TWO QUESTIONS

MICHAEL JOSWIG

1. What is a sparse matrix?

Problem 1. Find a rational number x with

$$2^{-1022} \;\leq\; x \;\leq\; 2^{1024} - 2^{971}$$

but without an exact representation as an IEEE 754 double. Are there also integers with the same property? In `Julia` find two numbers x, y which are correctly representable as type `Float64` but their sum $x + y$ is not.

Problem 2. The naive algorithm for multiplying two $n \times n$-matrices requires n^3 multiplications. Devise better algorithms

(1) for $n = 2$, requiring only $7 < 8 = 2^3$ multiplications,
(2) for $n = 3$, requiring fewer than $3^3 = 27$ multiplications. Can you do 23?

Problem 3. Write code to produce random matrices, whose coefficients are moderately sized integers (say, between -500 and 500), most of which are zero. Store such a matrix as four different types:

`SparseMatrixCSC{Int}`, `SparseMatrixCSC{Float}`, `SparseMatrixCSC{fmpq}`, `Matrix{fmpq}`.

Measure the timings for computing the squares and the third powers, using those representations and trying various matrix sizes and numbers of nonzeros.

In addition to `Oscar`[1] use the `SparseMatrix` package in `Julia`.

Problem 4. Write down an algorithm for multiplying two sparse matrices in the CSC format. What is its complexity?

Problem 5. This exercise requires using two different computers (different brands or builds). So you may want to work in pairs.

On Computer A determine the maximal number n such that a random $n \times n$-matrix (like in Problem 3) can be inverted in at most 30 seconds. Then estimate how long the same computation should take on Computer B. Do a reality-check on your prediction.

Problem 6. We consider the following decision problem: the input is a finite graph Γ, and the output is yes/no, answering the question whether or not Γ is connected.

(1) Create a few graphs with the `Graphs` package and measure the timings for deciding connectedness. Estimate the empirical complexity as a function of the number of nodes.
(2) What are reasonable ways to encode graphs in the computer? What are good algorithms for deciding connectedness? What is their theoretical complexity?

Date: 28 April 2022 (MPI-MiS Leipzig).

[1]`Oscar` has the type `SMat` for sparse matrices. It is optimized for solving systems of linear equations over certain fields and rings.

2. How good are convex hull algorithms?

Problem 7. Show that the cyclic polytopes are simplicial, i.e., each facet of a cyclic polytope is a simplex.

Problem 8. For any pair of integers (d, n) with $n \geq d + 1 \geq 3$ there is an integer $m = m(d, n)$ such that every d-polytope with n vertices has at most m facets, and that bound is tight. The precise values for m are given by McMullen's upper bound theorem.

For each d between 2 and 100 determine the maximal n such that $m(d, n) \leq 1{,}000{,}000$.

Problem 9. Consider the product of cyclic polytopes $P = C_3(20) \times C_3(20)$. Then P is 6-dimensional, with 400 vertices.

(1) How many facets does P have?
(2) Now consider two sequence of polytopes $P_0, P_1, P_2, \ldots$ and $Q_1, Q_2, Q_3, \ldots$ which are defined by the following rules: for $P_0 = P$ let Q_i be the intersection of the facet defining halfspaces of P_{i-1}, and P_i is the convex hull of the vertices of Q_i. Which polytopes do you get?
(3) Pick one exact and one inexact convex hull code and compute the first few polytopes $Q_1, P_1, Q_2, P_2, \ldots$ explicitly, with both codes.

Problem 10. A *cut* in a finite (undirected, connected) graph G is a subset of the edges whose removal disconnects the node set. The characteristic vectors of the cuts form 0/1-vectors, and their convex hull is the *cut polytope* of G.

Try several convex hull algorithms/codes and empirically decide which one performs best. In `Oscar` you can use constructions like this:

```
P = convex_hull([0 0; 1 0; 0 1])
Polymake.prefer("cdd") do
  @time nfacets(P)
end
```

Options for selecting (exact) convex hull algorithms include `beneath_beyond`, `cdd`, `lrs`, `libnormaliz`, `ppl`.

Problem 11. Pick points uniformly at random in some box in $\mathbb{R}^d$. The goal is to compute their (Euclidean) Voronoi diagram via a dual convex hull computation in $\mathbb{R}^{d+1}$.

Try several convex hull algorithms/codes and empirically decide which one performs best.

(Michael Joswig) Technische Universität Berlin, Chair of Discrete Mathematics/Geometry, Max-Planck Institute for Mathematics in the Sciences, Leipzig, joswig@math.tu-berlin.de