Exceptional service in the national interest

Foundations of Generalized Reversible Computing

Michael Frank
Center for Computing Research
Sandia National Laboratories

9th Conference on Reversible Computation
Kolkata, India
July 6th, 2017

Approved for Unclassified Unlimited Release
SAND2017-6910 C

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

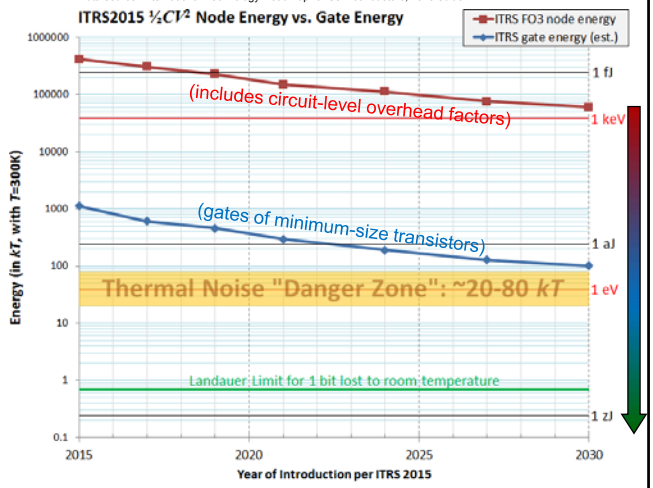
Talk Outline

- Motivation
  - Sustaining computer efficiency gains will require practical reversible HW
  - Modeling all reversible HW requires a more general theoretical foundation
- Fundamental physical foundations
  - Information/entropy, bijective dynamics, conditional entropy
- Formulating Landauer's Principle
  - A rigorous quantitative derivation from fundamental physics
- Redeveloping reversible computing theory from first principles
  - Traditional theory of unconditionally reversible computing
    - *Fundamental theorem of traditional reversible computing*
  - General theory of conditionally reversible computing
    - *Fundamental theorem of generalized reversible computing*
- Applications of Generalized Reversible Computing (GRC) theory
  - Examples of conditioned reversible operations
  - Modeling reversible hardware (*e.g.*, adiabatic circuits)
- Conclusion

Semiconductor Roadmap is Ending!

- Thermal noise on minimum-width segments of FET gate electrodes leads to channel fluctuations below ~1-2 eV
  - Increases leakage, impairs device performance
    - Thus, ITRS has minimum gate energy asymptoting to ~2 eV
- Also, real logic circuits incur many further overhead factors:
  - Transistor width 10-20× min.
  - Parasitic (junction, etc.) transistor capacitances (~2×)
  - Multiple (~2) transistors fed by each input to a given logic gate
  - Fan-out to a few (~3) logic gates
  - Parasitic wire capacitance (~2×)
- Due to all these overheads, the energy of each bit in real logic circuits is many times larger than the min.-width gate energy
  - 375-600× (!) larger in ITRS'15
    - ∴ Practical bit energy for irreversible logic asymptotes to **~1 keV**!
- Practical, real-world logic circuit designs can't just magically cross this ~500× architectural gap!
  - ∴ Thermodynamic limits imply much *larger* practical limits!
    - This is good news for our field…

Data source: International Technology Roadmap for Semiconductors, 2015 edition



Only reversible computing can take us from ~1 keV at the end of the CMOS roadmap, all the way down to $\ll kT$!

Motivation for this work

- To make reversible computing practical will be essential for sustaining the exponential growth of computer energy-efficiency over multi-decade timeframes, looking forward…
  - If we are successful at this, it can be reasonably expected to have a correspondingly enormous impact on future economic growth as well
    - Thus, there is a case for major investment, ***but*** we need a solid foundation!
- Unfortunately, the traditional (Landauer-Fredkin-Toffoli) theory of reversible logic circuits is not really adequate as a foundation for the design of real (physically) reversible hardware,
  - because it is *insufficiently expressive* to describe and explain the reversibility (at both the logical & physical levels) of the simplest real reversible hardware devices that we can build
    - We will see a number of examples and proofs of this later…
- Thus, we need a new, more general (and more expressive) theoretical foundation, which is suitable to serve as the basis for modeling any and all real reversible hardware devices & circuits…
  - We begin by developing a simple, deterministic framework for this…
    - Later, we can extend it to stochastic and/or quantum variants

Fundamental physical foundations

- It's widely known that Landauer's Principle (that information loss implies energy dissipation) is the rationale for studying reversible computing as a basis for energy efficiency gains…
  - But, it's perhaps less widely-understood how Landauer's Principle *itself* can be rigorously proved from more fundamental physics.
- We'll briefly review the following fundamental physical concepts that are necessary to fully understand the foundations of the thermodynamics of computation:
  - ***Entropy*** – What is it, how to understand its definition
  - ***Bijective dynamics*** – A core property of fundamental physics
  - ***Physical* vs. *computational* states** – And entropic implications
  - ***Conditional entropy*** – A visual illustration of its basic properties
- This will then set us up to rigorously derive a detailed quantitative form of Landauer's Principle…

Entropy in a Nutshell

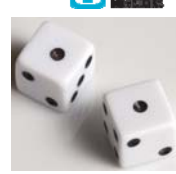
Basic review + coining some useful terminology

- Define the "surprisingness" or *surprise* $s(x)$ of any event x that has a 1 in m chance of occurring as $s = s(x) = s(m) = \log m$.
  - Call the $m \geq 1$ "improbability;" it can be a non-integer.
    - s is logarithmic b/c the improbabilities of independent surprises multiply.
  - *Indefinite* logarithm; dimensioned in *arbitrary* logarithmic units.
    - Some example units: $\log 2 = 1$ bit; $\log \mathrm{e} = 1$ nat $= k_{\mathrm{B}}$; $\log 10 = 1$ bel.
- In terms of event's *probability* $p = p(x) = p(m) = 1/m$,

$$s(p) = \log \frac{1}{p} = -\log p\,.$$

- Define event's "*heaviness*" $h = h(x) = h(p)$ (Hopefulness? Horribleness?) as its surprise, weighted by its probability:

$$h(p) = s/m = p \cdot s = p \log m = -p \log p.$$



Improbability: $m = 6 \times 6 = 36$

Surprise: $s = 2(\log 6)$

Heaviness: $h = \frac{s}{m} = \frac{2}{36} \log 6$

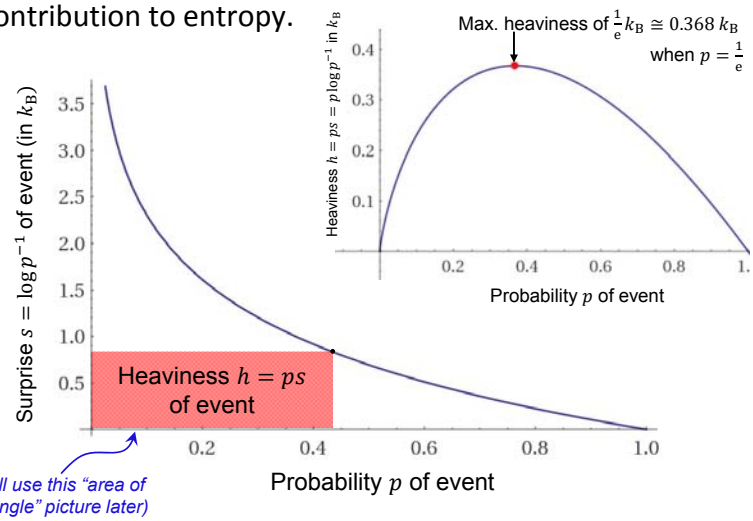
- Then for any probability distribution $p(x)$ over any mutually exclusive and exhaustive set of events $X = \{x_1, \ldots, x_n\}$, we have that the ***expected surprise*** $S(X) = \mathrm{E}_p[s(x)]$ and the ***total heaviness*** $H(X) = \sum_{x \in X} h(x)$ associated with that particular set of possible events are the same, and are given by:

$$S(X) = \sum_{x \in X} p(x) \cdot s(x) = H(X) = -\sum_{x \in X} p(x) \cdot \log p(x)\,.$$

- We call this quantity $H = S$ the *entropy* of the given epistemic situation.
  - By convention, we'll prefer H for "computational" entropy, S for "physical" entropy.

Surprise and Heaviness Functions

- For an individual state's contribution to entropy.

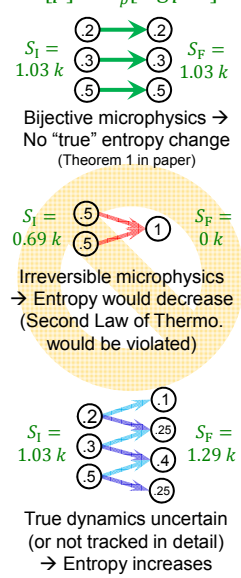


(We'll use this "area of rectangle" picture later)

Thermodynamics and Information

- *Physical* entropy quantifies *uncertainty about the detailed microstate of a physical system*.
  - First postulated by Boltzmann (in his H-theorem)
  - Integral to modern physics (Von Neumann entropy)
  - Depends on modeler's state of knowledge (Jaynes)
- The *reversibility (injectivity)* of microphysics <u>underlies</u> the Second Law of Thermodynamics.
  - States cannot merge as they evolve…
  - Thus, entropy of a closed system cannot decrease!
    - Conserved by unitary quantum time-evolution.
  - Entropy can *increase* if we have any uncertainty about the dynamics, or do not track it in detail
- At the most fundamental level, physical information *cannot be destroyed*.
  - Only *reversibly* transformed, and/or transferred between different subsystems…

$$S[p] = \mathrm{E}_p[\log p^{-1}]$$

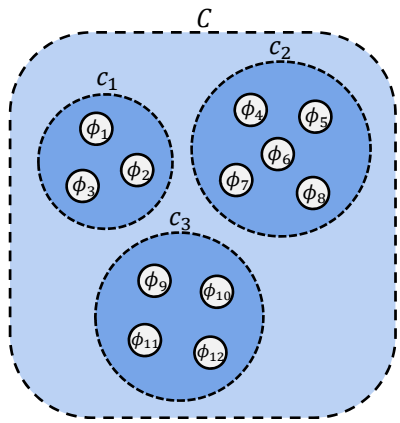


From Physics to Computation

- Thermodynamics and quantum mechanics show that any bounded physical system admits only a finite set $\Phi = \{\phi_1, \dots, \phi_n\}$ of measurably distinguishable detailed physical states (*microstates*).
  - *E.g.*, Φ could be any orthogonal set of basis vectors for the system's Hilbert space.
- We can *group* these microstates, that is, partition them into subsets c_j of micro-states that we consider as *equivalent* to each other for some designated purpose…
  - *e.g.*, for purposes of representing some specific *computational* information
- Any probability distribution $p(\phi_i)$ over the physical state space Φ induces a probability distribution P over the computational state space (subsystem) $C = \{c_j\}$ as well…

$$P(c_j) = \sum_{\phi_i \in c_j} p(\phi_i).$$

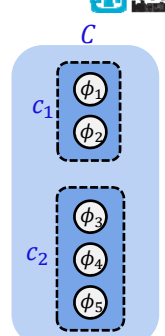
- This implies a *computational entropy* $H(C)$.

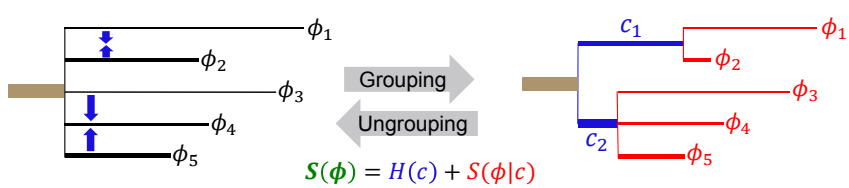


Example of a computational state space C consisting of 3 distinct computational states c_1, c_2, c_3, each defined as a set of equivalent physical states.

Visualizing Entropy of Grouped States

- Can represent a hierarchy of events in a tree structure…
  - Branch ***thickness*** = event probability p.
  - Branch ***length*** = *incremental surprise* Δs associated w. event,
    - relative to whatever base event it's branching off from.
  - Branch ***area*** = event's *incremental heaviness* $\Delta h = p\Delta s$, *i.e.*,
    - its contribution to total entropy, in addition to its base event's.
- *Grouping* events into larger events has these effects:
  - Thicknesses (probs.) of branches combine in parent branch
  - A corresponding part of the total length (surprise) of each branch is reassociated to parent (stem) branch.
  - Note: The total heaviness $\boldsymbol{H}$ of all branches and stems (total entropy $\boldsymbol{S}$) is not changed at all by any grouping/ungrouping!!

