Set intersection with minimal support

Jonas Seiler Cecilia Knäbchen

Abstract

Given a number $n \in \mathbb{N}$, let $k \in \mathbb{N}$ be the minimal number, such that one can find n sets $A_1, ..., A_n$ containing some integers from 1 to k such that any pair of sets A_i and A_j share exactly $|i-j|$ elements, that is:

$$A_i \subseteq \{1, ..., k\}, \quad 1 \leq i \leq n$$
$$|A_i \cap A_j| = |i-j|, \quad 1 \leq i < j \leq n$$

We present optimal values for k for $n \leq 22$ found by converting the problem to a suitable integer linear programm (ILP).
Furthermore we present upper and lower bounds for any n.

1. Introduction

For $n = 4$, $k = 5$ is optimal. An exemplary solution is the following:

$$A_1 = \{1, 2, 3, 4\}$$
$$A_2 = \{1, 5\}$$
$$A_3 = \{1, 2\}$$
$$A_4 = \{1, 3, 4, 5\}$$

```
A₁:  1  2  3  4
A₂:  1              5
A₃:  1  2
A₄:  1     3  4  5
```

Exemplary solutions for all $n \leq 22$ can be found in the relevant OEIS entry: `oeis.org/A381294`.

Given a solution, one can immediately get $k!$ other solutions by permuting the naming of the k elements. Additionally, one can "mirror" the sets, i.e. A_1 becomes A_n, A_2 becomes A_{n-1} and so on, to obtain new solutions as well.

To atleast partially normalize the structure of these solutions, we rename the elements in such a way, that the columns as seen in the solution above are lexicographically sorted. Note that is not invariant under mirroring.

The solutions listed in the OEIS are normalized with respect to this convention.

2. Optimal Values and Integer Programming

There are two ways to go about formulating this problem as an integer programm; First, testing if a specific combination of n and k is feasible, by having binary variables $x_{i,a}$ to denote that set A_i contains element a. A second strategy is not to check feasibility of a specific k but rather change the goal to try to minimize k. This second strategy worked better but is poorly scalable.

2.1. Feasibility of a specific (n, k) pair

As mentioned above, the formulation of this problem is easy:

$$\text{Variables: } x_{i,a} \in \{0, 1\} \qquad \forall i \in \{1, ..., n\}, a \in \{1, ..., k\}$$
$$\text{Subject to: } \sum_{a=1}^{k} x_{i,a} \cdot x_{j,a} = |j - i| \qquad \forall i, j \in \{1, ..., n\}$$

where $x_{i,a}$ denotes that set A_i contains element a. Two sets A_i and A_j share an element a, iff $x_{i,a} \cdot x_{j,a} = 1$, therefore we can count the size of the intersection by summing over those products.

Since our constraints are products over variables, this is no longer a linear binary programm and rather a quadratic binary programm. In our tests, the solving was far slower than the formulation as a linear integer program, even though this formulation has linear size in n in contrast to the exponential size of the next formulation.

2.2. Optimizing k itself

We can also write a linear programm that does not check for feasibility of a specific k but rather optimize k itself. We do this by constructing variables S_X for every subset $X \subseteq \{1, ..., n\}$ that encapsulates the number of elements that are shared by exactly the sets A_i for $i \in X$ and none else. k is therefore the sum of all these variables.

$$\text{Variables: } S_X \in \mathbb{N} \text{ for each } X \subseteq \{1, ..., n\}$$
$$\text{Minimize: } \sum_{X \subseteq \{1,...,n\}} S_X$$
$$\text{Subject to: } \sum_{\substack{X \subseteq \{1,...,n\} \\ i,j \in X}} S_X = |i - j| \ \forall i, j \in \{1, ..., n\}$$

Since there are 2^n subsets of $\{1, ..., n\}$, this linear programm has exponential size in n.

With this formulation, we have found optimal values for $n \leq 22$ as can be seen in Table 1.

One interesting observation is that for each $n \leq 22$, there exists at least one optimal solution where each elements $a \in \{1, ..., k\}$ appears in less than 6 sets. This additional heuristic drastically speeds up the search for new solutions but we have been unable to prove this observation formally and do not expect it to hold for larger n.

Code for this approach can be found on github:
`github.com/HerrPixel/A381294`

3. Upper bounds

We will give an upper bound by explicitely constructing a solution for any n. A trivial construction is the following:

Consider any pair (i, j) of indices $1 \leq i < j \leq n$ and their respective sets A_i and A_j. Add $j - i$ new elements to both sets and only those sets. Then their intersection has the correct size. Doing this for all pairs of indices, gives a

correct solution, albeit not optimal for $n \geq 3$. The number of elements used is the sum of all index differences:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} j - i = \sum_{i=1}^{n-1} \sum_{j=1}^{n-i} j \in \Theta(n^3)$$

This gives an asymptotic upper bound of $O(n^3)$ on the optimal k for any n.

One might get a better bound by considering individual step sizes instead of only pairwise indices. I.e. look at sets of the form $A_{1+1 \cdot i}$, for every $0 \leq i \leq n-1$ and add an element to each of those. Then look at sets of the form $A_{1+2 \cdot i}$ for every $0 \leq i \leq \frac{n-1}{2}$ and additionally $A_{2+2 \cdot i}$.

In general, one then has i different congruence classes of sets when considering step size i. For each congruence class, one needs to add one or more elements to "fix" their intersection size. Elements in a specific class already share elements for each proper divisor of i, therefore the number of new elements to be added will be

$$\text{new}(i) = i - \sum_{d \mid i} \text{new}(d)$$

This construction might lead to a better bound but will not be optimal for $n \geq 4$.

4. Lower bounds

Consider only intersections with the last set A_n. Set A_1 needs exactly $n-1$ elements in this intersection. Set A_2 needs exactly $n-2$ elements in this intersection of which at most one is already contained in A_1 since $|A_1 \cap A_2| = 1$. This means A_2 needs $n-3$ new elements for its intersection with A_n.

In general, a set A_i has $n-1$ elements in its intersection with A_n of which at most $\sum_j = 1^{i-1} |A_j \cap A_i| = \triangle(i-1)$ elements are not new. This procedure guarantees new elements until $\triangle(i-1) \geq n-i$. One can verify that this is the case for

$$i < \left\lfloor \frac{3\sqrt{n}+1}{2} \right\rfloor$$

We can then bound the number of elements needed from below with:

$$\begin{aligned}
\sum_{i=1}^{\left\lfloor \frac{3\sqrt{n}}{2} \right\rfloor} n - \triangle(i) &\geq \sum_{i=1}^{\sqrt{n}} n - \frac{i \cdot (i+1)}{2} \\
&= n\sqrt{n} - \frac{1}{2} \sum_{i=1}^{\sqrt{n}} i^2 + i \\
&= n\sqrt{n} - \frac{1}{2} \cdot \frac{2\sqrt{n}^3 + 3\sqrt{n}^2 + \sqrt{n}}{6} + \Theta(n) \\
&= n\sqrt{n} - \frac{n\sqrt{n}}{6} + \Theta(n) \\
&\in \Theta\left(n\sqrt{n}\right)
\end{aligned}$$

One might again get a better bound by not only considering A_n itself but also $A_{n-1}, A_{n-2}, \ldots$ and so on until there are no more guaranteed new elements.

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
k	0	0	1	2	5	9	16	24	36	50	70	91	120	150	189	231	280	336	398	468	547	630	728

Table 1: Known optimal values for k