Performance Improvements in 2016 release of ColdFusion

- A study in numbers -

Overview

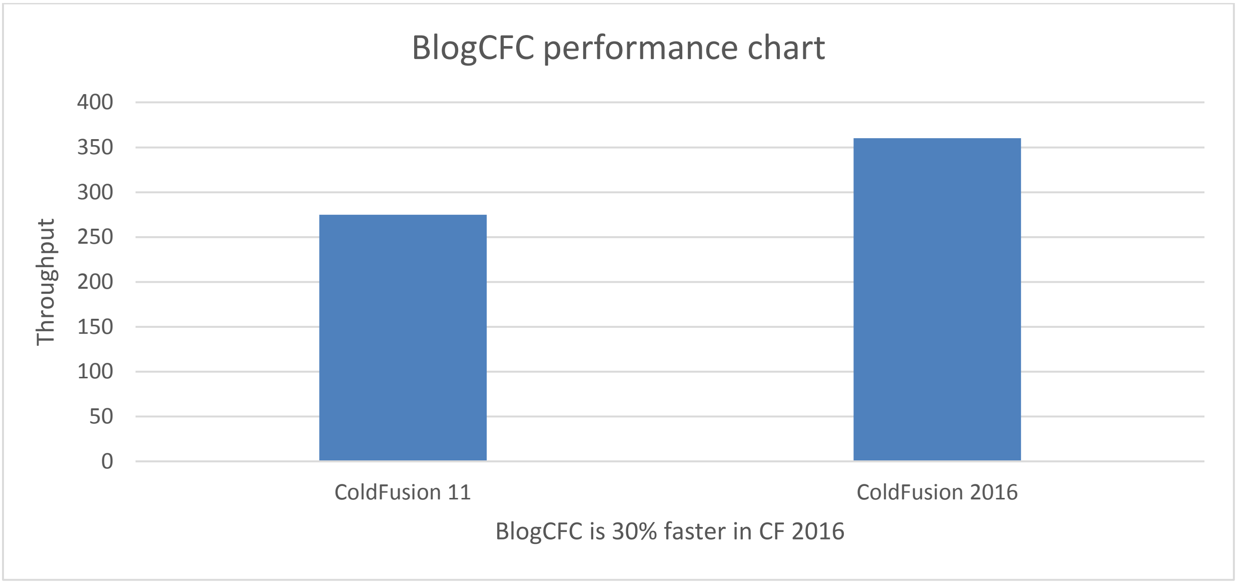
The 2016 release of Adobe ColdFusion with a wide range of performance improvement features, such as, array pass-by-reference, unsynchronized arrays, explicit scope lookup, and so on, to enable developers build high performing applications.

In addition to the new features, many existing features like CFQuery, file functions, and others have been enhanced for significant performance improvement.

Quick snapshot of performance improvements

ColdFusion applications

We chose BlogCFC as sample and upgraded it to the 2016 release of ColdFusion to measure the performance. We found that BlogCFC was 30% faster in the 2016 release.



Statistical highlights

Overall server performance

Application	Improvement over ColdFusion 11
BlogCFC	30%
ColdBox	5%

Overall feature performance

Feature	Improvement over ColdFusion 11
Query	822%
File Functions	67%
List Functions	66%
Whitespace management	25%
CFLoop	22%
Argument Validation	5%
***Arrays pass-by-reference**	2387%
***Explicit Scope lookup**	427%
***Unsynchronized Arrays**	92.80%

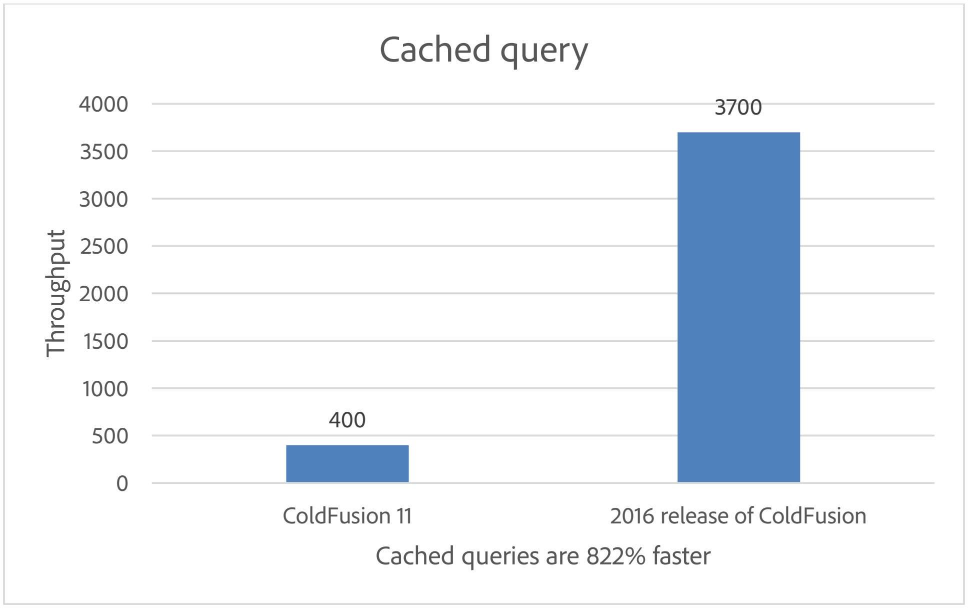
**New feature in the 2016 release*

Performance enhancements in existing features

We have enhanced existing features, such as, file functions, list functions, CFLoop, and Query. Developers or end-users will see these improvements when they upgrade to the 2016 release of ColdFusion, without changing the source code.

Cached Query

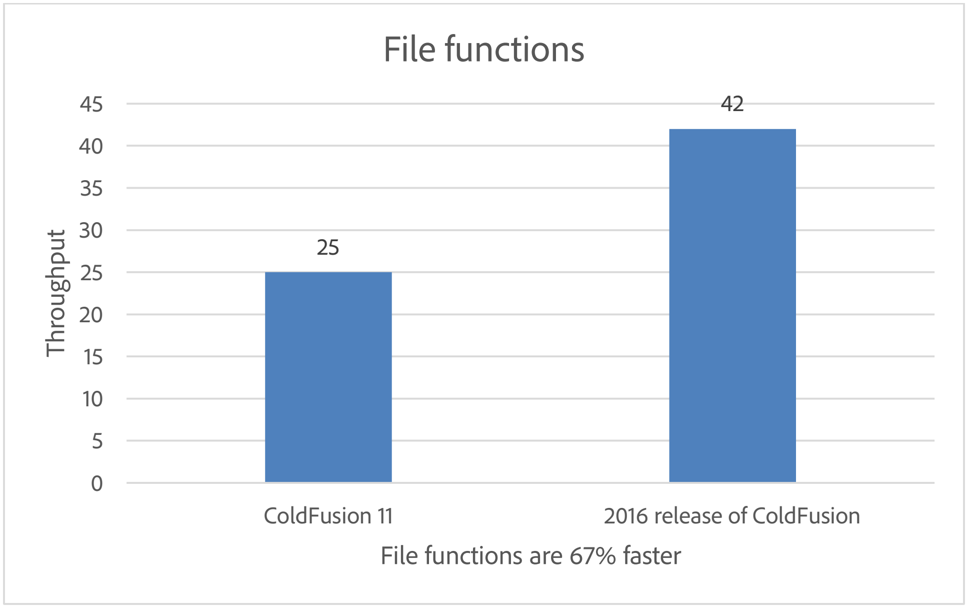
We have significantly improved the performance of cached queries with efficient handling of locking and by introducing immutability. When compared to ColdFusion 11, we found a performance improvement of 822%.



