## UNDERSTANDING DOC OF BSO

**What is BSO?**

Information employed for improving search results either by boosting the score of certain products to come up in the result set or by promoting products based on consumers' buying habits, brand preferences, and product usage.

**Prerequisites’**

JAVA VM of 3G

**Bso.sh Script flow**

* It creates directories (if they don’t exist) as below

*OUTPUT\_DIR -> $BSO\_DIR/output*

*LOG\_DIR -> $OUTPUT\_DIR/logs*

*ACTIONS\_DIR -> $OUTPUT\_DIR/actions*

*SEARS\_OUTPUT\_DIR -> $OUTPUT\_DIR/sears*

*KMART\_OUTPUT\_DIR -> $OUTPUT\_DIR/kmart*

*MYGOFER\_OUTPUT\_DIR -> $OUTPUT\_DIR/mygofer3*

*DEBUG\_DIR -> $OUTPUT\_DIR/debug*

* Input files for BSO arrive in: /home/solradm/bso/input/omni\_sears.txt.gz. The input files are then decompressed using gunzip command (Primarily, the Sears and Kmart input files). Since myGofer job is required only on Wednesday, the input file for myGofer will be handled only during Wednesday’s job run. The job fails if the old omni\_<STORENAME>.txt file still exists.

*Sears Input File -> $INPUT\_DIR/omni\_sears.txt*

*Kmart Input File -> $INPUT\_DIR/omni\_kmart.txt*

*MyGofer Input File -> $INPUT\_DIR/omni\_mygofer\_1.txt*

* The script then calls bso.jar to generate rankings for Sears and Kmart based on their input file. The bso.jar operates on the omniture input file and generates avgScore.txt file. This will be a call to this jar twice, once with Sears as parameter and once with Kmart.
* The next step is modifications for average score calculation revamp. This is a call to bsoWeightedScore perl script with parameters as *$BSO\_DIR $ACTIONS\_OUTPUT\_FILE $STORE\_SEARS* for Sears store and *$BSO\_DIR $ACTIONS\_OUTPUT\_FILE $STORE\_KMART* for Kmart store.
* The next call is to the nanny perl script to do a 30day decay and 30day rollover. This call again happens individually for each store i.e. Sears, Kmart.
* The script then calls validatebso perl script with parameters BSO\_DIR, ABS\_MIN\_CNT\_ORIG and ABS\_MIN\_CNT\_WTAVG
* The script then copies the weighted average file to their respective store locations.
* If the <STORENAME>\_CROWD\_SEARCH\_FLAG is set to TRUE, then mergeBsoFiles.pl perl script is invoked to merge the bso files with crowd search file.
* The script then generates the split files for bso. This is done via JAR call. The split happens for the respective store and the files are stored in their respective store output folder.
* The flow for MyGofer happens via a call to bso\_prepare Unix script.

**bsoWeightedScore Perl Script**

* This is a subroutine to compute score for every type/event.
* The event list used for computing score include:

"PURCHASE",

"PRODUCT\_VIEW",

"SHOPPING\_CART\_OPEN",

"SHOPPING\_CART\_ADD",

"SHOPPING\_CART\_REMOVE"

* The script calls computeScore function to perform the score computation based on the event list defined already.
* The input file is of the below format

*00041564v|SPM8765967713P|1|0*

*000p00340123000p00341462000p00340125000p00340126000p00340128000p00335333000p00335332000p003|07118136000P|1|0*

*00041564v|SP102A23526S6576491905P|1|0*

* This subroutine computeScore reads the input file (bso\_$TIMESTAMP.avgScore.txt) line by line, and computes eventScore as multiple of eventWeightage and eventOccurence. The avgScore is then computed as the sum of avgScore and eventScore for all the fields except the fourth field in the input file. In case of fourth field of the input file, the avgScore becomes the difference of avgScore and eventScore.
* If the computed avgScore is not 0, then along with the first two fields of the input file, the avgScore is also written as the third field to bso\_$TIMESTAMP.weighted\_average.txt output file.

**Nanny Perl Script**

* The nanny perl script is used to implement bso files roll over and decay.
* The script is invoked via perl nanny.pl [options] dirname and it accepts the below parameters

*Options:*

*-r, --rollover=ROLLOVER*

*Remove files older than ROLLOVER days in the past*

*-d, --decay=DECAY*

*Decay rankings based on DECAY days in the past*

*-a, --average*

*Calculate the weighted average for rankings across decayed files*

*-l, --lastmodified*

*Remove all files from dirname whose internal last-modified time is earlier than today*

*-v, --verbose*

*Enable verbose output*

*-s, --syslog*

*Direct info, debug and error messages to syslog instead of standard out*

* The function decay is to decay rankings based on DECAY days in the past. This is implemented by adjust\_rankings subroutine. The age of the file is computed by means of get\_age function. The clean operation begins with removing keywords with upper case, and retaining alphanumeric keywords. It also removes all those phrases with more than 1 space, all those phrases beginning with a space, all those spaces ending with a space, removing div-item & div/item searches, removing scores of zero, removing partnumber searches and removing multiple partnumber searches. It then removes duplicate entries i.e. keeping only unique phrase+partnumber combination. The phrases are then sorted alphabetically and score in descending order. The cleaned output file is then written. The function also keeps track of the below

*Rows upper case keywords : $upperPhraseCount*

*Rows without alphanumeric words : $alphaNum*

*Rows having keyword with more than one space* *: $moreSpaceCount*

*Rows starting with space : $preSpaceCount*

*Rows ending with space : $postSpaceCount*

*Rows with div-item search : $divitemSearchCount*

*Rows having scores of zero : $scoresOfZeroCount*

*Rows having partNumber search : $partnumberCount*

*Rows having multi partNumber search $multiPartnumberCount*

*Duplicate rows eliminated : $duplicateCount*

The cleaned file is then trimmed using trim. The trim happens based on cutoff value passed during the nanny perl script call (-c parameter) i.e. the first 25 (cutoff value)fields are trimmed to output file.

* The function rollover is to remove files older than ROLLOVER days in the past.
* The function average is to calculate the weighted average for rankings across decayed files.
* The parameter lastModified is to remove all files from dirname whose internal last-modified time is earlier than today

**validatebso Perl Script**

* This script does the validation of bso file and data. The perl script starts with a function call fetchFilesForValidation, which loads all rows from weighted\_average.txt file onto an array and all rows from orig.txt onto another array.
* These two array elements are then sorted based on timestamp.
* The files are then validated for record count.

**mergeBsoFiles perl Script**

* This script does the merging of bso file with the crowd search file.
* The script compares each row of bso file with the crowd search file, and data is found in one of them and not in the other, the data is merged as value from one file with second field value from the other file, both separated by |.
* When both the bso file and crowd search file have the same keyword – partnumber combination, then a boostValue (passed as parameter to this script) is applied and written as third field to the existing keyword, partnumber fields.

**bso\_prepare Script for MyGofer Store**

* This script will download the BSO data and prepares them for SOLR indexing.
* It connects to an ftp host and downloads the zipped version of BSO data for MyGofer.
* The file is then unzipped and named as BSO\_for\_Mygofer.csv, for use by SOLR indexing.