A study of practical deduplication

Dutch T. Meyer  
University of British Columbia  
Microsoft Research Intern

William Bolosky  
Microsoft Research
Why Dutch is Not Here
A study of practical deduplication

Dutch T. Meyer
University of British Columbia
Microsoft Research Intern

William Bolosky
Microsoft Research
Why study deduplication?

9ms per seek

$0.039$ per GB

$0.046$ per GB
When do we exploit duplicates?

It Depends.

- How much can you get back from deduping?
- How does fragmenting files affect performance?
- How often will you access the data?
Outline

- Intro
- Methodology
- “There’s more here than dedup” teaser
  (intermission)
- Deduplication Background
- Deplication Analysis
- Conclusion
Methodology

MD5(name)
Metadata
MD5(data)

Once per week for 4 weeks.
~875 file systems
~40TB
~200M Files

2011 Storage Developer Conference. © Dutch Meyer. All Rights Reserved.
There’s more here than dedup!

- We update and extend filesystem metadata findings from 2000 and 2004
- File system complexity is growing
- Read the paper to answer questions like:

  Are my files bigger now than they used to be?
Teaser: Histogram of file size

Since 1981!
There’s more here than dedup!

How fragmented are my files?
Teaser: Layout and Organization

- High linearity: only 4% of files fragmented in practice
  - Most windows machines defrag weekly
- One quarter of fragmented files have at least 170 fragments
Intermission

- Intro
- Methodology
- “There’s more here than dedup” teaser

(intermission)

- Deduplication Background
- Deplication Analysis
- Conclusion
Dedup Background

Whole file Deduplication

foo

01101010..... ....110010101

bar

01101010..... ....110010101
Dedup Background

Fixed Chunk Deduplication

Foo

\[01101010\ldots\quad\ldots\quad110010101\]

Bar

\[01101010\ldots\quad\ldots\quad110010101\]
Rabin Fingerprinting

```
<table>
<thead>
<tr>
<th>foo</th>
<th>01101010..</th>
<th>00100000...</th>
</tr>
</thead>
<tbody>
<tr>
<td>bar</td>
<td>01101010..</td>
<td>01010011..</td>
</tr>
</tbody>
</table>
```

```
# The Deduplication Space

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Parameters</th>
<th>Cost</th>
<th>Deduplication effectiveness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whole-file</td>
<td></td>
<td>Low</td>
<td>Lowest</td>
</tr>
<tr>
<td>Fixed Chunk</td>
<td>Chunk Size</td>
<td>Seeks</td>
<td>Middle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CPU</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Complexity</td>
<td></td>
</tr>
<tr>
<td>Rabin fingerprints</td>
<td>Average Chunk Size</td>
<td>Seeks</td>
<td>Highest</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CPU</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>More</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Complexity</td>
<td></td>
</tr>
</tbody>
</table>
What is the relative deduplication rate of the algorithms?
Dedup by method and chunk size

![Graph showing deduplication percentages for different chunk sizes and methods. The x-axis represents chunk sizes (64K, 32K, 16K, 8K), and the y-axis represents the percentage of space deduplicated. The graph shows three lines: Whole File (blue), Fixed-Chunk (red), and Rabin (green).]
What if I was doing full weekly backups?
Backup dedup over 4 weeks

- 8K rabin: Deduplicated Space
- Whole File + Sparse: Deduplicated Space
- Whole File: Deduplicated Space
How does the number of filesystems influence deduplication?
Dedup by filesystem count

Deduplication Domain Size (file systems)

- Whole File
- 64 KB Fixed
- 8KB Fixed
- 64KB Rabin
- 8KB Rabin

Space Deduplicated
So what is filling up all this space?
Bytes by containing file size

Percentage of Total Bytes

Containing File Size (Bytes), Power-of-2 bins

2000  2004  2009
What types of files take up disk space?
Disk consumption by file type

2000 | 2004 | 2009
--- | --- | ---
0%  | 0%  | 0%
10% | 10% | 10%
20% | 20% | 20%
30% | 30% | 30%
40% | 40% | 40%
50% | 50% | 50%
60% | 60% | 60%

File types: dll, pdb, vhd, exe, lib, pst, chm, cab, pch, mp3, wma, iso
Disk consumption by file type

- 2000
  - dll
  - cab
  - chm
  - lib
  - mp3
  - pch
  - pst
  - exe
  - pdb

- 2004
  - mp3
  - pst
  - cab
  - lib
  - wma
  - exe
  - pdb
  - vhd

- 2009
  - iso
  - wma
  - cab
  - pch
  - exe
  - pdb
  - vhd
  - dll
  - ø
Which of these types deduplicate well?
# Whole-file duplicates

<table>
<thead>
<tr>
<th>Extension</th>
<th>% of Duplicate Space</th>
<th>Mean File Size (bytes)</th>
<th>% of Total Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>dll</td>
<td>20%</td>
<td>521K</td>
<td>10%</td>
</tr>
<tr>
<td>lib</td>
<td>11%</td>
<td>1080K</td>
<td>7%</td>
</tr>
<tr>
<td>pdb</td>
<td>11%</td>
<td>2M</td>
<td>7%</td>
</tr>
<tr>
<td>&lt;none&gt;</td>
<td>7%</td>
<td>277K</td>
<td>13%</td>
</tr>
<tr>
<td>exe</td>
<td>6%</td>
<td>572K</td>
<td>4%</td>
</tr>
<tr>
<td>cab</td>
<td>4%</td>
<td>4M</td>
<td>2%</td>
</tr>
<tr>
<td>msp</td>
<td>3%</td>
<td>15M</td>
<td>2%</td>
</tr>
<tr>
<td>msi</td>
<td>3%</td>
<td>5M</td>
<td>1%</td>
</tr>
<tr>
<td>iso</td>
<td>2%</td>
<td>436M</td>
<td>2%</td>
</tr>
<tr>
<td>&lt;a guid&gt;</td>
<td>1%</td>
<td>604K</td>
<td>&lt;1%</td>
</tr>
</tbody>
</table>
What files make up the 20% difference between whole file dedup and sparse file, as compared to more aggressive deduplication?
Where does fine granularity help?

Percentage of difference vs. whole file + sparse

- 8K Fixed
- 8K Rabin
Last plea to read the whole paper

- ~4x more results in paper!
- Real world filesystem analysis is hard
  - Eight machines months in query processing
  - Requires careful simplifying assumptions
  - Requires heavy optimization
Conclusion

- The benefit of fine grained dedup is < 20%
  - Potentially just a fraction of that.
- Fragmentation is a manageable problem
- Read the paper for more metadata results

We’re releasing this dataset (when I finish the anonymization)