File functions

Our tests show that file functions in the 2016 release of ColdFusion are about 67% faster. We performed tests on the following functions:

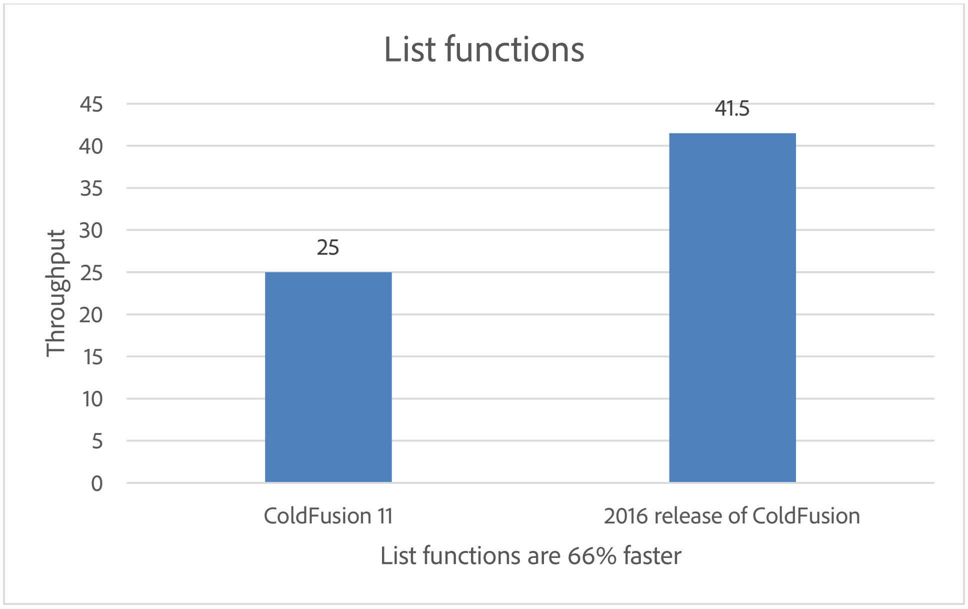
- fileExists
- directoryExists
- getFileFrompath
- getDirectoryFromPath
- getContextRoot
- getCurrentTemplatePath
- getBaseTemplatePath



List functions

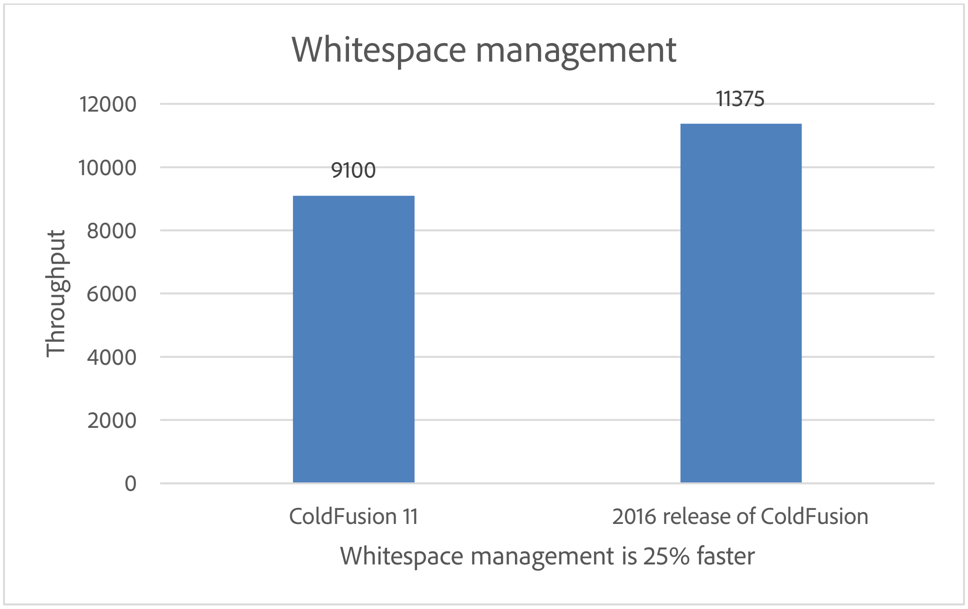
Our tests show that list functions in the 2016 release of ColdFusion show 66% performance improvement. Some of the functions that were part of our test-cases include the following:

