Vol.2012-EVA-37 No.1
2012/3/30

Dedup 技術の有効性の分析

小池　到[†]　古谷 友典[†]　山崎 寛人[†]　堀江真一郎[†]
木下 俊之[†]　早坂光雄[††]　櫻庭健年[††]
[†] 東京工科大学　バイオ・情報メディア研究科
[††](株)日立製作所 横浜研究所

あらまし　Dedup を利用することで，大規模なデータストレージに対して効率的にデータ削減を行うことができるが，データストレージの使用環境によっては効果が得られない問題がある．そこで本研究では，Dedup の特性を分析し，Dedup が有効となるデータストレージの要因を，利用者やデータの特徴から述べる．また，Dedup はストレージ全体で圧縮を行う手法のため，ファイル単位の圧縮手法と併用することができる．その際におけるデータ圧縮の有効性や最も効率のよい手法を述べる．
キーワード　Dedup，データ圧縮，シングルインスタンスストレージ

Analyse validity of Dedup technology

Itaru KOIKE[†]　Tomonori HURUYA[†]　Hiroto YAMAZAKI[†]
Shinichiro Horie[†]　Toshiyuki KINOSHITA[†]
Mitsuo HAYASAKA[††]　Taketoshi SAKURABA[††]
[†]Tokyo University of Technology
[††]Yokohama Research Laboratory, Hitachi, Ltd

Abstract　Dedup (De-duplication) technique can efficiently reduce a large scale data in a storage system. But, the reduction effect may depend on the user data stored in it. To analyze the validity of Dedup from the point of view of users and data characteristics, we measured the effectiveness of Dedup in the IT related research environment in a university. Also, we examine the effect of the combination of compression techniques with Dedup since Dedup is often used with data compression to reduce the total size of data.
Keyword　Dedup, data compression，Single Instance Storage

1. はじめに

情報システムにおけるデータストレージは，利用者の増加によるシステムの巨大化やバックアップの蓄積により，保持するデータ量は増加し続けている．増加するデータの中には，類似のデータが複数かさんで無駄なデータとなって膨らむ問題が起きている[1].

データストレージ全体の圧縮には幾つかの手法が考えられるが，Dedup(Data De-duplication)は複数個ある重複したデータを 1 つにまとめることで，全体のデータ量を削減する手法である．この手法では，バックアップ等の類似性の高いデータに対しては非常に高い効果を得られるが，類似性の低いデータには効果が得られず，対象の使用環境で効果が大きく異なるという特徴がある．このため Dedup を利用する際には，事前に有効性の確認が必要である[2]．本研究はこの Dedup の有効性について，ストレージの利用環境と対象のデータの特徴に基づいた分析を行う．

また Dedup はストレージ全体でデータ圧縮を行うが，ファイル毎に圧縮を行う手法と併用することで，類似性の低いデータへのデータ削減効果を持たせるなどの，Dedup の効果を上げる手法を検討する．

2. 関連研究

Dedup と類似した手法に，シングルインスタンスストレージがある．シングルインスタンスストレージは同一のファイルを 1 つにまとめる手法である（図 1）．この手法ではストレージ内に存在するファイルを 1 箇所にまとめ，元ファイルは格納先へのリンクを生成し利用可能にしている．Dedup は同一ファイルだけでなく，複数のファイル内に存在する同一データ部分の重複排除を行うことができる．これにより，一部を修正されたファイルや，追記されたデータに対しても重複排除を行うことができ，シングルインスタンスストレージより高いデータ削減が期待できる(図 2)．しかし，Dedup では重複の検出や削除のための処理時間が増えるため，ファイル内の部分的な重複排除をどの程度行えるかが重要な課題となっている[3].

