大数据开发技术

东北林业大学

卢洋

第四章

Hadoop数据压缩

4.2

MR支持的压缩编码

重点！！！！

不可分不适合用在map之前，因为不能进行数据切片

压缩格式	Hadoop自带?	算法	文件扩展名	是否可切分	换成压缩格式后，原来的程序是否需要修改
DEFLATE	是，直接使用	DEFLATE	.deflate	否	和文本处理一样，不需要修改
Gzip	是，直接使用	DEFLATE	.gz	否	和文本处理一样，不需要修改
bzip2	是，直接使用	bzip2	.bz2	是	和文本处理一样，不需要修改
LZO	否，需要安装	LZO	.lzo	是	需要创建索引，还需要指定输入格式
Snappy	否，需要安装	Snappy	.snappy	否	和文本处理一样，不需要修改

为了支持多种压缩/解压算法，Hadoop引入了编码/解码器

压缩格式	对应的编码/解码器
DEFLATE	org.apache.hadoop.io.compress.DefaultCodec
gzip	org.apache.hadoop.io.compress.GzipCodec
bzip2	org.apache.hadoop.io.compress.BZip2Codec
LZO	org.apache.hadoop.io.compress.LzopCodec
Snappy	org.apache.hadoop.io.compress.SnappyCodec

压缩性能的比较

压缩算法	原始文件大小	压缩文件大小	压缩速度	解压速度
gzip	8.3GB	1.8G	17.5MB/s	58MB/s
bzip2	8.3GB	1.1G	2.4MB/s	9.5MB/s
LZO	8.3GB	2.9G	49.3MB/s	74.6MB/s

http://google.github.io/snappy/
https://github.com/google/snappy

Performance

Snappy is intended to be fast. On a single core of a Core i7 processor in 64-bit mode, it compresses at about 250 MB/sec or more and decompresses at about 500 MB/sec or more. (These numbers are for the slowest inputs in our benchmark suite; others are much faster.) In our tests, Snappy usually is faster than algorithms in the same class (e.g. LZO, LZF, QuickLZ, etc.) while achieving comparable compression ratios.

Typical compression ratios (based on the benchmark suite) are about 1.5-1.7x for plain text, about 2-4x for HTML, and of course 1.0x for JPEGs, PNGs and other already-compressed data. Similar numbers for zlib in its fastest mode are 2.6-2.8x, 3-7x and 1.0x, respectively. More sophisticated algorithms are capable of achieving yet higher compression rates, although usually at the expense of speed. Of course, compression ratio will vary significantly with the input.

Although Snappy should be fairly portable, it is primarily optimized for 64-bit x86-compatible processors, and may run slower in other environments. In particular:

- Snappy uses 64-bit operations in several places to process more data at once than would otherwise be possible.
- Snappy assumes unaligned 32 and 64-bit loads and stores are cheap. On some platforms, these must be emulated with single-byte loads and stores, which is much slower.
- Snappy assumes little-endian throughout, and needs to byte-swap data in several places if running on a big-endian platform.

Experience has shown that even heavily tuned code can be improved. Performance optimizations, whether for 64-bit x86 or other platforms, are of course most welcome; see "Contact", below.