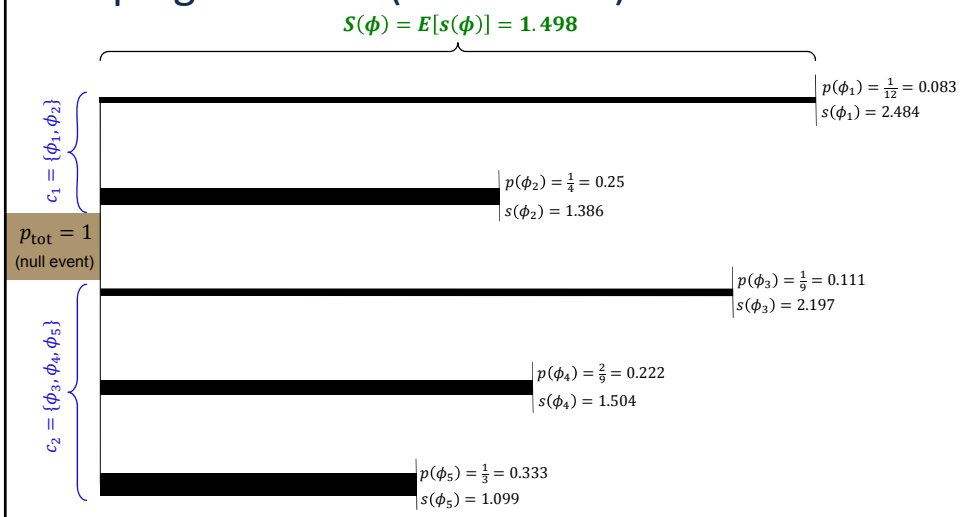




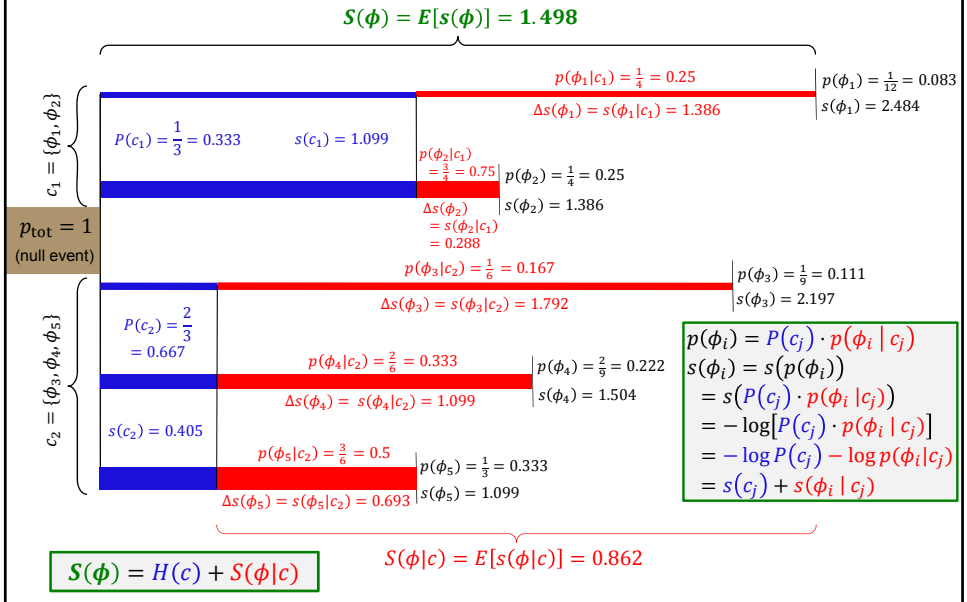
$$\boldsymbol{S}(\boldsymbol{\phi}) = H(c) + S(\phi|c)$$

Total system entropy = computational entropy + non-computational entropy

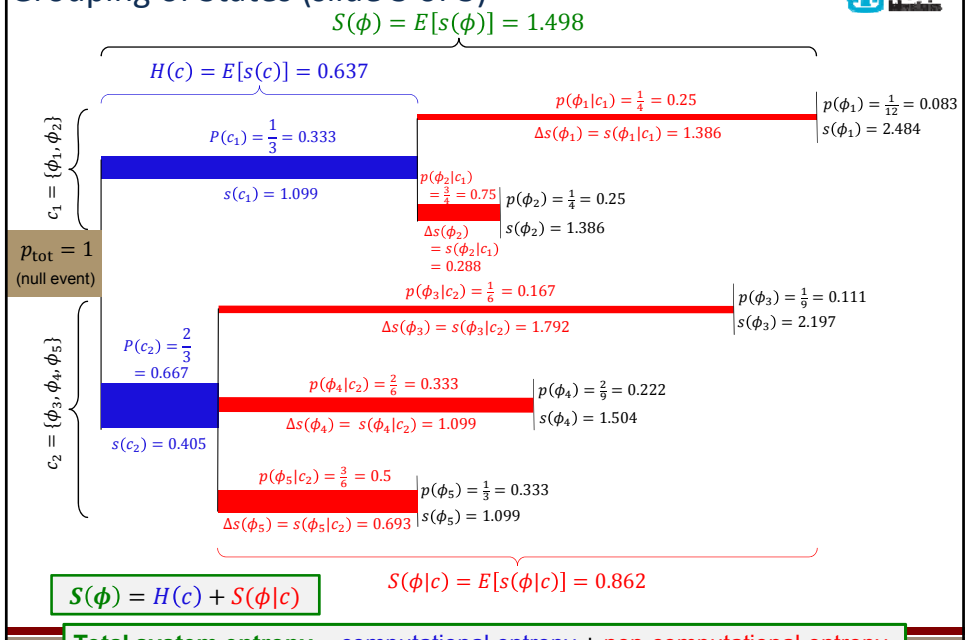
Grouping of States (slide 1 of 3)



Grouping of States (slide 2 of 3)



Grouping of States (slide 3 of 3)



$\boldsymbol{S(\phi)} = H(c) + S(\phi|c)$

Total system entropy = computational entropy + non-computational entropy

Proof of Landauer's Limit

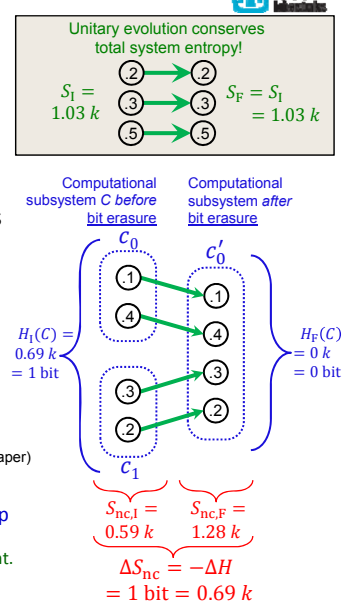
- We've seen that the total system entropy $S(\phi)$ for a given closed system cannot decrease at all…
  - So, what happens if we merge two *computational* states?
- Underlying probability distributions remain the same!
  - Only the identities of the physical states ϕ, and their groupings into computational states, can be changing
- Merging two computational states implies, removing a conceptual partition between groups of physical states
  - Same as the "ungrouping" operation we saw earlier
- The computational contribution $H(C)$ to the total entropy $\boldsymbol{S(\phi)}$ *cannot* simply vanish from existence…
  - Thus, it can only be *ejected* from the **computational** state into the ***non*-computational** state
- We define *non-computational entropy* as:

$$S_{\text{nc}}(\phi) = S(\phi|C) = \boldsymbol{S(\phi)} - H(C).$$

  - So, the *change* in $S_{\text{nc}}(\phi)$ from a merge operation is thus:

$$\Delta S_{\text{nc}}(\phi) = \Delta S(\phi|C) = -\Delta H(C).$$ (Theorem 2 in paper)

- To extent that "non-computational" = "uncontrolled,"
  - the extra non-computational entropy must ultimately end up in some thermal environment at some temperature T
    - We must thus emit at least heat $\Delta Q = T\Delta S$ to that environment. If $\Delta S = 1\text{ b} = k \ln 2$, then $\Delta Q = kT \ln 2$.



∴ **Landauer Limit**: $E_{\text{diss}} \geq kT \ln 2$ per bit lost. ■ (Corrollary 1 in paper)

Redeveloping Reversible Computing Theory

- We can characterize our task, in the development of the most fundamental aspects of Reversible Computing Theory, as follows:
  - We are given, as our starting point, this correct quantitative expression of Landauer's Principle, which follows rigorously from fundamental physics:

    $$\Delta S_{\mathrm{nc}}(\phi) = -\Delta H(C),$$

    - *i.e.*, for any computational operation, the change in the non-computational entropy of the physical state is exactly the negative of the change in the entropy of the computational state (*entropy ejected* from the computation).
      - Note this is referring the *minimum* non-computational entropy change necessitated by thermodynamics—there may be additional entropy increase from other causes.
  - Our task, then, is to infer, from this starting point, what *exactly* are the constraints on the *logical-level structure* of a computation that are necessary and sufficient conditions for this ***entropy ejected*** by a computation to attain or approach 0 (from above), $\Delta S_{\mathrm{nc}}(\phi) \rightarrow^{+} 0$.
    - This will then constrain the design of our physical mechanisms for computation.
- We can say that the following is the underlying essence of what we really mean (or should mean) when we talk about *logical reversibility*:
  - This phrase is, most generally, intended to express the *necessary and sufficient conditions*, at the logical level, under which the change in computational entropy approaches 0 (from below), $\Delta H(C) \rightarrow^{-} 0$.
    - However, Landauer's original definition of the phrase "logical reversibility" *did not actually accomplish this goal in the most general possible sense.*
      - The conditions he gave were sufficient, but, as we'll see *not necessary*.

Logical Reversibility, per Landauer

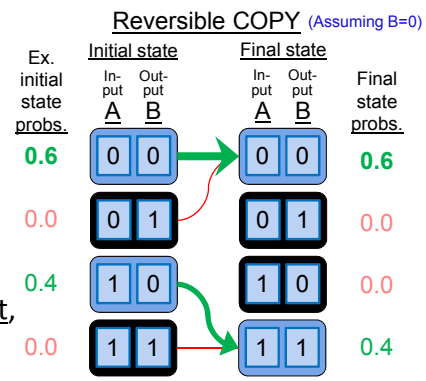
- Here, Landauer defines logical (ir)reversibility for an *N*-bit *device*…
  - Assumes the device is operating on the *entire* space of 2^N *combinatorially possible* initial states (or "inputs")

> … RESTORE TO ONE is an example of a logical truth function which we shall call *irreversible*. We shall call a device *logically irreversible* if the output of a device does not uniquely define the inputs.
>
> … Now assume that the computer is logically reversible. Then the machine cycle maps the 2^N possible initial states of the machine onto the same space of 2^N states, rather than just a subspace thereof.

- But, the probabilities of the initial states are also important!
  - DeBenedictis & Frank previously pointed this out, at the IEEE International Conference on Rebooting Computing 2016 (http://bit.ly/2hYWLdV)
- **Crucial:** If some initial states *have probability 0*, then *not* all of the 2^N combinatorially-possible initial states are *statistically* possible
  - In such operating contexts, a device's operation can transform the full combinatorially describable space of 2^N initial computational states onto a *smaller* set of final states, *while retaining* $\Delta S_{\mathrm{nc}} = 0$ (reversibility)!
    - Landauer's "logical reversibility" tragically obscured this critically important fact!

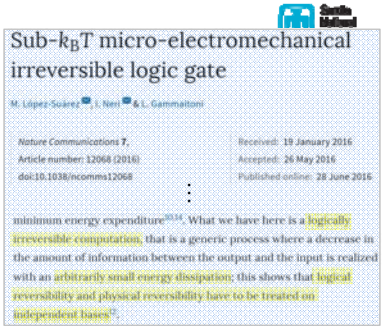
Logically reversible *computations* using "logically irreversible" *devices*

- The operation shown is "logically *ir*reversible" under Landauer's original, literal definition
  - Maps the 2^N=4 initial computational states to only 2 final states!
    - ∴ Merges some states!
- But in this specific operating context, *some initial probabilities are zero*.
  - Under such a distribution, note that the input *is* uniquely determined by the output, given the probabilities!
    - There are $<2^N$ *possible* (*i.e.*, nonzero-probability!) initial states of the device
    - This subset *is* mapped to a (different) set of states with the same size
  - This operation, *done in this context*, *is reversible*, because its $\Delta S_{\mathrm{nc}} = 0$!
    - Doing it does not eject any computational entropy into the environment!



Why this matters…

- Getting the definition of logical reversibility wrong is not without serious consequences!!
  - Example: An article was published in *Nature Communications* last year which claimed to empirically show that even logically irreversible computation doesn't require dissipation (!)


Sub-k_BT micro-electromechanical irreversible logic gate

M. López-Suárez, I. Neri & L. Gammaitoni

Nature Communications 7, Article number: 12068 (2016) doi:10.1038/ncomms12068

Received: 19 January 2016 Accepted: 26 May 2016 Published online: 28 June 2016

⋮

minimum energy expenditure. What we have here is a logically irreversible computation, that is a generic process where a decrease in the amount of information between the output and the input is realized with an arbitrarily small energy dissipation; this shows that logical reversibility and physical reversibility have to be treated on independent bases.

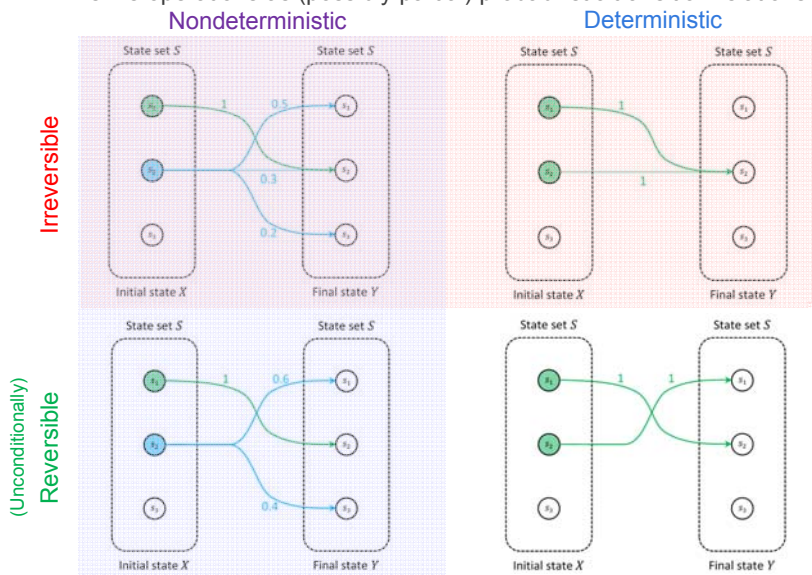
- Of course, the actual computation that they performed was, in fact, *logically reversible*, according to the *correct* definition…
  - It was an example of what I call a *conditioned reversible OR* operation, whose *precondition for reversibility* happened to be satisfied.
- But, we can hardly blame researchers for getting confused about this, when the definition of "logical reversibility" that we ourselves have used since Landauer is in fact *the wrong one*!!
  - Let's get this *right*, and avoid contributing to the widespread confusion…

Devices, Operations, Computations

- To set up the new theory, we'll find it helpful to distinguish several different concepts:
  - ***Device*** – Can perform one (or more) operations.
    - A given device has some associated local state information
      - Includes states of I/O terminals, internal states of device
  - ***Operation*** – a *(computational) operation* is a map O from *initial states* to *final states* (locally)
    - The terms "input" and "output" are too vague – avoid!
    - We can also consider partial maps (undefined=don't-care)
    - Generally speaking, the map O could also be stochastic…
      - Probabilistic transition rule $r_{ij} = \Pr[c_F = c_{Fj} \mid c_{\mathrm{I}} = c_{\mathrm{I}i}]$.
        - » However, that case is not our main focus at present
  - ***Computation*** – a computational operation *performed within a specific operating context*
    - Context specifies/constrains the initial state probabilities
      - These are <u>essential</u> for a meaningful thermodynamic analysis!

Types of Computational Operations

Define operations as (possibly partial) probabilistic transition relations



Operating Contexts & Entropy Ejection

- ***Operating contexts*** - A *(statistical) operating context* for a computational operation O is simply a probability distribution P_{I} over the distinguishable initial computational states of O.
  - A statistical situation in which the operation may be performed.
  - It carries an associated (computational) entropy $H_{\mathrm{I}} = H(P_{\mathrm{I}})$.
  - After performing the operation O in an operating context P_I, we will generally obtain some new ("final") probability distribution P_F,
    - with an associated final computational entropy $H_{\mathrm{F}} = H(P_{\mathrm{F}})$.

$$P_{\mathrm{F}}(c_{\mathrm{F}j}) = \sum_{i=1}^{|C_{\mathrm{I}}|} P_{\mathrm{I}}(c_{\mathrm{I}i}) \cdot r_{ij}$$

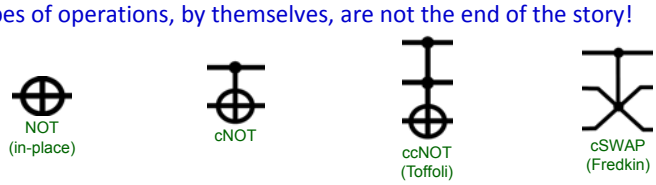
- ***Entropy-ejecting operations*** – A computational operation O is called *(potentially) entropy-ejecting* if and only if there is *some* operating context P_{I} such that, when the operation O is performed in that context, the entropy ejected from computational to non-computational form is positive,

$$\Delta S_{\mathrm{nc}} = -\Delta H = H_{\mathrm{I}} - H_{\mathrm{F}} > 0.$$

  - By convention, we will only call an operation O *non-entropy-ejecting* if (and only if) it is not even *potentially* entropy-ejecting.

Fundamental Theorem of Traditional Reversible Computing

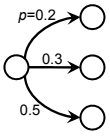
- **Theorem:** *(Theorem 3 in paper)* For any deterministic computational operation O, we have that O is non-entropy-ejecting (according to the prior definition) if and only if O is (unconditionally) reversible (*i.e.*, injective on its entire domain).
  - Proof can be carried out by considering what is the effect on computational entropy when two computational states w. nonzero probs. are merged.
    - However… What happens if a state with *zero* probability is merged with another?
      - This thought sets us up for GRC.
- There are of course many examples of deterministic, unconditionally-reversible operations; these are already very well-known in our field…
  - However, when it comes to *Generalized* Reversible Computing Theory, these types of operations, by themselves, are not the end of the story!



Conditionally Reversible Operations

- Focus here still on *deterministic* operations…
  - Could extend later to nondeterministic (randomizing) ops
- **Definition:** A (deterministic) computational operation O is called *conditionally (logically) reversible* if and only if there is *any* non-empty subset $A \subseteq \mathrm{dom}(O)$ of initial states (called an *assumed set*) that O maps one-to-one onto an equal-sized set of final states.
  - We can say, of any such A, that it is a *sufficient precondition for the (logical) reversibility of* O, or that it is *admissible*, for short.
    - We also refer to O_A (the concept of performing O in a context where A is satisfied) as a *conditioned reversible operation*.
- **Theorem:** All operations that are deterministically defined over some non-empty domain are conditionally reversible. (Slightly stronger version of Theorem 4 in paper.)
  - **Proof:** Consider any singleton set A that consists of any one of the initial states for which O is deterministically defined. ■
- **Theorem:** If O is deterministic, and has n reachable final states, then there's at least one admissible set $A \subseteq \mathrm{dom}(O)$ such that $|A| = n$.

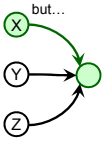
Nondeterministic



Final state is random

Deterministic

This operation is not reversible, but…



It's conditionally reversible, under the condition that the initial state is (say) X

Defining *computations*, and what entropy-ejection and logical reversibility mean for them

- ***Computations*** - A (deterministic) *computation* $\mathcal{C}$ performable by a device D is defined by a *pair* (O, P_{I}) of a deterministic operation O that D can do, *and* an operating context P_{I} for that operation.
  - This pair represents, performing the operation O *within the context* P_{I}.
  - O must be defined over at least all the nonzero-probability initial states
- ***Entropy-ejecting computations*** – A computation $\mathcal{C} = (O, P_{\mathrm{I}})$ is *(specifically) entropy-ejecting* if and only if, when O is performed within the *specific* operating context P_{I}, the entropy ejected from computational to non-computational form is positive, $\Delta S_{\mathrm{nc}} > 0$.
  - A computation (as opposed to an operation) shall be called *non-entropy-ejecting* as long as it is not *specifically* entropy-ejecting.
- ***Logically reversible computations*** – **(NEW, CORRECT DEFINITION)** A deterministic computation $\mathcal{C} = (O, P_{\mathrm{I}})$ shall be called *(specifically) logically reversible* if and only if the set A of all initial states that are assigned nonzero probability within the operating context P_{I} (a.k.a., the *active set*) is a sufficient precondition for the reversibility of O.

Fundamental Theorem of Generalized Reversible Computing

- **Theorem:** *(Theorem 5 in paper)* For any specific deterministic computation $\mathcal{C} = (O, P_{\mathrm{I}})$, we have that $\mathcal{C}$ is (specifically) non-entropy-ejecting if and only if $\mathcal{C}$ is (specifically!) logically reversible.
  - Note that here, we are using our ***new***, more general, contextualized definition of logical reversibility from the previous slide, ***not*** the traditional, more restrictive, context-independent definition!
    - The proof of the theorem is still similarly trivial, however.
- **Theorem:** *(Theorem 6 in paper)* Let O be any operation that is conditionally reversible under some assumed set A, and let the total probability assigned to A by P_{I} approach 1; then the entropy ΔS_{nc} ejected by the computation $\mathcal{C} = (O, P_{\mathrm{I}})$ approaches 0.
  - Thus, merely *approaching* 0 probability of violating some particular assumed one of the sufficient preconditions for reversibility is a sufficient logical-level condition for asymptotically approaching zero entropy ejection (*i.e.*, enabling an approach to thermodynamic reversibility).

Proof Sketch of Asymptotic Theorem

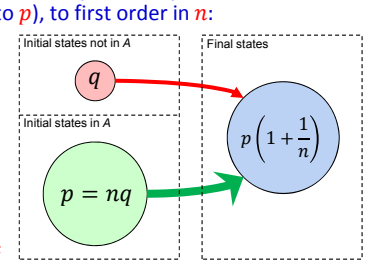
- **Theorem:** Let O be any operation that is conditionally reversible under some assumed set $A \subseteq \mathrm{dom}(O)$, and consider any progression $P_{\mathrm{I}1}, P_{\mathrm{I}2}, \ldots$ of operating contexts in which the total probability assigned to A by $P_{\mathrm{I}i}$ approaches 1 as $i \to \infty$; then the entropy ΔS_{nc} ejected by the computation $\mathcal{C} = (O, P_{\mathrm{I}i})$ approaches 0 in that progression, as $i \to \infty$.
  - By assumption, the sum of the probabilities of states not in A (that is, in the complementary set $B = \bar{A} = \mathrm{dom}(O) - A$) falls to 0:

$$\sum_{c \in B} P_{\mathrm{I}i}(c) \to 0 \text{ as } i \to \infty.$$

  - **Lemma:** For any state $c \in B$ with any probability $q = P_{\mathrm{I}i}(c) > 0$ that O merges with some state in A that has a larger probability $p = nq$ (where $n > 1$), the contribution Δs_{nc} of this state merger to the total entropy ΔS_{nc} ejected from the computation approaches the following expression as the probability ratio n increases (*i.e.*, as the probability q falls, relative to p), to first order in n:

$$\Delta s_{\mathrm{nc}} \to \frac{p}{n}(1 + \ln n) k_{\mathrm{B}}$$

    And, this value itself approaches 0, almost in proportion to $q = p/n$ as it falls.
  - Similarly, for states $c \in B$ merging with other states in B, their contributions Δs_{nc} to entropy ejected are upper-bounded by their heaviness $h = q \log q^{-1}$, which approaches 0 as $q \to 0$.
  - Since all the Δs_{nc} fall to 0, so does their sum ΔS_{nc}



Now, we can validly say this:

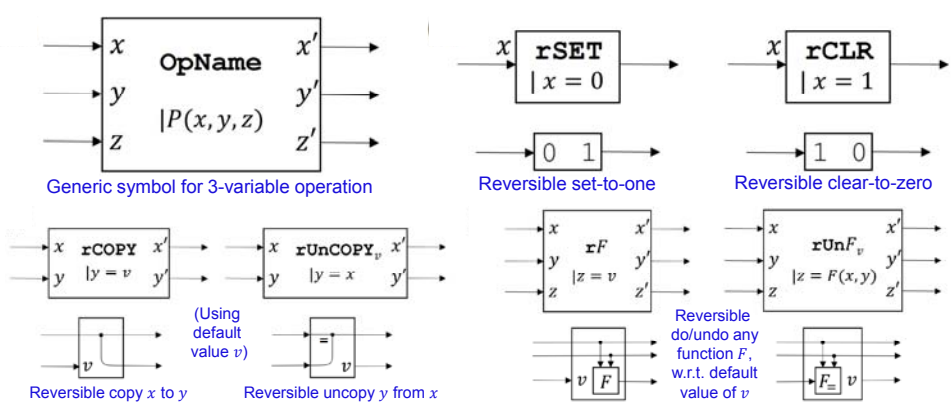
- **Assertion:** For any deterministic computation $\mathcal{C} = (O, P_{\mathrm{I}})$, that computation can be carried out in an asymptotically *thermodynamically* reversible way (within the context of some appropriately-designed family of implementable physical mechanisms) if <u>and only if</u> $\mathcal{C}$ is *specifically* logically reversible
  - …where, note that this assertion invokes our new, corrected definition of logical reversibility.
    - To prove the "if" part of this assertion (and make it a theorem) requires more technology development, but
    - The "only if" part already follows from Landauer's Principle.
- **NOTE:** The classic definition of logical reversibility is the ***<u>wrong one</u>***, because it does *not* actually satisfy (the "only if" part of) the above assertion!!