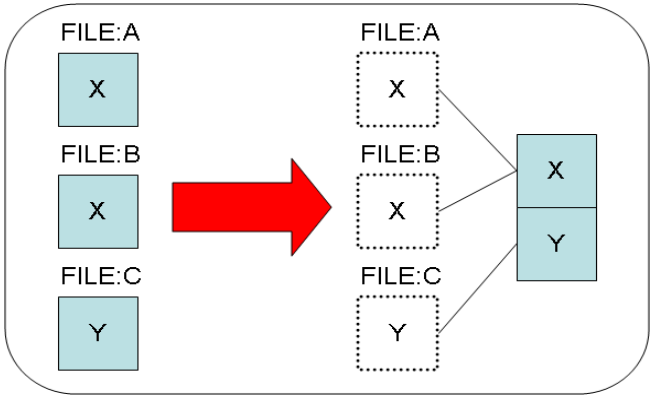

図 1　シングルインスタンスストレージの概念

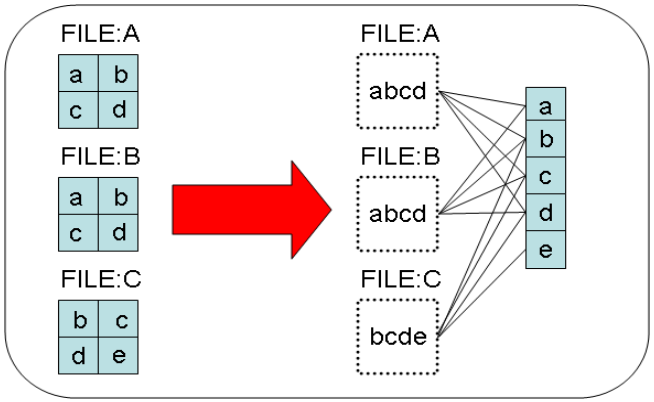

図 2　Dedup の概念

3.　実験方法

Dedup にはファイルを幾つかのデータ部分に分割するが（これをブロックと呼ぶ），ブロックサイズが固定長の場合と可変長の場合の 2 種類のブロック生成方式がある．固定長ブロックは処理時間が非常に短いが，データの一部が変更されていると，修正箇所以降のブロックは適切に重複検索が行われなくなる問題というがあり，限られた場合しか用いられない．そこで本研究では可変長ブロックと対象とする．

対象のデータベースには，東京工科大学木下研究室のファイルサーバを使用する．これは研究室に所属する学生の研究成果や研究に使用したデータの一括保存場所になっており，過去約 60 人が使用している．つまりバックアップのような使い方はしていないが，複数の学生が同じテーマの研究を行った場合に，類似のファイルが生成される可能性がある．このような大学の研究室で，バックアップ用途ではない通常の使い方をした場合の Dedup の有効性について検証する．また，幾つかのファイル形式毎に Dedup の結果をまとめ，重複排除に適するファイルの種類を分析する．

さらに，Dedup と次にあげる圧縮手法を併用した場合の効果について解析する．併用する圧縮手法は，Dedup の処理時間短縮のため Dedup を行う前に行った．実験により，併用する圧縮手法の違いによる Dedup の効果の違いについて評価する．

併用する圧縮手法

・連長圧縮
連続した記号を，[連続した記号][記号の数]に変換する．

・ハフマン符号
ファイル内の記号の頻度から，最適な符号化を行う．

・deflate
ファイル内の重複排除を行った後ハフマン符号化する．

4.　実験結果

4.1　Dedup 結果

前述のファイルサーバに Dedup を行った結果を表 1 に示す．

表 1　Dedup 結果

対象	Dedup 前 (byte)	Dedup 後 (byte)	重複排除率 (%)
ファイルサーバ	6,725,323,611 約 6.7G	5,145,314,064 約 5.1G	23.5

6.7Gbyte のサーバに対して，1.6Gbyte(約 23.5%)の容量が削減された．この内，重複排除率 100%のファイルは 12,629 個であり，全体(35,706 個)の約 30%であった．その容量は約 940Mbyte であり，削減された 1.6 Gbyte の約 60%になる．つまり，残り 40%の 660Mbyte は部分的な重複排除による削減であり，ファイルインスタンスストレージよりも十分にデータ削減効果が出ていることが分かる．

4.2 ファイル形式

データベース内に多く存在していたファイル形式は，主に画像形式，文書形式，プログラムファイルである．表 2 に，ファイルサーバ全体の Dedup を行った内のファイル形式別の結果を示す．

