Seven open problems in applied combinatorics[*]

Sinan G. Aksoy, Ryan Bennink, Yuzhou Chen, José Frías,
Yulia R. Gel, Bill Kay, Uwe Naumann, Carlos Ortiz Marrero,
Anthony V. Petyuk, Sandip Roy, Ignacio Segovia-Dominguez,
Nate Veldt, and Stephen J. Young

We present and discuss seven different open problems in applied combinatorics. The application areas relevant to this compilation include quantum computing, algorithmic differentiation, topological data analysis, iterative methods, hypergraph cut algorithms, and power systems.

While the development of combinatorial mathematics has always been spurred by applied problems, in recent years both the scope and specificity of applied combinatorics has increased dramatically. In concrete and varied ways, combinatorial approaches are tackling problems across the sciences: from using graph clustering to design reduced order models and controls for complex fluid flows [3, 4], employing Ramanujan graphs as interconnection topologies of linkages between elements of supercomputers [2, 7], predicting drug interactions using graph neural networks (GNNs) [6], to detecting anomalies in cybersecurity data using graph and hypergraph centrality measures [1, 5].

In celebration and promotion of this eclecticism, here we compile seven open problems in applied combinatorics posed by researchers in academia and government. The problems are organized in self-contained sections, authored by the attributed submitters. Each consists of a problem statement, discussion of relevant application areas, and any partial progress or prior work. The problems are as follows:

---

1. **The Dowker Complex in Metric Graphs**: Frías, Segovia-Dominguez, Chen, and Gel propose problems in topological data analysis (TDA) related to Dowker complexes.
2. **An Application of Probabilistic Combinatorics to Quantum Circuit Expressiveness**: Kay and Bennink introduce a problem in probabilistic combinatorics relevant to quantum circuits.
3. **The Computational Complexity of the 4-uniform Hypergraph Minimum s-t Cut Problem**: Veldt outlines a problem on the computational complexity of hypergraph minimum s-t cuts.
4-5. **The Edge and Vertex Elimination Problems in Directed Acyclic Graphs** and **Data Flow Reversal Problems**: Naumann poses several problems involving directed acyclic graphs, motivated by algorithmic differentiation of numerical programs.
6. **Price of Asynchrony**: Ortiz Marrero and Young pose problems to study how asynchronous updates affect the convergence rate of iterative methods for large-scale systems in scientific computing.
7. $(n-k)$**-contingent Zero-Forcing for Power Grids**: Aksoy, Petyuk, Roy, and Young propose problems on a variant of graph zero-forcing relevant to structural stability in power grids.

We have curated this sample of open problems with several criteria in mind. First, we have chosen problems that are timely and new, in the sense that they are motivated by recent research and not likely to be well-known by the larger community. Second, while far from comprehensive, our varied selection aims to illustrate the diversity of areas engaged by applied combinatorics. Third, while much of combinatorics plausibly "has applications", we have searched for problems having clear, specific impact to fields outside pure mathematics, including computing, industry, data science, and scientific software. We thank the reader for their interest.

References

[1] Sinan G Aksoy, Emilie Purvine, and Stephen J Young. "Directional Laplacian centrality for cyber situational awareness". In: *Digital Threats: Research and Practice (DTRAP)* 2.4 (2021), pp. 1–28.
[2] Sinan G Aksoy et al. "Ramanujan graphs and the spectral gap of supercomputing topologies". In: *The Journal of Supercomputing* 77 (2021), pp. 1177–1213.
[3] Muralikrishnan Gopalakrishnan Meena, Aditya G Nair, and Kunihiko Taira. "Network community-based model reduction for vortical flows". In: *Physical Review E* 97.6 (2018), p. 063103.

[4] Muralikrishnan Gopalakrishnan Meena and Kunihiko Taira. "Identifying vortical network connectors for turbulent flow modification". In: *Journal of Fluid Mechanics* 915 (2021), A10. MR4233199

[5] Cliff A Joslyn et al. "Hypergraph analytics of domain name system relationships". In: *Algorithms and Models for the Web Graph: 17th International Workshop, WAW 2020, Warsaw, Poland, September 21–22, 2020, Proceedings 17*. Springer. 2020, pp. 1–15.

[6] Carter Knutson et al. "Decoding the protein–ligand interactions using parallel graph neural networks". In: *Scientific Reports* 12.1 (2022), p. 7624.

[7] Stephen Young et al. "Spectralfly: Ramanujan graphs as flexible and efficient interconnection networks". In: *2022 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE. 2022, pp. 1040–1050.

1. The Dowker complex in metric graphs

José Frías, Ignacio Segovia-Dominguez, Yuzhou Chen, Yulia R. Gel

Background Topological data analysis (TDA) is a set of methods in computational topology that continues to receive increasing attention in data sciences. One of the most relevant tools in TDA is persistent homology (PH) [5, 9, 13]. The aim of PH is to obtain topological information of a finite metric space using a *filtered simplicial complex* $\mathcal{K}$ with vertices in the data set, namely, there exists a sequence of simplicial complexes $\mathcal{K}_0 \subset \mathcal{K}_1 \subset \cdots \subset \mathcal{K}_{n-1} \subset \mathcal{K}_n = \mathcal{K}$. A non-trivial element $\gamma \in H_p(\mathcal{K}_i)$, in the p-dimensional homology group of $\mathcal{K}_i$, is commonly called a *p-dimensional topological feature* and is associated with a point $(i_b, j_d) \in \mathbb{R}^2$, where i_b and j_d are the *birth* and *death* of γ, respectively (i.e., the indices of the simplicial complex at which γ is first and last observed, respectively, that is, when it becomes trivial in homology or is merged to another topological feature that was born before). The lifespan of the topological feature is $j_d - i_b$. The *persistence diagram PD* of the filtered simplicial complex $\mathcal{K}$ is the multiset $PD(\mathcal{K}) = \{(i_b, j_d) \in \mathbb{R}^2 \mid i_b < j_d\}$, where the multiplicity of (i_b, j_d) is the number of topological features in the filtered simplicial complex $\mathcal{K}$ that are born and die at i_b and j_d, respectively. A persistence diagram summarizes the evolution of the homology groups of $\mathcal{K}$ along the filtration. Two persistence diagrams can be compared using metrics such as the bottleneck or the Wasserstein distances (please see [3, 5, 9, 13] for a detailed exposition on persistence diagrams and distances in PH).

Metric spaces induced by weighted graphs are particularly interesting due to their combinatorial foundations and their applications in data analysis [1, 12]. A weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ contains a set of vertices $\mathcal{V}$, an edge set $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$, and a weight function $\omega : \mathcal{E} \to \mathbb{R}_+$. We say that a graph $\mathcal{G}$ is *simple* if it does not contain self-edges nor multiple edges. Given a path γ in a weighted graph $\mathcal{G}$, the *length* of γ is the sum of the weights of the edges in γ. A connected weighted graph $\mathcal{G}$ is then endowed with the *geodesic distance* $d_{\mathcal{G}} : \mathcal{V} \times \mathcal{V} \to \mathbb{R}_{\geq 0}$, defined on a pair of vertices $u, v \in \mathcal{V}$ as the minimum length among all the paths connecting u and v. The set $(\mathcal{V}, d_{\mathcal{G}})$ is a finite metric space.

One of the most widely used simplicial complexes in TDA is the Vietoris-Rips simplicial complex [3, 13]. One important property of this complex is that it is completely determined by its 1-skeleton, which makes it suitable for computations.

Definition 1.1 (Vietoris-Rips Complex). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ be a weighted simple graph with induced geodesic distance $d_{\mathcal{G}}$. For $\alpha \in \mathbb{R}_{\geq 0}$, we define the **Vietoris-Rips complex** $VR_\alpha(\mathcal{G})$ as the abstract simplicial complex with vertices in $\mathcal{V}$ and, for $k \geq 2$, a k-simplex $\sigma = [x_0, x_1, \dots, x_k] \in VR_\alpha(\mathcal{G})$ if and only if $d_{\mathcal{G}}(x_i, x_j) \leq \alpha$ for $0 \leq i \leq j \leq k$.

However, if the cardinality of $\mathcal{V}$ is large, the number of simplices in $VR_\alpha(\mathcal{G})$, for large values α, could be excessively large to be analyzed using computational tools. An alternative to deal with the scalability problem is to construct another simplicial complex with set of vertices $L \subset \mathcal{V}$, where $|L| < |\mathcal{V}|$. One of such simplicial complexes is the witness complex [1, 2, 7, 8].

Definition 1.2 (Witness Complex). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ be a weighted simple graph with induced geodesic distance $d_{\mathcal{G}}$, and take subsets $L, W \subset \mathcal{V}$. For $\alpha \in \mathbb{R}_{\geq 0}$, let $Wit_\alpha(W, L)$ be the abstract simplicial complex with set of vertices L, and a simplex $\sigma \in Wit_\alpha(W, L)$ if and only if for every $\tau \subseteq \sigma$ there exists $w \in W$ such that $d_{\mathcal{G}}(w, l) \leq d_{\mathcal{G}}(w, l') + \alpha$ for all $l \in \tau$ and $l' \in L \setminus \tau$. The complex $Wit_\alpha(W, L)$ is called the **witness complex** of $\mathcal{G}$ with set of **witnesses** W and **landmarks** L.

Given a sequence of non-decreasing values $0 \leq \alpha_0 \leq \alpha_1 \leq \cdots \leq \alpha_{n-1} \leq \alpha_n$, the associated Vietoris-Rips complexes constructed on $\mathcal{G}$ satisfy $VR_{\alpha_0}(\mathcal{G}) \subseteq VR_{\alpha_1}(\mathcal{G}), \subseteq \cdots \subseteq VR_{\alpha_n}(\mathcal{G})$, namely, the sequence of scale values define a filtered simplicial complex. If we select sets of landmarks and witnesses $L, W \subset \mathcal{V}$, there exists also a filtration $Wit_{\alpha_0}(L, W) \subseteq Wit_{\alpha_1}(L, W) \subseteq \cdots \subseteq Wit_{\alpha_n}(L, W)$. Then persistent homology can be computed in both filtered simplicial complexes. However, computation of witness
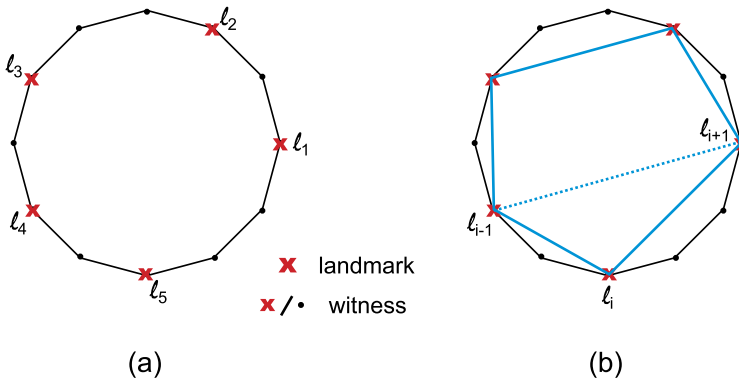
Figure 1: Landmarks in a cycle graph.

complexes on graphs is challenging due to the difficulty to determine a witness of some simplexes, as the next illustrative example shows.

Example 1.3. Suppose we are interested in the 1-dimensional persistent homology of the 1-weighted cycle graph $\mathcal{C}_m$, $m \geq 4$, with vertices set $\mathcal{V}$. Let $L \subset \mathcal{V}$ be a set of landmarks and $W = \mathcal{V}$ be a set of witnesses (see Figure 1a). It is not difficult to determine the smallest scale value $\alpha > 0$ such that $Wit_\alpha(L, W)$ contains the 1-simplex $\{l_i, l_{i+1}\}$, for any two consecutive landmarks $l_i, l_{i+1} \in L$ (in Figure 1b the 1-simplexes are represented by edges connecting consecutive landmarks). These 1-simplexes form a 1-dimensional topological feature γ that is born at α. However, given a sequence of three consecutive landmarks $l_{i-1}, l_i, l_{i+1} \in L$, it is combinatorially more difficult to determine a witness and a scale value at which the 1-simplex $\{l_{i-1}, l_{i+1}\}$ (dotted edge in Figure 1b), appears in the filtered witness complex, or, even more difficult, when γ vanishes.

The Dowker complex [3, 4] is a construction obtained after relaxing the definition of the witness complex, Definition 1.2, to make it easier to compute:

Definition 1.4 (Dowker Complex). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ be a weighted simple graph with induced geodesic distance $d_\mathcal{G}$, and let $L, W \subset \mathcal{V}$. For $\alpha \in \mathbb{R}_{\geq 0}$, let $Dow_\alpha(W, L)$ be the abstract simplicial complex with set of vertices L and a simplex $\sigma \in Dow_\alpha(W, L)$ if and only if there exists $w \in W$ such that $d_\mathcal{G}(w, l) \leq \alpha$ for all $l \in \sigma$. The simplicial complex $Dow_\alpha(W, L)$ is called the **Dowker complex** of $\mathcal{G}$ with sets of witnesses and landmarks W and L, respectively.

