

# Reversible and Computation

Seminar at Appalachian State University

Clément Aubert

06/02/2017

---

## Abstract

Students know that the flow of a program is a combination of sequential processing, branches, and loops. The introduction of exception and their handling, as well as of parallel threads, gives a more fine-grained view on the variations in a program's execution. There is one last variation, of critical impact, that won't be treated in the [CS2440](#) lecture: reversibility. Indeed, recently emerged a completely different way of handling the flow of a program, by allowing the program to go *back and forth*.

This requires every operation, i.e., statement, in the program to be *invertible*, so that any function, for instance, can seamlessly go from the input to the output, and from the output to the input. Allowing a program to go back and forth offers several advantages:

- As odd as it may seem, it *saves energy*, due to [Landauer's principle](#) that states that “*If no information is erased, computation may in principle be achieved which is thermodynamically reversible, and require no release of heat*”, i.e., no consumption of energy.
- It allows to re-use code: the program that computes the  $n$ th element of the [Fibonacci sequence](#) is the same as the one that, given a Fibonacci number, gives you its position!
- It forces to adopt a programming discipline where no data is ever lost: given an output, one may always ‘undo’ the computation to read back the input, so that the preservation of information is guaranteed.
- Bug tracking becomes easy!
- It also opens the door to a completely new way of writing and thinking algorithms.

I will provide a quick tour of the motivations and fundamentals of reversible computing during ~30 minutes, and will happily answer your questions for the rest of the lecture. Some material will be posted at <https://lacl.fr/~caubert/ASU/>.

## When and Where?

### When

Monday, February 6, 2017, 10:00 PM – 10:50 PM

### Where

[Appalachian State University, Anne Belk Hall](#), Room 325

---

## To Go Further

### Reversible Programming Languages

- [Janus](#) is probably the oldest and most robust reversible programming language. Its [playground](#) is unfortunately broken, but should be fixed soon.
- [Joule](#) is an object-oriented variation on Janus.
- [rfun](#) is an experimental, functional and reversible programming language, with an interpreter for Haskell.
- [Boomerang](#) is a ‘bidirectional programming language for ad-hoc, textual data’.
- [JsonGrammar](#) is a bidirectional ‘Haskell library for converting between Haskell datatypes and JSON ASTs’.

### Libraries

Code for reversible programming languages is hard to find, with one notable exception: [Sarah Vang Nøhr](#), published [the code](#) that resulted from her [her Master’s thesis](#) (*Reversible Graph Algorithms*, January 2015). Her pioneer work in the adaptation of graph algorithms for reversible computation is well-documented, solid, and enlightening. The code is written in Janus.

### Readings and Viewings

**Video** Holger Bock Axelsen, from the University of Copenhagen, gave an excellent 10-minutes introduction to [Reversible Computing](#).

### Textbook

- [Introduction to Reversible Computing](#), by [Kalyan S. Perumalla](#), is ‘envisioned to be suitable at the senior undergraduate and graduate levels.’

The same author gave a [tutorial](#) in 2014, that gives a rough idea of the extend of the topic, along with some useful references.

- An excellent introduction and panorama of the field is covered by [Michael Kirkedal Carøe](#)'s Ph.D. thesis *Design of Reversible Computing Systems*.

## Research Papers

- *Interpretation and Programming of the Reversible Functional Language RFUN*
- *Reversible Computation and Reversible Programming Languages*
- *Elements of a Reversible Object-Oriented Language* gives elements to extend Janus with object-oriented features.
- *Time, space, and energy in reversible computing* features an excellent introduction that surveys results of a quarter century of work on reversible computation.

## Traveling

Make sure to submit your work to [Reversible Computation](#) (the next edition will take place in India!).

---

## Misc.

- Download a [pdf version](#) or [the md source](#) of this page
- [HTML5](#) and [CSS3](#) valid
- [Creative Commons Attribution 4.0 International License](#)
- Contact: [aubertc@appstate.edu](mailto:aubertc@appstate.edu)
- Created with [debian](#), [pandoc](#), [latex](#) and [emacs](#).