表 2 のファイル数は Dedup の対象としたファイル数であり，結果として重複排除されなかったファイルも含まれる．また重複排除率は，Dedup を行ったファイルがどれ程データ削減されたかであり，元のファイルサイズ/Dedup 後ファイルサイズで求めている．図 2 に表 2 の中で重複排除が行われたファイル数の割合を，図 3 に重複排除率を示す．

表 2 ファイル形式毎の結果

拡張子	ファイル数	重複排除したファイル数		重複排除率(%)
bmp	1,623	766	47.1%	42.6
eps	88	75	85.2%	29.6
gif	2,297	433	18.9%	12.9
jpg	585	106	18.1%	24.9
png	418	67	16.0%	13.0
c	4,875	2198	45.1%	43.0
java	268	174	65.0%	62.7
cpp	603	403	66.8%	85.0
ppt	199	119	59.8%	28.0
doc	392	96	24.5%	21.0
rtf	113	60	53.1%	22.9
pdf	84	43	51.2%	15.8
html	1,319	227	17.2%	10.7
txt	635	235	37.0%	40.9
tex	62	12	19.4%	30.6
exe	487	352	72.3%	44.1

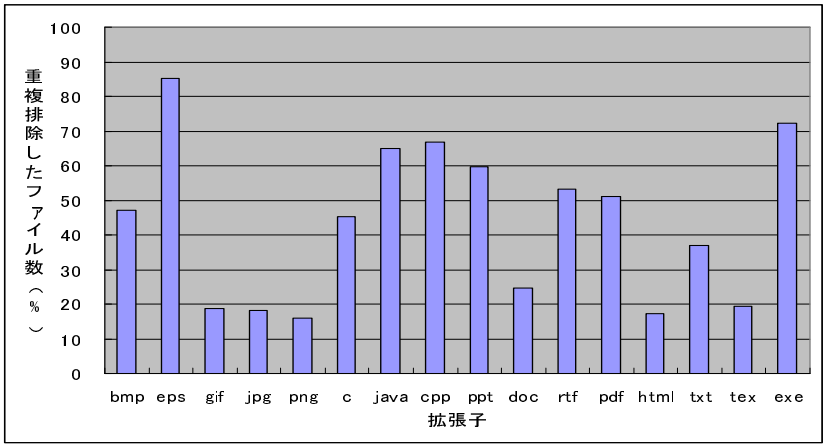

図 3 重複排除したファイル数

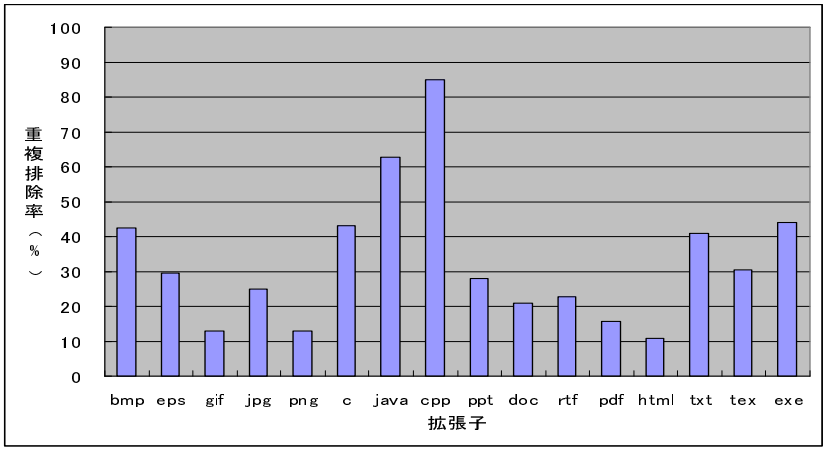

図 4 重複排除率

図 3 によると，重複排除されたファイル数が多いのは，eps や exe，java，cpp 形式であり(65%以上)，少ないもの(24.5%以下)はすでに圧縮された画像形式の gif や jpg，png 形式の他に文章ファイルでは doc，pdf，html 形式である．

