



名称	修改日期	类型	大小
ArmV10	2022-04-07 10:07	文件夹	
linux.x64	2022-01-11 13:13	文件夹	
SUSE	2022-11-02 19:54	文件夹	
workspace	2023-08-16 21:03	文件夹	
X86V10	2022-04-07 10:07	文件夹	

信创版本

非信创版本

磁盘 (F:) > UFTDB\_NEW > UFTDB\_NEW > instance > ArmV10 > lib

名称

修改日期

类型

libdata\_transfer.so

2023-01-19 10:28

SO 文件

libfsc\_comparedb.so

2023-01-19 10:28

SO 文件

libfsc\_dbserver.so

2023-01-19 10:28

SO 文件

libfsc\_todb.so

2023-07-27 10:54

SO 文件

libfsc\_uft\_routerchange.so

2023-01-19 10:28

SO 文件

libfsc\_uftdb.so

2023-08-16 21:03

SO 文件

libfsc\_uftdb\_logrecv.so

2023-01-19 10:28

SO 文件

libfsc\_uftdb\_logsend.so

2023-01-19 10:28

SO 文件

libs\_compt\_uftdb.so

2023-01-19 10:28

SO 文件

libs\_glbfunction\_uft.so

2023-01-19 10:28

SO 文件

libs\_libpublic\_uft.so

2023-07-27 10:38

SO 文件

libs\_uftdb\_rftread.so

2023-06-15 9:16

SO 文件

libuftdb.so

2023-01-19 10:28

SO 文件

libuftdb\_dbless.so

2023-08-16 21:03

SO 文件

libuftdb\_dm.so

2023-08-16 21:03

SO 文件

libuftdb\_kafka\_compt.so

2023-07-27 10:54

SO 文件

libuftdb\_lightdb.so

2023-08-16 21:03

SO 文件

libuftdb\_mysql.so

2023-08-16 21:03

SO 文件

libuftdb\_ob\_oracle.so

2023-08-16 21:03

SO 文件

libuftdb\_parse.so

2023-01-19 10:28

SO 文件

libuftdb\_sqlite\_vtab.so

2023-01-19 10:28

SO 文件

本地磁盘 (E:)

