into $g(k, r) \cdot n^{\mathcal{O}(1)}$ algorithm to find r diverse solutions.

- **Colorful Matching:** $1.618^k n^{\mathcal{O}(1)}$. *(Gupta et al, 2019)*

The Framework

If we are capable to find a k-colorful variation of a structure Π in $f(k) \cdot n^{\mathcal{O}(1)}$ time, we can extend this into $g(k, r) \cdot n^{\mathcal{O}(1)}$ algorithm to find r diverse solutions.

- **Colorful Matching:** $1.618^k n^{\mathcal{O}(1)}$. *(Gupta et al, 2019)*
- **Colorful Interval scheduling:** $2^k n^{\mathcal{O}(1)}$. *(Halldórsson et al, 2006)*

The Framework

If we are capable to find a k-colorful variation of a structure Π in $f(k) \cdot n^{\mathcal{O}(1)}$ time, we can extend this into $g(k, r) \cdot n^{\mathcal{O}(1)}$ algorithm to find r diverse solutions.

- **Colorful Matching:** $1.618^k n^{\mathcal{O}(1)}$. *(Gupta et al, 2019)*
- **Colorful Interval scheduling:** $2^k n^{\mathcal{O}(1)}$. *(Halldórsson et al, 2006)*
- **Bounded Treewidth Subgraph Isomorphism.**

Table of Contents

Approximately Diverse?

Finding maximum diversity solutions is always harder than finding disjoint solutions. Mostly we hit the NP-complete from the latter one.

Approximately Diverse?

Finding maximum diversity solutions is always harder than finding disjoint solutions. Mostly we hit the NP-complete from the latter one.
We can anticipate the "approximation" of diversity in polynomial time.

Approximately Diverse?

Finding maximum diversity solutions is always harder than finding disjoint solutions. Mostly we hit the NP-complete from the latter one.
We can anticipate the "approximation" of diversity in polynomial time.
This part is highly contextual, so I'll quickly skim through.

Approximate diversity

Now we suppose the weighted variant of d_{sum}. Given a ground set E and the cost $c : E \to \mathbb{R}_{\geq 0}$, define $d_c(X, Y) := \sum_{e \in X \triangle Y} c(e)$, and naturally extend to $d_{c-\mathrm{sum}}(X_1, \cdots, X_r) := \sum_{1 \leq i < j \leq r} d_c(X_i, X_j)$.

Approximate diversity

Now we suppose the weighted variant of d_{sum}. Given a ground set E and the cost $c : E \to \mathbb{R}_{\geq 0}$, define $d_c(X, Y) := \sum_{e \in X \triangle Y} c(e)$, and naturally extend to $d_{c-\mathrm{sum}}(X_1, \cdots, X_r) := \sum_{1 \leq i < j \leq r} d_c(X_i, X_j)$.
Note that the further discussions do not work well with the other diversity metrics.

The Machinery

Here's the machinery for a general MAX-SUM-DIVERSIFICATION. Given a universe $\mathcal{A} \subseteq 2^E$ and we have to find $\mathcal{B} = \{B_1, \cdots, B_r\} \subseteq \mathcal{A}$ such that $d_{c-\text{sum}}(B_1, \cdots, B_r)$ is maximized.

The Machinery

Here's the machinery for a general MAX-SUM-DIVERSIFICATION. Given a universe $\mathcal{A} \subseteq 2^E$ and we have to find $\mathcal{B} = \{B_1, \cdots, B_r\} \subseteq \mathcal{A}$ such that $d_{c-\mathrm{sum}}(B_1, \cdots, B_r)$ is maximized.

Local Search Algorithm

Initialize $\mathcal{B}$ as an arbitrary r element of $\mathcal{A}$, and *find $B \in \mathcal{B}$, $C \in \mathcal{A} \setminus \mathcal{B}$ such that $d_{c-\mathrm{sum}}(\mathcal{B} - B + C)$ is maximized. If the metric grown, $\mathcal{B} \leftarrow \mathcal{B} - B + C$.*

The Machinery

Here's the machinery for a general MAX-SUM-DIVERSIFICATION. Given a universe $\mathcal{A} \subseteq 2^E$ and we have to find $\mathcal{B} = \{B_1, \cdots, B_r\} \subseteq \mathcal{A}$ such that $d_{c-\mathrm{sum}}(B_1, \cdots, B_r)$ is maximized.

Local Search Algorithm

Initialize $\mathcal{B}$ as an arbitrary r element of $\mathcal{A}$, and *find $B \in \mathcal{B}$, $C \in \mathcal{A} \setminus \mathcal{B}$ such that $d_{c-\mathrm{sum}}(\mathcal{B} - B + C)$ is maximized. If the metric grown, $\mathcal{B} \leftarrow \mathcal{B} - B + C$.*

Theorem (Cevallos, 2016)

Iterating the blue part $\mathcal{O}(r \log r)$ times gives an $(1 - \frac{2}{r})$ approximation for MAX-SUM-DIVERSIFICATION. Furthermore, this gives a PTAS for the same problem.