図 4 によると，重複排除率が高いのは cpp や java，c のソースプログラム形式のものである．これは研究室内で開発キットを共有していたり，プログラムを引継いでいたり，過去バージョンのバックアップなどが要因にあげられる．これにより，プログラム開発を行う研究機関では類似コードが多く存在し，Dedup が有効な環境であるといえる．

一方，重複排除率の低いのは，gif，png，html 形式であり，これらは重複排除されたファイル数も少ないため Dedup 向けの形式ではないと考えられる．

4.3 部分重複排除率

前節 4.2 の Dedup 結果には，同一のファイル(重複排除率 100%)と，ファイルの一部が重複排除された部分重複排除(重複排除率 0.1～99.9%)のファイルが含まれている．この内, 同一のファイルを除き, 部分重複排除が行われた Dedup の結果を表 3 に示す．

表 3　部分重複排除

拡張子	ファイル数	部分重複排除ファイル数		部分重複排除率の平均(%)
bmp	1,623	295	18.2%	52.4
eps	88	67	76.1%	61.6
gif	2,297	13	0.6%	62.0
jpg	585	18	3.1%	45.7
png	418	6	1.4%	30.9
c	4,875	119	2.4%	50.4
java	268	0	0.0%	0.0
cpp	603	88	14.6%	78.0
ppt	199	89	44.7%	35.5
doc	392	4	1.0%	0.0
rtf	113	43	38.1%	28.6
pdf	84	11	13.1%	24.4
html	1,319	112	8.5%	27.8
txt	635	30	4.7%	39.0
tex	62	2	32.3%	24.3
exe	487	194	39.8%	35.4

表 3 によると，部分重複排除されたファイル数が抜きん出て多いのが eps 形式であり，他の画像形式では bmp 形式以外はとても少ない．ソースプログラム形式でも c と java 形式は部分重複されたファイルが圧倒的に少なく，表 2 で重複排除率が比較的高いのは部分重複排除時ではなく同一のファイルが多いためであることを示している．cpp は部分重複排除されたファイル数は 14.6%と高くないが分重複排除率が 78%と非常に高く，これはよく似たファイルが多いことを示している．文章形式では ppt 形式がよく部分重複排除されているが, 他の文章形式（特に doc 形式）では非常に少ない．以上より，部分重複排除は画像ファイルでは bmp 形式と eps 形式，文章形式では ppt 形式と pdf 形式で多いが，それ以外では少ないといえる．

eps 形式では部分重複排除ファイル数が多いが，その原因を 4.4 で分析する．

4.4 eps 形式の部分重複排除率

eps 形式では部分重複排除が多い．この要因は 2 つ考えられる．1 つは他のファイル形式の画像の同一部分と重複排除が行われている可能性である，eps 形式では一部的な重複が存在しやすいことである．前者について表 4 に，eps 形式のみで Dedup を行った場合と，4.2 節のデータベース全体で Dedup を行った結果の比較を示す．

表 4　eps 形式の比較

対象範囲	重複排除率(%)	部分重複排除ファイル数
eps のみ	28.4	65
データベース	29.6	67

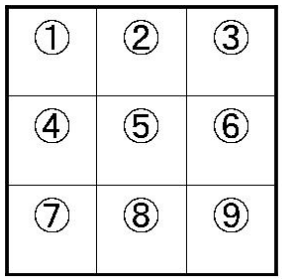

図 3　画像 A

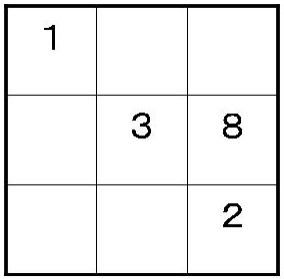

図 4　画像 B

表 5　類似した eps 画像

画像	ファイルサイズ (byte)	重複排除サイズ (byte)	重複排除率 (%)
eps 画像 A	504,928	0	0.0
eps 画像 B	504,928	315,616	62.5
jpg 画像 A	14,062	0	0.0
jpg 画像 B	8,333	0	0.0