AlgoServerForWin

AlgoServerForWin0301

app

AresCode

AresWork

AresWork\_01

AresWork\_06

AresWork\_资讯平台

bin

bin\_测试问题

BlogNote

DingTalkAppData

Java

Pictrue

python

QQMusicCache

qqpcmgr\_docpro

Software

SVN

Tools

Windows Kits

名称

修改日期

类型

大小

libalgo\_calculator.so

2023-07-19 10:19

SO 文件

28,204 KB

libclickhouse-cpp-lib.so

2023-07-19 10:19

SO 文件

6,965 KB

libcurl.so

2023-08-30 10:53

SO 文件

3,819 KB

libcurl.so.4

2023-08-30 10:53

4 文件

3,819 KB

libcurl.so.4.7.0

2023-08-30 10:53

0 文件

3,819 KB

libf\_os.so

2023-07-19 10:19

SO 文件

403 KB

libfsc\_comparedb.so

2023-07-19 10:19

SO 文件

3,083 KB

libfsc\_curlqt.so

2023-07-19 10:19

SO 文件

513 KB

libfsc\_f2config.so

2023-07-19 10:19

SO 文件

258 KB

libfsc\_hscmmmsg.so

2023-07-19 10:19

SO 文件

2,930 KB

libfsc\_license.so

2023-07-19 10:19

SO 文件

202 KB

libfsc\_todb.so

