HASHBUFFER

By: Christopher Lane (Lane@Sumex-Aim.Stanford.Edu)

Uses: HASH

HASHBUFFER combines *hash files* with *hash arrays* in order to improve hash file performance when keys are accessed multiple times. This module also defines two functions for moving data between hash files and hash arrays.

The functions below are used in place of the hash file routines. When a hash file is opened, a hash array is created, of a complimentary size. When requests for keys are made, the array is searched, and if a value is found, it is returned. If a value is not found, the file is searched and if a value is found there, it is stored in the array and returned. If a value is not found, a marker is put in the array so that the file is not searched again.

(OPENHASHBUFFER *FILE [ACCESS MINKEYS OVERFLOW HASHBITSFN EQUIVFN]*) [Function]

Opens an existing hash file and returns a hash buffer datum which must be given to the other hash buffer functions. Only the *FILE* argument is required; the *MINKEYS* argument is used for the size of the hash array and if not supplied the size of the hash file is used. Setting *MINKEYS* smaller than the size of the hash file allows a fast, small hash array window onto a larger, slower hash file. The *OVERFLOW*, *HASHBITSFN* and *EQUIVFN* arguments are passed to `HASHARRAY`.

(CREATEHASHBUFFER *FILE [VALUETYPE ITEMLENGTH #ENTRIES ^*
OVERFLOW HASHBITSFN EQUIVFN]) [Function]

Like OPENHASHBUFFER but creates a new hash file. The *FILE*, *VALUETYPE* and *ITEMLENGTH* arguments are passed to CREATEHASHFILE; the *OVERFLOW*, *HASHBITSFN* and *EQUIVFN* arguments are passed to HASHARRAY. The *#ENTRIES* argument is used for both the fIle and array.

(CLOSEHASHBUFFER *HASHBUFFER [FILEONLY?]*) [Function]

Closes the hash file and sets the hash array to NIL so that it can be reclaimed. If *FILEONLY?* is non-NIL then only the hash file is closed, the hash array will be left alone.

(GETHASHBUFFER *KEY HASHBUFFER*) [Function]

(PUTHASHBUFFER *KEY VALUE HASHBUFFER*) [Function]

Retrieve and store *VALUE* for *KEY* in the hash buffer. If the hash file is only open for input, then storing a key will only affect the hash array. If the hash file is open for output, then storing a key will put it in both the hash array and hash file. If *VALUE* is NIL, then a delete is performed.

(HASHARRAY.TO.HASHFILE *HASHARRAY HASHFILE [TESTFN]*) [Function]

Uses MAPHASH to move the contents of *HASHARRY* into a hash file. If *HASHFILE* is a file name, CREATEHASHFILE is called; if *HASHFILE* is an open hash file datum, it is used and left open. *TESTFN*, if supplied, is called before each PUTHASHFILE on (KEY VALUE HASHARRAY HASHFILE) and if it returns non-NIL, the key and value are copied to the file.

(HASHFILE.TO.HASHARRAY *HASHFILE [HASHARRAY TESTFN]*) [Function]

Uses MAPHASHFILE to move the contents of *HASHFILE* into a hash array. *If HASHARRAY* is not supplied a new hash array is created. *TESTFN* is called before each PUTHASH on (KEY VALUE HASHFILE HASHARRAY) and if it returns non-NIL, the key and value are copied to the array.