BAG 1.4.0 – Compression Proposal

2011-07-07
Webb McDonald, SAIC

Background:

File size of BAGs has been identified as a limitation in import to various software products, file transfer time costs, archival space costs, and portability concerns. The *Open Navigation Surface Working Group* Meeting Summary from 2011-04-27 details actions that were agreed upon to address these concerns.

Investigative Summary:

The first goal was to identify default parameters for the compression scheme in HDF5 that would enable a significant space savings without a costly run-time performance hit. The two variables that control HDF5 compression are the *compression level* (0-9) and the *chunk size*. The chunk size is an “atomic object” on which the compression algorithm is executed. In BAG terms this is a chunk of surface nodes on which all disk I/O will be performed. A *B-Tree* structure in HDF5 maps the chunk addresses to the actual file addresses, so there is a slight file size overhead with the chunk size, but our results show that the file compression savings more than makes up for it.

After testing various combinations of parameters it was determined that a chunk size of `100x100` nodes, with a compression level of `1`, provides the best default performance with considerable file space savings. This same chunk size is referenced in the HDF5 reference manual (Dataset Chunking Issues). Applications may choose to set a different *chunk size* along with *compression level* through the *bagData* variables used to create the BAG file. The underlying principle to this compression scheme is that variability between nearby nodes is typically low for hydrographic data sets. This is especially true for sparse data, and may benefit from *chunk sizes* other than the default. The full results of these tests are in the final section of this report.

The last point is that these compression parameters are merely file creation settings for HDF5, and impose no modifications to any of the BAG API for reading BAGs. BAG software versions v1.3.0 and up will be able to read the compressed BAGs, and new versions of software will be able to read older uncompressed BAGs without a conflict.

Proposed API Changes:

- Add *u8 compressionLevel* to *struct bagData / bagDataOpt,* where compressionLevel is an unsigned integer value ranging between 0 and 9.
- Add u32 *chunkSize* to *struct bagData / bagDataOpt,* to specify cache size of blocks of nodes on which disk I/O and file compression is executed.
- Add a macro for end-user programs, *BAG_DEFAULT_COMPRESSION = 1.*
- *bag_hdf.c* and *bag_opt_surfaces.c* will set the data transfer property list to enable chunking of `100x100` nodes by default or use the bagData *chunk sze* if provided
- *bag_hdf.c* and *bag_opt_surfaces.c* will set deflate parameter equal to the bagData *compression level* if greater than 0. If the dataset has less than 100 rows or columns the chunking will reduce to `10x10`. Less than 10 will ignore compression.
- The *bagcreate* example program will be updated to demonstrate setting of the bagData compression to *BAG_DEFAULT_COMPRESSION* before execution of bagFileCreate().

Test Metrics:

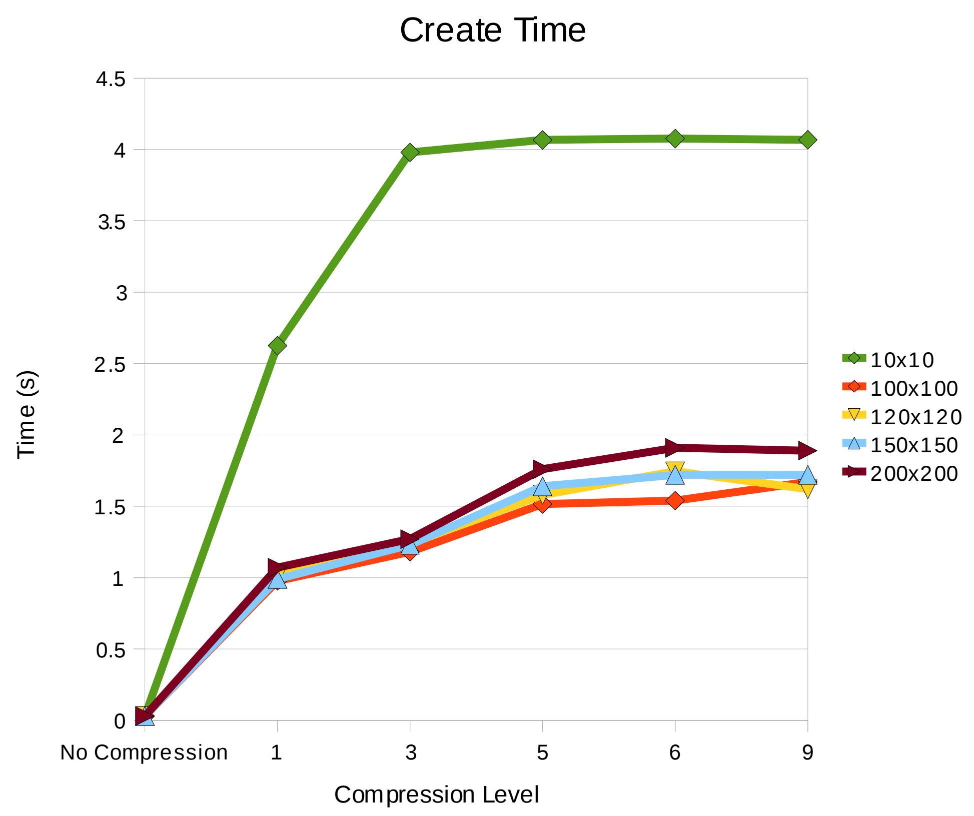
In the following tables, five trials were run for each data point with a `1200x1200` grid of homogenous test data. Each test was run with a fully populated and repeated with a sparse data set. Space savings is shown to be more profound at higher compression levels in more uniform hydrographic data. However, the main purpose of this generic test was to understand the penalties to creation and access time. An upper limit of *chunk size* was found above `200x200`. Data points were impractical to chart above this limit, as the time to complete was in the minutes, rather than tenths of a second below that threshold.