表 4 によると，重複排除率は，データベース全体で Dedup を行った場合のほうが，eps 形式のみより 1.2%高く，ファイル数は 2 つ多い．しかしこれは，eps 形式において部分重複排除が多い原因としては微々たるものである．よって eps 形式で部分重複排除ファイル数が多い理由は，eps 形式同士で部分重複排除が頻繁に起こるためである．試しに，データベース内にある視覚的に類似した 2 種類の画像（図 3，図 4）の eps 形式と jpg 形式に Dedup を行った結果を，表 5 に示す．

eps 形式の画像 B では 62.5%が重複排除されたが，jpg 形式では部分重複排除されなかった．このように，eps 形式の画像は，視覚的に似た画像の部分重複排除を適切に行えることが分かる．jpg 形式や gif 形式，png 形式では圧縮操作が行われるため，画像が似ていてもうまく部分重複排除が行えないと考えられる．

4.5　他の圧縮手法と併用

連長圧縮，ハフマン符号，deflate 圧縮の 3 種の圧縮手法をファイルサーバ内の全ファイルに用いた後に，Dedup を使用した結果を表 6 に示す．

表 6 によると，圧縮と併用した場合の重複排除率は何れも 7%以上減少したが，Dedup 後のサイズは何れも Dedup のみより小さくなっているため，圧縮手法と Dedup の併用は有効であるといえる．ハフマン符号は圧縮併用の中で最も重複排除率が高いが，圧縮効果が最低のため，Dedup 後のサイズは最大なった．

連長圧縮では，重複排除率がハフマン符号と比較して若干低下したが，Dedup 前の圧縮効果により全体のファイルサイズは小さくなった．Deflate 圧縮では，他の圧縮手法よりも大きく圧縮した上に，他と殆ど同じ重複排除率だったため Dedup 後のサイズは最小になり，最も効率が高かった．deflate 圧縮後の Dedup の拡張子毎の重複排除結果を表 7，図 5，図 6 に示す．

表 6　Dedup の併用

手法	圧縮後 (byte)	Dedup 後 (byte)	重複排除率 (%)
Dedup のみ	6,725,323,611 約 6.7G	5,145,314,064 約 5.1G	23.5
ハフマン符号	5,471,588,743 約 5.5G	4,580,943,219 約 4.6G	16.3
連長圧縮	5,393,133,848 約 5.4G	4,528,120,960 約 4.5G	16.0
deflate	4,496,887,768 約 4.5G	3,783,867,126 約 3.8G	15.9

表 7 deflate 圧縮後の拡張子

拡張子	ファイル数	重複排除したファイル数		重複排除率 (%)
bmp	1,620	597	36.9%	40.9
eps	88	15	17.0%	7.8
gif	2,295	426	18.6%	12.6
jpg	585	92	15.7%	24.4
png	418	62	14.8%	11.0
c	4,875	2146	43.9%	42.9
java	268	174	65.0%	63.1
cpp	603	380	63.0%	77.0
ppt	151	37	24.5%	8.9
doc	309	47	15.2%	14.5
rtf	113	16	14.2%	11.9
pdf	84	33	39.3%	19.7
html	1,319	119	9.0%	7.0
txt	618	215	34.8%	41.8
tex	62	11	17.7%	27.0
exe	487	197	40.5%	41.8