2023-07-19 10:19

SO 文件

2,745 KB

libfsc\_uft\_routerchange.so

2023-07-19 10:19

SO 文件

804 KB

libfsc\_uftdb.so

2023-07-19 10:19

SO 文件

8,977 KB

libh5api.so

2023-08-24 19:19

SO 文件

3,625 KB

libh5sdk.so

2023-07-19 10:19

SO 文件

6,574 KB

libh5sdk\_quotelog.so

2023-07-19 10:19

SO 文件

1,210 KB

libhs\_best\_message.so

2023-07-19 10:19

SO 文件

1,288 KB

libhsfixeng20.so

2023-07-19 10:19

SO 文件

9,794 KB

liblua.so

2023-07-19 10:19

SO 文件

601 KB

libnlopt.so

2023-07-19 10:19

SO 文件

1,333 KB

libnlopt.so.0

2023-07-19 10:19

0 文件

1,333 KB

libnlopt.so.0.8.2

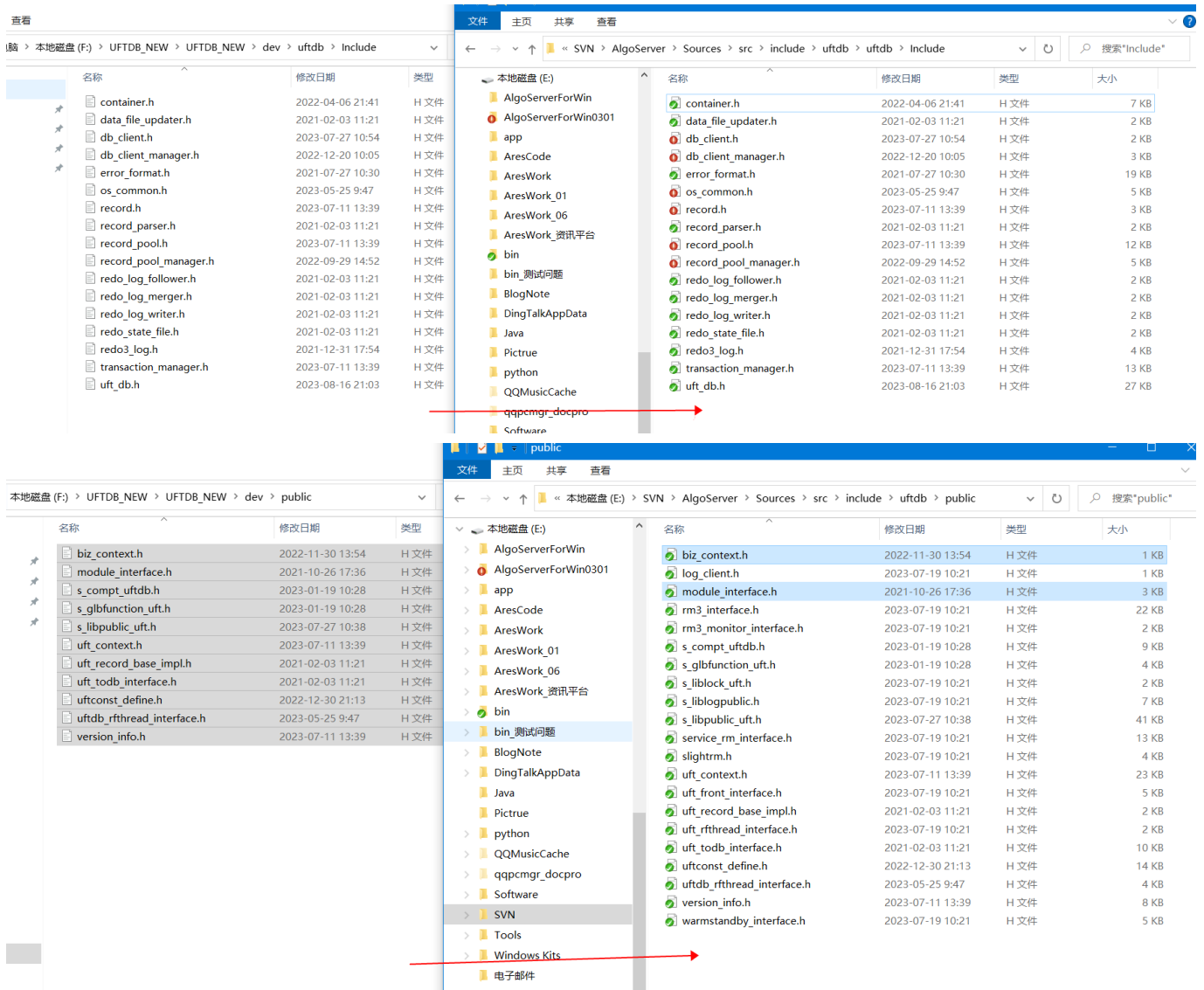
2023-07-19 10:19

2 文件

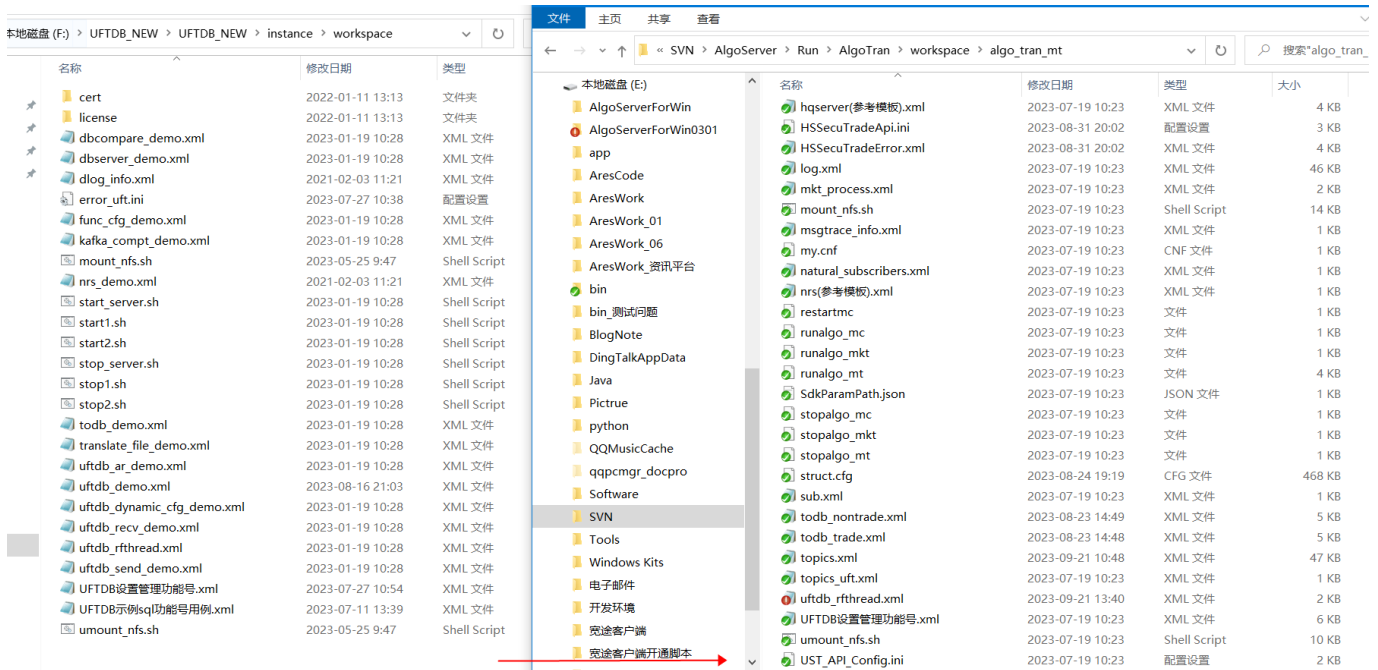
1,333 KB

一定要注意：修改source/src下的makefile文件，将新增的so加入到STP\_INSTALL中

4. 覆盖instance\linux.x64\lib(非信创)、instance\ArmV10\lib(信创)下的so到Run\ASAR\bin和Run\ASAR\_V10ARM\bin下
5. 替换dev\下的头文件