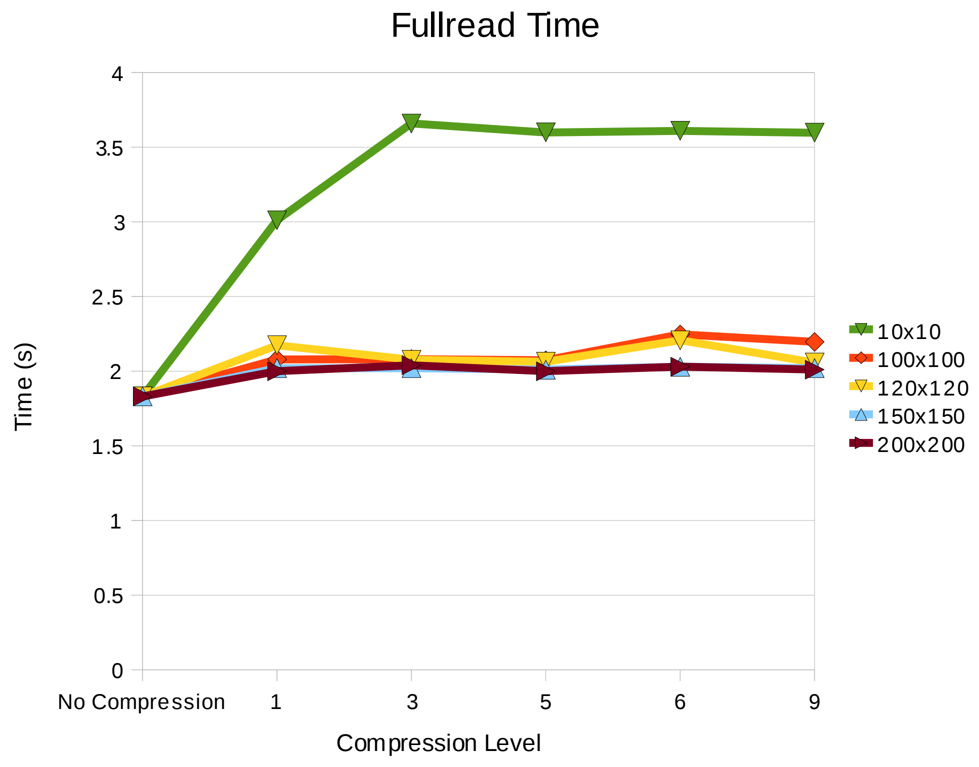
“Create time” was the time to write sample data *true_n_nominal.xml* with *bagcreate.* “Full-read time” was the time to read *true_n_nominal.bag* with *bagread* example program modified to access every node. “File size” and “File compression” were charted to demonstrate the amount of savings across the different compression parameters.