- listLen
- listSort
- listFind
- listGetAt
- listDeleteAt
- listAppend
- listToArray
- arrayToList



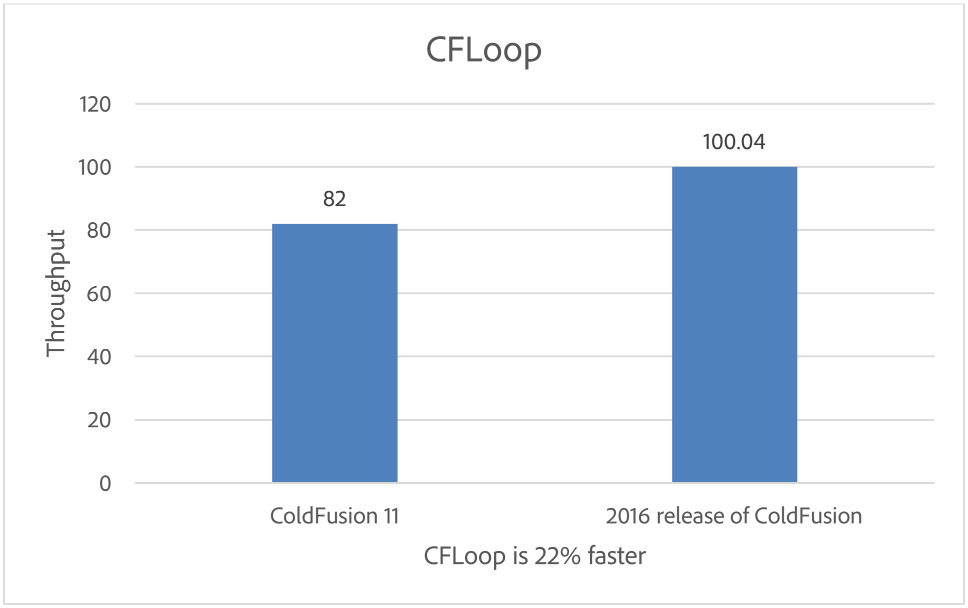
Whitespace management

Users can see up to 25% improvement in application performance when they switch on whitespace management in ColdFusion Administrator. This has been achieved by moving some of the whitespace management operations that were earlier handled at run time to compile time.



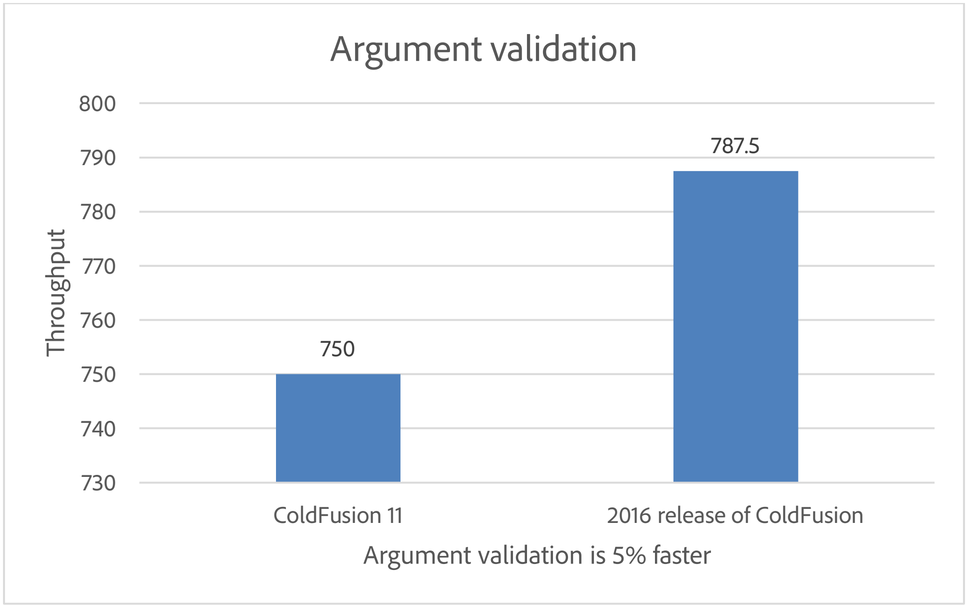
CFloop performance

Users can see up to 22% in performance improvement when they use CFloop in the 2016 release of ColdFusion. We have performed a wide range of optimizations to make the tag execute faster.



