How to configure Solr PostingsFormat block size

< Previous Topic Next Topic >

Classic List Threaded 11 messages Options

Tom Burton-West-2 Reply | Threaded | More

Jan 12, 2015; 3:37pm **How to configure Solr PostingsFormat block size**

341 posts

Hello all,

Our indexes have around 3 billion unique terms, so for Solr 3, we set TermIndexInterval to about 8 times the default. The net effect of this is to reduce the size of the in-memory index by about 1/8th. (For background see for http://www.hathitrust.org/blogs/large-scale-search/too-many-words-again,)

We would like to do something similar for Solr4. T

he Lucene 4.10.2 JavaDoc for setTermIndexInterval suggests how this can be done by setting the minimum and maximum size for a block in Lucene code (http://lucene.apache.org/core/4_10_2/core/org/apache/lucene/index/IndexWriterConfig.html#setTermIndexInterval%28in)
"For example, Lucene41PostingsFormat <http://lucene.apache.org/core/4_10_2/core/org/apache/lucene/codecs/lucene41/Lucene41PostingsFormat.html> implements the term index instead based upon how terms share prefixes. To configure its parameters (the minimum and maximum size for a block), you would instead use Lucene41PostingsFormat.Lucene41PostingsFormat(int, int) <http://lucene.apache.org/core/4_10_2/core/org/apache/lucene/codecs/lucene41/Lucene41PostingsFormat.html#Lucene4 which can also be configured on a per-field basis"

How can we configure Solr to use different (i.e. non-default) mimum and maximum block sizes?

Tom

Michael Sokolov-3 Reply | Threaded | More

Jan 12, 2015; 3:54pm **Re: How to configure Solr PostingsFormat block size**

263 posts

It looks like this is a good starting point:

http://wiki.apache.org/solr/SolrConfigXml#codecFactory

-Mike

On 01/12/2015 03:37 PM, Tom Burton-West wrote:
... [show rest of quote]

Chris Hostetter-3 Reply | Threaded | More

Jan 12, 2015; 4:46pm **Re: How to configure Solr PostingsFormat block size**

6705 posts

: It looks like this is a good starting point:
:
: http://wiki.apache.org/solr/SolrConfigXml#codecFactory

The default "SchemaCodecFactory" already supports defining a diff posting format per fieldType - but there isn't much in solr to let you "tweak" individual options on specific posting formats via configuration.

So what you'd need to do is write a small subclass of Lucene41PostingsFormat that called "super(yourMin, yourMax)" in it's constructor.

: On 01/12/2015 03:37 PM, Tom Burton-West wrote:
: > Hello all,
: >
: > Our indexes have around 3 billion unique terms, so for Solr 3, we set
: > TermIndexInterval to about 8 times the default. The net effect of this is
: > to reduce the size of the in-memory index by about 1/8th. (For background
: > see for
: > http://www.hathitrust.org/blogs/large-scale-search/too-many-words-again,)
: >
: > We would like to do something similar for Solr4. T
: >
: > he Lucene 4.10.2 JavaDoc for setTermIndexInterval suggests how this can be
: > done by setting the minimum and maximum size for a block in Lucene code (
: >
http://lucene.apache.org/core/4_10_2/core/org/apache/lucene/index/IndexWriterConfig.html#setTermIndexInterval%28in
: >)
: > "For example, Lucene41PostingsFormat
: > <http://lucene.apache.org/core/4_10_2/core/org/apache/lucene/codecs/lucene41/Lucene41PostingsFormat.html>
: > implements the term index instead based upon how terms share prefixes. To
: > configure its parameters (the minimum and maximum size for a block), you
: > would instead use Lucene41PostingsFormat.Lucene41PostingsFormat(int, int)
: >
<http://lucene.apache.org/core/4_10_2/core/org/apache/lucene/codecs/lucene41/Lucene41PostingsFormat.html#Lucene4
: > which can also be configured on a per-field basis"
: >
: > How can we configure Solr to use different (i.e. non-default) mimum and
: > maximum block sizes?
: >
: > Tom
: >
:
:

-Hoss
http://www.lucidworks.com/

Tom Burton-West-2 Reply | Threaded | More

Jan 13, 2015; 1:22pm **Re: How to configure Solr PostingsFormat block size**

341 posts

Thanks Michael and Hoss,

assuming I've written the subclass of the postings format, I need to tell
Solr to use it.

