Some Notes About Reversibility

Abstract: A typical computer user knows the difference between what can be undone on a computer, and what cannot. They may be familiar with the "undo" feature of text editors but understand the impossibility of recovering an unsaved document after an emergency shutdown. Creating programs guaranteeing that any action can be undone requires to design and implement reversible programming languages. While such languages come with interesting built-in security features (because any past action can be investigated), they also raise challenges when it comes to concurrency. Indeed, undoing an action that involved synchronization between multiple actors requires all actors to agree to undo said action. This talk offers to discuss current trends in solving the aforementioned problem, and to highlight some of the benefits that could result from well-designed concurrent and reversible programming languages.

Intro

A computer generally transforms an *input* into an *output*:

Input	Program	Output
`html` document	pandoc	`pdf` document
$\sqrt{16}$	calculator	4
A message and a private key	gpg	An encrypted message

Brief Presentation of Reversible Computation

Distinguish between causal consistent and non-causal consistent. Refer to various surveys about "flavors" of reversible computation.

Why Is Reversibility of Interest?

- It is connected to quantum computing,
- It is related to bio-computing
- It may open the possibility of low-energy computers,

- It will simplify the development and proof of correctness of some programs,
- It opens new perspective theoretically.

Quantum Computing

Definition of unitary operation. Is it just on the surface that quantum computation is reversible?

Bio-computing (chemical reactions)

Low-Energy Computers

Software-Engineering Benefits

Writing a proper file archiver (like 7-Zip, Ark or TAR) requires to:

- write a program to compress files into an archive,
- write a program to extract files from an archive,
- gain evidence that compressing then extracting files gives back the original files.

If that same program is written using a *reversible* programming language, then

- compressing means executing it forward,
- extracting means executing it backward,
- that compressing then extracting is the identity comes for free!

Another example is pandoc, or, actually, any program that performs lossless compression (frow `wav` to `flac`, but also in cryptography).

References