Top r oracles to the diversity maximization

In this case, the only thing matters is that the size of $\mathcal{A}$ could be exponential to the ground set size $|E|$. Once we decide the exile B, we need to find such $C \in \mathcal{A} \setminus \mathcal{B}$ maximizing $d_{c-\mathrm{sum}}(\mathcal{B} - B + C)$ in polynomial time.

Top r oracles to the diversity maximization

In this case, the only thing matters is that the size of $\mathcal{A}$ could be exponential to the ground set size $|E|$. Once we decide the exile B, we need to find such $C \in \mathcal{A} \setminus \mathcal{B}$ maximizing $d_{c-\mathrm{sum}}(\mathcal{B} - B + C)$ in polynomial time.

- Define $\mathcal{B}' := \mathcal{B} - B$, and $\mathrm{In}(e) := |\{X \in \mathcal{B}' : e \in X\}|, \mathrm{Ex}(e) := |\{X \in \mathcal{B}' : e \notin X\}|$.

Top r oracles to the diversity maximization

In this case, the only thing matters is that the size of $\mathcal{A}$ could be exponential to the ground set size $|E|$. Once we decide the exile B, we need to find such $C \in \mathcal{A} \setminus \mathcal{B}$ maximizing $d_{c-\mathrm{sum}}(\mathcal{B} - B + C)$ in polynomial time.

- Define $\mathcal{B}' := \mathcal{B} - B$, and $\mathrm{In}(e) := |\{X \in \mathcal{B}' : e \in X\}|, \mathrm{Ex}(e) := |\{X \in \mathcal{B}' : e \notin X\}|$.
- Then we need to choose C such that $\sum_{e \in C} c(e)\mathrm{Ex}(e) + \sum_{e \notin C} c(e)\mathrm{In}(e) = \sum_{e \in C} c(e)\left(\mathrm{Ex}(e) - \mathrm{In}(e)\right) + \sum_e c(e)\mathrm{In}(e)$ is maximized.

Top r oracles to the diversity maximization

In this case, the only thing matters is that the size of $\mathcal{A}$ could be exponential to the ground set size $|E|$. Once we decide the exile B, we need to find such $C \in \mathcal{A} \setminus \mathcal{B}$ maximizing $d_{c-\mathrm{sum}}(\mathcal{B} - B + C)$ in polynomial time.

- Define $\mathcal{B}' := \mathcal{B} - B$, and $\mathrm{In}(e) := |\{X \in \mathcal{B}' : e \in X\}|, \mathrm{Ex}(e) := |\{X \in \mathcal{B}' : e \notin X\}|$.
- Then we need to choose C such that $\sum_{e \in C} c(e)\mathrm{Ex}(e) + \sum_{e \notin C} c(e)\mathrm{In}(e) = \sum_{e \in C} c(e)\left(\mathrm{Ex}(e) - \mathrm{In}(e)\right) + \sum_e c(e)\mathrm{In}(e)$ is maximized.
- The term only relies on e and $\mathcal{B}'$, so we find C such that maximizing the new weight $c'(e) := c(e)(\mathrm{Ex}(e) - \mathrm{In}(e))$. If we choose topmost $r+1$ elements along c', we have at least one element belonging to $\mathcal{A} \setminus \mathcal{B}$.

PTAS for diversity approximation

So if we have top-k enumeration algorithm, we have PTAS for diversity maximization.

- (s, t)-path: enumeration algorithm is given by the famous paper of Eppstein.
- Common bases of two matroids
- Minimum cuts
- Matchings

References & Further readings

- *Tesshu Hanaka, Masashi Kiyomi, Yasuaki Kobayashi, Yusuke Kobayashi, Kazuhiro Kurita, Yota Otachi: "A framework to design approximation algorithms for finding diverse solutions in combinatorial problems". In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2023), 37(4), pp. 3968–3976, AAAI Press, 2023.*
- *Tesshu Hanaka, Yasuaki Kobayashi, Kazuhiro Kurita, Yota Otachi. "Finding diverse trees, paths, and more", In: Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI 2021), 35(5), pp. 3778-3786, AAAI Press, 2021.*
- *Eppstein, David. "k-best enumeration." arXiv preprint arXiv:1412.5075 (2014).*
- *de Berg, Mark, Andrés López Martínez, and Frits Spieksma. "Finding Diverse Minimum s-t Cuts." arXiv preprint arXiv:2303.07290 (2023).*