Do I just do something like:

<fieldType name="ocr" class="solr.TextField" postingsFormat="MySubclass" />

Is there a way to set this for all fieldtypes or would that require writing
a custom CodecFactory?

Tom

On Mon, Jan 12, 2015 at 4:46 PM, Chris Hostetter <[hidden email]>
wrote:

>
> : It looks like this is a good starting point:
> :
> : http://wiki.apache.org/solr/SolrConfigXml#codecFactory
> >
> > The default "SchemaCodecFactory" already supports defining a diff posting
> > format per fieldType - but there isn't much in solr to let you "tweak"
> > individual options on specific posting formats via configuration.
> >
> > So what you'd need to do is write a small subclass of
> > Lucene41PostingsFormat that called "super(yourMin, yourMax)" in it's
> > constructor.
> >
> >
> >
> >

Chris Hostetter-3 Reply | Threaded | More

Jan 13, 2015; 2:16pm **Re: How to configure Solr PostingsFormat block size**


6705 posts

: assuming I've written the subclass of the postings format, I need to tell
: Solr to use it.
:
: Do I just do something like:
:
: <fieldType name="ocr" class="solr.TextField" postingsFormat="MySubclass" />

the postingFormat xml tag in schema.xml just refers to the "name" of the

postingFormat in SPI -- which is discussed in the PostingFormat
javadocs...

https://lucene.apache.org/core/4_10_0/core/org/apache/lucene/codecs/PostingsFormat.html

...the nuts & bolts of it is that the PostingFormat baseclass should take
care of all the SPI "name" registration that you need based on what you
pass to the super() construction ... allthough now that i think about it,
i'm not sure how you'd go about specifying your own name for the
PostingFormat when also doing something like subclassing
Lucene41PostingsFormat ... there's no Lucene41PostingsFormat constructor
you can call from your subclass to override the name.

not sure what the expectation is there in the java API.

: Is there a way to set this for all fieldtypes or would that require writing
: a custom CodecFactory?

SchemaCodecFactory uses the Lucene default for any fieldType that doesn't
define it's own postingFormat -- so if you wnated to change the
postingFormat or *every* fieldType, then yes: you'd need to override the
CodecFactory itself.

-Hoss
http://www.lucidworks.com/

Chris Hostetter-3 Reply | Threaded | More

Jan 13, 2015; 3:16pm **Re: How to configure Solr PostingsFormat block size**


6705 posts

: ...the nuts & bolts of it is that the PostingFormat baseclass should take
: care of all the SPI "name" registration that you need based on what you
: pass to the super() construction ... allthough now that i think about it,
: i'm not sure how you'd go about specifying your own name for the
: PostingFormat when also doing something like subclassing

: Lucene41PostingsFormat ... there's no Lucene41PostingsFormat constructor
: you can call from your subclass to override the name.
:
: not sure what the expectation is there in the java API.

ok, so i talked this through with mikemccand on IRC...

in 4x, the API is actaully really dangerous - you can subclass things like
Lucene41PostingsFormat w/o overriding the name used in SPI, and might
really screw things up as far as what class is used to read back your
files later.

in the 5.0 APIs, these non-abstract codec related classes are all final to
prevent exactly this type of behavior - but you can still use the
constructor args to change behavior related to *writing* the index, and
the classes all are designed to be smart enough that when they are loaded
by SPI at search time, they can make sense of what's on disk (regardlessof
wether non-default constructor args were used at index time)

but the question remains: where does that leave you as a solr user who
wants to write a plugin, since Solr only allows you to configure the SPI
name (no constructor args) via 'postingFormat="foo"'

the anwser is that instead of writing a subclass, you would have to write
a small proxy class, something like...

```
public final class MyPfWrapper extends PostingFormat {
  PostingFormat pf = new Lucene50PostingsFormat(42, 99999);
  public MyPfWrapper() {
```

```
    super("MyPfWrapper");
  }
  public FieldsConsumer fieldsConsumer(SegmentWriteState state) throws IOException {
    return pf.fieldsConsumer(state);
  }
  public FieldsConsumer fieldsConsumer(SegmentWriteState state) throws IOException {
    return pf.fieldsConsumer(state);
  }
  public FieldsProducer fieldsProducer(SegmentReadState state) throws IOException {
    return pf.fieldsProducer(state);
  }
}
```