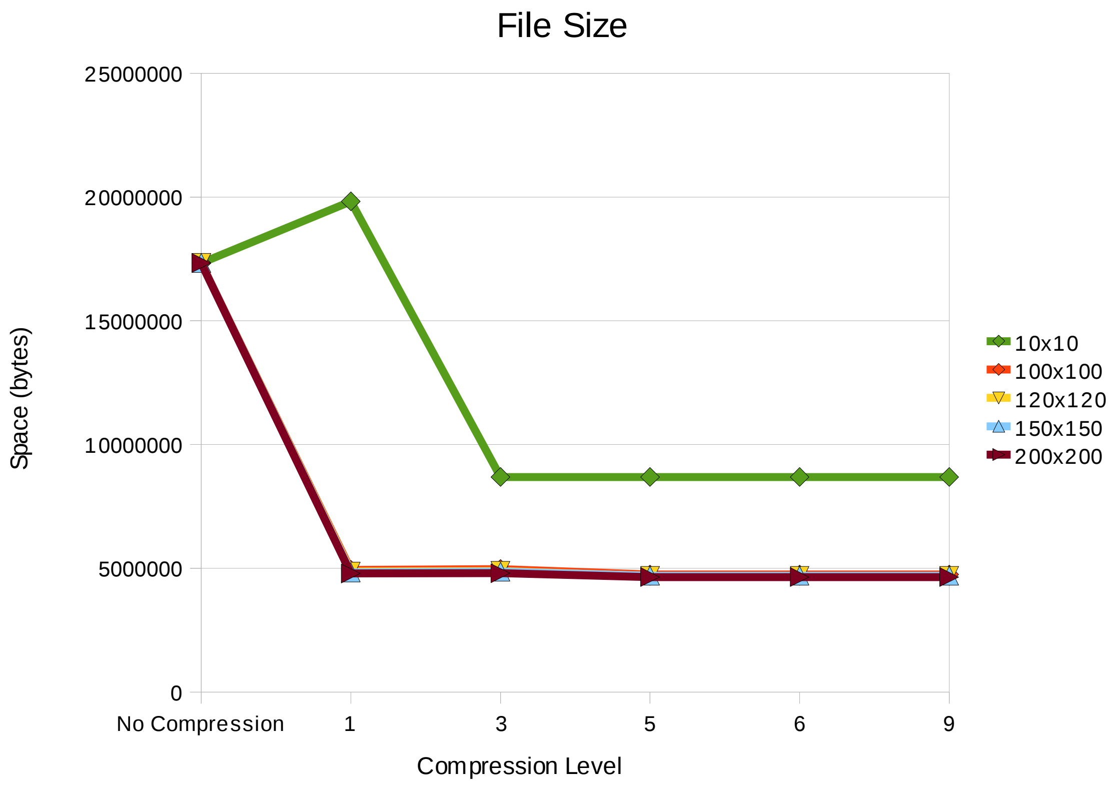
Fully Populated 1200x1200 Grid

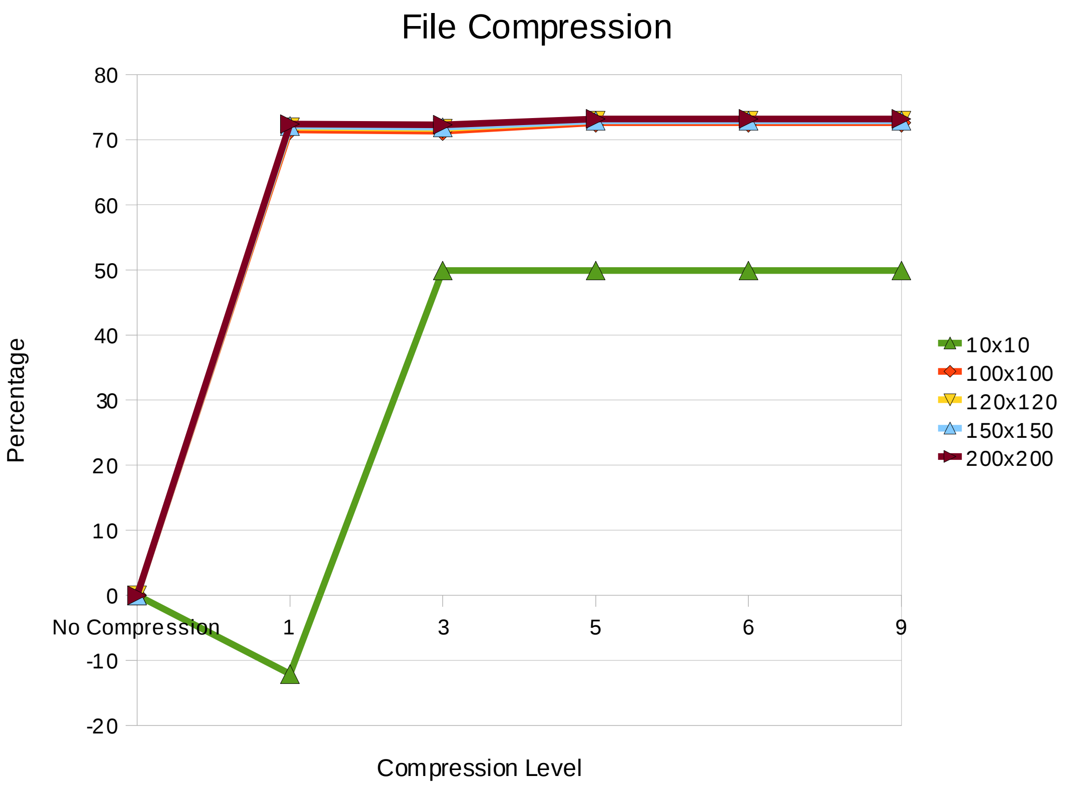
100% of grid populated

CompressLvl	ChunkSize	Create time (s)	Fullread time (s)	%	Space (bytes)
0	0	0.03	1.83	0	17336123
1	10	2.63	3.01	-12.1	19829195
3	10	3.98	3.66	49.9	8691486
5	10	4.07	3.6	49.9	8691512
6	10	4.08	3.61	49.9	8691512
9	10	4.07	3.6	49.9	8691511
0	0	0.03	1.83	0	17336123
1	100	0.98	2.08	71.5	4938770
3	100	1.18	2.08	71.3	4973018
5	100	1.52	2.07	72.6	4749939
6	100	1.54	2.25	72.6	4749953
9	100	1.67	2.2	72.6	4749955
0	0	0.03	1.83	0	17336123
1	120	1.02	2.17	71.9	4872451
3	120	1.24	2.08	71.7	4900325
5	120	1.58	2.06	72.9	4690541
6	120	1.75	2.21	72.9	4690558
9	120	1.62	2.06	72.9	4690558
0	0	0.03	1.83	0	17336123
1	150	0.99	2.02	72.1	4838680
3	150	1.23	2.02	71.9	4870204
5	150	1.64	2.01	72.9	4704985
6	150	1.72	2.03	72.9	4705015
9	150	1.72	2.02	72.9	4705015
0	0	0.03	1.83	0	17336123
1	200	1.07	2	72.4	4791325
3	200	1.27	2.04	72.3	4804810
5	200	1.76	2	73.2	4649062
6	200	1.91	2.03	73.2	4649125
9	200	1.89	2.01	73.2	4649125


Create Time
Time (s)
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
No Compression
1
3
5
6
9
Compression Level
10x10
100x100
120x120
150x150
200x200


Fullread Time
Time (s)
4
3.5
3
2.5
2
1.5
1
0.5
0
No Compression
1
3
5
6
9
Compression Level
10x10
100x100
120x120
150x150
200x200