Note that witness and Dowker complexes constructions strongly depend on the selection of the sets L and W. In [7], two algorithms to define the set of landmarks L are developed: *random* and *maxmin* algorithms. More recently, [1] proposes the ϵ-nets algorithm to obtain a set of landmarks $L \subset \mathcal{V}$ satisfying both properties of sparsity and proximity to the set of points $\mathcal{V} \setminus L$, thereby allowing for a trade-off between computational complexity and potential information loss.

Definition 1.5 (ϵ-net). Let $(\mathcal{V}, d_{\mathcal{G}})$ be the finite metric space obtained from the weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$. Given $L = \{u_1, u_2, \ldots, u_l\} \subset \mathcal{V}$ and $\epsilon \geq 0$ then:

  (i) The set L is an ϵ-**sample** of $\mathcal{G}$ if the collection $\{\mathcal{N}(u_i)\}_{i=1}^{l}$ of closed ϵ-neighborhoods of points in L covers $\mathcal{V}$, i.e. for any $v \in \mathcal{V}$ there exists $u_j \in L$ such that $d_{\mathcal{G}}(v, u_j) \leq \epsilon$.
 (ii) L is ϵ-**sparse** if for any two distinct points $u_i, u_j \in L$, their distance $d_{\mathcal{G}}(u_i, u_j) > \epsilon$.
(iii) The set L is an ϵ-**net** of $\mathcal{G}$ if it is an ϵ-sample of $\mathcal{G}$ and is ϵ-sparse.

Open problems   Along the present section, $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ is a weighted simple graph and $d_{\mathcal{G}}$ is the geodesic distance induce by $\mathcal{G}$ on $\mathcal{V}$. The construction of Dowker simplicial complexes on the metric space $(\mathcal{V}, d_{\mathcal{G}})$ relies on the selection of the sets of landmarks and witnesses. Usually the set of witnesses is taken to be the set of vertices, $W = \mathcal{V}$, or the complement of landmarks, $W = \mathcal{V} \setminus L$. Furthermore, there are numerous possibilities of selecting a set of landmarks. The existence of an ϵ-net defining the set of landmarks guarantees some combinatorial and geometric properties, as shown in [1]. Furthermore, [1] proved that for a weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ and a particular ϵ, there exists an ϵ-net for $(\mathcal{V}, d_{\mathcal{G}})$ whose cardinality admits a bound depending on ϵ, the diameter of the graph and the number of vertices (Theorem 3). However, this bound does not pretend to be any close to a sharp bound, while its further analysis has a number of important implication for graph learning applications in machine learning and statistics. As a result, we formulate the first open problem on the Dowker complexes on graphs as follows:

Open Problem 1.1. Given a weighted simple graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ and $\epsilon > 0$, determine optimal upper and lower bounds for the number of elements of an ϵ-net for $\mathcal{G}$.

Three different methods to obtain ϵ-nets in metric graphs, namely, the greedy, iterative and *SPTpruning* algorithms, are proposed in [1]. These al-

gorithms are experimentally compared with respect to the number of land-marks they produce for the same graph, including their execution time. As such, the next natural question arises:

Open Problem 1.2. For a weighted simple graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$, propose an algorithm to construct an ϵ-net for $\mathcal{G}$ such that:

(i) It is based on a lower number of landmarks than that of the existing algorithms;

(ii) The time of execution of the algorithm is lower than the existing algorithms.

Since the real-world datasets are usually noisy and incomplete, a suit-able choice of landmarks in the Dowker complexes has multi-fold potential benefits in applications. First, it tends to improve performance of graph learning algorithms by focusing on the most essential topological charac-teristics (i.e. the "data skeleton"). Second, it helps enhance robustness to noise/data perturbations. Third, by reducing computational costs, it opens a perspective of utilizing TDA for downstream tasks in graph learning. In-deed, as computing persistence homology on large graphs is often infeasible due to prohibitive computational costs, the Dowker complex construction offers a promising alternative by the means of the suitable landmark selec-tion. Additionally, the data separation in landmarks and witnesses trigger an intrinsic spatial decomposition that match with the natural local divisions needed when dealing with modern massive parallel computing [11].

Given the current proliferation of the Vietoris-Rips complex in TDA, the natural question to address is the comparison of persistent homology obtained from the Vietoris-Rips and Dowker complexes constructions on the same weighted graph $\mathcal{G}$. We denote with $PD^i(\mathcal{K})$ the persistence diagram of dimension i for the filtered simplicial complex $\mathcal{K}$. In the case of metric graphs, 1-dimensional homology is relevant. It is known that for a large enough scale value α, the persistence diagram $PD^1(VR_\alpha(\mathcal{G}))$ contains as many points (counting multiple points) as the genus of $\mathcal{G}$, the cardinality of a minimal system of cycles of $\mathcal{G}$ [10]. A main concern on the selection of landmarks to construct the witness or Dowker complexes is the loss of relevant topological information. Then, the next question arises:

Open Problem 1.3. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ be weighted simple graph. Take the set of witnesses $W = \mathcal{V}$.

(i) Determine $\epsilon > 0$ and the way to select an ϵ-net $L \subset \mathcal{V}$ such that $PD^1(Dow_\alpha(W, L))$ has cardinality equal to the genus of $\mathcal{G}$, for a suf-ficiently large value α.

(ii) Assuming that an ϵ-net for $\mathcal{G}$ satisfying (i) is given. Find a bound for $d_B(PD^1(VR_\alpha(\mathcal{G})), PD^1(Dow_\alpha(W, L)))$, in terms of the lengths of cycles in $\mathcal{G}$ and ϵ, where d_B is the bottleneck distance in persistence diagrams.

Finally, another important but largely unexplored topic for PH obtained from the Dowker complexes is stability. Stability ensures that after applying a "small perturbation" to a metric graph $\mathcal{G}$ in order to obtain a new metric graph $\mathcal{G}'$, the corresponding persistence diagrams are close with respect to some distance, for instance the bottleneck or Wasserstein distance. By perturbation we refer to any transformation that may include edge deletion, addition, cleaving or contraction, or changing the weight function (which automatically modifies the distance $d_\mathcal{G}$). In the case of complexes depending on a selection of landmarks and witnesses a perturbation could be changing the selection of these two sets. For the Dowker complex, some stability results have been obtained, including the Dowker interleaving and the Dowker duality [3, see Section 4.2.3]. In particular, Dowker duality can be interpreted as the property such that if $L, W \subset \mathcal{V}$ are the sets of landmarks and witnesses of $\mathcal{G}$, then $Dow_\alpha(W, L)$ and $Dow_\alpha(L, W)$ have the same homotopy type and, then, the two filtered simplicial complexes have the same persistent homology.

Open Problem 1.4. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$ be a weighted graph and let $L, W \subset \mathcal{V}$ be sets of landmarks and witnesses for $\mathcal{G}$.

(i) Let $\mathcal{G}' = (\mathcal{V}, \mathcal{E}, \omega')$ be the metric graph with the same set of vertices and edges of $\mathcal{G}$, but another weight function ω'. Find a bound for the bottleneck distance of the persistence diagrams corresponding to the Dowker complexes of $\mathcal{G}$ and $\mathcal{G}'$ in terms of the supremum distance between ω and ω'.

(ii) If two sets of landmarks L and L' for $\mathcal{G}$ are ϵ-nets. Find conditions on the landmarks, such that $Dow_\alpha(W, L)$ and $Dow_\alpha(W, L')$ have the same homotopy type.

Recent studies have addressed some properties of the Dowker complexes in directed graphs. For instance, a complete characterization of the Dowker persistence diagrams for cycle networks is presented, and some stability properties have been proven for pair swaps by [4] and for conjugate and shift equivalent relations [6]. While both the theoretical properties and practical utility of the Dowker complexes on graphs yet remain largely unexplored, we believe that the Dowker complexes have a potential to address many bottlenecks that currently preclude broader applications of TDA in analysis of real-world graph datasets.

References

[1] N. A. Arafat, D. Basu, and S. Bressan. "ϵ-net Induced Lazy Witness Complexes on Graphs". In: (2020). DOI: 10.48550/ARXIV.2009.13071. URL: https://arxiv.org/abs/2009.13071.

[2] J. D. Boissonnat, L. Guibas, and S. Oudot. "Manifold reconstruction in arbitrary dimensions using witness complexes". In: *Proceedings of the twenty-third annual symposium on Computational geometry*. 2007, pp. 194–203. MR2469165

[3] F. Chazal, V. De Silva, and S. Oudot. "Persistence stability for geometric complexes". In: *Geometriae Dedicata* 173.1 (2014), pp. 193–214. MR3275299

[4] S. Chowdhury and F. Mémoli. "A functorial Dowker theorem and persistent homology of asymmetric networks". In: *Journal of Applied and Computational Topology* 2 (2018), pp. 115–175. MR3873182

[5] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. "Stability of persistence diagrams". In: *Proceedings of the twenty-first annual symposium on Computational geometry*. 2005, pp. 263–271. MR2460372

[6] D. Côté. *Dowker Complexes and filtrations on self-relations*. 2023. DOI: 10.48550/ARXIV.2301.03739. URL: https://arxiv.org/abs/2301.03739.

[7] V. De Silva and G. Carlsson. "Topological estimation using witness complexes." In: *PBG*. 2004, pp. 157–166.

[8] T. K. Dey, F. Fan, and Y. Wang. "Graph induced complex on point data". In: *Proceedings of the twenty-ninth annual symposium on Computational geometry*. 2013, pp. 107–116.

[9] H. Edelsbrunner, D. Letscher, and A. Zomorodian. "Topological persistence and simplification". In: *Proceedings 41st annual symposium on foundations of computer science*. IEEE. 2000, pp. 454–463. MR1931842

[10] E. Gasparovic et al. "A complete characterization of the one-dimensional intrinsic Čech persistence diagrams for metric graphs". In: *Research in Computational Topology* (2018), pp. 33–56. MR3905000

[11] R. Lewis and D. Morozov. "Parallel computation of persistent homology using the blowup complex". In: *Proceedings of the 27th ACM Symposium on Parallelism in Algorithms and Architectures*. 2015, pp. 323–331.

[12] X. Liu et al. "Dowker complex based machine learning (DCML) models for protein-ligand binding affinity prediction". In: *PLoS Computational Biology* 18.4 (2022), e1009943.

[13] A. Zomorodian and G. Carlsson. "Computing persistent homology". In: *Proceedings of the twentieth annual symposium on Computational geometry.* 2004, pp. 347–356. MR2121296

2. An application of probabilistic combinatorics to quantum circuit expressiveness

Bill Kay, Ryan Bennink

Introduction   Quantum computing is a computing system in which certain physical properties are leveraged for computational advantages for some classes of problems. Recently, access to some quantum computing devices has grown. Many of the accessible devices are small and noisy, and the *variational approach* in which a traditional computer modifies parameters of some quantum circuit towards maximizing a function of its output has found success [3, 2, 5]. The purpose of this document is to introduce a purely combinatorial problem, and to explain how it is related to a problem in quantum computing. We note that we are casting the analysis in [1] to a probabilistic setting regarding random binary vectors. Hence, progress in the combinatorial problem has an interpretation which impacts quantum computing. We then present a sequence of examples, preliminary results, and open questions.

Commutative quantum circuits, originally introduced as "Instantaneous Quantum Polytime" circuits [4], are a special class of quantum circuits that are relatively simple yet exhibit non-trivial quantum behavior. For our purposes a commutative quantum circuit may be described as a circuit that first creates a uniform superposition of all 2^n computational states of n qubits, then applies parity-dependent phases to selected subsets of qubits. Each subset is specified by a binary vector $a \in \{0,1\}^n$ and its associated phase is specified by a real angle θ.

Definition 2.1 (Commutative Quantum Circuit)**.** A commutative quantum circuit specified by the binary vectors $a_1, \ldots, a_m \in \{0,1\}^n$ and associated parameters $\theta_1, \ldots, \theta_m \in \mathbb{R}$ creates a quantum state of the form

$$\frac{1}{2^{n/2}} \left(e^{i\phi_{0\ldots0}}, \ldots, e^{i\phi_{1\ldots1}} \right) \in \mathbb{C}^{2^n}$$

where

$$\phi_x = \sum_{i=1}^{m} \theta_i (-1)^{\langle a_i, x \rangle}$$

and $\langle \cdot, \cdot \rangle$ denotes the inner product in $GF(2)^n$.