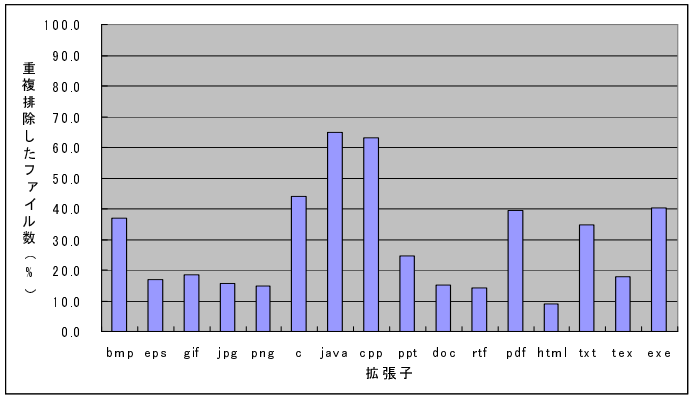

図 5　deflate 圧縮後の重複排除したファイル数

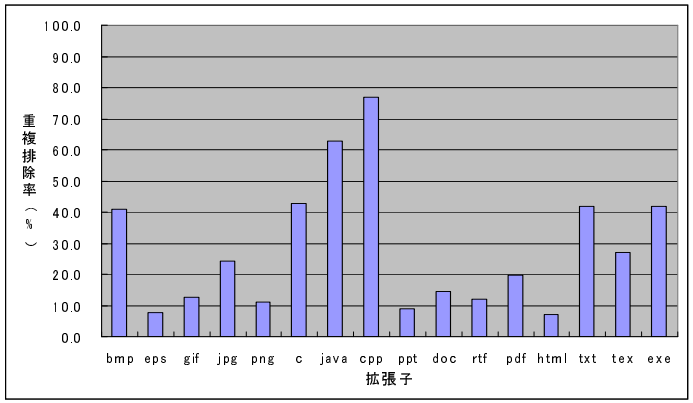

図 6 deflate 圧縮後の重複排除率

Dedup のみの結果では（表 2），eps 形式が最も重複排除されたファイル数が多かったが，deflate 圧縮後では java 形式が最も多くなっている．java 形式の重複排除されたファイル数は Dedup 前と変わっていないが，他のものは全て少なくなっている．
重複排除率でも，java を除いて全体的に下がっている．特に下がっているのは eps 形式(21.8%)と ppt 形式(19.1%)である．これらは，部分重複排除されたファイル数が多い形式であり（表 3），このことから圧縮された画像形式同様に deflate 圧縮を行なったことで，部分重複排除がうまく行われないことが考えられる．

5. 結論

Dedup が有効となる環境には，バックアップ目的のストレージの他に，共通のプログラムを多く使用する研究機関等があげられる．java，c，cpp といったプログラムのソースコードでは，同一のソースコードや，古いバージョンの保存，一部変更したソースコードを複数人が使用するためである．また，プログラムの実行形式である exe ファイルも重複排除が多くできたことが確認できた．Dedup の重複排除率が高い形式には他に，bmp 形式や eps 形式の無圧縮の画像ファイルがあげられる．これらに対する重複排除のデータ削減効率は jpg 圧縮などに及ばないが，画質を落とさず保存できる点で Dedup が優れているといえる．特に eps 形式では，視覚的に似ている画像には重複排除が有効に働くという特徴が見られる．また画像圧縮形式である jpg，gif，ping 形式では，同一のファイルにしか殆ど重複排除されないため Dedup 向きではなく，シングルインスタンスストレージが向いていると考えられる．

データ削減効率を上げるためにデータ圧縮と Dedup を併用することは，どの圧縮技術についてもデータ削減効率を上げることができた．その中でも，deflate 圧縮との併用が最も効果が高かった．これは deflate 圧縮は他の圧縮手法よりも圧縮効果が高いのに，Dedup と併用したとき重複排除率が殆ど下がらないためである．しかし eps 形式や ppt 形式では，圧縮と併用することで Dedup の部分重複排除の効率が大きく減ってしまった．

今後の課題として，ファイル形式に応じて Dedup のブロックの区切り方を変更することがあげられる．例えば，前述の jpg 等の圧縮画像形式に対しては，ブロックの区切りをより小さくすることで，部分的な重複を的確に検知できるようになると考えられる．また，重複排除率が高く部分重複排除率が低かった java 等のソースプログラム形式では，ブロックの区切りを大きくすることで Dedup の処理時間を短くすることが有効と考えられる．圧縮と Dedup の併用については，Dedup と併用していることを前提としたより有効な圧縮手法の検討や，ファイル内で符号化されたデータをファイル間で重複排除する Dedup アルゴリズムの開発を行う必要がある．

参考文献

1)　EMC NEWS 肥大化する DB の最適なデータ管理手法とは？
http://japan.emc.com/microsites/japan/techcommunity/sol/appl/db-ilm.htm
2)　JDSF データ・マネジメント・ソリューション部会 Dedup 講座
http://www.jdsf.gr.jp/de-dupe/04.html
3)　Netapp 重複排除機能
http://www.netapp.com/jp/communities/tech-ontap/tot-back-to-basics-deduplication-1104-ja.html