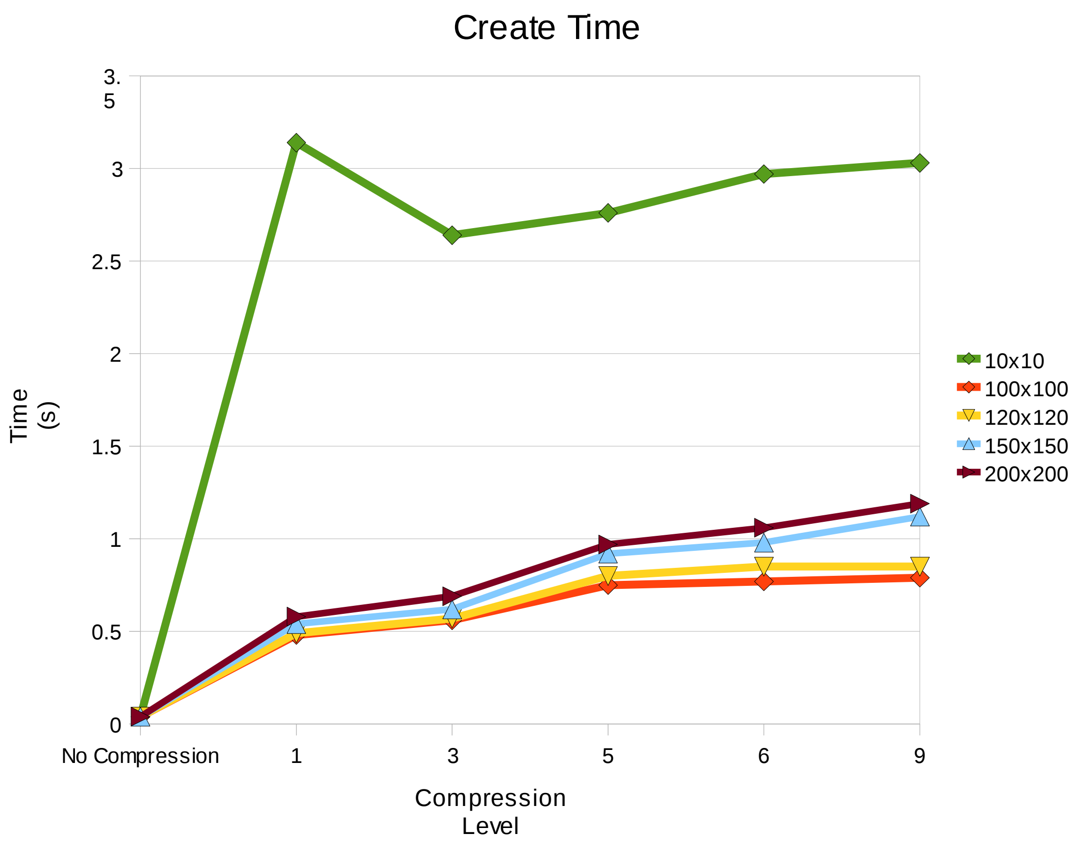

File Size
Space (bytes)
25000000
20000000
15000000
10000000
5000000
0
No Compression
1
3
5
6
9
Compression Level
10x10
100x100
120x120
150x150
200x200

