Week 13: Audacity

Roger B. Dannenberg
Professor of Computer Science and Art
Carnegie Mellon University

Introduction

- **Audacity**
- Audacity Implementation
- The Nyquist Plug-in Architecture

Audacity

- Graphical Audio Editor
- Cross Platform: Win32, Mac, Linux
- Currently for mono and stereo (but more channels possible)
- Good for large files
- Free and Open Source
- Active development team

Types of Audio Editors

In-Place

- Original samples are modified on disk.
- For example:
  - Adobe Audition (CoolEdit)

Non-Destructive

- Original files are left alone.
- For example:
  - Cubase
  - ProTools
  - Logic
  - Digital Performer

In-Place, Non-Destructive, and Audacity

- In-Place Features:
  - You see results of operations
  - Conceptually simple: direct manipulation
  - Precomputes audio: no real-time problems
  - Non-causal, out-of-time operations possible
- Non-Destructive Features:
  - Large files can be handled efficiently
  - Effect parameters can be adjusted without undoing other effects
- Audacity does *In-Place* with *efficiency.*

Introduction

- Audacity
- **Audacity Implementation**
- The Nyquist Plug-in Architecture

The Sequence Data Structure

Get(i, l): **Retrieve l consecutive samples from the ith sample.**

Set(i, l): **Change l consecutive samples from the ith sample.**

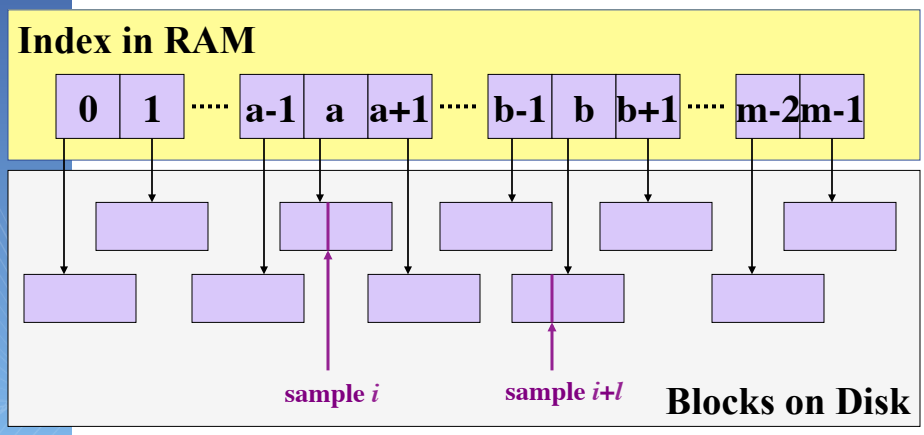
Insert(i, l): **Insert l consecutive samples before the ith sample.**

Delete(i, l): **Delete l consecutive samples from the ith sample.**

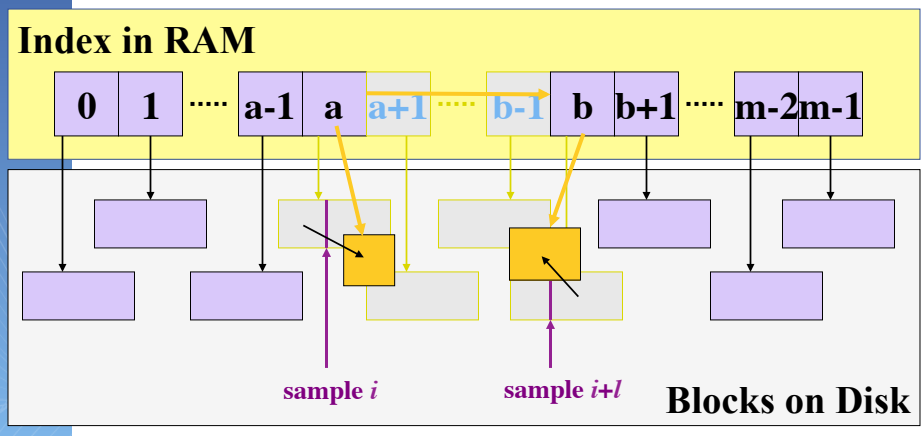
Our Sequence Implementation

- For some k, split the sequence into blocks with sizes between k and $2k$.
- When editing, always preserve this k-$2k$ property by rearranging the data within blocks.
- Any Sequence operation can be performed with this restriction in only constant (disk) time.