Notations and Examples

- For conditioned reversible operations; general form:

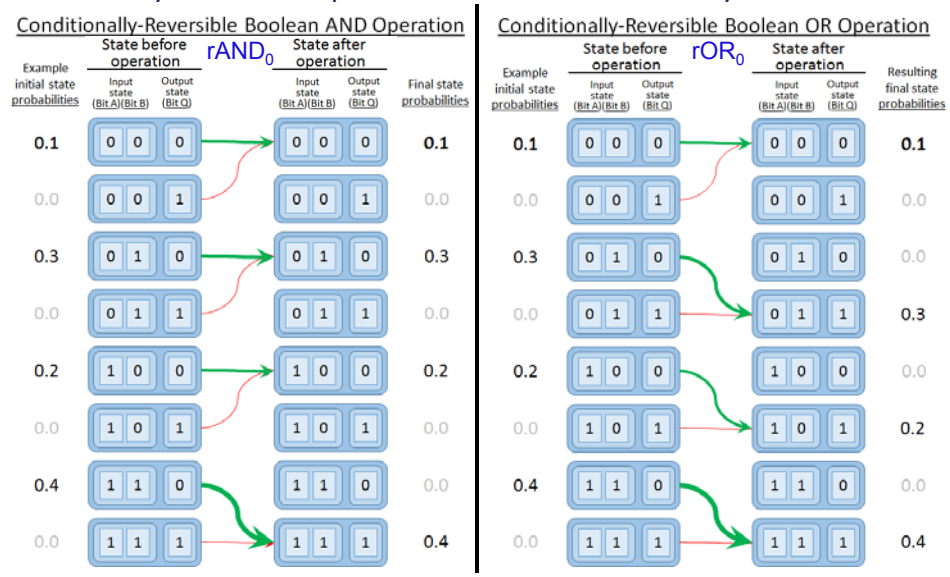
$$\texttt{OpName}(x, y, z \mid P(x, y, z))$$

  - Operation named **OpName**, operating on (typically binary) state variables x, y, z, with a precondition for reversibility of $P(x, y, z)$,
    - *I.e.*, the assumed set of possible initial states is $A = \{(x, y, z) \mid P(x, y, z)\}$.



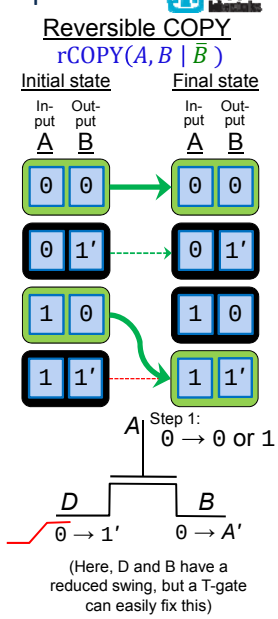
Conditionally-Reversible Operations are Useful!

Universality does *not* require unconditional reversibility



Conditionally-reversible operations can have very simple implementations!

- E.g.: Even *a single MOSFET* (operated adiabatically) can do a certain (conditioned) reversible COPY operation…
  - Operation sequence is as follows:
    0. Driving node D is initially statically held at 0, input *A* also 0.
    1. Input A is externally supplied (D&B connected iff *A* is high)
    2. Externally transition driver D from 0 to (weak) logic high 1′
    3. Voltage level on node B follows D iff A is strong logic high (**1**)
       - B is then afterwards logically equal to A (with a weak swing)
  - Note: Given a (strong) assumed precondition of $\bar{B}$,
    - *i.e.*, if all initial states with $B = 1$ have prob. 0,
    - this indeed performs a reversible COPY operation, $\mathrm{rCOPY}(A, B \mid \bar{B})$.
  - Note: The output in this case is not full-swing,
    - In this diagram, primes (′) denote reduced-voltage logic high signals
      - If we need full-swing outputs, we can use a transmission gate (parallel nFET/pFET pair) with complementary controls
  - A notation precisely describing this operation's semantics is:
    - $[\overline{AB}]$ if $B = 0$ then $B := A$ *(else, leave state unchanged)*
      - The expression $\overline{AB}$ in brackets gives the ***precondition for reversibility*** for the entire operation (the operation is both logically reversible and asymptotically thermodynamically reversible unless $A = B = 1$).
      - The remainder of the statement describes exactly how the state will be transformed in all cases (even if the precondition is not met).
- Note: Traditional reversible computing theory based on unconditionally reversible operations is insufficient to model the logical/physical reversibility of this operation!

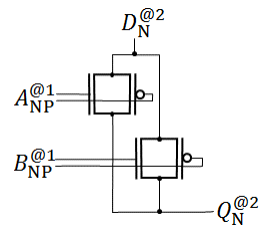


(Here, D and B have a reduced swing, but a T-gate can easily fix this)

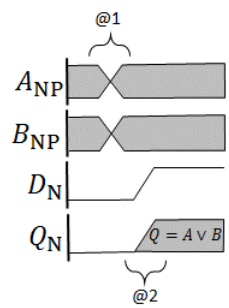
Two-Level Adiabatic Logic (2LAL)

Invented at the University of Florida, circa 2000

- Uses CMOS transmission gates for full-swing switching
  - Logic signals implicitly dual-rail (complementary PN pairs)
  - Logic NOT can be done by simply swapping (relabeling) rails
- Series/parallel T-gate combinations can do Boolean logic…
  - The parallel gate pair at right can be used to (conditionally) reversibly compute the output Q as $A \vee B$ (logic OR)
  - By DeMorgan's law, by complementing both inputs and output of this structure, we can also get: $Q = \overline{\overline{A} \vee \overline{B}} = A \wedge B$ (logic AND).
  - By cascading such gates, we can compute any binary function.
- Operation sequence:
  1. Precondition: Output signal Q is initially at logic 0
  2. Driving signal D is also initially logic 0
  3. At time 1 (@1), inputs A, B transition to new levels
     - Connecting D to Q if and only if A or B is logic 1
  4. At time 2 (@2), driver D transitions from 0 to 1
     - Q follows it to 1 if and only if A or B is logic 1
     - Now Q is the logical OR of inputs A,B
- Additional reversible steps that we can do afterwards:
  - Restore A, B to 0 (latches output Q in place at its current level)
    - 2LAL was the first adiabatic logic family capable of doing both logic and latching in the same structure!
  - Or, simply perform the above operation sequence in reverse



