COAP 2019 Best Paper Prize: Paper of Andreas Tillmann

Each year, the editorial board of Computational Optimization and Applications selects a paper from the preceding year's publications for the Best Paper Award. Two of the 92 papers published by the journal in 2019 tied for the award. This article highlights the research related to the award winning paper of Andreas Tillmann (Technische Universität Braunschweig) whose award winning paper "Computing the spark: mixed-integer programming for the (vector) matroid girth problem" was published in volume 74, pages 387–441.

Tillmann's paper [9] provides an in-depth investigation of the problem of computing the smallest number of linearly dependent columns of a given matrix, known as the *spark* of the matrix or as the *girth* (that is, length of a shortest circuit) of the associated matroid. The work identifies polynomially solvable special cases of the generally NP-hard problem, reviews related complexity results as well as upper and lower bounding procedures, proposes novel exact integer and mixed-integer programming (MIP) formulations, and develops a dedicated branch-and-cut algorithm involving tailored primal heuristics, domain propagation routines, and cutting planes. Extensive numerical experiments show that significant computational gains (compared to black-box MIP approaches) can be achieved by means of exploiting problem-specific structure that is based in no small part on the connection to matroid theory.

The spark of a matrix $A \in \mathbb{R}^{m\times n}$ is defined as

$$\operatorname{spark}(A) := \min\{\|x\|_0 : Ax = 0,\ x \neq 0\}, \tag{1}$$

where $\|x\|_0 := |\{j : x_j \neq 0\}|$. It is essential in the context of compressed sensing and the core sparse recovery problem (P_0):= $\min\{\|x\|_0 \ :\ Ax = b\}$, since every k-sparse vector $\hat{x}$ is the unique sparsest solution to $Ax = b$ with $b := A\hat{x}$ if and only if $2k < \operatorname{spark}(A)$ (see, e.g., [4]). Moreover, conditions involving the spark are used to identify ambiguities in linear sensor arrays [7], check the existence of unique tensor decompositions [6], verify k-stability for matrix completion [11], and more.

Computing the spark is NP-hard in general [10], and prior to [9], known exact methods were either essentially brute-force enumerative schemes (e.g., [2]) or based on reformulations that are much less amenable to exploiting problem-specific structural knowledge (e.g., [5,1]). In contrast, the direct approach taken by the author allowed to utilize fast heuristics and lower bounds from the rich field of compressed sensing and inference/propagation rules rooted in the matroid viewpoint to aid the branch-and-bound search process.

The MIP model proposed for general-purpose (black-box) solvers reads

$$\min_{x\in\mathbb{R}^n;\,y,z\in\{0,1\}^n} \quad \mathbb{1}^\top y \qquad \text{s.t.} \qquad Ax = 0,\ \ -y + 2z \leq x \leq y,\ \ \mathbb{1}^\top z = 1. \tag{2}$$

Here, scalability of nullspace vectors (x with $Ax = 0$) is exploited to enforce bounds $\pm\mathbb{1}$ on x via auxiliary binary variables y that model the support of x (so that $\|x\|_0 = \mathbb{1}^\top y$ in the optimum) and the binary vector z is used to fix one of the x-variables to 1, enforcing that $x \neq 0$. The author reports that related matroid-dual approaches (from [5] and [1], as well as a novel alternative) solved significantly slower that (2). Similarly, MIP solver performance using (2) was observed to be superior to both a sequential approach (fixing each x-variable to 1 in turn, the minimum of the resulting sequence of (P_0)-problems coinciding with the spark) and one where the nontriviality constraint $x \neq 0$ was realized by fixing $\|x\|_1 = 1$.

Moreover, based on the observation that in a matroid $\mathcal{M}$, every dependent set has nonempty intersection with the complements of all bases, a pure binary IP was introduced for matroid girth computation:

$$\min_{y\in\{0,1\}^n} \quad \mathbb{1}^\top y \qquad \text{s.t.} \qquad y(B^c) \coloneqq \sum_{j\notin B} y_j \leq 1 \quad \forall \,\text{bases } B \text{ of } \mathcal{M}, \tag{3}$$

where for spark(A), the bases are the column-index sets of all $m\times m$ regular submatrices of A, assuming without loss of generality that $\operatorname{rank}(A) = m < n$. Thus, (3) treats the constraints $Ax = 0$ and $x \neq 0$ implicitly, and works solely with the auxiliary support variables y. While there can, in general, be exponentially many bases of a matroid, [9] proceeds to show that the separation probem for the so-called covering inequalities in (3)—i.e., given some $\hat{y} \in [0,1]^n$, either find a (maximally) violated covering inequality or conclude that they are all satisfied—can be solved in (independence-)oracle-polynomial time for general matroids, and in time $\mathcal{O}(n(\log(n) + m^3))$ for the vector matroid (spark) case, by a simple greedy scheme: Build a basis by traversing the entries of $\hat{y}$ in non-increasing order, adding indices so that independence of the constructed set is maintained. These results laid the groundwork for the specialized spark problem solver developed subsequently.

