Analog Sorting

Theory and evaluation

Lusa Zhan
Columbia University
Department of Computer Science
Department of Mathematics
lz2371@columbia.edu

Yipeng Huang
Columbia University
Department of Computer Science
yipeng@cs.columbia.edu

Table 1: Frequency of Special Characters

Non-English or Math	Frequency	Comments
Ø	1 in 1,000	For Swedish names
π	1 in 5	Common in math
\$	4 in 5	Used in business
Ψ_1^2	1 in 40,000	Unexplained usage

ABSTRACT

Keywords

linear algebra; ordinary differential equations; algorithms

1. INTRODUCTION

2. BACKGROUND

2.1 Classical sorting algorithms

Table for comparing important efficient sorting algorithms, for example quick & merge sort.

How does analog sorting compare to other sorting algorithms in terms of complexity?

2.2 The QR algorithm

The analog sorting method we show is related to the classical QR algorithm. While the QR algorithm is a discrete algorith operating step-by-step, the analog sorting algorithm does so in continuous time [5, 3, 4].

The QR algorithm finds the eigenvalues and eigenvectors of a square matrix. The eigenvalue problem is as follows:

$$A_0 x = \lambda x$$

The QR algorithm operates as follows: Given the QR decomposition, each step proceeds as:

$$A_k = Q_k R_k$$

$$A_{k+1} = R_k Q_k$$

 ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

A side effect of the QR algorithm is that the eigenvalues of the original matrix end up in sorted order along the diagonal. This is a useful property in eigenvalue problems, where users of the algorithm are interested in finding the largest few eigenvalues. But few researchers have pointed out that this may be useful in itself, for sorting. Other algorithms for finding the eigenvalues and eigenvectors include the Jacobi eigenvalue algorithm, and the divide-and-conquer algorithm. It seems that the QR algorithm is the algorithm among these that sorts the eigenvalues.

2.3 Hamiltonian systems

The finite Toda lattice system of ODEs belongs to special class of ODEs called Hamiltonian systems. Hamiltonian systems are an important and efficient way to describe classical mechanics. Because of their importance, special ODE solvers called symplectic solvers have been developed specifically to solve Hamiltonian systems.

A Hamiltonian system is characterized by a total energy scalar $\hat{\mathcal{H}}$. The components of the system are described by vectors p for momenta and q for positions. The system obeys the laws of motion:

$$dp/dt = -dH/dq = f(q) dq/dt \qquad = dH/dp = g(p) \tag{1}$$

Conceptually, we encode the keys we like to sort as the momenta of particles. The momenta of the particles carries them to a final state that represents the sorted system.

2.4 Toda lattice, double bracket

The finite Toda lattice is an ODE system that is related to the QR algorithm. If you plot the evolution of the Toda lattice ODE with respect to time, the values of the ODE at integer time steps is the intermediate states of the QR algorithm. The common property of the finite Toda lattice and the QR algorithm is that both preserve the eigenvalues of the matrix they operate on [1].

The finite Toda lattice is a Hamiltonian system with the form:

$$\hat{\mathcal{H}}(p,q) = \frac{1}{2}\sum_{1}^{n} p_k^2 + \sum_{1}^{n-1} exp(q_k - q_{k+1})$$

This basic form can changed in two ways, using change of variables, into the equations for analog sorting described in [2].

The central idea of analog sorting as described in the papers is using the Toda Lattice, which is described using the Hamiltonian

$$H(p,q) = \frac{1}{2}\sum_{1}^{n} p_k^2 + \sum_{1}^{n-1} exp(q_k - q_{k+1})$$

A system of equations are associated with this, defined as the Hamiltonian equations (can be derived using total differential)

$$\dot{p_k} = -\frac{\partial H}{\partial q_k}$$

$$\dot{q_k} = \frac{\partial H}{\partial p_k}.$$

Calculating this, we get

$$\dot{p_k} = exp(q_{k-1} - q_k) - exp(q_k - q_{k+1})$$

$$\dot{q_k} = p_k$$

where $e^{x_0 - x_1} = e^{x_n - x_{n+1}} = 0$

2.5 Toda Lattice - ODE form

This system of ODEs can be changed to the system described in (http://hrl.harvard.edu/analog/) using a change of variables.

$$x_k = -\frac{1}{2}p_k$$

$$y_k = \frac{1}{2}e^{(q_k - q_{k+1})/2}$$

In that case,

$$\begin{aligned}
\dot{x_k} &= -\frac{1}{2}\dot{p_k} \\
&= -\frac{1}{2}(exp(q_{k-1} - q_k) - exp(q_k - q_{k+1})) \\
&= -\frac{1}{2}(4y_{k-1}^2 + 4y_k^2) \\
&= 2y_k^2 - 2y_{k-1}^2
\end{aligned}$$

and

$$\begin{aligned}
\dot{y_k} &= \frac{1}{2}e^{(q_k - q_{k+1})/2}(\frac{\dot{q}_k - \dot{q}_{k+1}}{2}) \\
&= y_k\frac{p_k - \dot{p}_{k+1}}{2} \\
&= y_k(x_{k+1} - x_k)
\end{aligned}$$

Taking into account the boundary conditions $y_0 = y_n = 0$ and we get the desired system of ODEs.