..and then refer to it with postingFormat="MyPfWrapper"

at index time, Solr will use SPI to find your "MyPfWrapper" class, which will delegate to an instance of Lucene50PostingsFormat constructed with the overriden constants, and then at query time the SegmentReader code paths will use SPI to find MyPfWrapper by name as well, and it will again delegate to Lucene50PostingsFormat for reading back the index.

or at least: that's how it *should* work :)

-Hoss
http://www.lucidworks.com/

Tom Burton-West-2 Reply | Threaded | More

Jan 13, 2015; 4:55pm **Re: How to configure Solr PostingsFormat block size**

341 posts

Thanks Hoss,

This is starting to sound pretty complicated. Are you saying this is not doable with Solr 4.10?
>>...or at least: that's how it *should* work :) makes me a bit nervous about trying this on my own.

Should I open a JIRA issue or am I probably the only person with a use case for replacing a TermIndexInterval setting with changing the min and max block size on the 41 postings format?

Tom

On Tue, Jan 13, 2015 at 3:16 PM, Chris Hostetter <[hidden email]> wrote:

> : ...the nuts & bolts of it is that the PostingFormat baseclass should take
> : care of all the SPI "name" registration that you need based on what you
> : pass to the super() construction ... allthough now that i think about it,
> : i'm not sure how you'd go about specifying your own name for the
> : PostingFormat when also doing something like subclassing
> : Lucene41PostingsFormat ... there's no Lucene41PostingsFormat constructor
> : you can call from your subclass to override the name.
> :
> : not sure what the expectation is there in the java API.
>
> ok, so i talked this through with mikemccand on IRC...
>
> in 4x, the API is actaully really dangerous - you can subclass things like
> Lucene41PostingsFormat w/o overriding the name used in SPI, and might
> really screw things up as far as what class is used to read back your
> files later
... [show rest of quote]

Chris Hostetter-3 Reply | Threaded | More

Jan 13, 2015; 5:01pm **Re: How to configure Solr PostingsFormat block size**

: This is starting to sound pretty complicated. Are you saying this is not

6705 posts

: doable with Solr 4.10?

it should be doable in 4.10, using a wrapper class like the one i
mentioned below (delegating to Lucene51PostingsFormat instead of
Lucene50PostingsFormat) ... it's just that the 4.10 APIs are dangerous and
let malicious/foolish java devs do scary things they shouldn't do. but
what i outlined before (Below) is intended to work, and should continue to
work in 5.x.

: >>...or at least: that's how it *should* work :) makes me a bit nervous
: about trying this on my own.

...worst case scenerio, i overlooked something - but all it would take to
verify that it's working is to try it at small scale: write the class,
configure it, index a handful of docs, shutdown & restart solr, and see if
your index opens & is correctly searchable -- if it is, then i didn't
overlook anything, if it isn't then there is a bug somewhere and details
of your experiement with your custom posting format (ie wrapper class)
source in JIRA would be helpful.