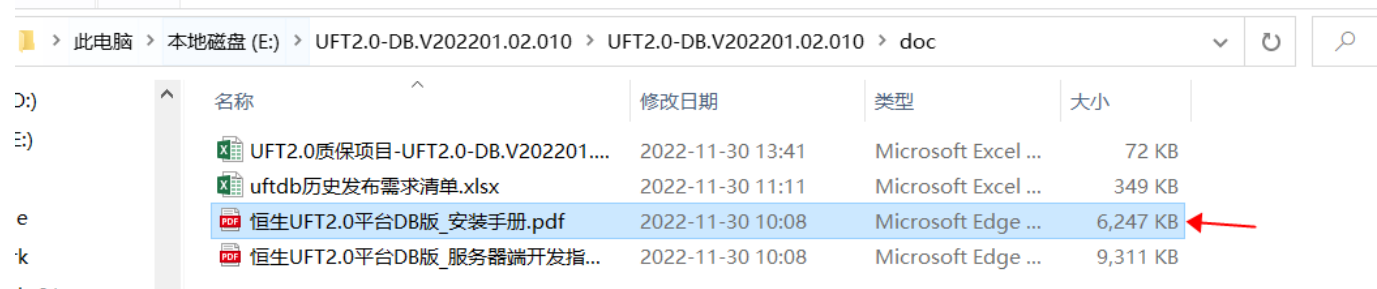


## 6. 替换instance\workspace下的配置文件



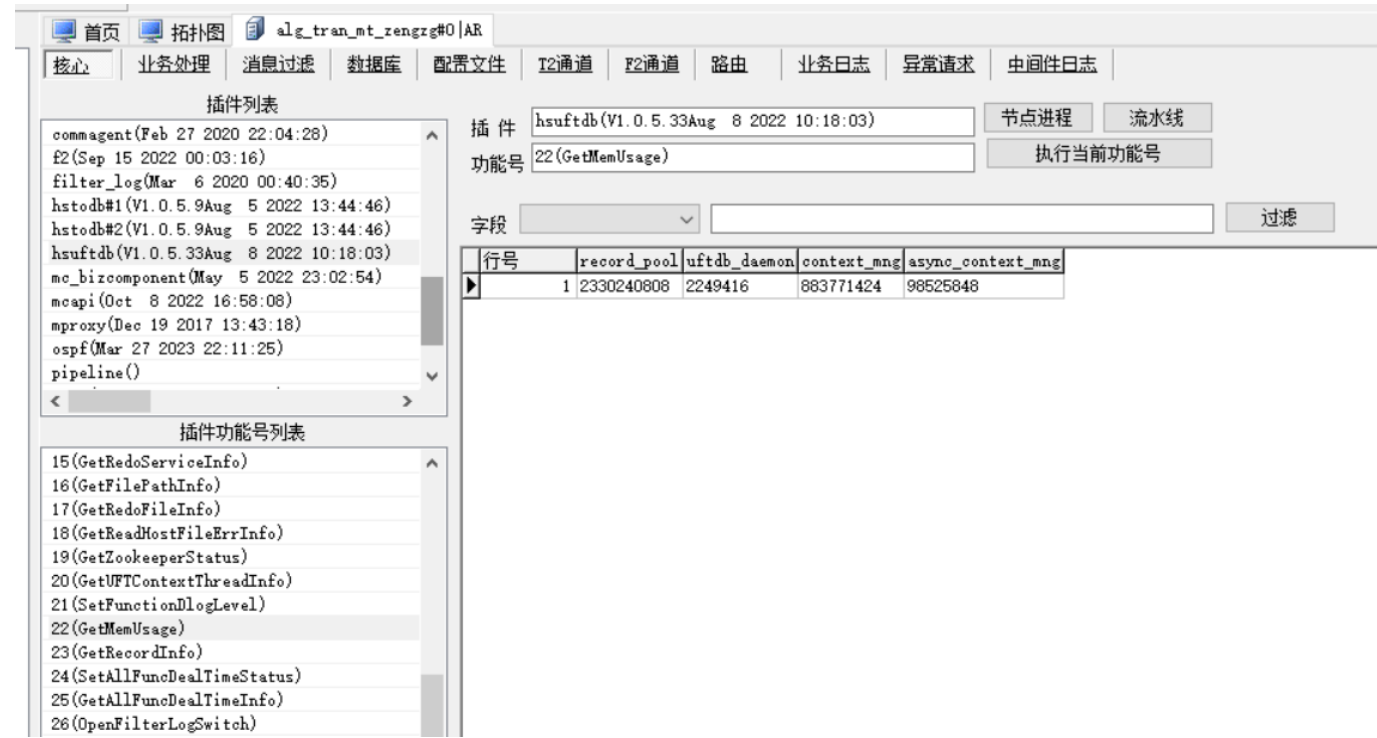
一般来说，如果algotran/workspace下没有的配置文件不需要更新，对已有的xml文件进行覆盖更新就行

7. 查看内存使用大小，这些函数可以在下载制品的pdf中查找



4.22 GetMemUsage 获取内存使用信息

功能名称	GetMemUsage	版本号	v1.0.0.0	更新日期	20171023
功能备注	获取内存使用信息				
功能说明	获取内存使用信息				
入参说明	字段名	说明			类型
	无				
出参说明	字段名	说明			类型
	record_pool	记录池使用内存(单位：字节)			s-字符串
	uftdb_daemon	Uftdb 守护进程使用内存(单位：字节)			s-字符串
	context_mng	上下文管理器使用内存(单位：字节)			s-字符串
	async_context_mng	异步上下文管理器使用内存（单位：字节）			s-字符串