The problem-specific branch-and-cut solver merged (2) and (3)—note that the y-variables have the same meaning in both models, so the constraints from (3) can be added as cutting planes during the solving process of (2). Various other solver components were addressed and specialized besides separation of covering inequalities. For example, the propagation routine ensures that the index set of y-variables fixed to 1 is always independent, and as soon as a check finds that adding one more column (index) yields a dependent set (i.e., a feasible solution), the remaining branch is cut off immediately. The implementation also includes efficient primal heuristics (mostly based on ℓ_1-norm minimization), other classes of cuts (in particular, generalized cycle inequalities as known from set covering, and inequalities based on the sparsity pattern of A), and a branching rule that eliminates the z-variables at the first level of the search tree.

The spark solver was implemented in the open-source framework SCIP and compared against the commercial solvers Cplex and Gurobi (applied to (2)) on a test set of 100 instances of sizes up to well over 1000 columns. The best-performing variant of the author's specialized code solved 82 instances (versus 66 and 58 for Cplex and Gurobi, resp.) within one hour, with significantly lower average running times and search nodes, and also achieved much lower optimality gaps (about 200% versus about 300%) on the remaining 18 instances where all solvers hit the time limit.

The paper [9] has since inspired work on adapting the methodology to the special case of binary matroids, which leads to the fundamental coding-theoretical problems of minimum Hamming distance computation and certain maximum-likelihood decoding tasks for binary linear codes ([8], and ongoing). Moreover, the approach is currently being extended to finding sparsest solutions to underdetermined inhomogeneous equation systems (i.e., (P_0), which corresponds to seeking shortest matroid circuits containing a specific element), and has initiated research into matroid-based MIP methods for matrix sparsification (cf., e.g., [3]), where spark computation appears as a subproblem.

References

1. Arellano, J.D., Hicks, I.V.: Degree of Redundancy of Linear Systems using Implicit Set Covering. IEEE Trans. Automation Sci. Eng. **11**(1), 274–279 (2014)
2. Cho, J.J., Chen, Y., Ding, Y.: On the (Co)Girth of a Connected Matroid. Discrete Appl. Math. **155**(18), 2456–2470 (2007)
3. Egner, S., Minkwitz, T.: Sparsification of Rectangular Matrices. J. Symb. Comput. **26**(2), 135–149 (1998)
4. Foucart, S., Rauhut, H.: A Mathematical Introduction to Compressive Sensing. Appl. Numer. Harmon. Anal. Birkhäuser (2013)
5. Kianfar, K., Pourhabib, A., Ding, Y.: An Integer Programming Approach for Analyzing the Measurement Redundancy in Structured Linear Systems. IEEE Trans. Automation Sci. Eng. **8**(2), 447–450 (2011)
6. Kolda, T.G., Bader, B.W.: Tensor Decompositions and Applications. SIAM Rev. **51**(3), 455–500 (2009)
7. Manikas, A., Proukakis, C.: Modeling and Estimation of Ambiguities in Linear Arrays. IEEE Trans. Signal Process. **46**(8), 2166–2179 (1998)
8. Puchert, C., Tillmann, A.M.: Exact separation of forbidden-set cuts associated with redundant parity checks of binary linear codes. IEEE Commun. Lett. **24**(10), 2096–2099 (2020)
9. Tillmann, A.M.: Computing the spark: mixed-integer programming for the (vector) matroid girth problem. Comput. Optim. Appl. **74**(2), 387–441 (2019)
10. Tillmann, A.M., Pfetsch, M.E.: The Computational Complexity of the Restricted Isometry Property, the Nullspace Property, and Related Concepts in Compressed Sensing. IEEE Trans. Inf. Theory **60**(2), 1248–1259 (2014)
11. Zhu, Z., So, A.M.C., Ye, Y.: Fast and Near-Optimal Matrix Completion via Randomized Basis Pursuit. In: L. Ji, Y.S. Poon, L. Yang, S.T. Yau (eds.) Proc. 5th ICCM, *AMS/IP Stud. Adv. Math.*, vol. 51, pp. 859–882. AMS and International Press (2012)

Andreas Tillmann is currently a postdoctoral researcher at the Institute for Mathematical Optimization at TU Braunschweig. He received his doctoral degree (Dr. rer. nat.) from TU Darmstadt in December 2013, and was an interim professor of mathematical optimization at TU Braunschweig in winter 2014/15. As a postdoc, he was at TU Darmstadt (until 2016), at RWTH Aachen University (where he headed the unit "Data" of the Computational Science and Engineering profile area, 2017–2019), and a visiting researcher at Polytechnique Montréal (summer 2019). His research interests include theoretical and practical aspects of (discrete) optimization, mixed-integer programming, and machine learning in areas like compressed sensing, signal and image processing, and operations research.