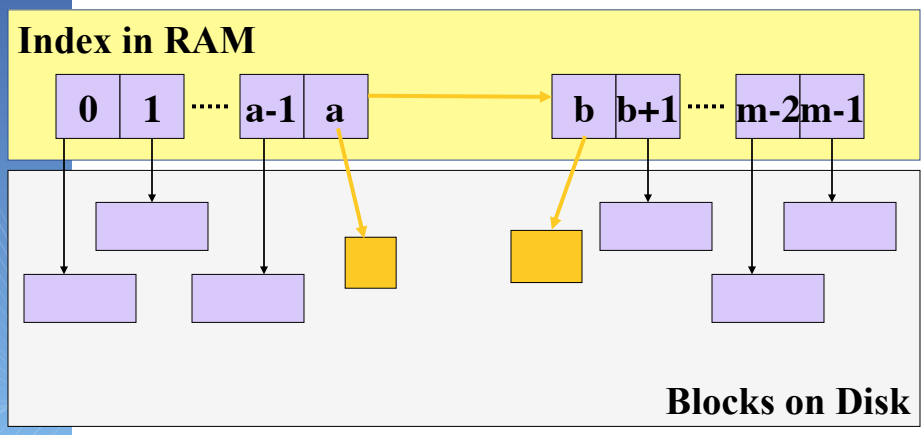
Example: Delete(*i*, *l*)



Example: Delete(*i*, *l*)



Example: Delete(*i*, *l*)



Advantages of a Sequence

- Speed (editing operations are fast, taking constant disk time).
- Easy to implement Undo by reference-counting the blocks.
- With reference-counting, the same block can appear in the Sequence more than once, making duplication/loops easy to implement with low storage overhead.

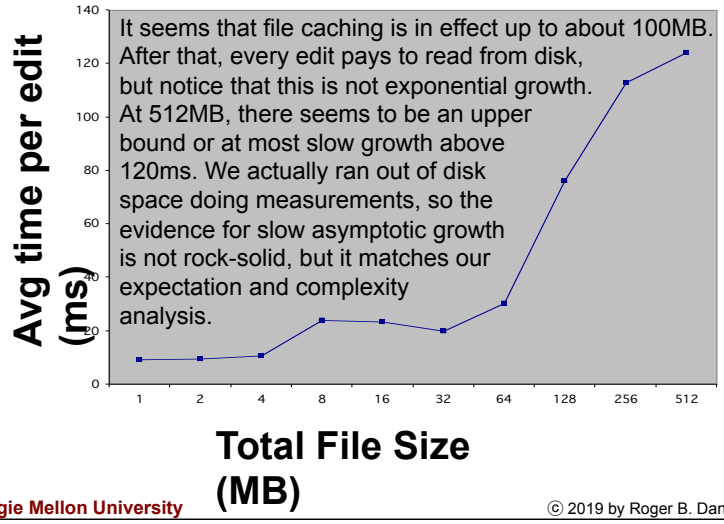
Disadvantages of a Sequence

- Each block is stored in a separate file. To move an audio project from one location to another, hundreds of small files must be moved.
- Soon, Audacity will use SQLite, a single-file, single-process SQL database that is very efficient with large objects. E.g. Photoshop uses it for thumbnails and other data – apparently doing a query to retrieve a thumbnail image is faster than going through directories using ordinary file systems.
- Using SQLite for Audacity projects, we'll use the same sequence-of-blocks implementation, but all blocks will be in a single project file which will be an SQLite database.

Fast Redisplay

- Sample amplitudes are summarized at two zoom levels
- And cached at head of blocks on disk
- Simplifies implementation
- Quite fast in practice
  - Avoids scanning actual samples
  - Only read data that appears on display
  - Discussion: would it be better to put sample amplitudes in separate files?

Performance Measurements (2001)



It seems that file caching is in effect up to about 100MB. After that, every edit pays to read from disk, but notice that this is not exponential growth. At 512MB, there seems to be an upper bound or at most slow growth above 120ms. We actually ran out of disk space doing measurements, so the evidence for slow asymptotic growth is not rock-solid, but it matches our expectation and complexity analysis.

Introduction

- Audacity
- Audacity Implementation
- **The Nyquist Plug-in Architecture**

Running Nyquist Within Audacity