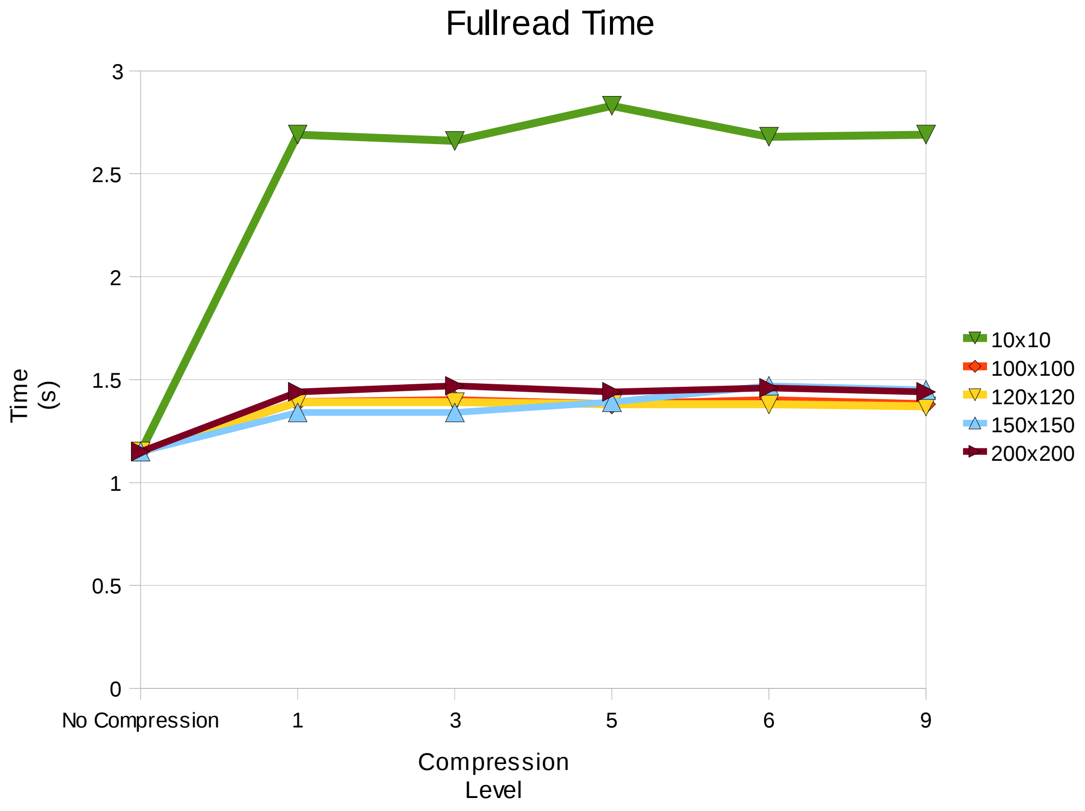

File Compression
Percentage
80
70
60
50
40
30
20
10
0
-10
-20
No Compression
1
3
5
6
9
Compression Level
10x10
100x100
120x120
150x150
200x200