In the context of variational quantum computing, the vectors $a_1, \ldots, a_m$ specify a fixed circuit structure and the angles $\theta_1, \ldots, \theta_m$ are the variational parameters. A fundamental question concerns the *expressiveness* of such: given the vectors $a_1, \ldots, a_m$, how much can the output state be varied by varying the angles $\theta_1, \ldots, \theta_m$? Intuitively, the expressiveness can be quantified by the expected (dis)similarity of two output states with randomly chosen parameter values: the more expressive the circuit, the more likely two random output states will be dissimilar. As discussed in [1], for commutative quantum circuits this expectation value can be reduced to a combinatoric problem involving binary matrices described below. There it is shown that in the special case that the circuit is maximal, i.e. there is an independent phase for each nonempty subset of qubits, the expressiveness reduces to the problem of counting *abelian squares*. Here we consider the more general problem of computing the expressiveness of commutative quantum circuits involving arbitrary collections of qubit subsets. In the next section we cast this problem in purely combinatorial language. We conclude with some examples, special cases, and open problems.

A combinatorial problem In this section we will have several definitions where prescribed notations will aid in clarity. Here we fix several such notations for the section for ease of exposition. Fix $n, t \in \mathbb{Z}^+$ and let $1 \leq m \leq 2^n - 1$.

- Let $\mathbf{x} = (x_1^T, \ldots, x_t^T)$ where $x_i \in \{0,1\}^n$ for $1 \leq i \leq t$.
- Let $\mathbf{y} = (y_1^T, \ldots, y_t^T)$ where $y_i \in \{0,1\}^n$ for $1 \leq i \leq t$.
- $\mathbf{A} = (a_1^T, \ldots, a_m^T)$ where $a_i \in \{0,1\}^n$ for $1 \leq i \leq m$. Moreover, the $\{a_i\}_{i=1}^m$ are non-zero and pairwise distinct.

For clarity, $\mathbf{x}$ and $\mathbf{y}$ are $n \times t$ binary matrices with specified column labels, and $\mathbf{A}$ is an $n \times m$ binary matrix with *distinct, non-zero* columns with specified labels. For our specific application, we take the columns of $\mathbf{A}$ to be given in lexicographic order. We remark that each a_i is a length n binary vector and has a natural correspondence with subsets of indices of length n vectors.

We now need the following definition:

Definition 2.2 (Expressiveness Indicator). Given $\mathbf{A}, \mathbf{x}$ as above and $1 \leq i \leq m$, we define the ith *expressiveness indicator* $m_i^{\mathbf{A}}(\cdot)$ as:

$$m_i^{\mathbf{A}}(\mathbf{x}) := \sum_{j=1}^{t} \langle a_i, x_j \rangle_2$$

where the inner product $\langle \cdot, \cdot \rangle_2$ is *mod* 2 but the outer sum is integral.

When $\mathbf{A}$ is clear from context we simply write $m_i(\cdot)$. As these are not standard definitions, and there is a mix of modular and integral arithmetic, we include a clarifying example:

Example 2.3. Let

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} ; \mathbf{x} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Then we have:

$$\begin{aligned} m_5(\mathbf{x}) &= \langle a_5, x_1 \rangle_2 + \langle a_5, x_2 \rangle_2 + \langle a_5, x_3 \rangle_2 + \langle a_5, x_4 \rangle_2 \\ &= 0 + 1 + 1 + 0 \\ &= 2 \end{aligned}$$

We are now ready to pose a general combinatorial question and relate it back to computing expressiveness of commutative quantum circuits.

Computational Problem. *Given $\mathbf{A}$, let $\mathbf{x}$ and $\mathbf{y}$ be sampled uniformly at random. Compute:*

$$\mathbb{P}\left((m_i(\mathbf{x}) = m_i(\mathbf{y})) \ \forall i\right)$$

While the Computational Problem is interesting in its own right, we would like to clarify how it is related to computing expressiveness of commutative quantum circuits. Recall from Definition 2.1 that the expressiveness of a commutative quantum circuit takes as input a collection of subsets of indices of a state vector. Further, we have remarked that for a length n state vector, there is a natural correspondence between a subset of indices and binary length n vectors (namely, the indicator vector of which indices are selected). Indeed, one can see in the analysis in [1] that if we encode the subsets of indices in the state vector we are interested in as $\mathbf{A}$, then the computation given in Computational Problem is the critical computation

for finding the desired expressiveness. In particular, it is shown that expressiveness of *maximal* commutative quantum circuits is equivalent to taking $\mathbf{A}$ to be all $2^n - 1$ non-zero length n binary vectors, and that the probability in Computational Problem is precisely resolved by counting abelian squares.

Examples, preliminary results, and open problems Notice that for each choice of $\mathbf{A}$, there is a different instance of Computational Problem. For some fixed choices of $\mathbf{A}$, there is a natural interpretation of Computational Problem.

Example 2.4. Let $\mathbf{A}$ be the collection of all weight 1 length n vectors (i.e., the $n \times n$ identity matrix). Let $\mathbf{x}$ and $\mathbf{y}$ be $n \times t$ binary matrices. Then

$$(m_i(\mathbf{x}) = m_i(\mathbf{y})) \ \forall i$$

if and only if $\mathbf{x}$ and $\mathbf{y}$ have the same row sums.

This is clear, as $\langle a_j, x_i \rangle_2 = 1$ if and only if x_i is 1 in coordinate j. Using this combinatorial interpretation, we can resolve Computational Problem when $\mathbf{A}$ is the $n \times n$ identity matrix by the following (equivalent) lemma:

Lemma 2.5. *Let $\mathbf{x}$ and $\mathbf{y}$ be $n \times t$ binary matrices sampled uniformly at random. Then the probability that $\mathbf{x}$ and $\mathbf{y}$ have all the same row sums is*

$$\binom{2t}{t}^n \frac{1}{2^{2tn}}.$$

Proof. We will argue for a single pair of uniform random binary vectors of length t. Since we are computing row sums for $\mathbf{x}$ and $\mathbf{y}$, we argue first for a fixed row and conclude by taking a product of independent probabilities. Summing over possible row sums, the probability that a pair of length t binary vectors have the same row sum is given by:

$$\sum_{k=0}^{t} \binom{t}{k}^2 (1/2)^{2k}(1/2)^{2(t-k)} = \frac{1}{2^{2t}} \sum_{k=0}^{t} \binom{t}{k}^2$$

$$= \frac{1}{2^{2t}} \sum_{k=0}^{t} \binom{t}{k}\binom{t}{t-k}$$

$$= \frac{1}{2^{2t}} \binom{2t}{t}$$

where the last equality can be seen by counting the number of length $2t$ binary vectors with precisely t ones. The right hand side counts this quantity

directly, while the left hand side counts the number of ways for k ones in the first half with $t - k$ ones in the second half. Taking the product over all n (independent) rows of $\mathbf{x}$ and $\mathbf{y}$ gives the desired probability. $\qquad\square$

Each column of $\mathbf{A}$ provides an additional constraint on possible choices of $\mathbf{x}$ and $\mathbf{y}$ in Computational Problem. In fact, the collections of $(\mathbf{x}, \mathbf{y})$ pairs which satisfy the computation in Computational Problem are *monotone* in the columns of $\mathbf{A}$, in that if the set of columns $\mathbf{A}_2$ contains the set of columns of $\mathbf{A}_1$, then

$$(m_i^{\mathbf{A}_2}(\mathbf{x}) = m_i^{\mathbf{A}_2}(\mathbf{y})) \ \forall i \Rightarrow (m_i^{\mathbf{A}_1}(\mathbf{x}) = m_i^{\mathbf{A}_1}(\mathbf{y})) \ \forall i.$$

For example, if $\mathbf{A}$ has among its columns all weight 1 length n vectors, if $\mathbf{x}$ and $\mathbf{y}$ satisfy the condition in Computational Problem then $\mathbf{x}$ and $\mathbf{y}$ necessarily have the same row sums. We provide combinatorial interpretation of Computational Problem for another choice of $\mathbf{A}$, and then exploit the monotonicity property to state and prove Computational Problem for another choice of $\mathbf{A}$.

Example 2.6. Let $a_{i,j}$ be the length n binary vector with a 1 in positions i and j, 0 otherwise. Let x be any length n binary vector. $\langle a_{i,j}, x \rangle_2$ is 1 if the ith and jth entries of x disagree, 0 otherwise. We call this an i, j-*mismatch* in x. Hence, if $\mathbf{A}$ consists of all length n binary vectors of weight at most 2, and $\mathbf{x}$ and $\mathbf{y}$ are $n \times t$ binary matrices, then

$$(m_i(\mathbf{x}) = m_i(\mathbf{y})) \ \forall i$$

if and only if $\mathbf{x}$ and $\mathbf{y}$ have the same row sums, and for each pair of rows i and j the number of columns in which $\mathbf{x}$ and $\mathbf{y}$ have an i, j-mismatch are the same.

Computational Problem for the family presented in Example 2.6 remains open. However, we present the following:

Lemma 2.7. *Let $\mathbf{A}$ be the collection of binary vectors weight 1 and weight 2 with a 1 in the first coordinate. Let $\mathbf{x}$ and $\mathbf{y}$ be $n \times t$ binary matrices sampled uniformly at random. Then:*

$$\mathbb{P}\left((m_i(\mathbf{x}) = m_i(\mathbf{y})) \ \forall i\right)$$

is given by:

$$\frac{1}{2^{2tn}} \sum_{k=0}^{t} \left(\binom{2k}{k} \binom{2(t-k)}{t-k} \right)^{n-1} \binom{t}{k}^2$$

Proof. For $\mathbf{x}$ and $\mathbf{y}$ sampled uniformly at random, we want to compute the probability that $\mathbf{x}$ and $\mathbf{y}$ have the same row sums, and for each $1 < j \le n$, the number columns which have a $1, j$-mismatch is the same. We claim that

$$\sum_{k=0}^{t} \binom{t}{k}^2 \left(\sum_{j=0}^{k} \sum_{\ell=0}^{t-k} \binom{k}{j}^2 \binom{t-k}{\ell}^2 \right)^{n-1}$$

counts the number of $(\mathbf{x}, \mathbf{y})$ pairs for which

$$(m_i(\mathbf{x}) = m_i(\mathbf{y})) \; \forall i$$

holds. We now count such pairs. First, fix a first row for each of $\mathbf{x}$ and $\mathbf{y}$. Since the row sums must be the same, these first rows have some number of 1s, say k. For each row $1 < r \le n$, there are some number of 1s in $\mathbf{x}$ and $\mathbf{y}$. Since the row sums are the same for $\mathbf{x}$ and $\mathbf{y}$, this number is the same. We note that $\mathbf{x}$ and $\mathbf{y}$ need to have the same number of $1, r$ matches as well as mismatches, and this can be counted by enumerating the number of ways to place 1s in columns with a 1 in the first entry, and then columns with 0 in the first entry. There are $j + \ell$ 1s in row r, with j in columns with 1 in the first entry and ℓ in columns with 0 in the first entry. Hence, for each fixed choice of j and ℓ there are

$$\binom{k}{j}^2 \binom{t-k}{\ell}^2$$

ways to place these 1s. Summing over choices of j and ℓ we see that for each fixed pair of first rows for $\mathbf{x}$ and $\mathbf{y}$ there are

$$\sum_{j=0}^{k} \sum_{\ell=0}^{t-k} \binom{k}{j}^2 \binom{t-k}{\ell}^2$$

choices to place the 1s in row r. These choices are independent for rows $1 < r \le n$, and so there are

$$\left(\sum_{j=0}^{k} \sum_{\ell=0}^{t-k} \binom{k}{j}^2 \binom{t-k}{\ell}^2 \right)^{n-1}$$

choices for $\mathbf{x}$ and $\mathbf{y}$ for a fixed first row. For each k, there are $\binom{t}{k}$ ways to place 1s in the first row of $\mathbf{x}$ and $\mathbf{y}$. Summing over each k, the total number

of desired $(\mathbf{x}, \mathbf{y})$ pairs is

$$\sum_{k=0}^{t} \binom{t}{k}^2 \left(\sum_{j=0}^{k} \sum_{\ell=0}^{t-k} \binom{k}{j}^2 \binom{t-k}{\ell}^2 \right)^{n-1}$$

Applying analysis similar to that in the proof of Lemma 2.5 shows that this is

$$\sum_{k=0}^{t} \left(\binom{2k}{k} \binom{2(t-k)}{t-k} \right)^{n-1} \binom{t}{k}^2$$

There are 2^{2tn} total possible pairs $(\mathbf{x}, \mathbf{y})$, and dividing through yields the claim in the statement of Lemma 2.7. $\qquad\square$

The methods in the proofs of Lemmas 2.5 and 2.7 depend on row-wise independence. However, these methods do not directly extend to $\mathbf{A}$ for, say, all binary vectors of weight at most 2, as for $1 \leq i < j < k \leq m$, the configurations of i, j-mismatches and the configurations of j, k-mismatches affect the possible configurations of i, k- mismatches. More broadly, once the family $\mathbf{A}$ becomes rich enough, the step where we appeal to independence is not applicable and more sophisticated analysis is required. Hence, we have the following (open) questions:

Open Problem 2.1. Fix m and $k \leq m$. Can we answer Computational Problem when $\mathbf{A}$ consists of all vectors of weight at most k?