Argument validation

We have moved argument validation that includes validation of attribute values of all CF tags to compile time wherever possible. As a result, argument validation is now 5% faster with the 2016 release.



Performance enhancements in new features

We have introduced significant performance enhancements in new features in ColdFusion. Developers or end-users can leverage the enhancements with minimal code changes.

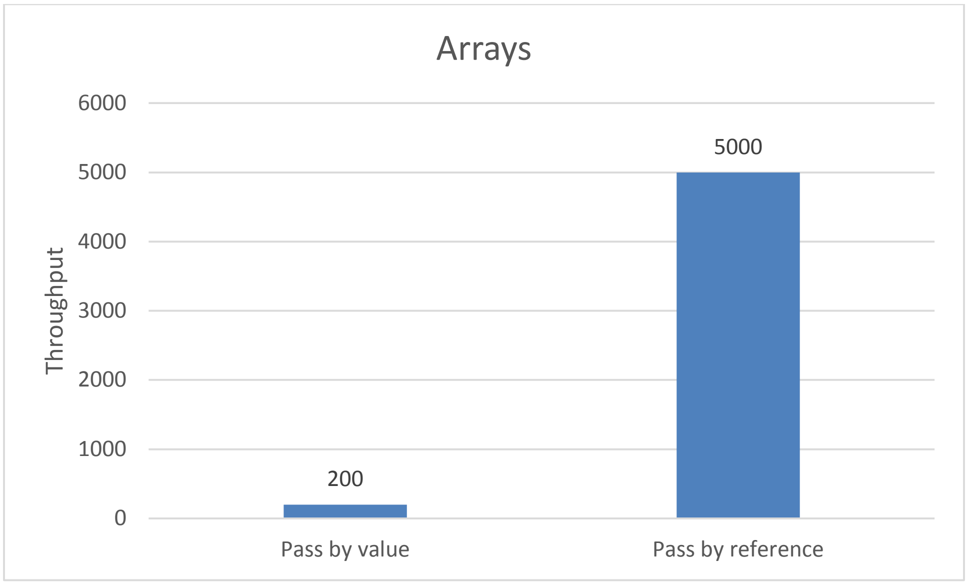
Pass arrays by reference

In ColdFusion, by default, arrays are passed by value. In the 2016 release, users can pass arrays by reference, which greatly improves the performance of UDFs.

You can control this behavior via the application level setting:

this.passArraybyReference=true/false; (false by default)

Setting this to true can improve the performance up to 25 times.



