Algorithmic complexity of XML parsers/validators

Asked 16 years, 3 months ago Modified 5 years, 5 months ago

Viewed 2k times

15

I need to know how the performance of different XML tools (parsers, validators, XPath expression evaluators, etc) is affected by the size and complexity of the input document. Are there resources out there that document how CPU time and memory usage are affected by... well, what? Document size in bytes? Number of nodes? And is the relationship linear, polynomial, or worse?

Update

In an article in IEEE Computer Magazine, vol 41 nr 9, sept 2008, the authors survey four popular XML parsing models (DOM, SAX, StAX and VTD). They run some very basic performance tests which show that a DOM-parser will have its throughput halved when the input file's size is increased from 1-15 KB to 1-15 MB, or about 1000x larger. The throughput of the other models is not significantly affected.

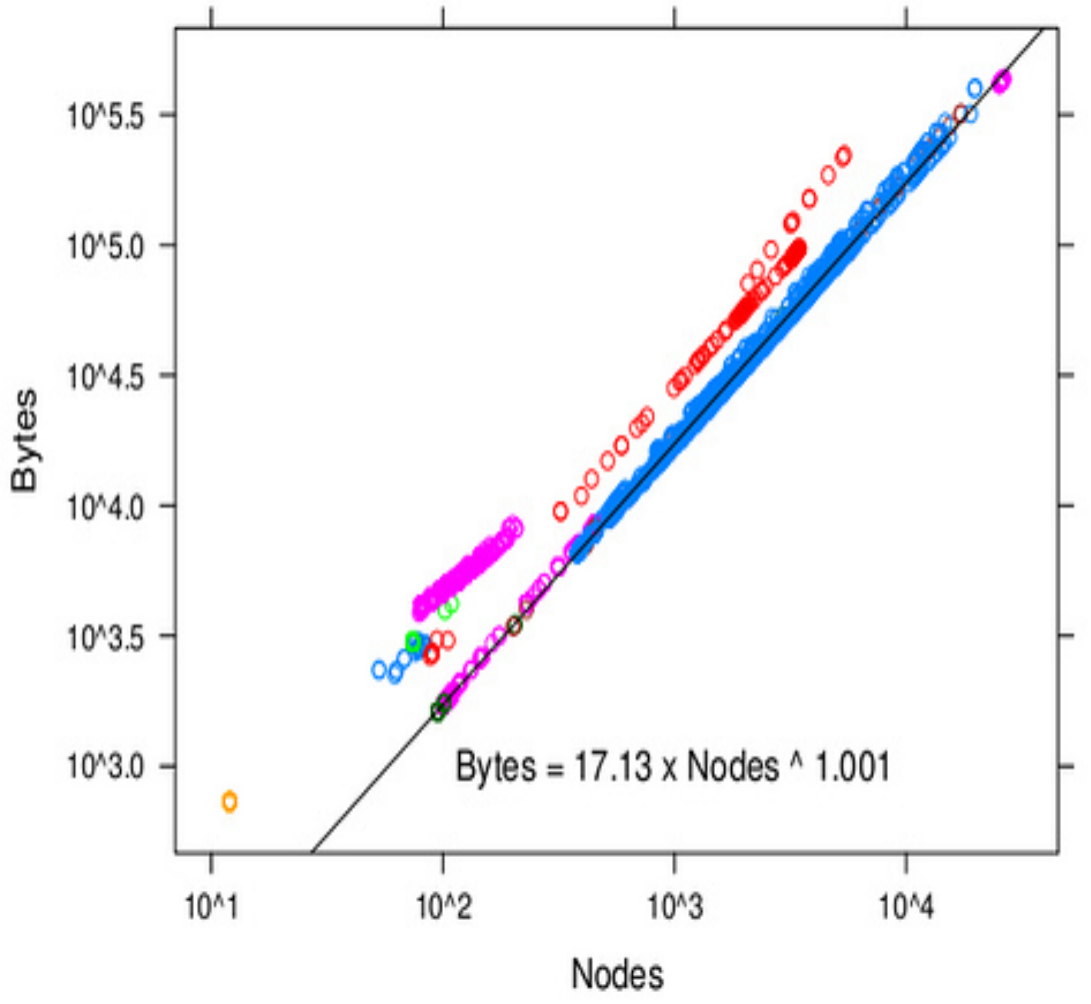
Unfortunately they did not perform more detailed studies, such as of throughput/memory usage as a function of number of nodes/size.