: Should I open a JIRA issue or am I probably the only person with a use case
: for replacing a TermIndexInterval setting with changing the min and max
: block size on the 41 postings format?

you're the only person i've ever seen ask about it :)

```
: > public final class MyPfWrapper extends PostingFormat {
: >   PostingFormat pf = new Lucene50PostingsFormat(42, 99999);
: >   public MyPfWrapper() {
: >     super("MyPfWrapper");
: >   }
: >   public FieldsConsumer fieldsConsumer(SegmentWriteState state) throws
: > IOException {
: >     return pf.fieldsConsumer(state);
: >   }
: >   public FieldsConsumer fieldsConsumer(SegmentWriteState state) throws
: > IOException {
: >     return pf.fieldsConsumer(state);
: >   }
: >   public FieldsProducer fieldsProducer(SegmentReadState state) throws
: > IOException {
: >     return pf.fieldsProducer(state);
: >   }
: > }
: >
: > ..and then refer to it with postingFormat="MyPfWrapper"
```

-Hoss
http://www.lucidworks.com/

Michael Sokolov-3 Reply | Threaded | More

Jan 14, 2015; 11:50am **Re: How to configure Solr PostingsFormat block size**

263 posts

As a foolish dev (not malicious I hope!), I did mess around with
something like this once; I was writing my own Codec. I found I had to
create a file called META-INF/services/org.apache.lucene.codecs.Codec in
my solr plugin jar that contained the fully-qualified class name of my
codec: I guess this registers it with the SPI framework so it can be
found by name? I'm not clear, but I think you might need to do
something similar to plug in a PostingsFormat as well.

-Mike

On 01/13/2015 05:01 PM, Chris Hostetter wrote:

> : This is starting to sound pretty complicated. Are you saying this is not
> : doable with Solr 4.10?
>
> it should be doable in 4.10, using a wrapper class like the one i
> mentioned below (delegating to Lucene51PostingsFormat instead of
> Lucene50PostingsFormat) ... it's just that the 4.10 APIs are dangerous and
> let malicious/foolish java devs do scary things they shouldn't do. but
> what i outlined before (Below) is intended to work, and should continue to
> work in 5.x.
>
> : >>...or at least: that's how it *should* work :) makes me a bit nervous
> : about trying this on my own.
>
> ...worst case scenerio, i overlooked something - but all it would take to
> verify that it's working is to try it at small scale: write the class,
> configure it, index a handful of docs, shutdown & restart solr, and see if
> your index opens & is correctly searchable -- if it is, then i didn't
... [show rest of quote]

Chris Hostetter-3 Reply | Threaded | More

Jan 14, 2015; 6:05pm **Re: How to configure Solr PostingsFormat block size**


6705 posts

: As a foolish dev (not malicious I hope!), I did mess around with something
: like this once; I was writing my own Codec. I found I had to create a file
: called META-INF/services/org.apache.lucene.codecs.Codec in my solr plugin jar
: that contained the fully-qualified class name of my codec: I guess this
: registers it with the SPI framework so it can be found by name? I'm not

Yep, that's how SPI works - the important bits are mentioned/linked in the
PostingsFormat (and other SPI related classes in lucene) javadocs...

https://lucene.apache.org/core/4_10_2/core/org/apache/lucene/codecs/PostingsFormat.html

https://docs.oracle.com/javase/7/docs/api/java/util/ServiceLoader.html?is-external=true

-Hoss
http://www.lucidworks.com/

Tom Burton-West-2 Reply | Threaded | More

Mar 12, 2015; 1:40pm **Re: How to configure Solr PostingsFormat block size**


341 posts

Hi Hoss,

I created a wrapper class, compiled a jar and included an
org.apache.lucene.codecs.Codec file in META-INF/services in the jar file
with an entry for the wrapper class :HTPostingsFormatWrapper. I created a
collection1/lib directory and put the jar there. (see below)

I'm getting the dread "ClassCastException Class.asSubclass(Unknown Source"
error (See below).

This is looking like a complex classloader issues. Should I put the file
somewhere else and/or declare a lib directory in solrconfig.xml?

Any suggestions on how to troubleshoot this?.