Lemma 2.5 answers Open Problem 2.1 when $k = 1$, and the maximal case in [1] answers Open Problem 2.1 when $k = m$. These boundary cases are the only values of k for which Open Problem 2.1 has been answered.

Open Problem 2.2. For which choices of $\mathbf{A}$ can we answer Computational Problem exactly? For which choices of $\mathbf{A}$ can we answer Computational Problem efficiently?

To rephrase the second part of Open Problem 2.2, we ask:

Open Problem 2.3. Is Computational Problem NP-Complete for arbitrary $\mathbf{A}$?

Finally, we remark that exact answers to Computational Problem provide answers to expressiveness of commutative quantum circuits, approximate answers to Computational Problem provide approximations to expressiveness of commutative quantum circuits. Hence we ask:

Open Problem 2.4. For which choices of $\mathbf{A}$ can we approximate answers to Computational Problem?

References

[1] Ryan S. Bennink. *Counting Abelian Squares for a Problem in Quantum Computing.* Aug. 2022. DOI: 10.48550/arXiv.2208.02360. arXiv: arXiv:2208.02360.

[2] Alicia B Magann et al. "From pulses to circuits and back again: A quantum optimal control perspective on variational quantum algorithms". In: *PRX Quantum* 2.1 (2021), p. 010101.

[3] Jarrod R McClean et al. "The theory of variational hybrid quantum-classical algorithms". In: *New Journal of Physics* 18.2 (2016), p. 023023.

[4] Dan Shepherd and Michael J. Bremner. "Temporally Unstructured Quantum Computation". In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 465.2105 (Feb. 2009), pp. 1413–1439. DOI: 10.1098/rspa.2008.0443. MR2500810

[5] Xiao Yuan et al. "Theory of variational quantum simulation". In: *Quantum* 3 (2019), p. 191.

3. The computational complexity of the 4-uniform hypergraph minimum *s-t* cut problem

Nate Veldt

Introduction Finding a minimum *s-t* cut in a graph is one of the most fundamental and well-known combinatorial optimization problems, studied and applied widely in mathematics, computer science, operations research, and data science. The input to the problem is a graph $G = (V, E)$ with a distinguished source node s and sink node t. The goal is to delete the minimum number of edges to destroy all paths from s to t, or equivalently partition the nodes in a way that separates s and t while minimizing a cut penalty (see Figure 2). A polynomial time solution for this problem has been known since the 1950s [1], and many other algorithms are now taught regularly as a standard part of an undergraduate algorithms course.
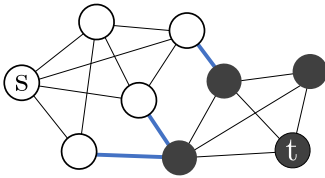
Figure 2: Graph *s-t* cut.

Two formulations of the graph *s-t* cut problem

1. Find a minimum size set of edges whose deletion separates s from t (e.g., three blue edges in Figure 2).
2. Partition V into $S \subseteq V$ and $\bar{S} = V \backslash S$ (e.g., white and black nodes in Figure 2), with $s \in S$ and $t \in \bar{S}$, in order to minimize the number of *cut* edges (i.e., edges crossing the partition).

The *s-t* cut problem can also be defined when the input is a hypergraph $\mathcal{H} = (V, E)$, in which case each edge $e \in E$ can have two *or more* nodes. If we generalize one way of formulating the graph *s-t* cut problem, the goal is simply to delete a minimum sized set of hyperedges to separate a source node s from a sink node t. A polynomial time solution for this problem was given half a century ago by Lawler [3], which works by reducing the hypergraph to an *s-t* cut problem in a *directed* graph with an augmented node set. However, generalizing the second formulation of the graph *s-t* cut problem (partition nodes to separate s and t while minimizing a cut penalty) leads to alternative problems that are not equivalent to this edge-deletion problem. When a hyperedge has more than two nodes, there is more than one way to separate those nodes across two node sets S and $\bar{S}$. In a 4-uniform hypergraph (see Figure 3), we can treat 1-vs-3 hyperedge splits differently from 2-vs-2 splits. Recently there has been a growing interest in solving hypergraph cut problems under these generalized types of cut penalties [5, 6, 7, 8, 4], since different ways of splitting up the nodes of a hyperedge may
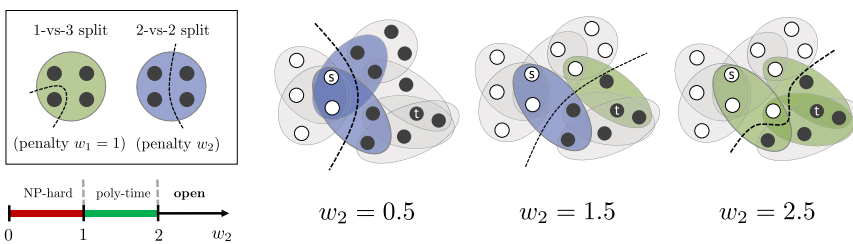

Figure 3: Optimal *s-t* cuts in a 4-uniform hypergraph, depending on the penalty w_2 assigned to 2-vs-2 hyperedge splits (i.e., two nodes on each side of a bipartition). All 1-vs-3 hyperedge splits have a penalty $w_1 = 1$. Previous results have shown the problem is NP-hard for $w_2 \in [0, 1)$ and polynomial-time solvable when $w_2 \in [1, 2]$. The complexity of the problem remains open for $w_2 > 2$.

be more or less desirable depending on the application. In many machine learning and data mining applications, for example, hyperedges represent evidence that a set of data objects are related and should be associated with the same label or cluster. In these cases, it is more desirable to split hyperedges in such a way that most (even if not all) nodes are on the same side of a cut. At the same time, using different cut penalties leads to drastic differences in the underlying computational complexity of the problem, even when hyperedges are very small. We specifically consider an open question on the computational complexity of a generalized hypergraph s-t cut problem in 4-uniform hypergraphs.

The cardinality-based hypergraph s-t cut problem Let $\mathcal{H} = (V, \mathcal{E})$ be a hypergraph with a source node s and sink node t. Given a bipartition $\{S, \bar{S}\}$ of the nodes, hyperedges can be classified based on the number (i.e., cardinality) of nodes on the small side of the cut. Edges with i nodes on the small side of the cut are denoted by $\partial_i(S) = \{e \in \mathcal{E}: \min\{|e \cap S|, |e \cap \bar{S}|\} = i\}$. We take the minimum between $|e \cap S|$ and $|e \cap \bar{S}|$ to ensure the resulting hypergraph cut function is symmetric, generalizing the fact that a graph cut function is symmetric. Let $w_i \geq 0$ be the penalty assigned to each hyperedge in $\partial_i(S)$. The value w_0 is set to 0 to ensure that a hyperedge only has a penalty if it is actually cut. The cardinality-based hypergraph s-t cut problem is given by

$$(1) \qquad \min_{S \subseteq V} \sum_i w_i |\partial_i(S)|, \qquad \text{subject to } s \in S \text{ and } t \in \bar{S}.$$

When $w_i = 1$ for every $i > 0$, this is equivalent to finding a minimum number of hyperedges to remove to separate s from t, and can be solved in polynomial time using the reduction of Lawler [3]. If $\mathcal{H}$ is 3-uniform, any way of cutting a hyperedge places exactly one node in S or exactly one node in $\bar{S}$. Since all hyperedge cut penalties are the same, this is solved by Lawler's algorithm.

Results for 4-uniform hypergraphs When $\mathcal{H}$ is 4-uniform, there is a distinction between 2-vs-2 splits (penalty of w_2) and 1-vs-3 splits (penalty of w_1). As long as $w_1 > 0$, parameters can be scaled without loss of generality so that $w_1 = 1$. For certain choices of w_2, Veldt, Benson, and Kleinberg [7] showed how to reduce the hypergraph problem to a graph s-t cut problem, by replacing each hyperedge with a *gadget* involving auxiliary vertices and directed edges. The aim is to design gadgets so that cut penalties in the

reduced directed graph match cut penalties in the original hypergraph. In the 4-uniform case, it turns out this is possible if *and only if* $1 \leq w_2 \leq 2$. If $0 \leq w_2 < 1$, the problem can be shown to be NP-hard via reduction from the optimization version of maximum cut, one of Karp's 21 NP-complete problems [2]. If $G = (V, E)$ is an unweighted graph representing an instance of maximum cut, this reduction constructs a hypergraph with the same node set plus two additional nodes s and t. For each edge $(u, v) \in E$, a hyperedge $\{u, v, s, t\}$ is introduced, which must be cut in the resulting hypergraph s-t cut problem since it contains both s and t. If $w_2 < w_1 = 1$, then a 2-vs-2 split of the hyperedge is cheaper than a 1-vs-3 split. Therefore, the minimum s-t cut is the cut maximizing the number of 2-vs-2 hyperedge splits, which is equivalent to finding a bipartition of nodes in G that maximizes the number of cut edges.

Open questions and motivation   Given the above notation and terminology, our open question can be stated formally as follows.

Open Problem 3.1. What is the complexity of the cardinality-based 4-uniform hypergraph s-t cut problem when $w_2 > 2$ and $w_1 = 1$?

Nothing is known about the computational complexity of the problem except that the graph reduction strategy no longer applies. A reasonable first step is to address the special case obtained by setting $w_1 = 1$ and taking the limit as $w_2 \to \infty$. This limit converges to the NO-EVEN-SPLIT 4-uniform s-t cut problem: minimize the number of 1-vs-3 hyperedge splits when separating s and t, without making any 2-vs-2 hyperedge splits. Placing node s in a cluster by itself always provides one way to separate s from t without having even splits, but the complexity of finding the *minimum* number of 1-vs-3 splits is unknown.

Open Problem 3.2. What is the complexity of the NO-EVEN-SPLIT 4-uniform s-t cut problem?

This special case of Open Problem 3.1 is particularly interesting given its close relationship to an s-t cut problem with a very simple solution. More specifically, this problem at first appears equivalent to setting $w_2 = 1$ and taking a limit as $w_1 \to 0$, but there is a subtle and interesting difference. Setting $w_2 = 1$ and $w_1 = 0$ results in a degenerate problem that is easily optimized, since separating s from the rest of the nodes has a cut penalty of 0. However, if $w_1 = 1$ and $w_2 \to \infty$, it may never be optimal to separate s (or t) from all other nodes. Despite its close relationship to a degenerate and

easily solved problem, it is not clear if NO-EVEN-SPLIT 4-uniform s-t cut is NP-hard, or if there is a strategy leading to a polynomial time solution.

A solution to the NO-EVEN-SPLIT problem would hopefully shed light on other polynomial time algorithms or hardness results for the parameter region $w_2 \in (2, \infty)$. In hypergraphs where the maximum hyperedge size is greater than 4, there is an even larger gap between known NP-hardness results and polynomial time solutions. When hyperedges are of arbitrary size and cut penalties satisfy $w_i = f(i)$ for some increasing concave function f, the hypergraph problem can be reduced to a directed graph s-t cut problem [7]. There are also some parameter regimes where the problem is known to be NP-hard, but there are many parameter settings where neither hardness results nor polynomial time solutions are known. New techniques for the 4-uniform case would hopefully help close these gaps as well.

In addition to their theoretical value, answers to these questions have the potential to advance the state of the art in practical hypergraph algorithms for downstream applications. Generalized hypergraph cut problems are already being used in data mining and machine learning applications such as node classification [4, 5, 8] and image segmentation [6], but existing methods focus only on the regime where cut penalties are submodular. Cut penalties are also cardinality-based in most applications [7], in which case this submodularity property exactly corresponds to cut penalties defined by a concave function f. If, for example, penalties were instead chosen so that $w_i = g(i)$ for an increasing and *strictly convex* function g, this would provide even more incentive to assign nodes from the same hyperedge to the same cluster or node label. However, the computational complexity of finding minimum cuts in this setting is unknown. The 4-uniform hypergraph s-t cut problem with $w_2 > 2$ is the simplest example of this type of convex-penalty hypergraph cut problem. An answer to Open Problem 3.1 would provide a needed first step in understanding what is algorithmically possible for this and many other types of generalized hyperedge cut penalties.

References

[1] Lester Randolph Ford and Delbert R Fulkerson. "Maximal flow through a network". In: *Canadian Journal of Mathematics* 8 (1956), pp. 399–404. MR0079251

[2] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 2010.

[3] E. L. Lawler. "Cutsets and partitions of hypergraphs". In: *Networks* 3.3 (1973), pp. 275–285. DOI: 10.1002/net.3230030306. eprint:

https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.3230030306. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/net.3230030306.

[4] Pan Li and Olgica Milenkovic. "Submodular Hypergraphs: p-Laplacians, Cheeger Inequalities and Spectral Clustering". In: *Proceedings of the 35th International Conference on Machine Learning.* Vol. 80. ICML '18. PMLR, 2018, pp. 3014–3023. URL: http://proceedings.mlr.press/v80/li18e.html.

[5] Nate Veldt, Austin R. Benson, and Jon Kleinberg. "Minimizing Localized Ratio Cut Objectives in Hypergraphs". In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* KDD '20. 2020.

[6] Nate Veldt, Austin R Benson, and Jon Kleinberg. "Approximate Decomposable Submodular Function Minimization for Cardinality-Based Components". In: *Advances in Neural Information Processing Systems* 34 (2021).

[7] Nate Veldt, Austin R Benson, and Jon Kleinberg. "Hypergraph cuts with general splitting functions". In: *SIAM Review* 64.3 (2022), pp. 650–685. MR4461566

[8] Yu Zhu and Santiago Segarra. "Hypergraph cuts with edge-dependent vertex weights". In: *Applied Network Science* 7.1 (2022), pp. 1–20.

4. The edge and vertex elimination problems in directed acyclic graphs

Uwe Naumann

Introduction Impressive progress in the development of computer hard- and software has been made over the past decades. Consequently, numerical simulation has become one of the pillars of science and engineering. Practically relevant real-world phenomena are modelled mathematically. Their numerical evaluation yields multivariate vector functions $F : \mathbb{R}^n \to \mathbb{R}^m$ implemented as often highly complex computer programs.

Algorithmic Differentiation [7, 11] of numerical programs plays a central role in numerous areas of computational science and engineering including error estimation, uncertainty quantification, parameter sensitivity analysis, model calibration and optimization. The resulting derivative programs can be used, for example, to compute the Jacobian (matrix) of (a differentiable) F.

The data flow in a numerical program induces a directed acyclic graph (dag) $G = (V, E)$ with integer vertices representing all input, intermediate

and output values v_i, $i = 1, \ldots, |V|$ of the program and directed edges $E \subseteq V \times V$ modelling data dependence. Association of local partial derivatives $d_{i,j} = \frac{\partial v_j}{\partial v_i}$ with all edges yields the chain rule of differentiation as

$$(2) \qquad \frac{dv_t}{dv_s} = \sum_{(s,*,t)} \prod_{(i,j) \in (s,*,t)} d_{i,j} \, ,$$

where summation is over all paths $(s, *, t)$ connecting a vertex s with a vertex t [1]. (Partial) Derivatives are thus defined for arbitrary pairs of values represented by both vertices.

Formal statement of the problem

Definition 4.1 (Edge Elimination). Let $G = (V, E)$ and $(i, j) \in E$. *Front-elimination* of (i, j) yields $G - (i, j) \equiv G^f = (V^f, E^f)$ such that

$$V^f = \begin{cases} V & \text{if } |P_i| > 1 \\ V \setminus j & \text{otherwise} \end{cases} \quad \text{and} \quad E^f = E \cup \{(i, k) : k \in S_j\}$$

at the cost of $|S_j|$.

    Back-elimination of (i, j) yields $G - (j, i) \equiv G^b = (V^b, E^b)$ such that

$$V^b = \begin{cases} V & \text{if } |S_i| > 1 \\ V \setminus i & \text{otherwise} \end{cases} \quad \text{and} \quad E^b = E \cup \{(k, j) : k \in P_i\}$$

at the cost of $|P_i|$.

    Front-[back-]eliminatable edges are required to have non-empty successor [predecessor] sets. By the chain rule of differentiation, all *complete* edge elimination sequences transform G into a bipartite DAG representing the Jacobian matrix of the underlying numerical program. The cost of an edge elimination sequence is the sum of the costs of the individual edge eliminations. We aim to minimize the cost over all complete edge elimination sequences.

Open Problem 4.1 (EDGE ELIMINATION). Given a directed acyclic graph G and a positive integer $k \geq 0$, is there a complete edge elimination sequence with cost less than or equal to k?

    The elimination of a vertex is equivalent to front-elimination of its in-edges. Similarly, it is equivalent to back-elimination of its out-edges.

Definition 4.2 (Vertex Elimination)**.** Let $G = (V, E)$ and $j \in V$. *Elimination* of j yields $G - j \equiv G' = (V', E')$ such that

$$V' = V \setminus j \quad \text{and} \quad E' = E \cup \{(i, k) : i \in P_j \text{ and } k \in S_j\}$$

at the cost of $|P_j| \cdot |S_j|$.

Eliminatable vertices are required to have non-empty predecessor and successor sets. By the chain rule of differentiation, all *complete* vertex elimination sequences transform G into a bipartite DAG representing the Jacobian matrix of the underlying numerical program. The cost of a vertex elimination sequence is the sum of the costs of the individual vertex eliminations. We aim to minimize the cost over all complete vertex elimination sequences.

Open Problem 4.2 (VERTEX ELIMINATION)**.** Given a directed acyclic graph G and a positive integer $k \geq 0$, is there a complete vertex elimination sequence with cost less than or equal to k?

What is [not] known   Both edge and vertex elimination terminate. Structural and numerical correctness follows by the chain rule [9].

Theorem 4.3. JACOBIAN ACCUMULATION *is NP-complete.*

The proof, see [10], uses reduction from ENSEMBLE COMPUTATION [4]. It exploits potential algebraic dependences (in particular, equality) among local partial derivatives (labels on edges in the DAG). Under the same assumptions both EDGE ELIMINATION and VERTEX ELIMINATION turn out to be NP-complete. The computational complexity of the purely structural formulations in Defs. 4.1 and 4.2 is unknown.

Exiting heuristics for computing good vertex and edge elimination sequences are based on the structural formulations, for example [2, 3, 5, 12].

Incomplete elimination sequences are required in the case of *scarcity* [6]. The computational complexity of the associated combinatorial MINIMUM EDGE COUNT problem asking for a (matrix-free) representation of the Jacobian as a dag with a minimal number of edges is unknown [8]. Ultimately, combinations EDGE ELIMINATION and VERTEX ELIMINATION with MINIMUM EDGE COUNT should be considered.

References

[1] W. Baur and V. Strassen. "The Complexity of Partial Derivatives". In: *Theoretical Computer Science* 22 (1983), pp. 317–330. issn: 0304-3975. MR0693063

[2] J. Chen et al. "An Integer Programming Approach to Optimal Derivative Accumulation". In: *Recent Advances in Algorithmic Differentiation*. Ed. by S. Forth et al. Vol. 87. Lecture Notes in Computational Science and Engineering. Berlin: Springer, 2012, pp. 221–231. isbn: 978-3-540-68935-5. DOI: 10.1007/978-3-642-30023-3_20.

[3] S. Forth et al. "Jacobian Code Generated by Source Transformation and Vertex Elimination can be as Efficient as Hand-Coding". In: *ACM Transactions on Mathematical Software* 30.3 (2004), pp. 266–299. URL: http://doi.acm.org/10.1145/1024074.1024076. MR2124393

[4] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. First Edition. W. H. Freeman, 1979. MR0519066

[5] A. Griewank and U. Naumann. "Accumulating Jacobians as chained sparse matrix products". In: *Mathematical Programming, Ser. A* 95.3 (2003), pp. 555–571. issn: 0025-5610. URL: http://dx.doi.org/10.1007/s10107-002-0329-7. MR1969765

[6] A. Griewank and O. Vogel. "Analysis and Exploitation of Jacobian Scarcity". In: *Modeling, Simulation and Optimization of Complex Processes*. Ed. by H. Bock et al. Berlin: Springer, 2005, pp. 149–164.

[7] A. Griewank and A. Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. 2nd. Other Titles in Applied Mathematics 105. Philadelphia, PA: SIAM, 2008. isbn: 978–0–898716–59–7. MR2454953

[8] V. Mosenkis and U. Naumann. "On optimality preserving eliminations for the minimum edge count and optimal Jacobian accumulation problems in linearized DAGs". In: *Optimization Methods and Software* 27.2 (2012), pp. 337–358. MR2901965

[9] U. Naumann. "Optimal accumulation of Jacobian matrices by elimination methods on the dual computational graph". In: *Mathematical Programming, Ser. A* 99.3 (2004), pp. 399–421. issn: 0025-5610. URL: http://dx.doi.org/10.1007/s10107-003-0456-9.

[10] U. Naumann. "Optimal Jacobian accumulation is NP-complete". In: *Mathematical Programming, Ser. A* 112.2 (2008), pp. 427–441. issn: 0025-5610. URL: http://dx.doi.org/10.1007/s10107-006-0042-z.

[11] U. Naumann. *The Art of Differentiating Computer Programs: An Introduction to Algorithmic Differentiation*. Software, Environments, and Tools 24. Philadelphia, PA: SIAM, 2012. isbn: 978–1–611972–06–1. URL: http://bookstore.siam.org/se24. MR2894688

[12] J. Pryce and M. Tadjouddine. "Fast Automatic Differentiation Jacobians by Compact LU Factorization". In: *SIAM Journal on Scientific*

5. Data flow reversal problems

Uwe Naumann

Introduction Impressive progress in the development of computer hard-
and software has been made over the past decades. Consequently, numerical
simulation has become one of the pillars of science and engineering. Prac-
tically relevant real-world phenomena are modelled mathematically. Their
numerical evaluation yields multivariate vector functions $F : I\!R^n \to I\!R^m$:
$\mathbf{y} = F(\mathbf{x})$ implemented as often highly complex computer programs.

Evaluation of a numerical program at a given input induces a directed
acyclic graph (DAG) $G = (V, E)$. Its integer vertices $V = (X, Z, Y)$ consist
of $n \geq 1$ sources $X = \{1, \ldots, n\}$ representing $\mathbf{x} \in I\!R^n$, $p \geq 0$ intermediate
vertices $Z = \{n + 1, \ldots, n + p\}$ and $m \geq 0$ sinks $Y = \{n + p + 1, \ldots, n + p + m\}$ representing $\mathbf{y} \in I\!R^m$. Intermediate vertices and sinks model calls to
elemental functions

$$\mathbf{v}_j = F_j(\mathbf{v}_i)_{(i,j)\in E} \quad \text{for } j = n + 1, \ldots, n + p + m .$$

Directed edges $E \subseteq V \times V$ are due to corresponding data dependence within
the program. A topological order of the vertices is implied. DAGs of numer-
ical simulations become very large. They are nonpersistent in the sense that
they cannot be stored in memory.

Algorithmic Differentiation (AD) [7, 13] of numerical programs in ad-
joint mode plays a central role in numerous areas of computational science
and engineering including error estimation, uncertainty quantification, pa-
rameter sensitivity analysis, model calibration and optimization. It yields
efficient ($\mathcal{O}(m)$) gradients ($m = 1$) with machine accuracy which would
otherwise have to be approximated numerically with a cost of $\mathcal{O}(n)$ where
often $n \gg 1$. Refer to www.autodiff.org for links to AD projects and research
groups as well as for an extensive bibliography on the subject.

The *algorithmic adjoint* evaluates $\bar{\mathbf{x}} = \bar{\mathbf{x}} + F'(\mathbf{x})^T \cdot \bar{\mathbf{y}}$ as

$$\bar{\mathbf{v}}_i = \bar{\mathbf{v}}_i + F'_j(\mathbf{v}_i)_{(i,j)\in E} \cdot \bar{\mathbf{v}}_j \quad \text{for } j = n + p + m, \ldots, n + 1$$

by accessing the nonpersistent $\mathbf{v}_i$ in reverse order. Equivalently, the DAG
G is reversed by accessing its vertices in reverse order. Values lost due to

overwriting of program variables in the given implementation of F need to be recovered. Naively, all overwritten *required* [1, 9] values could be pushed onto a stack. The persistent memory required for data flow reversal becomes maximal ($\mathcal{O}(n + p)$). Alternatively, all overwritten required values could be recomputed from the persistent inputs, which yields minimal persistent memory requirement ($\mathcal{O}(n)$). Reversal of the flow of data in nonpersistent memory results in quadratic computational cost ($\mathcal{O}(p^2)$). Each intermediate value needs to be recomputed from $\mathbf{x}$. Without loss of generality, unit cost can be assumed for all F_j. The combinatorial DAG REVERSAL problem aims to minimize the computational cost of data flow reversal in limited ($\ll \mathcal{O}(p^2)$) persistent memory.

Applications of data flow reversal beyond algorithmic adjoints include reverse debugging of large-scale numerical simulations [4] and checkpointing for resilience of computer systems [16].

Formal statement of the problem

Open Problem 5.1 (DAG REVERSAL)**.** Given a DAG G and two integers $M \geq n$ and $O \geq 0$ can G be reversed within persistent memory of size less than or equal to M with computational cost less than or equal to O?

What is [not] known

Theorem 5.1. DAG REVERSAL *is NP-complete.*

The proof can be found in [12]. The related CALL TREE REVERSAL problem is also known to be NP-complete [11].