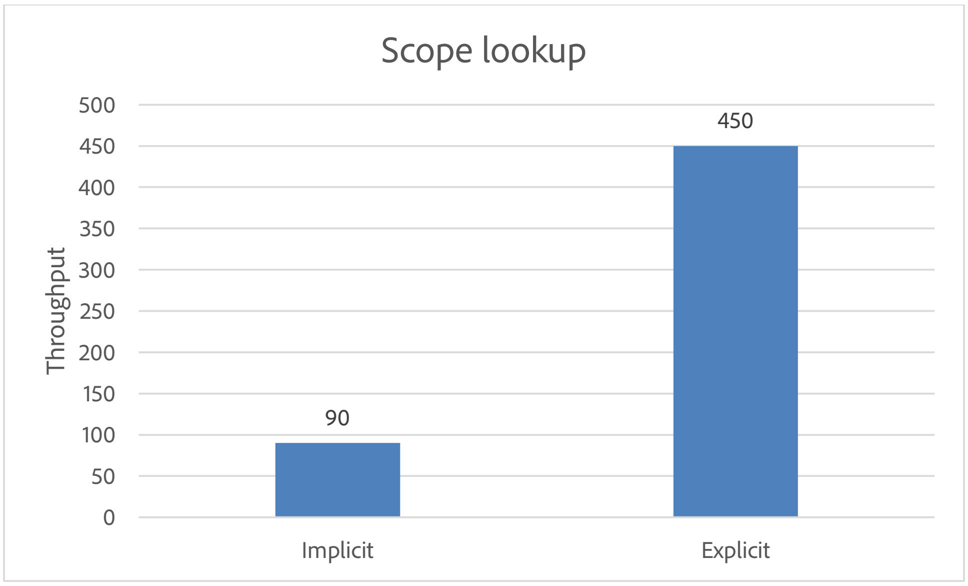
Scope lookup

By default, ColdFusion for an unscoped variable in all the implicit scopes (CGI, URL, FORM, Cookie, File, Client scopes, and so on). This is an expensive operation. Now developers can disable this implicit scope lookup programmatically and improve performance significantly.

This can be controlled with the application level setting:

searchImplicitScopes = true/false. (false by default)

Setting this to *true* can improve the performance up to 5 times.

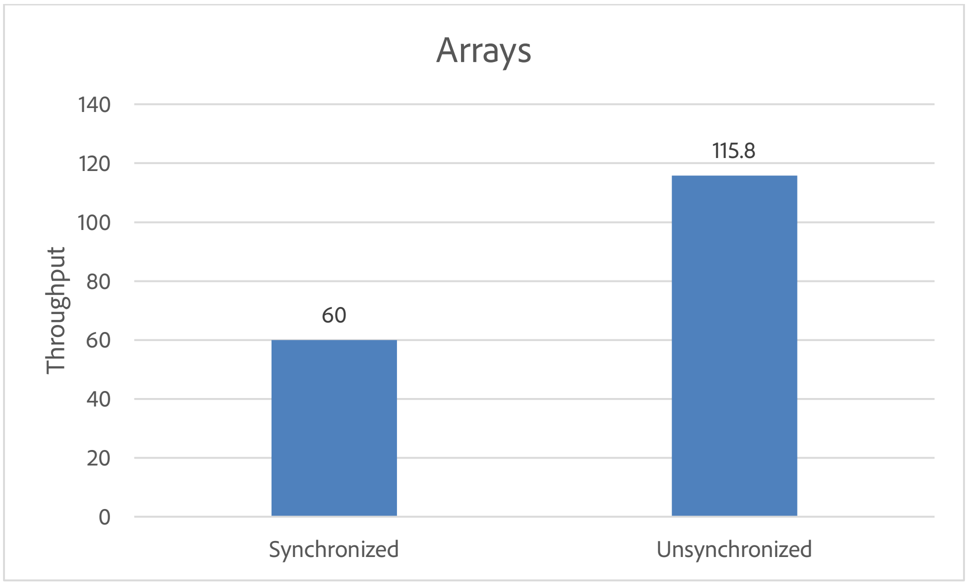


Unsynchronized arrays

The 2016 release of ColdFusion comes with support for unsynchronized arrays. Users can create unsynchronized arrays by passing a special flag, ***isSynchronized***, to the *arrayNew* method. By default, the arrays are synchronized for backward compatibility. Set the flag to ***false*** to create an unsynchronized array. Unsynchronized arrays are about 93% faster due to lock avoidance. Please note that they are not thread safe. You should use them when you do not need thread safety.

Sample code to create an unsynchronized array:

ar_obj = arrayNew(numeric dimension, boolean isSynchronized);



Appendix

Testing configuration and tools	JVM settings	Server settings
• Dell PowerEdge R720 server • Windows Server 2012 R2 • Xeon E5-2643 (4 cores) @ 3.3 GHz – 2 CPU sockets • RAM: 32 GB • Client: distributed Apache jMeter-2.13. 2 clients managed by one server • 500 concurrent users	• Java version 1.8.0_25 64-bit • Xms 1024m • Xmx 1024m • Maxmetaspace 192m • useParallelGC	• Max no. of simultaneous templates: 100 • WhiteSpace Management enabled • Caching disabled • RDS disabled • Line debugging disabled