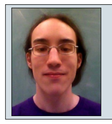
MADALGO
CENTER FOR MASSIVE DATA ALGORITHMICS


Jayson Lynch
MIT CSAIL


Erik Demaine
MIT CSAIL

Danmarks Grundforskningsfond
Danish National Research Foundation

Faster, Cheaper, Cooler, Longer: Energy Efficient Algorithms

Abstract

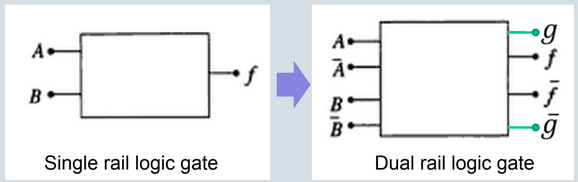
Energy is an obvious resource in any computation. Historically, the only resources considered in the asymptotic analysis of algorithms are computation time and space. Work in information theory and the physics of computation show that information, entropy, and energy are fundamentally connected [4]. We propose two models of computation. The first is based on current methods of representing logical values as high and low voltages in a wire. The second is based on the Landauer Limit which relates information destruction to waste energy.

Motivation

- **Cheaper Computation**
  - 2005: 26M Servers → 14.0 GW ≈ $8.3 billion / year [7]
- **Smaller Environmental Impact**
  - 2010: 31M Servers → 23 − 31 GW
    ≈ 1.1 − 1.5% of worldwide energy usage [7]
- **Longer Battery Life**
- **Faster CPUs** – Cooling is the limiting factor in clock speed.
  - Less energy → less waste heat → faster computer [6]

Bit-Flip Model

- Transitioning from a 1 to 0 requires a unit of energy.
- Theorem: Any w-bit RAM algorithm can be made conservative using O(w) extra space and a constant factor increase in time.
  - **Conservative Logic** – Every gate has an equal number of 1 outputs as 1 inputs. [3]
  - **Dual-Rail Logic** – Encode every bit with two wires as a pair (1,0) or (0,1).
  - Convert every gate to Dual-Rail Logic and added extra outputs to make it Conservative.



[Uyemura: CMOS Logic Circuit Design, 1999]

- The same argument can be used to prove results about multi-processor systems.
- **Future Work:**
  - Do simulations show any benefit of using these principles to reduce energy consumption? What about security settings where dual-rail logic is already used?
  - Can other symmetric encoding schemes yield better constant factor overheads?

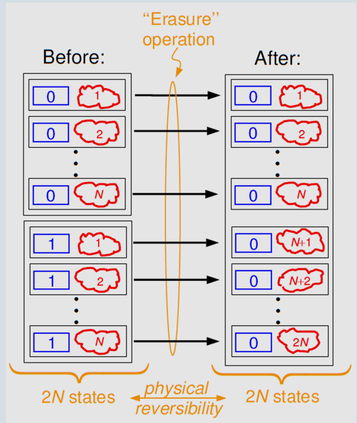
Reversible Model

Landauer's Principle: Increasing entropy by 1 bit requires $kT \ln 2$ energy where k is Boltzman's Constant and T is the temperature. [4]

$$kT \ln 2 \approx 3 \cdot 10^{-21} \text{ J} \approx 8 \cdot 10^{-25} \text{ Wh}$$

This limit can be circumvented with **reversible computation**, which preserves entropy and can be undone to reset energy. Unfortunately the best theoretical bounds require a **quadratic increase in** or **exponential increase in time**. [1]

What is the best tradeoff between energy, time, and space if we allow reversible and irreversible operations?

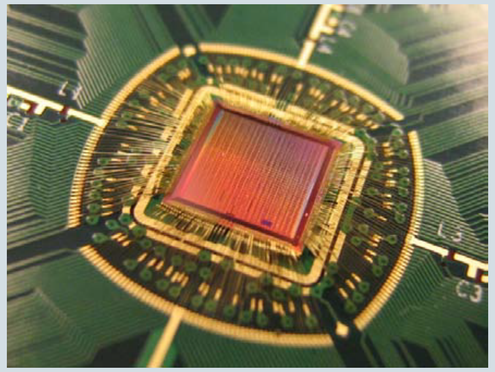


Michael Frank, 1999. [2]

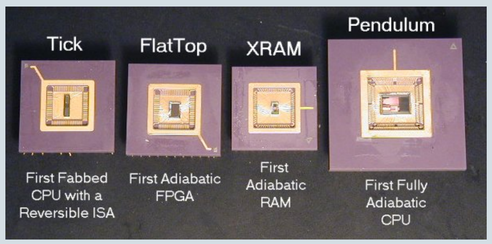
Reversible Analysis Techniques

- **Instruction Level Abstraction** – By working out the energy consumption of irreversible operations and the amount of garbage produced by reversible operations, we can work at the Word RAM level of abstraction.
- **Assembly Composition** – In the worst case, we can use our known reversible and irreversible instructions to write out assembly which can be analyzed.
- **Memory Diagram** – Reversibility often depends on whether a piece of data has been acted on by an irreversible operation. A diagram of where a piece of information is used is often easier than trying to create assembly for some pseudo-code.
- **Unwindable Block** – We can notate pseudo-code with blocks that are guaranteed to be reversible. Whenever these are executed, we then know this section can be unwound.

Real World Reversible



Adiabatic [reversible] clock developed by
Cyclos Semiconductor to be used in new AMD GPU
www.cyclos-semi.com/pdfs/time_to_change_the_clocks.pdf



The Pendulum, developed in the Knight Lab at MIT,
is a fully reversible CPU [2]
www.eng.fsu.edu/~mpf/MPF-SEALeR-talk.pdf

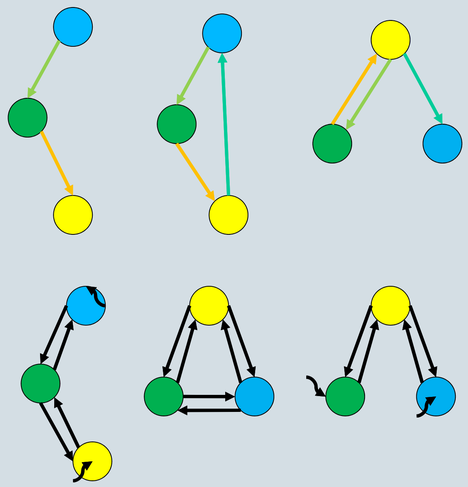
Energy Reduction Techniques

Instruction Optimization

Word level operations can sometimes be made more efficient than the corresponding bit operations would suggest.
Comparing two bits takes a bit of energy; however, comparing two w-bit words should only create 1 bit of entropy. With the right circuit, comparison takes $O(1)$ energy instead of $O(w)$.

For Loop Unwinding

The counter in a for loop can be decremented and thus unwound independently of whether the content in the for loop was reversible. Characterizing when situations allow partial unwinding in a program is a large open question.



Data Structure Rebuilding

Naively trying to make many data structures reversible leads to a space blow up dependent on the total number of operations. Intuitively, we should only need to get rid of the things actually in the data structure. By periodically unwinding and rebuilding our data structure, we can get the space needed to scale only with the number of insertions.

Pointer Swapping

Creating a new pointer takes $O(w)$ energy. Swapping them around is free. Thus if we can augment a data structure to enable swapping instead we can save energy.

Algorithm Results

Algorithm	Time	Space	Energy
Comparison Sort	$\Theta(n \lg n)$	$\Theta(n)$	$\Theta(n \lg n)$
Counting Sort	$\Theta(n+k)$	$\Theta(n+k)$	$\Theta(n+k)$
Counting Sort (reversible)	$\Theta(n+k)$	$\Theta(n+k)$	0
Bellman-Ford	$\Theta(VE)$	$\Theta(V)$	$\Theta(VEw)$
Floyd-Warshall	$\Theta(V^3)$	$\Theta(V^2)$	$\Theta(V^3 w)$
Reversible Floyd-Warshall [2]	$\Theta(V^3 \lg v)$	$\Theta(V^2 \lg V)$	0
Standard AVL Trees (build)	$O(n \lg n)$	$O(n)$	$O(w \cdot n \lg n)$
(find)	$O(\lg n)$	$O(1)$	$O(\lg n)$
(insert)	$O(\lg n)$	$O(1)$	$O(w \lg n)$
(delete)	$O(\lg n)$	$O(1)$	$O(w \lg n)$
Augmented AVL Trees (build)	$O(n \lg n)$	$O(n)$	$O(w \cdot n)$
(find)	$O(\lg n)$	$O(1)$	0
(insert)	$O(\lg n)$	$O(1)$	$O(1)$
(delete)	$O(\lg n)$	$O(1)$	$O(\lg n)$
Reversible AVL Trees (build)	$O(n \lg n)$	$O(n)$	0
(find)	$O(\lg n)$	$O(1)$	0
(insert)	$O(\lg n)$	$O(1)$	0
(delete)	$O(\lg n)$	$O(1)$	0
Binary Heap (Build)	$O(n \lg n)$	$O(n)$	$O(n \lg n)$
(insert)	$O(\lg n)$	$O(1)$	$O(\lg n)$
(delete)	$O(\lg n)$	$O(1)$	$O(\lg n)$
Reversible Binary Heap (Build)	$O(n \lg n)$	$O(n \lg n)$	0
(insert)	$O(\lg n)$	$O(\lg n)$	0
(delete)	$O(\lg n)$	$O(1)$	0

Future Work

- **A Whole New Field for Algorithms**
  - Fast Fourier Transforms
  - Implicit Data Structures
  - Dynamic Memory Allocation
- **High level language**
  - Notation for reversible vs. irreversible operations
  - Ways of keeping track of what sections cannot be unwound
- **Refine Model**
  - How should memory storage work?
  - What are the costs of RAM operations?

References

[1] C. Bennett. *Time/space trade-offs for reversible computation*. SIAM J. Computing, 1989.
[2] M. Frank. *Reversibility for efficient computing*. PhD Thesis MIT,1999.
[3] T. Tooli, E. Fredkin. *Conservative logic*. 1982.
[4] R. Landauer. *Irreversibility and heat generation in the computing process*. IBM Journal of Research and Development, 1961.
[5] G. Kissin. *Measuring energy consumption in VLSI circuits: A foundation*. STOC 1982.
[6] J. Koomey, S. Berard, M. Sanchez, H. Wong. *Assessing trends in the electrical efficiency of computation over time*. Annals of the History of Computing, 2009.
[7] J. Koomey. *Growth in Data center electricity use 2005 to 2010*. Analytics Press, 2011.