The state of the art in methods for data flow reversal is mostly driven by solutions for special cases due to given simulation scenarios [3, 15, 17]. Optimal checkpointing methods for certain types of evolutions (DAG is a simple chain) were developed [5, 6]. Generalizations include mixed-integer programming for CALL TREE REVERSAL [10] and divide-and-conquer approaches [14]. Refer to [8] for additional information.

The combinatorial data flow reversal problems are expected to benefit from further formal theoretical analysis. In particular, there are no approximation methods available.

References

[1] B. Dauvergne and L. Hascoët. "The Data-Flow Equations of Checkpointing in Reverse Automatic Differentiation". In: *Computational Science – ICCS 2006*. Ed. by V. Alexandrov et al. Vol. 3994. Lecture Notes in Computer Science. Heidelberg: Springer, 2006, pp. 566–573.

[2] C. Bischof et al., eds. *Advances in Automatic Differentiation*. Lecture Notes in Computational Science and Engineering (LNCSE) 64. Springer, 2008.

[3] I. Charpentier. "Checkpointing Schemes or Adjoint Codes: Application to the Meteorological Model Meso-NH". In: *SIAM Journal on Scientific Computing* 22.6 (2001), pp. 2135–2151. MR1856306

[4] J. Engblom. "A review of reverse debugging". In: *Proceedings of the 2012 System, Software, SoC and Silicon Debug Conference* (2012), pp. 1–6.

[5] A. Griewank. "Achieving Logarithmic Growth of Temporal and Spatial Complexity in Reverse Automatic Differentiation". In: *Optimization Methods and Software* 1 (1992), pp. 35–54.

[6] A. Griewank and A. Walther. "Algorithm 799: Revolve: An Implementation of Checkpointing for the Reverse or Adjoint Mode of Computational Differentiation". In: *ACM Trans. Math. Softw.* 26.1 (Mar. 2000), pp. 19–45.

[7] A. Griewank and A. Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. 2nd. Other Titles in Applied Mathematics 105. Philadelphia, PA: SIAM, 2008. isbn: 978–0–898716–59–7. MR2454953

[8] L. Hascoët. "Reversal Strategies for Adjoint Algorithms". In: *From Semantics to Computer Science. Essays in memory of Gilles Kahn*. Ed. by Y. Bertot et al. Cambridge University Press, 2009, pp. 487–503.

[9] L. Hascoët, U. Naumann, and V. Pascual. "To-Be-Recorded Analysis in Reverse Mode Automatic Differentiation". In: *Future Generation Computer Systems* 21 (2005), pp. 1401–1417.

[10] J. Lotz, U. Naumann, and S. Mitra. "Mixed Integer Programming for Call Tree Reversal". In: *Seventh SIAM Workshop on Combinatorial Scientific Computing*. Proceedings. SIAM, 2016, pp. 83–91.

[11] U. Naumann. "Call Tree Reversal is NP-complete". In: *[2]*. Springer, 2008, pp. 13–22. MR2531676

[12] U. Naumann. "DAG Reversal is NP-complete". In: *Journal of Discrete Algorithms* 7 (2009), pp. 402–410. MR2558997

[13] U. Naumann. *The Art of Differentiating Computer Programs: An Introduction to Algorithmic Differentiation*. Software, Environments, and Tools 24. Philadelphia, PA: SIAM, 2012. isbn: 978–1–611972–06–1. URL: http://bookstore.siam.org/se24. MR2894688

[14] J. Siskind and B. Pearlmutter. "Divide-and-conquer checkpointing for arbitrary programs with no user annotation". In: *Optimization Methods & Software* 33.4–6 (2018), pp. 1288–1330. MR3853239

[15] W. Symes. "Reverse time migration with optimal checkpointing". In: *GEOPHYSICS* 72.5 (Sept. 2007), SM213–SM221.

[16] A. Walther and S. Narayanan. "Extending the Binomial Checkpointing Technique for Resilience". In: *2016 SIAM Workshop on Combinatorial Scientific Computing (CSC16)*. extended abstract.

[17] Q. Wang, P. Moin, and G. Iaccarino. "Minimal Repetition Dynamic Checkpointing Algorithm for Unsteady Adjoint Calculation". In: *SIAM Journal on Scientific Computing* 31.4 (2009), pp. 2549–2567. MR2520289

6. Price of asynchrony

Carlos Ortiz Marrero, Stephen J. Young

Introduction   Finding the solution x^* of a large-scale system of linear equations $Ax = b$ is one of the most common computation primitives in scientific computation, forming the backbone of numerical approaches to a variety of applications including computational fluid dynamics, optimization, and atmospheric modeling. While there are many different approaches to solve such systems (for example, Gaussian elimination, Krylov subspace methods, or conjugate gradient methods), iterative approaches are popular because of their straight-forward nature and ease of implementation. Broadly speaking, iterative methods proceed by cleverly selecting an auxiliary matrix M and using M to generate a sequence of approximate solutions $x^{(0)}, x^{(1)}, x^{(2)}, \dots$ with the recurrence $Mx^{(k+1)} = (M - A)x^{(k)} + b$. Thus if systems of the form $My = c$ can be easily solved, then the approximation can be refined for essentially the cost of one matrix-vector multiplication. For example, the (damped) Jacobi iteration is when M is chosen to be the (scaled) diagonal of A and Gauss-Seidel iteration is when M chosen to be the lower-triangular portion of M.

Typically, the analysis of these methods work with the sequence of residuals $r^{(k)} = x^{(k)} - x^*$, rather than directly with the sequence of approximate solutions. It is easy to see that the residual solutions satisfy a homogeneous version of the recurrence which defines the approximate solutions, i.e. $Mr^{(k+1)} = (M - A)r^{(k)}$. Indeed, if M is invertible, then the sequence of residuals is given by $r^{(k)} = C^k r^{(0)}$ where $C = I - M^{-1}A$ is known as the iteration matrix. As a consequence, the iterative approaches to solving $Ax = b$ can be understood in terms of repeated matrix multiplication. Indeed, an iterative method converges independent of the initial approximate solution if and of if the spectral radius of C, $\rho(C)$, is strictly less than

1. Even more is true, specifically, if $\rho(C) < 1$ then $-\log_{10}(\rho(C))$ gives an lower bound on the rate of increase (in terms of number of iterations) of the approximate solutions, i.e. the number decimals of precision of $x^{(k)}$ is approximately $-\log_{10}(\rho(C))k + \log_{10}(\|r^{(0)}\|)$.

While iterative methods are quite effective and practical for solving linear systems, there are significant challenges that arise when considering large scale systems $Ax = b$ such as those that arise in many applications, such as the modeling of wind turbines, the modeling of reactor processes, computational chemistry for catalysis design, and many others. In particular, the matrix A can be so large that it can not be effectively stored in memory (RAM) on a single compute node – this necessitates either taking a significant performance hit by repeatedly transferring portions of A in and out of memory or dividing the system into parts and using many compute nodes to build the solution.

More concretely, suppose that $A \in \mathbf{R}^{n \times n}$ and $b \in \mathbf{R}^n$ and we wish to use compute nodes labeled $1, 2, \ldots, \ell$ to solve the system $Ax = b$. One approach is to partition the rows of A into ℓ sets resulting in rectangular matrices $A_i \in \mathbf{R}^{k_i \times n}$ and have a matching partition for M, x and b. In cases such as Jacobi iteration it is easy to extend the iterative scheme by considering the same partition on the diagonal matrix M, i.e. $M_i x_i^{(k+1)} = (M_i - A_i)x^{(k)} + b_i$, where $x^{(k)}$ is the approximate solution at iteration k reconstructed from the local solutions $x_1^{(k)}, \ldots, x_\ell^{(k)}$. For non-diagonal M, such as with Gauss-Seidel iteration, the naïve approach would be to iteratively solve for the $x_i^{(k+1)}$ using the lower-triangular form and then "pass" the information on $x_i^{(k+1)}$ onto the subsequent compute nodes so that the lower-triangular form can be used to solve $x_{i+1}^{(k+1)}, x_{i+2}^{(k+1)}, \ldots, x_\ell^{(k+1)}$. However, both of these approaches result in computational delays resulting from the time need to communicate between different compute nodes.

The problem is especially acute in large high-performance computing (HPC) systems such as Aurora, Fugaku, and LUMI, where size of the system and over all network congestion can cause significant inter-node communication delays. Indeed, for some scientific computations it is estimated that over 50% of the computational time is taken up by communication requirements [1]. In other contexts it is known that by performing work *asynchronously*, that is, without waiting for the results of computation or work on remote nodes, can significantly increase the performance of HPC systems [3, 4, 2]. However, little is known about how iterative methods for solving linear systems are effected by asynchrony. More generally, we wish to understand the *price of asynchrony*, that is, the effect of asynchronous updates on the convergence rate of iterative methods.

Problem statement As a first step towards understanding the price of asynchrony, we propose a simple model of asynchronous updates. To begin, we assume that the iterative method can be viewed through the recurrence $x^{(k+1)} = Mx^{(k)}$ where M is a $n \times n$ real matrix with $\rho(M) < 1$. To build a simple model of asynchronous updates for M, we partition the matrix M into an $\ell \times \ell$ array of blocks and for each $i \in [\ell]$ define the update scheme

$$\tilde{x}_i^{(k+1)} = \sum_j M_{ij} \tilde{x}_j^{(k-\delta_{ji})},$$

where δ_{ji} represents the "delay" in information from compute node j reaching compute node i (and so we assume that $\delta_{ii} = 0$). The *price of asynchrony* can be viewed as the difference between the rates at which $\left\| x^{(k)} \right\|$ and $\left\| \tilde{x}^{(k)} \right\|$ converge to 0. With this framing, for any $\kappa \geq \max_{i,j} \delta_{ij}$ the asynchronous updates can be viewed as a linear operator on $\mathbf{R}^{(\kappa+1)n}$ defined by $(\tilde{x}^{(\kappa)}, \ldots, \tilde{x}^{(0)})$ is mapped to $(\tilde{x}^{(\kappa+1)}, \ldots, \tilde{x}^{(1)})$ where $\tilde{x}_i^{(\kappa+1)} = \sum_j M_{ij} \tilde{x}_j^{(\kappa-\delta_{ji})}$ for all blocks i. We denote the matrix for this linear operator by $M^{(\delta,\kappa)}$ and note that $M^{(0,\kappa)}$ corresponds to the synchronous update which maintains a history of κ steps.

Lemma 6.1. *Let M be a $n \times n$ real matrix with $\rho(M) < 1$ partitioned into an $\ell \times \ell$ array of blocks. Let δ be an element of $\mathbf{N}^{\ell \times \ell}$ such that $\delta_{ii} = 0$ for all i. Suppose that $\max_{ij} \delta_{ij} \leq \kappa \leq \kappa'$, then $\rho(M^{(\delta,\kappa)}) = \rho(M^{(\delta,\kappa')})$.*

Proof. Suppose that (μ, v) is an eigenpair of $M^{(\delta,\kappa)}$. Since $M^{(\delta,\kappa)}$ takes $(\tilde{x}^{(\kappa)}, \ldots, \tilde{x}^{(0)})$ to $(\tilde{x}^{(\kappa+1)}, \ldots, \tilde{x}^{(1)})$, there is some vector $v_0 \in \mathbf{R}^n$ such that $v = (\mu^\kappa v_0, \mu^{\kappa-1} v_0, \ldots, \mu v_0, v_0)$. Further, since v is an eigenvector of $M^{(\delta,\kappa)}$ we have that $\mu^{\kappa+1}(v_0)_i = \sum_j \mu^{\kappa-\delta^{ji}} v_0$. Now define the vector $v' = (\mu^\kappa v_0, \mu^{\kappa-1} v_0, \ldots, \mu v_0, v_0, \mu^{-1} v_0, \ldots, \mu^{\kappa-\kappa'} v_0)$ and consider $M^{(\delta,\kappa')} v'$. By construction, we have that construction $M^{(\delta,\kappa')} v' = (w', \mu^\kappa v_0, \ldots, \mu^{\kappa-\kappa'+1} v_0)$ for some $w' \in \mathbf{R}^n$. Furthermore, $w_i' = \sum_j \mu^{\kappa-\delta_{ji}} v_0 = \mu^{\kappa+1}(v_0)_i$. As a consequence (μ, v') is a eigenpair for $M^{(\delta,\kappa')}$. A similar argument shows that any eigenpair for $M^{(\delta,\kappa')}$ has a corresponding eigenpair for $M^{(\delta,\kappa)}$. In particular, the two matrices are co-spectral and hence have the same spectral radius. $\square$

As an immediate consequence of Lemma 6.1, we have that $\rho(M^{(0,\kappa)}) = \rho(M)$ for any κ. Furthermore, it suffices to restrict our attention to the spectral radius of $M^{(\delta,\kappa)}$ for any system of delays δ and $\kappa = \max \delta$. Thus, we will abuse notation slightly denote by $M^{(\delta)}$ any member of the family $\left\{ M^{(\delta,k)} \mid k \geq \max \delta \right\}$.