The article is here.

Update

I was unable to find any formal treatment of this problem. For what it's worth, I have done some experiments measuring the number of nodes in an XML document as a function of the document's size in bytes. I'm working on a warehouse management system and the XML documents are typical warehouse documents, e.g. advanced shipping notice etc.

The graph below shows the relationship between the size in bytes and the number of nodes (which should be proportional to the document's memory footprint under a DOM model). The different colors correspond to different kinds of documents. The scale is log/log. The black line is the best fit to the blue points. It's interesting to note that for all kinds of documents, the relationship between byte size and node size is linear, but that the coefficient of proportionality can be very different.



(source: flickr.com)

xml algorithm performance

Share

Improve this question

Follow

edited Jul 12, 2019 at 22:25

Glorfindel

22.6k ● 13 ● 89 ● 116

asked Aug 28, 2008 at 8:01

lindelof

35.2k ● 31 ● 102 ● 144

uhhh Graphs! Always nice. Good updates! – svrist Jan 27, 2009 at 18:57

4 Answers

Sorted by: Highest score (default)

3

If I was faced with that problem and couldn't find anything on google I would probably try to do it my self.

Some "back-of-an-evelope" stuff to get a feel for where it is going. But it would kinda need me to have an idea of how to do a xml parser. For non algorithmical benchmarks take a look here:

- http://www.xml.com/pub/a/Benchmark/exec.html
- http://www.devx.com/xml/Article/16922
- http://xerces.apache.org/xerces2-j/faq-performance.html

Share Improve this answer

Follow

answered Aug 28, 2008 at 8:21

svrist

7,110 ● 7 ● 46 ● 67

1

I think there are too many variables involved to come up with a simple complexity metric unless you make a lot of assumptions.

A simple SAX style parser should be linear in terms of document size and flat for memory.

Something like XPath would be impossible to describe in terms of just the input document since the complexity of the XPath expression plays a huge role.

Likewise for schema validation, a large but simple schema may well be linear, whereas a smaller schema that has a much more complex structure would show worse runtime performance.

As with most performance questions the only way to get accurate answers is to measure it and see what happens!

Share Improve this answer

Follow

answered Aug 29, 2008 at 18:54

Rob Walker
47.4k ● 15 ● 100 ● 137

1

Rob Walker is right: the problem isn't specified in enough detail. Considering just parsers (and ignoring the question of whether they perform validation), there are two main flavors: tree-based—think DOM—and streaming/event-based—think SAX (push) and StAX (pull). Speaking in huge generalities, the tree-based approaches consume more memory and are slower (because you need to finish parsing the whole document), while the streaming/event-based approaches consume less memory and are faster. Tree-based parsers are generally considered easier to use, although StAX has been heralded as a huge improvement (in ease-of-use) over SAX.

answered Sep 3, 2008 at 11:47

Share Improve this answer

Follow

Hank Gay

71.8k ● 36 ● 161 ● 222

0

I was planning to load extremely large XML files in my application. I asked the question here on Stack Overflow: Fastest Possible XML handling for very large documents.

And yes, it was the parsing part, that was the bottleneck.

I ended up not using XML parsers at all. Instead, I parsed characters one by one as efficiently as possible optimizing for speed. This resulted in speeds of 40 MB per second on a 3 GHz Windows PC for the reading, parsing and loading of the internal data structure.

I would be very interested in hearing how the various XML parsing modes compare to this.

Share Improve this answer

Follow

edited May 23, 2017 at 10:29

Community Bot

1 ● 1

answered Jan 12, 2009 at 17:44

lkessler

20.1k ● 36 ● 137 ● 208