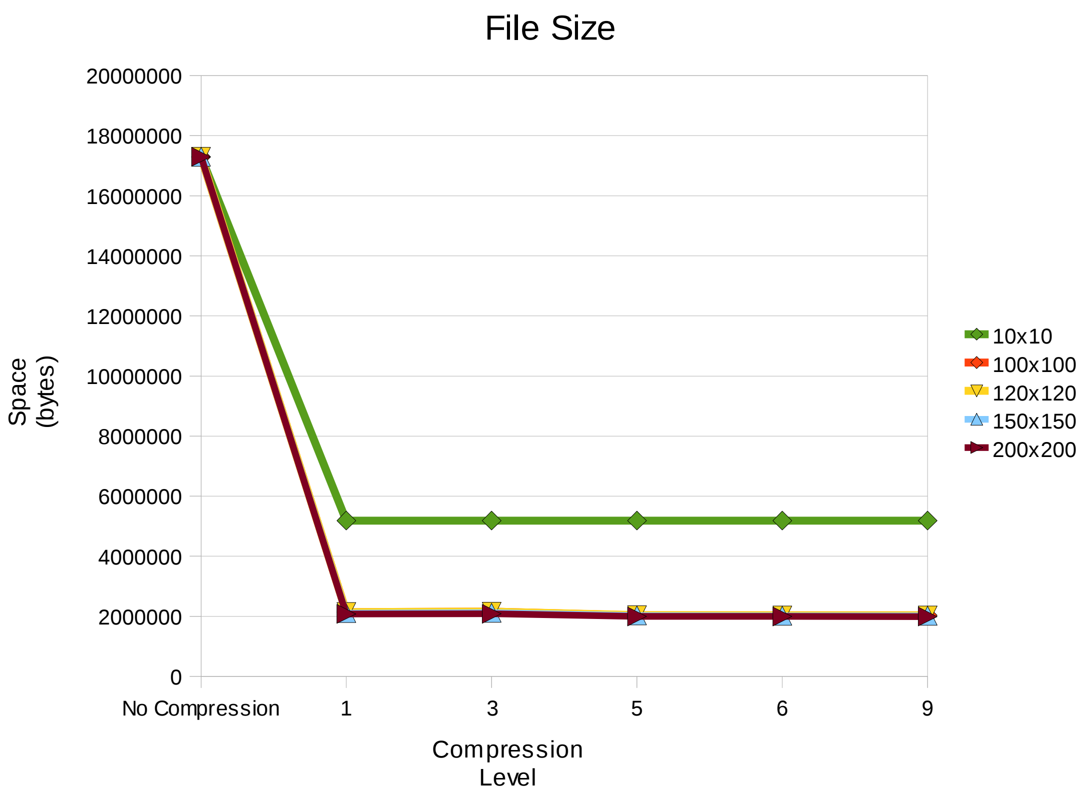
Sparse 1200x1200 Grid

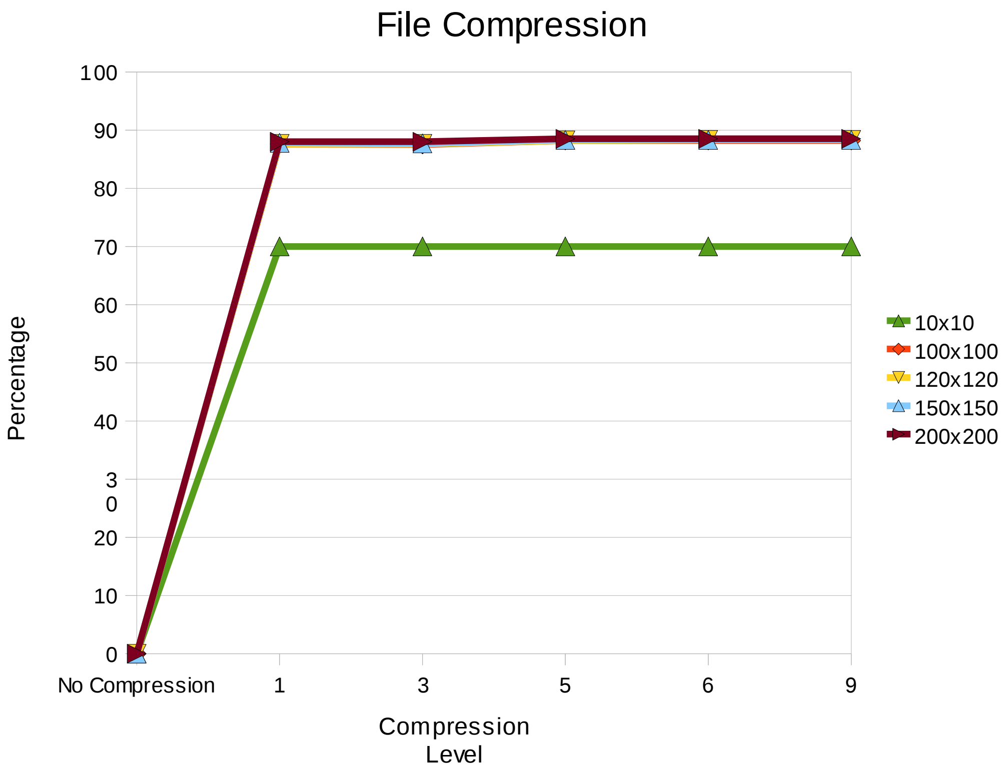
42% of grid populated

CompressLvl	ChunkSize	Create time (s)	Fullread time (s)	%	Space (bytes)
0	0	0.04	1.15	0	17295576
1	10	3.14	2.69	70	5186107
3	10	2.64	2.66	70	5186554
5	10	2.76	2.83	70	5188580
6	10	2.97	2.68	70	5188580
9	10	3.03	2.69	70	5188580
0	0	0.04	1.15	0	17295576
1	100	0.48	1.39	87.6	2141002
3	100	0.56	1.4	87.5	2154937
5	100	0.75	1.38	88.2	2033161
6	100	0.77	1.4	88.2	2033083
9	100	0.79	1.38	88.2	2033131
0	0	0.04	1.15	0	17295576
1	120	0.49	1.39	87.6	2140957
3	120	0.57	1.39	87.6	2153926
5	120	0.8	1.38	88.2	2035954
6	120	0.85	1.38	88.3	2030569
9	120	0.85	1.37	88.3	2030569
0	0	0.04	1.15	0	17295576
1	150	0.54	1.34	87.8	2106667
3	150	0.62	1.34	87.7	2121262
5	150	0.92	1.39	88.3	2025391
6	150	0.98	1.47	88.3	2016190
9	150	1.12	1.45	88.3	2016190
0	0	0.04	1.15	0	17295576
1	200	0.58	1.44	88	2078032
3	200	0.69	1.47	88	2083075
5	200	0.97	1.44	88.5	1993891
6	200	1.06	1.46	88.5	1993999
9	200	1.19	1.44	88.5	1990516


Create Time
Time (s)
3.5
3
2.5
2
1.5
1
0.5
0
No Compression
1
3
5
6
9
Compression Level
10x10
100x100
120x120
150x150
200x200


Fullread Time
Time (s)
3
2.5
2
1.5
1
0.5
0
No Compression
1
3
5
6
9
Compression Level
10x10
100x100
120x120
150x150
200x200


File Size
Space (bytes)
20000000
18000000
16000000
14000000
12000000
10000000
8000000
6000000
4000000
2000000
0
No Compression
1
3
5
6
9
Compression Level
10x10
100x100
120x120
150x150
200x200


File Compression
Percentage
100
90
80
70
60
50
40
3
0
20
10
0
No Compression
1
3
5
6
9
Compression Level
10x10
100x100
120x120
150x150
200x200