Intuitively, it seems that if we fix a maximum delay k, then $\rho(M^{(\delta)})$ is maximized when the delay between any pair of nodes achieves the maximum delay. In this case, uniformity of the delays allows for an explicit calculation of $\rho(M^{(\delta)})$.

Lemma 6.2. *Let M be an $n \times n$ real matrix with $\rho(M) < 1$ partitioned into an $\ell \times \ell$ array of blocks. Let δ be an element of $\mathbf{N}^{\ell \times \ell}$ such that $\delta_{ij} = k$ for all non-zero blocks of M, then $\rho(M^{(\delta)}) = \rho(M)^{1/k+1}$.*

Proof. As noted in Lemma 6.1, any eigenpair (μ, v) of $M^{(\delta,\kappa)}$ has the form $v = (w, \mu^{-1}, \ldots, \mu^{-\kappa}w)$, where $\mu w_i = \sum_j \mu^{-\delta_{ij}} M_{ij} w_j$. However, since $\delta_{ij} = k$ for any non-zero block of M, this implies that $\mu w = \mu^{-k} M w$ and (μ^{k+1}, w) is an eigenpair for M. It is easy to see that a similar construction will take an eigenpair for M to an eigenpair for $M^{(\delta,\kappa)}$. The desired result on the spectral radius follows immediately. $\qquad\square$

This leads naturally to the following open questions:

Open Problem 6.1. Let M be a $n \times n$ real matrix with $p(M) = \lambda < 1$ partitioned into an $\ell \times \ell$ array of blocks. Is it the case that for all elements δ of $[k]^{\ell \times \ell}$ with zero diagonals, $\rho(M^{(\delta)}) \leq \lambda^{1/k+1}$? If not, is there some function $f \colon (0,1) \times \mathbf{N} \to (0,1)$ such that $\rho(M^{(\delta)}) \leq f(\lambda, k)$?

Intuitively, it seems as if this is the worst possible case for the convergence rate, that is, if the all the delays are at most k.

Experimental results As a first pass towards resolving this problem, consider a random matrix M selected from one of three different ensembles derived from a matrix X with independent normally distributed entries with mean zero; $M = X$, the Gaussian Orthonormal Ensemble where $M = X + X^T$, and the Wishart ensemble where $M = XX^T$. For notational convenience, we assume that M is re-scaled to having unit spectral radius. We also consider, for each of these matrix ensembles, the iteration matrix associated with a block Jacobi iteration (again, normalized to have norm 1), that is, the diagonal blocks are zero and the off-diagonal blocks are given by $M_{ii}^{-1} M_{ij}$ where M_{xy} denotes the x, y block in M. In each of these six, cases we also consider two different delay patterns with 4 compute nodes; a single node with a delay of 5 and all others having a delay of 0 (Figure 4a) and randomly generated Poisson delays with a mean of 3 (Figure 4b). In Figure 5, we plot the relationship between $\rho((cM)^{(\delta)})$ versus $c \in (0,1)$ for each of these graphs over 50 different trials.

These experiments would seem to provide at least some support for the conjecture that $\rho(M^{(\delta)}) \leq \rho(M)^{1/\max \delta}$, at least when $\rho(M)$ is sufficiently
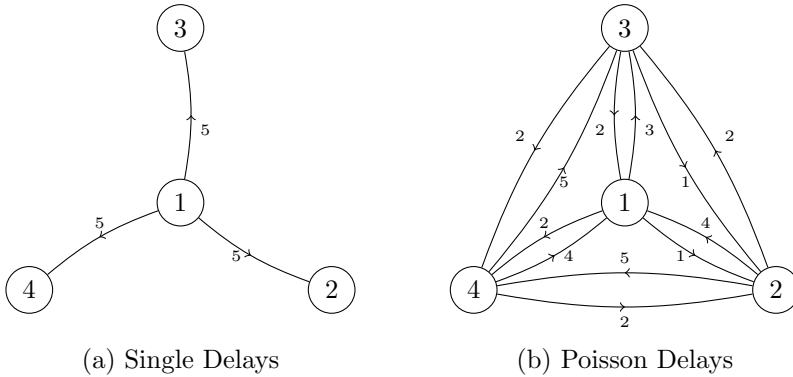
(a) Single Delays (b) Poisson Delays

Figure 4: Delay patterns for asynchronous computation. A weight d edge directed from j to i represents a delay of d time steps in information propagating from j to i, i.e. $\delta_{ji} = d$.

small. In the case where $\rho(M)$ is close to one, we expect that at least part of the violation of the conjecture may be as a result of numerical instabilities in the spectral radius calculation. In particular, from the proofs of Lemma 6.1 and 6.2, it easy to see that despite $M^{(\delta,\kappa)}$ being an $(\kappa+1)n \times (\kappa+1)n$ matrix, the sum of the geometric multiplicities is at most n. To further complicate matters, $M^{(\delta)}$ is non-Hermitian and has entries of both positive and negative signs, increasing the likelihood that the eigenvalue which yields the spectral radius is complex.

It is also interesting to note that there is relatively little difference between the behavior of the full matrix iteration and the block Jacobi iteration when there is a single constant delay, while for Poisson delays the block Jacobi iteration has a significantly higher spectral radius. One possible explanation is to consider the average effective delays (define as the mean delay over non-zero blocks of M) which is $15/16$ for a single delayed node, $15/12$ for Jacobi iteration with a single delayed node, $33/16$ for Poisson delays, and $33/12$ for Jacobi iteration with Poisson delays. As a result, one might speculate that the important parameter for understanding $\rho(M^{(\delta)})$ is the mean delay, perhaps weighted by the spectral radius of the individual blocks.

(a) Single Delay

(b) Poisson Delays

(c) Single Delay, Block Jacobi
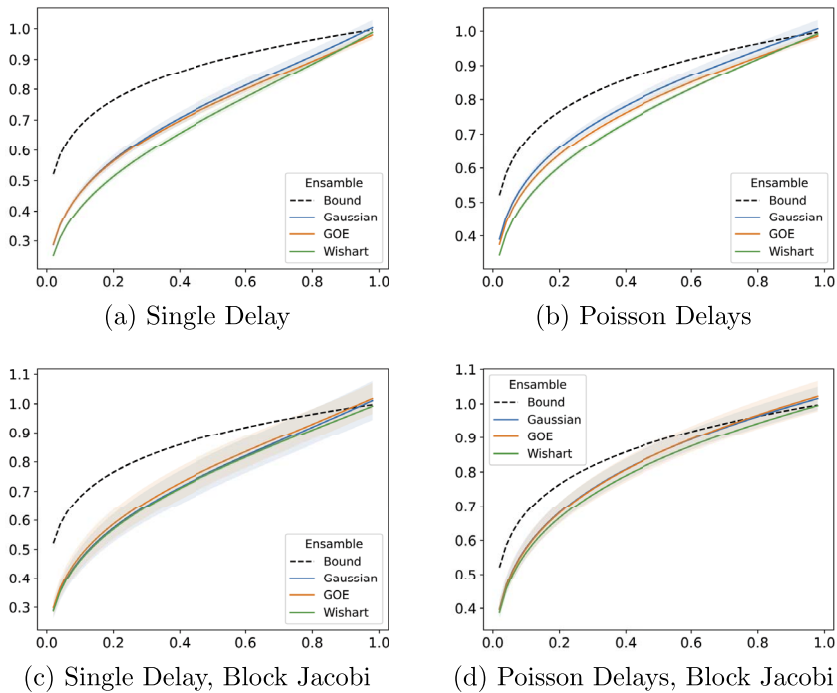
(d) Poisson Delays, Block Jacobi

Figure 5: Price of Asynchrony. For each of the three matrix ensembles, the solid line represents the average spectral radius over 50 different samples from the ensemble, while the shaded area represents a range of two standard deviations. The dashed line is the conjectured upper bound in terms of the spectral radius of the synchronous update matrix.

References

[1] Nikhil Jain. "Optimization of communication intensive applications on HPC networks". PhD thesis. University of Illinois at Urbana-Champaign, 2016.

[2] Joshua D. Suetterlein et al. "Extending the Roofline Model for Asynchronous Many-Task Runtimes". In: *2016 IEEE International Conference on Cluster Computing (CLUSTER)*. 2016, pp. 493–496. DOI: 10.1109/CLUSTER.2016.47.

[3] Joshua Suetterlein et al. "Asynchronous Runtimes in Action: An Introspective Framework for a Next Gen Runtime". In: *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. 2016, pp. 1744–1751. DOI: 10.1109/IPDPSW.2016.191.

[4] Joshua Suetterlein et al. "On the Marriage of Asynchronous Many Task Runtimes and Big Data: A Glance". In: *2020 IEEE 27th International Conference on High Performance Computing, Data, and Analytics (HiPC)*. 2020, pp. 233–242. DOI: 10.1109/HiPC50609.2020.00037.

7. $(n-k)$-contingent zero-forcing for power grids

Sinan G. Aksoy, Anthony V. Petyuk, Sandip Roy, Stephen J. Young

Introduction With the increasing penetration of microgrids, renewable energy sources, power-generation at the edge, and distributed energy resources, maintaining the stability of power grid transmission has changed dramatically. For example, the majority of grid power generation used to be achieved by spinning large turbines (e.g. hydro-electric turbines spun by falling water, steam powered turbines in gas/coal/nuclear power plants). However, in 2022 almost 14% of the utility scale generation was from wind or solar [14]. In contrast to turbine-based generation, neither wind nor solar power has any "inertia" and so is subject to rapid changes in the total generation which are not controlled by operators. This puts significant pressure on control schemes designed to manage the stability of the power grid through the regulation of power generation. Such control schemes are also made challenging by the increasing penetration of microgrids in the power system: these are small, regional areas of the power grid which contain sufficient internal generation that disconnect themselves from the broader power grid in order to improve local stability. As microgrids change their connectivity to the power grid, the fundamental electrical equations which govern power flow on the grid also change, impacting control schemes.

The changing grid not only poses challenges for stable control, but also presents several opportunities to increase both resilience and efficiency. For example, phasor measurement units (PMUs) have been deployed at selected places within the power-grid over the last 25 years, providing measures of the local power grid state (i.e., voltage magnitude and phase angle) up to 120 times a second – a significant increase over previous methods yielding one measurement every few seconds. Another opportunity to increase the resilience and stability of the grid is afforded by the increasing penetration

of distributed energy resources (DERs), such as grid-scale battery installations. Not only do these large battery installations provide a means to temporally arbitrage power from renewable sources (e.g., store excess power generated during the day by solar panels to use at night), they also provide alternative points of control for the power grid via selective charging, islanding, and discharging to rapidly inject power from the overall grid system. Furthermore, because of the significantly smaller footprint of grid-scale batteries (as compared to traditional power plants) it is possible to deploy these resources in locations which are advantageous to control and stability.

Given this variety of power-grid structures and operating points, it might seem nearly intractable to decide whether a collections of PMUs provide sufficient observability of the state of grid, or whether a collection DERs are able to provide stabilization for the grid. Fortunately, both of these problems can be reduced to a combinatorial problem on the structure of the grid known as zero-forcing.

Definition 7.1 (Zero-Forcing Move). Given a graph $G = (V, E)$ and a set $S \subseteq V$ of vertices colored blue. A vertex $v \in V - S$ can be colored by a *zero-forcing move* if there is a vertex s such that $\{v\} = N(s) - S$, that is, if there is a vertex $s \in S$ such that v is the unique neighbor of s not colored blue. If v is colored via a zero-forcing move that relies on s, we will say that s is used to color v.

Definition 7.2 (Zero-Forcing Set). Given a graph $G = (V, E)$, a set $Z \subseteq V$ is called a *zero-forcing set* if there is a sequence of zero-forcing moves such that all vertices in V are colored.

There is an extensive literature around determining the size and structure of minimal zero-forcing sets (see for instance [9] and the references therein). Concerning power-grid applications, [11] shows the linear dynamics associated to the power-flow equations are controllable by controllers at a zero-forcing set, and [4, 13] shows if phasor-measurement units are placed at a power-dominating set[1] then the phase of every bus in the network is recoverable. Consequently, distributed energy resource placement (such as grid-scale batteries) and phasor-measurement units may enable the real-time steering of the power-grid to increase efficiency and resiliency. In addition, recent work [12] has shown zero-forcing sets can be used to identify control points for microgrids. However, these efforts fail to take into account the dynamic nature of the grid. In particular, zero-forcing is defined for a static

---

[1]A power-dominating set is a set of vertices S such that, combined with their neighborhood, they form a zero-forcing set.

graph which does not allow for changes in the underlying network topology, limiting its applicability to the observability and control of the power grid.

Contingent zero-forcing To address the limitations of zero-forcing in the context of the power grid, we propose the following extension of the zero-forcing to identify resilient sets which can achieve the observability and control of the grid.