Tom

---

error:
by: java.lang.ClassCastException: class
org.apache.lucene.codecs.HTPostingsFormatWrapper
 at java.lang.Class.asSubclass(Unknown Source)
 at org.apache.lucene.util.SPIClassIterator.next(SPIClassIterator.java:141)

Contents of the jar file:

C:\d\solr\lucene_solr_4_10_2\solr\example\solr\collection1\lib>jar -tvf
HTPostingsFormatWrapper.jar
    25 Thu Mar 12 10:37:04 EDT 2015 META-INF/MANIFEST.MF
  1253 Thu Mar 12 10:37:04 EDT 2015
org/apache/lucene/codecs/HTPostingsFormatWrapper.class
  1276 Thu Mar 12 10:49:06 EDT 2015
META-INF/services/org.apache.lucene.codecs.Codec

Contents of META-INF/services/org.apache.lucene.codecs.Codec in the jar
file:
org.apache.lucene.codecs.lucene49.Lucene49Codec
org.apache.lucene.codecs.lucene410.Lucene410Codec
tbw adds custom wrapper here per Hoss e-mail
org.apache.lucene.codecs.HTPostingsFormatWrapper

---
log file excerpt with stack trace:

12821 [main] INFO org.apache.solr.core.CoresLocator – Looking for core
definitions underneath C:\d\solr\lucene_solr_4_10_2\solr\example\solr
12838 [main] INFO org.apache.solr.core.CoresLocator – Found core

collection1 in C:\d\solr\lucene_solr_4_10_2\solr\example\solr\collection1\

12839 [main] INFO org.apache.solr.core.CoresLocator – Found 1 core definitions
12841 [coreLoadExecutor-5-thread-1] INFO org.apache.solr.core.SolrResourceLoader – new SolrResourceLoader for directory: 'C:\d\solr\lucene_solr_4_10_2\solr\example\solr\collection1\'
12842 [coreLoadExecutor-5-thread-1] INFO org.apache.solr.core.SolrResourceLoader – Adding 'file:/C:/d/solr/lucene_solr_4_10_2/solr/example/solr/collection1/lib/HTPostingsFormatWrapper.jar' to classloader
12870 [coreLoadExecutor-5-thread-1] ERROR org.apache.solr.core.CoreContainer – Error creating core [collection1]: class org.apache.lucene.codecs.HTPostingsFormatWrapper
java.lang.ClassCastException: class org.apache.lucene.codecs.HTPostingsFormatWrapper
at java.lang.Class.asSubclass(Unknown Source)
at org.apache.lucene.util.SPIClassIterator.next(SPIClassIterator.java:141)
at org.apache.lucene.util.NamedSPILoader.reload(NamedSPILoader.java:65)
at org.apache.lucene.codecs.Codec.reloadCodecs(Codec.java:119)
at org.apache.solr.core.SolrResourceLoader.reloadLuceneSPI(SolrResourceLoader.java:206)
at org.apache.solr.core.SolrResourceLoader.<init>(SolrResourceLoader.java:142)
at org.apache.solr.core.ConfigSetService$Default.createCoreResourceLoader(ConfigSetService.java:144)
at org.apache.solr.core.ConfigSetService.getConfig(ConfigSetService.java:58)
at org.apache.solr.core.CoreContainer.create(CoreContainer.java:489)
at org.apache.solr.core.CoreContainer$1.call(CoreContainer.java:255)
at org.apache.solr.core.CoreContainer$1.call(CoreContainer.java:249)
at java.util.concurrent.FutureTask.run(Unknown Source)
at java.util.concurrent.ThreadPoolExecutor.runWorker(Unknown Source)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(Unknown Source)
at java.lang.Thread.run(Unknown Source)

On Wed, Jan 14, 2015 at 6:05 PM, Chris Hostetter <[hidden email]> wrote:

> : As a foolish dev (not malicious I hope!), I did mess around with something
> : like this once; I was writing my own Codec. I found I had to create a file
> : called META-INF/services/org.apache.lucene.codecs.Codec in my solr plugin jar
> : that contained the fully-qualified class name of my codec: I guess this
> : registers it with the SPI framework so it can be found by name? I'm not
>
> Yep, that's how SPI works - the important bits are mentioned/linked in the PostingsFormat (and other SPI related classes in lucene) javadocs...
>
> https://lucene.apache.org/core/4_10_2/core/org/apache/lucene/codecs/PostingsFormat.html
>
... [show rest of quote]

« Return to Solr - User | 496 views