2.6 Toda Lattice - Jacobi Matrix

The connection to the Double Bracket notation can be made through the Jacobi Matrix form of the Toda Lattice. The Jacobi Matrix is given by

$$H = \begin{bmatrix} x_1 & y_1 & 0 & \ldots & 0 \\ y_1 & x_2 & y_2 & \ldots & 0 \\ & & \ddots & & \\ & & y_{n-2} & x_{n-1} & y_{n-1} \\ 0 & \ldots & & y_{n-1} & x_n \end{bmatrix}$$

This is also the form that H is in for analog sorting.

In order to get the double bracket form, we need a diagonal matrix $N = diag(n, n-1, \ldots, 1)$, so we need

$$N = \begin{bmatrix} n & 0 & \ldots & 0 \\ 0 & n-1 & & \\ \vdots & & \ddots & \vdots \\ 0 & \ldots & & 1 \end{bmatrix}$$

We then get

$$HN = \begin{bmatrix} nx_1 & (n-1)y_1 & \ldots & 0 \\ ny_1 & (n-1)x_2 & \ldots & 0 \\ \vdots & & \ddots & 0 \\ & & 2x_{n-1} & y_{n-1} \\ 0 & \ldots & 2y_{n-1} & x_n \end{bmatrix}$$

$$NH = \begin{bmatrix} nx_1 & ny_1 & \ldots & 0 \\ (n-1)y_1 & (n-1)x_2 & \ldots & 0 \\ \vdots & & \ddots & 0 \\ & & 2x_{n-1} & 2y_{n-1} \\ 0 & \ldots & y_{n-1} & x_n \end{bmatrix}$$

$$HN - NH = \begin{bmatrix} 0 & -y1 & \ldots & & 0 \\ y_1 & 0 & -y_2 & \ldots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ & & y_{n-2} & 0 & -y_{n-1} \\ 0 & & \ldots & y_{n-1} & 0 \end{bmatrix}$$

$$H[H,N] = \begin{bmatrix} y_1^2 & -x_1y_1 & -y_1y_2 & 0 & \ldots & 0 \\ x_2y_1 & -y_1^2 + y_2^2 & -x_2y_2 & 0 & \ldots & \\ y_1y_2 & \ddots & & \ddots & & 0 \\ \vdots & & & & y_{n-1}^2 - y_{n-2}^2 & -x_{n-1}y_{n-1} \\ 0 & \ldots & & & x_ny_{n-1} & -y_{n-1}^2 \end{bmatrix}$$

$$[H,N]H = \begin{bmatrix} y_1^2 & -x_2y_1 & -y_1y_2 & 0 & \ldots & 0 \\ x_1y_1 & y_1^2 - y_2^2 & -x_3y_2 & 0 & \ldots & \\ y_1y_2 & \ddots & & \ddots & & 0 \\ \vdots & & & & -y_{n-1}^2 + y_{n-2}^2 & -x_ny_{n-1} \\ 0 & \ldots & & & x_{n-1}y_{n-1} & y_{n-1}^2 \end{bmatrix}$$

Therefore, we get

$$[H,[H,N]] = \begin{bmatrix} 2y_1^2 & y_1(x_2 - x_1) & 0 & & \ldots \\ y_1(x_2 - x_1) & -2y_1^2 + 2y_2^2 & y_2(x_3 - x_2) & 0 & \ldots \\ 0 & \ddots & & \ddots & \\ \vdots & & & & 2y_{n-1}^2 - 2y \\ 0 & \ldots & & & y_{n-1}(x_n - x \end{bmatrix}$$

which is exactly $\dot{H}$

3. RELATED WORK

Below is a list of articles that are related to this topic:

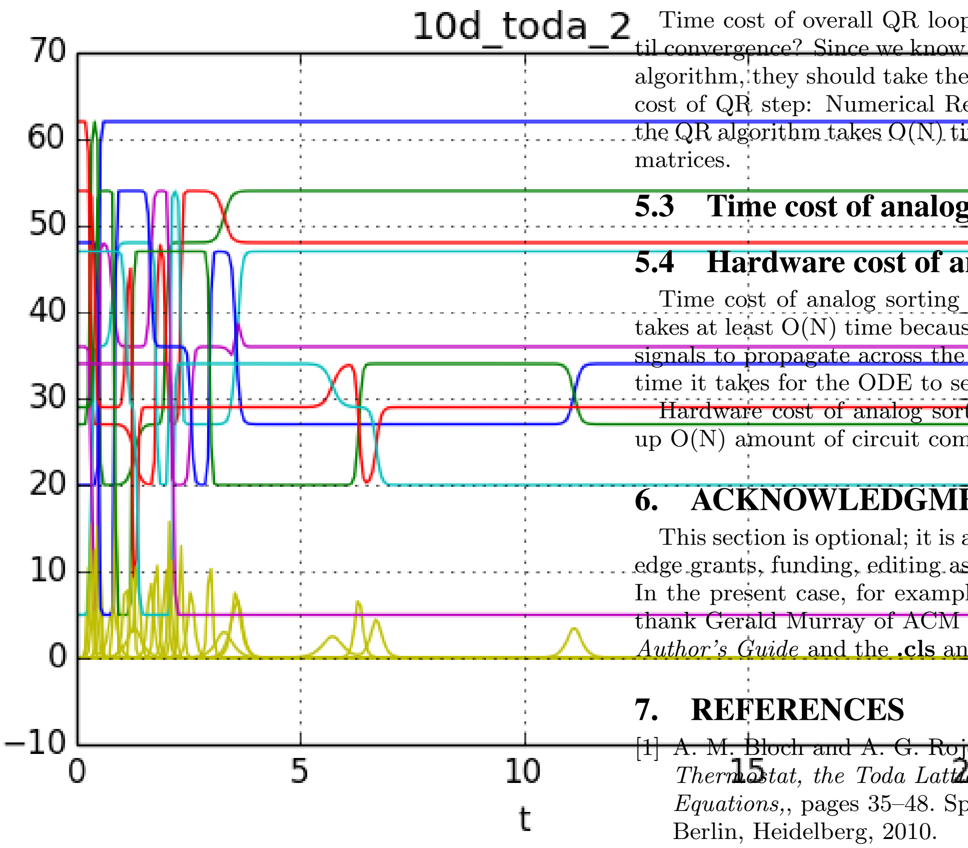

Figure 1: A sample black and white graphic.

- Bloch, Anthony M., and Alberto G. Rojo. "Sorting: The Gauss Thermostat, the Toda Lattice and Double Bracket Equations,." *Three Decades of Progress in Control Sciences* (2010): 35-48.
- Guseinov, Gusein Sh. "A Class of Complex Solutions to the Finite Toda Lattice." *Mathematical and Computer Modelling* 57.5-6 (2013): 1190-202.
- Tomei, Carlos. "The Toda Lattice, Old and New." *JGM Journal of Geometric Mechanics* 5.4 (2013): 511-30. Web.

3.1 Realizing the analog sorter

4. METHODOLOGY

We validate the functionality of analog sorting using a prototype analog computer chip. In order to explore how analog sorting works in larger systems, we use an ODE solver built on the odeint solver in the Python SciPy library.

5. EVALUATION

How does analog sorting compare to other sorting algorithms for example quick, merge etc in terms of complexity?

5.1 Functionality validation of analog sorting

5.2 Time cost of the discrete QR algorithm

Time cost of overall QR loop: how many iterations of qr til convergence? Since we know the ODE is analogous to QR algorithm, they should take the same amount of time. Time cost of QR step: Numerical Recipes 3rd Edition p585 says the QR algorithm takes O(N) time for symmetric tridiagonal matrices.

5.3 Time cost of analog sorting

5.4 Hardware cost of analog sorting

Time cost of analog sorting In terms of time, the sorter takes at least O(N) time because of the time it takes just for signals to propagate across the circuit. Another issue is the time it takes for the ODE to settle to its final value.

Hardware cost of analog sorting The analog sorter takes up O(N) amount of circuit components to sort N elements.

6. ACKNOWLEDGMENTS

This section is optional; it is a location for you to acknowledge grants, funding, editing assistance and what have you. In the present case, for example, the authors would like to thank Gerald Murray of ACM for his help in codifying this *Author's Guide* and the **.cls** and **.tex** files that it describes.

7. REFERENCES

[1] A. M. Bloch and A. G. Rojo. *Sorting: The Gauss Thermostat, the Toda Lattice and Double Bracket Equations,*, pages 35–48. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[2] R. W. Brockett. Dynamical systems that sort lists, diagonalize matrices and solve linear programming problems. In *Proceedings of the 27th IEEE Conference on Decision and Control*, pages 799–803 vol.1, Dec 1988.

[3] M. T. Chu. On the continuous realization of iterative processes. *SIAM Review*, 30(3):375–387, 1988.

[4] M. T. Chu. A list of matrix flows with applications. In *in Hamiltonian and Gradients Flows, Algorithms and Control*, pages 87–97, 1994.

[5] P. Deift, T. Nanda, and C. Tomei. Ordinary differential equations and the symmetric eigenvalue problem. *SIAM Journal on Numerical Analysis*, 20(1):1–22, 1983.

APPENDIX