Definition 7.3 $((n - k)$-contingent Zero Forcing)**.** Given a graph $G = (V, E)$, a set $Z \subseteq V$ is a $(n - k)$-contingent zero-forcing set if for every set of edges E' of size at most k, Z is a zero-forcing set for $G' = (V, E - E')$.

We note the edge-leaky zero-forcing number introduced independently by Alameda, Kritschgau, and Young [2] is essentially equivalent to this definition[2]. Since the transmission and distribution level of power grids frequently exhibit "tree-like" structure, it is natural to first consider nature of the $(n - k)$-contingent zero-forcing sets on trees. To explore this, we first recall the following fact about the zero-forcing sets of trees.

Theorem 7.4 ([6])**.** *If T is a tree with $t \geq 2$ leaves, then any collection of at least $t - 1$ leaves is a zero-forcing set for T.*

In the setting of $(n - k)$-contingent zero-forcing, we apply this to prove the following lemma.

Lemma 7.5. *Let $T = (V, E)$ be a tree. For any $k \geq 1$, a set $Z \subseteq V$ is a $(n - k)$-contingent zero-forcing set if and only if Z contains all vertices of degree at most k.*

Proof. If Z is a $(n - k)$-contingent zero-forcing set for any graph, then it must contain all vertices of degree at most k. In particular, if any vertex v of degree at most k is not in Z then removing all of the incident edges to v results in a graph where Z is not a zero-forcing set.

Now let Z be the set of vertices of degree at most k in T and let F be the forest formed by removing a set of k edges from T. We first note that any isolated vertex in F has degree at most k in T as we removed at most k

---

[2]In contrast to $(n - k)$-contingent zero-forcing, edges in the edge-leaky zero-forcing process aren't removed from the graph, rather their use is prohibited. Consequently, it is possible that in some stage of the zero-forcing process the contingent zero-forcing process can force a strict superset of the vertices forcible in the equivalent edge-leaky process. However, Alameda, Kritschgau, and Young go on to show the ℓ-edge-leaky zero-forcing sets are the same as ℓ-leaky zero-forcing sets. Thus, by Lemma 7.7 ℓ-edge leaky zero-forcing is equivalent to $(n-\ell)$-contingent zero-forcing.

edges. Suppose there is some tree T' which has two leaves $u, v \notin Z$. We note that since u and v are in the same tree in F, there edge $\{u, v\}$ is not in T. Thus, in order for both u and v to be leaves in F at least $k + k > k$ edges must be removed, contradicting the construction of F. Thus, every tree in F is either an isolated vertex belong to Z or a tree with at most one leaf not in Z. Thus, by Theorem 7.4, every tree in F is forcible by Z.   □

The idea of $(n-k)$-contingent zero-forcing is the not the first attempt to develop a notion of zero-forcing that is resilient to changes in the underling graph. For example, Dillman and Kenter [6] consider the resilience in the context of water flows where water can "leak" out at the joints preventing vertices from participating in the forcing process. More formally, they give the following definition:

Definition 7.6 (ℓ-leaky Zero-Forcing). Given a graph $G = (V, E)$, a set $Z \subseteq V$ is an ℓ-leaky zero-forcing set if for every set of vertices $L \subseteq V$ of size at most ℓ, Z is a zero-forcing set of G', where G' is formed by adding a pendant vertex to each vertex in L.

Observe that any vertex $v \in L$ can not be used to color any other vertex via a zero-forcing move. In other words, the zero-forcing process "leaks" out of the graph on the vertices in L so they are only able to result in the coloring of new pendant vertices. In many ways, the leaky zero-forcing process can be thought of as an vertex version of the contingent zero-forcing process: in leaky zero-forcing an unknown set of vertices can not participate in the zero-forcing process, while in the contingent zero-forcing process an unknown set of edges can not participate in the zero-forcing process. With this observation, it is unsurprising these two processes are effectively equivalent, as we show below.

Lemma 7.7. *A set S is a $(n-k)$-contingent zero-forcing set if and only if it is an k-leaky zero-forcing set.*

Proof. Let G be a graph and suppose Z is a $(n-k)$-contingent zero-forcing set for G. Fix an arbitrary set of L leaks in the graph G, where $|L| \leq k$. We will identify a set $\mathcal{E}$ of at most ℓ edges such that there is a forcing sequence of $G - \mathcal{E}$ which does not use any vertex in L. To this end, we iteratively build the set $\mathcal{E}$ of removed edges based on Z. Note that as long as $|\mathcal{E}| \leq k$, the forcing process from Z can continue as Z is a $(n-k)$-contingent zero-forcing set. Now, to identify these edges of $\mathcal{E}$, consider sequentially applying the forcing process until the first time a vertex $v \in L$ is used to color u. At this point, add the edge $\{v, u\}$ to $\mathcal{E}$ and remove it from the graph G. Since all

of the neighbors of v are now colored, there is no future step in the forcing process in which v colors a neighbor. Thus we may effectively remove v from L. Repeating this process results in a set $\mathcal{E}$ of size at most k and a sequence of zero-forcing moves for $G - \mathcal{E}$, such that no vertex in L forces any other vertex. Thus S is a forcing set for G with leaks at all the vertices in L.

The converse proceeds similarly. Let Z be a k-leaky zero-forcing set for G and fix an arbitrary set of $\mathcal{E}$ edges in G. We again interatively apply the zero-forcing process and build a set of leaks L as needed. Specifically, for each edge $\{u, v\} \in E$ we add the first vertex from the edge that is colored to the set of leaks L. Clearly, this ensures that the edge $\{u, v\}$ is not used in the forcing process and adds at most k vertices to the set L. $\qquad \square$

In general, determining the size of the minimal zero-forcing set is $\mathcal{NP}$-complete, even for highly-restricted classes of graphs (see [8], for instance). Thus, it is exceedingly likely that determining the size of the minimal $(n-k)$-contingent zero-forcing number is also $\mathcal{NP}$-complete. However, the class of graphs corresponding to power-grid networks is known to have a number of unusual structural properties (see, for instance [1, 15]), including average degree between 1 and 2, diameter scaling like the square root of the number of vertices, and the presence of long-cycles. In other words, typical power-grid networks are tree-like by Lemma 7.5 the minimal $(n - k)$-contingent zero-forcing sets are known exactly for trees. This leads naturally to our first two open problems:

Open Problem 7.1. Is there a structural characterization of the minimal-sized $(n - k)$-contingent zero-forcing sets for graphs with few cycles? For sparse graphs with large diameter?

Open Problem 7.2. Is there an efficient algorithm to determine the minimal-sized $(n - k)$-contingent zero-forcing set for graphs with few cycles? For sparse graphs with large diameter?

Even if the structural properties of power-grid networks are insufficient to yield an efficient means of computing a minimal, or nearly so, $(n - k)$-contingent zero-forcing set, if such a set is sufficiently small the additional computational effort may be worth while. However, in order to assess the value of identifying the minimal $(n-k)$-contingent zero-forcing sets, it would be helpful to have a rough estimate of size of the resulting set.

Open Problem 7.3. What is the typical size, in terms of the number of vertices and edges, of the minimal $(n - k)$-contingent zero-forcing sets in graphs exhibiting structural properties characteristic of power grids?

While this question may be answerable by restricting attention to the class of graphs with a few hallmark structural properties (e.g., limitation on density, diameter, maximum degree, etc.) it may be more useful to build off the recent work characterizing the zero-forcing number for the Erdős-Renyí random graph [7] and random regular graphs [3, 10]. While the structural properties of these random graph models are far from those of the power-grid, there are a few recent models such as the Chung-Lu Chain [1] and the Geometric Delaunay [5] models which have been shown to capture an array of structural properties of the power grid.

References

[1] Sinan G Aksoy et al. "A generative graph model for electrical infrastructure networks". In: *Journal of Complex Networks* 7.1 (Aug. 2018), pp. 128–162. MR3910485

[2] Joseph S Alameda, Juergen Kritschgau, and Michael Young. "Generalizations of Leaky Forcing". In: *arXiv preprint arXiv:2009.07073* (2020).

[3] Deepak Bal et al. "Zero-forcing in random regular graphs". In: *J. Comb.* 12.1 (2021), pp. 85–116. issn: 2156-3527. doi: 10.4310/JOC.2021.v12.n1.a4. url: https://doi.org/10.4310/JOC.2021.v12.n1.a4. MR4195585

[4] Dennis J. Brueni and Lenwood S. Heath. "The PMU placement problem". In: *SIAM J. Discrete Math.* 19.3 (2005), pp. 744–761. issn: 0895-4801. doi: 10.1137/S0895480103432556. url: https://doi.org/10.1137/S0895480103432556.

[5] Asim K. Dey, Stephen J. Young, and Yulia R. Gel. "From Delaunay Triangulation to Topological Data Analysis: Generation of More Realistic Synthetic Power Grid Networks". `under review`. 2023.

[6] Shannon Dillman and Franklin Kenter. "Leaky forcing: a new variation of zero forcing". In: *arXiv preprint arXiv:1910.00168* (2019).

[7] Sean English, Calum MacRury, and Paweł Prałat. "Probabilistic zero forcing on random graphs". In: *European Journal of Combinatorics* 91 (2021), p. 103207. MR4161801

[8] Shaun Fallat, Karen Meagher, and Boting Yang. "On the complexity of the positive semidefinite zero forcing number". In: *Linear Algebra Appl.* 491 (2016), pp. 101–122. ISSN: 0024-3795. DOI: 10.1016/j.laa.2015.03.011. URL: https://doi.org/10.1016/j.laa.2015.03.011. MR3440126

[9] Leslie Hogben, Jephian C.-H. Lin, and Bryan L. Shader. *Inverse problems and zero forcing for graphs.* Vol. 270. Mathematical Surveys and Monographs. American Mathematical Society, Providence, RI, 2022, pp. xi+287. ISBN: 978-1-4704-6655-8. DOI: 10.1090/surv/270. URL: https://doi.org/10.1090/surv/270. MR4478249

[10] Thomas Kalinowski, Nina Kamčev, and Benny Sudakov. "The zero forcing number of graphs". In: *SIAM J. Discrete Math.* 33.1 (2019), pp. 95–115. ISSN: 0895-4801. DOI: 10.1137/17M1133051. URL: https://doi.org/10.1137/17M1133051.

[11] Nima Monshizadeh, Shuo Zhang, and M Kanat Camlibel. "Zero forcing sets and controllability of dynamical systems defined on graphs". In: *IEEE Transactions on Automatic Control* 59.9 (2014), pp. 2562–2567. MR3254556

[12] Sandip Roy et al. "Structural Controllability Assessment for Inverter-Based Microgrids". In: *2021 North American Power Symposium (NAPS).* 2021, pp. 1–6. DOI: 10.1109/NAPS52732.2021.9654687.

[13] Logan A Smith and Illya V Hicks. "Optimal Sensor Placement in Power Grids: Power Domination, Set Covering, and the Neighborhoods of Zero Forcing Forts". In: *arXiv preprint arXiv:2006.03460* (2020). MR4421294

[14] Brady Tyra et al. *Electric Power Monthly.* Tech. rep. U.S. Energy Information Administration, Feb. 2023. URL: https://www.eia.gov/electricity/monthly/current_month/february2023.pdf.

[15] Stephen J. Young et al. "Topological Power Grid Statistics from a Network-of-Networks Perspective". In: *2018 IEEE Power Energy Society General Meeting (PESGM).* 2018, pp. 1–5.

Sinan G. Aksoy
Pacific Northwest National Laboratory
USA
E-mail address: sinan.aksoy@pnnl.gov

Ryan Bennink
Oak Ridge National Laboratory
USA
E-mail address: benninkrs@ornl.gov

Yuzhou Chen
Temple University
USA
E-mail address: yuzhou.chen@temple.edu

José Frías
University of Texas at Dallas
USA
E-mail address: frias4@cimat.mx

Yulia R. Gel
University of Texas at Dallas
USA
E-mail address: ygl@utdallas.edu

Bill Kay
Pacific Northwest National Laboratory
USA
E-mail address: william.kay@pnnl.gov

Uwe Naumann
RWTH Aachen University
Germany
E-mail address: naumann@stce.rwth-aachen.de

Carlos Ortiz Marrero
Pacific Northwest National Laboratory
USA
E-mail address: carlos.ortizmarrero@pnnl.gov

Anthony V. Petyuk
Hanford High School
USA
E-mail address: anthony.petyuk@gmail.com

Sandip Roy
Washington State University
USA
E-mail address: sandip@wsu.edu

Ignacio Segovia-Dominguez
University of Texas at Dallas
Jet Propulsion Laboratory, Caltech
USA
E-mail address: Ignacio.SegoviaDominguez@UTDallas.edu

Nate Veldt
Texas A&M University
USA
E-mail address: nveldt@tamu.edu

Stephen J. Young
Pacific Northwest National Laboratory
USA
E-mail address: stephen.young@pnnl.gov