Saxon - Bug #2103

High memory usage running with a MessageListener

2014-07-11 17:41 - Bob Williams

Status:	Closed	**Start date:**	2014-07-11
Priority:	High	**Due date:**	
Assignee:	Michael Kay	**% Done:**	100%
Category:	Performance	**Estimated time:**	0:00 hour
Sprint/Milestone:		**Spent time:**	0:00 hour
Legacy ID:		**Fixed in Maintenance Release:**	
Applies to branch:		**Platforms:**	
Fix Committed on Branch:			

Description

When running a transformation using Saxonica 9.4.0.6 we run into very high memory usage. Over 5.5 GB RAM for the attached input transformation. We are using .Net api and have code like this to run the transformation. We send it to a memory stream. The actual size of the input xml file is 62 MB and the transformed file it about 198 MB.

var builder = _processor.NewDocumentBuilder();

```
            var input = builder.Build(inputXml);

            var destination = new Serializer();


            var transformer = _executable.Load();
```

transformer.InputXmlResolver = _resolver;

```
            transformer.InitialContextNode = input;
```

using (var stream = new MemoryStream())

```
                {

                    destination.SetOutputStream(stream);

                    transformer.Run(destination);

                    xslTransformationResult.Transformation =

                        CreateTransformationDocument(stream);

                 }
```

Is there anything we can do on the Saxonica API to make the memory usage more efficient?

History

#1 - 2014-07-11 18:15 - Michael Kay

Thanks for reporting it. We'll do some profiling to see if we can identify what it is that's using all this memory.

#2 - 2014-07-11 21:18 - Michael Kay

- Status changed from New to In Progress

- Assignee changed from Bob Williams to Michael Kay

There seems to be a missing file code-files.xml needed to run the transformation. Could you supply it please?

#3 - 2014-07-11 22:12 - Bob Williams

- File code-files.zip added

#4 - 2014-07-11 22:13 - Bob Williams

Michael Kay wrote:

> There seems to be a missing file code-files.xml needed to run the transformation. Could you supply it please?

I've submitted the files.

#5 - 2014-07-11 23:10 - Michael Kay

OK, thanks, it now runs. On the Java platform with 9.5, from the command line, it runs successfully using 383Mb of heap (taking 6 minutes), which is entirely reasonable. I'll try it on 9.4 to see if that makes a difference, and if it works there then I'll try it on .NET when I'm back in the office with access to a Windows machine.

I had Java CPU profiling switched on for this run and it didn't reveal any obvious anomalies.

You raised questions about the API calls you are making: does that mean it runs successfully for you from the command line? I can't see anything obviously wrong with the API calls, but obviously using a MemoryStream for a 198Mb output stream is going to add to the memory requirements.

#6 - 2014-07-11 23:24 - Michael Kay

I get similar data for Saxon-EE 9.4.0.9 (Java platform):

Execution time: 5m 1.26s (301260ms)

Memory used: 262721008

(probably faster because Java profiling wasn't on for this run)

#7 - 2014-07-13 03:07 - Bob Williams

We found that attaching the message listener seemed to cause the memory to swell. Some of transformations render a lot of messages (over 100k) and I guess there is an object created for every such message that isn't released from memory until after transformation is done.

Any way to optimize this through Saxonica or do we need to be more mindful of this in our transformations?

#8 - 2014-07-13 12:40 - Michael Kay

OK, thanks, that's probably the explanation -- or at least, part of it. In the .net Saxon.Api, if you supply an IMessageListener, the Saxon.API code creates a MessageListenerProxy which bridges the Java MessageListener interface to the C# IMessageListener interface. The MessageListenerProxy extends SequenceWriter, which builds a TinyTree instance for each message. This is potentially large, even if the message is small, because Saxon uses a learning algorithm to decide how much space to allocate, based on the size of previous trees. The code for the xsl:message instruction does not appear to call close() on the SequenceWriter, which means that TinyTree.condense() will not be called to reclaim unused space on the tree.

However, I'm not sure this is the whole explanation. I can't see anything in Saxon that would cause messages to be retained in memory after they are passed to the application-supplied IMessageListener, and I don't think the memory used for a single message would account for the amount of heap usage you are seeing. So it may be worth checking the logic of your MessageListener to see if there is anything that causes old messages to be retained.

#9 - 2014-07-13 14:18 - Michael Kay

I've stepped my way through a Java test case which uses a similar MessageListener, and I've found that it is indeed accumulating all the messages in a single TinyTree structure. There's an "optimization" that deliberately tries to append lots of small documents to a single TinyTree rather than creating a new TinyTree for each document. There are cases where this gives a valuable saving because the space and time overhead for creating a new TinyTree for each small document. It's supposed to limit this to something like 50 documents per tree to avoid excessive memory use; I need to explore whether this is happening or whether it grows unlimited.

#10 - 2014-07-13 17:04 - Michael Kay

- Tracker changed from Support to Bug

- Found in version set to 9.5

I think the simplest fix for this is for the MessageListenerProxy (both Java and .NET versions) to build the message XML using the LinkedTree model rather than the TinyTree model. Although the TinyTree uses less space for large documents, the space allocation is more dynamic and incremental for the LinkedTree, which means that for small documents it is probably better -- and the output of xsl:message will almost invariaby be small. I will make this change on the 9.5 and 9.6 branches. We have no plans to issue any further maintenance releases for 9.4.

#11 - 2014-10-31 10:33 - O'Neil Delpratt

- Status changed from In Progress to Resolved

Bug fix committed to subversion.

#12 - 2014-10-31 17:26 - O'Neil Delpratt

- Status changed from Resolved to Closed

- % Done changed from 0 to 100

- Fixed in version set to 9.5.1.8

bug fix applied to the Saxon 9.5.1.8 maintenance release

#13 - 2014-11-01 00:11 - Michael Kay

- Subject changed from Memory usage is very high running a transformation to High memory usage running with a MessageListener

Files

FileForTransform.zip	11.2 MB	2014-07-11	Bob Williams
code-files.zip	8.6 KB	2014-07-11	Bob Williams