Item 310
git_comments:

1. But if more threads compete, then waiting a bit (random delay) can improve our chances. The delay should in theory grow as the number of concurrent threads attempting updates increase, but we don't know that number, so doing exponential backoff instead. max wait 2^13ms=8.192 sec

git_commits:

1. **summary:** SOLR-14928: add exponential backoff wait time when Compare And Swap fails in distributed cluster state update due to concurrent update (#2438)
   message: SOLR-14928: add exponential backoff wait time when Compare And Swap fails in distributed cluster state update due to concurrent update (#2438)

github_issues:

github_issues_comments:

github_pulls:

1. **title:** SOLR-14928: add exponential backoff for distributed cluster state updates
   body: add exponential backoff wait time when Compare And Swap fails in distributed cluster state update due to concurrent update

github_pulls_comments:

1. It appears you merged this PR like 7 hours after you filed it without a review? Normally we either wait for an approved code review, or merge in 2 business days, whichever comes first. If you wish to commit without review, it's also best to comment on your intention to do so stating "lazy consensus" (perhaps saying so at the very start).
2. > It appears you merged this PR like 7 hours after you filed it without a review? Normally we either wait for an approved code review, or merge in 2 business days, whichever comes first. If you wish to commit without review, it's also best to comment on your intention to do so stating "lazy consensus" (perhaps saying so at the very start). I agree with you @dsmiley in general about letting the PR's some baking time and opportunity for comments, but this is a minor change (applying a well known pattern) to a previous large PR that didn't get much attention even though I left it out 11 days and requested feedback (Mike Drob did have a look and Noble did make a comment, but no real review happened). Initial [PR 2285](https://github.com/apache/lucene-solr/pull/2285), and I used a rebased one later for merge [PR 2364](https://github.com/apache/lucene-solr/pull/2364). If you do have concerns or comments about the change in this PR, please don't hesitate. Having merged it doesn't mean it can't be changed!
3. FYI https://cwiki.apache.org/confluence/display/LUCENE/Commit+Process+Guidelines#CommitProcessGuidelines-ExceptionsforMinorChanges Wether this change is "minor" or not is, I suppose, subjective. Don't bother recapitalizing; it's not worth yet another commit (commits are for-ever). Perhaps maybe if we touch this in the future for something else.
4. It occurred to me why this raised a red flag to me -- it's because it's a PR. If you truly felt the change in question was so minor then simply commit directly to master. I certainly do this on occasion for typos, javadocs, or maybe a fix to a failing test. But the creation of the PR suggests you have an interest in a code review? Then please wait more than 7 hours for one if you want one :-)
5. @dsmiley I don’t want to push directly. I want to be able to review what will be committed to the upstream repo in case I made a mistake. So I always do PR’s.
6. I review what I commit locally in my IDE tooling. I think IntelliJ does a nice job of this.

github_pulls_reviews:

1. My instinct would be to lowercase this field name because it's not really some constant. It's rather rare in the codebase to use all-caps in other situations.
2. This variable was moved here as part of a refactoring, see [PR 2390](https://github.com/apache/lucene-solr/pull/2390). When introduced in [2014 in OverseerCollectionMessageHandler] (https://github.com/apache/lucene-solr/commit/3f999895448c9524c77ff33b801f43cfb6f8b8bd#diff-942546903f5862bde980b5392f8a4cf99e9b16fe1d48f7d1a0e61661dc36a173R167) this variable already had

that uppercase name. Do you want me to lowercase it? Likely matter of personal taste, but to me it seems ok uppercase.

jira_issues:

jira_issues_comments: