

Updating national access nodes [Draft 01] (Work packages 1 and 2)

C. Goutorbe

June 26, 2000

1 Introduction

One of the goals of the LIMES project is to establish a network of Zentralblatt-MATH servers in participating countries. These servers should be kept in synchronization as far as possible, i.e they should hold the same data. Database updates must therefore be propagated from the master production server to the national servers, on a regular (predictable) basis. A frequency of (at least) one month has been proposed.

1.1 What does updating mean ?

We have to handle all three possible kinds of updates:

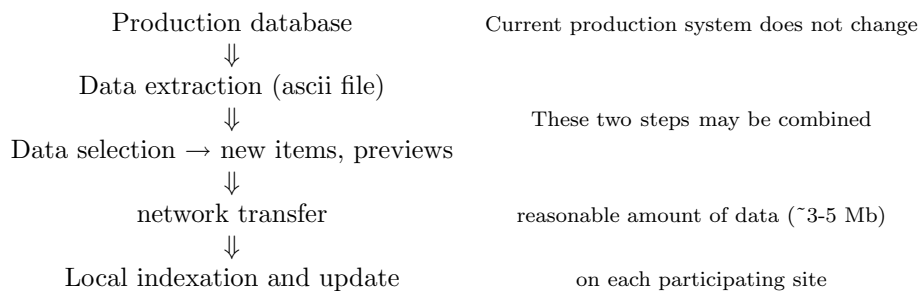
1. addition of new items, both new reviews and new previews
2. corrections to existing items
3. deletion of items that have changed from the preview state to the reviewed state

1.2 What are the constraints ?

1. We need a high enough frequency of updates
2. Servers should be updated at about the same time
3. Production of update data should not be a costly and lengthy operation

1.3 How can we cope ?

In order to meet these requirements, we propose the following general scheme for the updating process.



2 Some statistics

These figures were obtained by comparing states n and $n + 1$ of the database, in the following way:

1. for each item in db_{n+1} :
 - (a) if the item is new (its accession number does not exist in db_n), add it to the *new items* file.
 - (b) if the item exists in db_n , compare the two items, write differences to the *modified items* file.
2. for each item in db_n : if the item does not exist in db_{n+1} , write its accession number to the *deleted items* file.
3. examine raw, gzipped-compressed, and bzip2-compressed file sizes.

Note that these 3 files include all information needed to create db_{n+1} from db_n . These figures are not one hundred percent accurate, because we will have to include some “service bytes” (such as flags indicating which fields have been modified in the case of a modified item).

Update (total db size, Mb)	New items		Modified items		Deleted items		Transfer size (Mb)		
	count	size (Mb)	count	size (Mb)	count	size (Kb)	raw	gzipped	bzipped
912-914 (257,950,949)	10,909	7,732,365	17,638	6,018,103	4,613	46,13	13,796,598	4,313,081	3,240,964
914-916 (263,830,974)	8,912	6,749,182	11,145	5,625,521	4,453	44,53	12,419,233	4,035,796	3,057,320
916-918 (270,886,946)	11,755	8,246,926	11,954	5,903,831	4,665	46,65	14,197,407	4,476,092	3,376,312
918-920 (275,427,077)	10,788	7,727,126	10,987	3,446,471	5,048	50,48	11,224,077	3,534,430	2,678,429
920-922 (281,249,791)	13,167	8,239,379	7,458	2,620,138	4,838	48,38	10,907,897	3,468,570	2,611,975
922-924 (292,455,880)	28,664	15,453,158	10,419	4,055,380	12,151	121,51	19,630,048	6,118,670	4,588,077
924-924/final (266,108,713)	7	0,5755	63,609	3,127,084	59,763	597,63	3,730,469	1,065,400	0,703,776
929-931 (50,200,049)	13,905	9,362,735	12,777	6,717,972	4,737	47,37	16,128,077—	5,221,858	3,964,951
931-933 (56,033,115)	12,828	8,903,609	13,934	3,446,839	6,005	60,05	12,410,498	4,072,799—	3,087,437

(The 922/924 update includes the “special” volume 925)

3 Details of the updating process

The updating process is a two phases process.

3.1 Phase 1: production of update data.

The goal of this step is to produce file(s) containing new items, modified items, and a list of deleted items. This step is to be executed once on the main production (Berlin) server.

3.1.1 Constraints and requirements

1. keep updating information to the necessary minimum, so that files can be transmitted over the network in a reasonable amount of time.
2. simplify the local update phase as much as possible. In particular the output of this phase should be already sorted by the “natural” display order of ZMATH. (XXX add details): due to the current structure of data files, the local updates must be done by a merging procedure.

3.1.2 Proposals

Depending on the capabilities of the production system, this may be done in a variety of ways.

1. Assuming *no changes* in the production system, the main server will hold states n and $n+1$ of the database. We can then simply extract new, modified and deleted items by a comparison process similar to the one used for gathering the above statistics.
New items must obviously be included in full.
In the case of modified items, we need transmit only *changed fields* .
In the case of deleted items, we need transmit only the ZMATH accession number to identify them.
2. It may be simpler for the current production system to generate db_{n+1} in the form of a “big ascii file” . This does not affect the logic of the extraction process, but puts extra requirements on it:
 - (a) this file may have to be sorted
 - (b) new and modified items can still be extracted by a sequential scan of this file
 - (c) for performance reasons, the identification of deleted items requires some kind of indexing (which may be done during the sequential scan phase)
3. Note that extraction of new and modified data, and identification of deleted items really belong to the production system (WP_{-1}). The new input system should be able to produce information needed for updates (e.g. by using time stamping techniques and / or auxiliary tables).

3.1.3 To do list

1. Make a decision about the format of db_{n+1} . (one ore more pfs database (db.bib), one or more ascii files, sorted or unsorted, or even already extracted data). (FIZ)
2. detail specifications of the format of transmitted data (CMD)

3.2 Local indexing and merging

This step is to be executed once on each participating site. Its goal is to produce db_{n+1} from db_n and update files.

3.2.1 Constraints and requirements

1. This process must use the same indexing rules and ordering criteria that are used on the main production server, in order to produce consistent results. (documentation needed)
2. The current structure of data files does not allow updating them in place. The local updating process is actually a merging process. (and thus needs more disk space).
3. Software requirements include: the development of an indexer and merging/updating procedures.

3.2.2 To do list (before software development)

1. document existing indexing rules and ordering criteria (FIZ)