(and similarly for Q_P)



Unconditionally-Reversible Operations are only a special case…

More critiques of Landauer '61…

- It was Landauer who first introduced what's now called a *Toffoli gate operation*, or controlled-controlled-NOT (ccNOT), an unconditionally logically reversible operation:

  $$r := r \oplus pq.$$

  - Landauer describes (correctly) that AND can be embedded into this operation. (Given initial $r = 0$)
- However, his statement here that the AND operation "is not, in itself, reversible" is somewhat misleading!
  - That would only be true if:
    - The input bits were *consumed*…
      - But, in modern technologies such as CMOS, gates never actually consume their inputs!
    - Or, if the output bit was *destructively overwritten* with the result…
      - But, doing that is not necessary either!

> Consider, for example, a particular three-input, three-output device, i.e., a small special purpose computer with three bit positions. Let p, q, and r be the variables before the machine cycle. The particular truth function under consideration is the one which replaces r by $p \cdot q$ if $r = 0$, and replaces r by $\overline{p \cdot q}$ if $r = 1$. The variables p and q are left unchanged during the machine cycle. We can consider r as giving us a choice of program, and p, q as the variables on which the selected program operates. This is a logically reversible device, its output always defines its input uniquely. Nevertheless it is capable of performing an operation such as AND which is not, in itself, reversible.

- The approach Landauer takes here, of XOR'ing the result into the output bit, is indeed one that is logically reversible in *all* operating contexts.
  - But, it is rather complex to implement…
    - The simpler, conditionally-reversible setting of the output (w.r.t. an assumed default value) also works fine, in suitably restricted operating contexts!
- In my opinion, as a community, we should avoid making statements such as "AND and OR are irreversible," since this can be misleading…
  - Unless we include all of the appropriate caveats.

All truly, fully adiabatic circuits are conditionally reversible!

- "Dry switching" rules for designing truly adiabatic circuits:
  - Never close a switch when there's a voltage ≠0 between its terminals
    - *E.g.*, don't turn on a transistor when $V_{\mathrm{DS}} \neq 0$.
  - Never open a switch when there's a current passing through it.
    - *E.g.*, don't turn off a transistor when $I_{\mathrm{DS}} \neq 0$.
      - Only exception to this rule: If there's an alternate path for the current.
  - Never pass current through diodes (which have a voltage drop)
- Violating any of these rules leads to significant dissipation!
- **Theorem:** The operation of a switching circuit carries out a (conditionally) logically reversible computation, in any operation context where the above rules are always satisfied.
  - *It's impossible to erase information in any truly, fully adiabatic logic operation.* → Logically-reversible computing is key to adiabatic design
    - But, the *right* definition of "logically reversible" is our generalized one!
      - The same structures are *not* reversible/adiabatic if the rules are violated

Nondeterministic GRC Theory

- Let's briefly consider now the case of nondeterministic (here meaning, stochastic) computational operations…
  - These tend to increase the entropy in the computational state, that is, we have a positive change in computational entropy, $\Delta H > 0$,
    - Thus, they tend to *decrease* the non-computational entropy, so that we can have a *negative* change in non-computational entropy, $\Delta S_{\mathrm{nc}} < 0$…
      - Operations can be entropy-absorbing, as opposed to entropy-ejecting!
        - » *C.f. paramagnetic cooling* as a real physical example of this effect
- Even if a stochastic computation is logically "irreversible" (in the sense of, not injective over its active set), it could *still* be non-entropy-ejecting, or even entropy-absorbing…
  - As long as the increase in computational entropy due to any nondeterminism at least "breaks even," with respect to counteracting any decrease in computational entropy due to its irreversibility
    - Such cases still need to be analyzed and characterized in more detail

Some ideas for future work…

1. Give more examples of conditioned reversible operations;
   - Find other simple, natural primitives that are easy to implement;
2. Illustrate detailed physical implementations of devices for performing such operations;
3. Further develop the theory for the nondeterministic case;
   - Explore potential applications, *e.g.*, in randomized algorithms
4. Develop additional descriptive frameworks at higher levels, building upon the GRC foundation, *e.g.*:
   - Formal/algebraic representation theory for GRC constructs
     - Notations distinguishing preconditions for correctness *vs.* reversibility
   - GRC-based hardware description languages
     - GRC-based circuit architectures for useful functions
   - GRC-based programming languages
     - GRC-based software algorithms for useful functions

Conclusion

- By examining precisely how and why the validity of Landauer's Principle follows rigorously from fundamental physics,
  - we can clearly infer that there exists a much broader range of computations that can avoid ejecting entropy from the computational to the non-computational state than is traditionally recognized within reversible computing theory.
    - This leads to a corresponding *new definition* of logical reversibility that applies to appropriately-*conditioned* computational operations, as well as to specific computations operating within any statistical contexts that meet those conditions.
- It is the resulting theory of ***Generalized Reversible Computing*** (GRC), and *not* the traditional theory of reversible computing alone, that is the most appropriate foundation for the design of real reversible hardware, because:
  - It well models the (generally only conditionally-reversible!) nature of the reversible hardware devices that we actually know how to build;
    - Such as, for example, adiabatic circuits constructed out of MOSFETs;
  - It broadens the design space for constructing reversible architectures and algorithms,
    - And allows us to compose them out of simpler primitives than we have been using so far;
- Understanding GRC theory also helps resolve some longstanding, widespread confusions about the (correct, but widely misunderstood) connection between logical and physical reversibility, and between information and entropy.
  - We hope that this improved understanding will facilitate the emergence of reversible computing as the dominant foundation for 21st-century computing…
    - Since this will be absolutely required in order for the efficiency and economic impact of computing to continue increasing by many more orders of magnitude

CALL FOR PAPERS

IEEE Transactions on Emerging Topics in Computing

Special Issue on

Design of Reversible Computing Systems

Michael P. Frank and Marco Ottavi
Special Issue Editors

Submission deadline: March 1, 2018
Publication date: First issue of 2019

Full CFP online at: computer.org/tetc