http://mail-archives.apache.org/mod\_mbox/lucene-dev/200110.mbox/%3c3BD954E2.3090902@earthlink.net%3e

From lucene-dev-return-286-apmail-jakarta-lucene-dev-archive=jakarta.apache.org@jakarta.apache.org Sat Oct 27 20:26:23 2001

Return-Path: <lucene-dev-return-286-apmail-jakarta-lucene-dev-archive=jakarta.apache.org@jakarta.apache.org>

Delivered-To: apmail-jakarta-lucene-dev-archive@jakarta.apache.org

Received: (qmail 90602 invoked by uid 500); 27 Oct 2001 20:26:23 -0000

Mailing-List: contact lucene-dev-help@jakarta.apache.org; run by ezmlm

Precedence: bulk

Reply-To: lucene-dev@jakarta.apache.org

list-help: <mailto:lucene-dev-help@jakarta.apache.org>

list-unsubscribe: <mailto:lucene-dev-unsubscribe@jakarta.apache.org>

list-post: <mailto:lucene-dev@jakarta.apache.org>

Delivered-To: mailing list lucene-dev@jakarta.apache.org

Received: (qmail 90533 invoked from network); 27 Oct 2001 20:26:23 -0000

Message-ID: <3BD954E2.3090902@earthlink.net>

Date: Fri, 26 Oct 2001 06:19:46 -0600

From: Dmitry Serebrennikov <dmitrys@earthlink.net>

User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:0.9.2) Gecko/20010726 Netscape6/6.1

X-Accept-Language: en-us

MIME-Version: 1.0

To: lucene-dev@jakarta.apache.org

Subject: Performance in TermInfosReader

Content-Type: text/plain; charset=us-ascii; format=flowed

Content-Transfer-Encoding: 7bit

X-Spam-Rating: daedalus.apache.org 1.6.2 0/1000/N

Greetings,

I'm doing some stress testing and optimization for out application for

high concurrency rates and I'm seeing a lot of contention over the

synchronization monitor in TermInfosReader.terms(Term). Our application

tends to do a lot of navigation through the term dictionary to resolve

each user's request. This probably isn't a typical situation for Lucene,

but has anyone seen this?

I'm using OptimizeIt profiler (it's GREAT!) and this is how I know.

Tomorrow I'm going to look at what can be done either on the application

side or in Lucene to ease this contention. Does anyone have any ideas /

suggestions / experience in this area?

More specific info:

The actual operation that the application is performing involves

searching for a large number of terms (100-200) in the dictionary, which

may or may not be there. These terms are sorted by term number (and thus

lexicographically too). At first, I tried to have a single TermEnum and

scroll through it. This turned out to be very slow. Creating new enum

using terms(Term) seems to work better. There were many other

bottlenecks all over the place that I had to clear out and now I'm back

at this same issue.

Doug, what would be an approach for making TermEnums "seekable" in an

efficient manner?

On the term vector support:

I made some substantial changes in order to improve performance. The

interface is now different. It is more like an enum, so that you seek to

a particular document and then access its term vector. Then you move to

another one. This significantly cuts down on needless memory allocation

since no TermVector objects need to be created. If anyone had a chance

to take a look at the code I released previously, feedback would be

welcome! :)

Dmitry