### 4.23 GetRecordInfo 获取记录信息

功能名称	GetRecordInfo	版本号	v1.0.0.0	更新日期	20171023
功能备注	获取记录信息				
功能说明	获取记录信息				
入参说明	字段名	说明			类型
	无				
出参说明	字段名	说明			类型
	category	记录号			I-整形
	record_name	表名			S-字符串
	memory_usage	使用内存(单位: KB)			S-字符串
	record_count	记录条数			I-整形
	load_record_time	加载记录耗时			I-整形

插件		hsuftdb(V1.0.6.31Aug 1 2023 22:18:27)				节点进程	流水线
功能号		23(GetRecordInfo)				执行当前功能号	
字段						过滤	
	行号	category	record_name	memory_usage	record_count	load_record_time(us)	
	1	100	tstp_stockinfo	33088	48427	0	
	2	1001	tstp_stockinfo_hq	6976	47728	0	
	3	1002	tstp_bondproperty_ext	0	0	0	
	4	101	tstp_futureinfo	2880	25	0	
	5	102	tstp_priceinterval	65728	315	0	
	6	103	tstp_dictionary	25280	829	0	
	7	104	tstp_o32service	1344	3390	0	
	8	105	tstp_optionproperty	5056	11	0	
	9	106	tstp_etfbasicinfo	0	0	0	
	10	107	tstp_etfstocklistinfo	0	0	0	
	11	108	tstp_bondproperty	23232	2	0	
	12	109	tstp_goldtcontractparam	0	0	0	
	13	110	tstp_gzinfo	1728	10485	0	
	14	201	tstp_investunit	11584	626	0	
	15	202	tstp_capitalunit	133952	626	0	
	16	203	tstp_tinstancestock	8128	34	0	
	17	204	tstp_fieldreflect	147136	3159	0	
	18	205	tstp_operator	5056	83	0	
	19	206	tstp_instance	7744	626	0	
	20	207	tstp_daytradeprofit	0	0	0	
	21	208	tstp_systemparaminfo	71104	32	0	
	22	209	tstp_etfrevisedprice	0	0	0	
	23	210	tstp_etfportfolio	0	0	0	
	24	2100	tstp_ctpposition	0	0	0	
	25	2101	tstp_entrustwithdrawal	0	0	0	
	26	2102	tstp_pendentrust	0	0	0	
	27	2103	tstp_mktbestpush	0	0	0	
	28	2104	tstp_mktpendentrust	0	0	0	
	29	2105	tstp_instruction	152768	20	0	
	30	2106	tstp_instructionstock	13120	15	0	
	31	2107	tstp_investadvice	0	0	0	
	32	2108	tstp_investadvicestock	0	0	0	
	33	2109	tstp_investriskinfo	0	0	0	
	34	211	tstp_positiondetail	3648	48	0	
	35	2110	tstp_investflowinfo	0	0	0	
	36	2111	tstp_inshgmortagage	0	0	0	
	37	2112	tstp_clickpresetparam	0	0	0	
	38	212	tstp_schemecodereflect	6592	1	0	

8. 最后，在algo的Run\ASAR\bin\CRES2.0历史发布需求清单.xlsx文件中新增记录

uft内存优化方式



每次清流重启mt，都会将内存表的数据备份到./workspace/uftdata/data/目录下，然后重新将表数据加载到内存表。

tip:也会将.dat文件压缩备份到workspace下 (\*.tar.gz) [参考链接](#)

低版本的uft每加载一张非空的内存表，都会分配300M的内存，而其中有很多小表，根本用不了300M，因此造成表空间内存浪费非常大，如./workspace/uftdata/data/下的表数据只有85M,但却分配了1.5G的内存

优化后的uft每次加载内存表的数据时，若发现已有内存不足，只会分配一个2M的小块，总的分配的内存为200M左右

## 查看内存

使用top命令查看mt启动后的占用内存信息：

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
293321	algotran	20	0	26.638g	2.349g	56828	S	28.3	0.9	3:27.44	hsserver
265999	algotran	20	0	4214636	191964	6836	S	5.7	0.1	490:35.92	hsserver
267598	algotran	20	0	1907972	107648	4708	S	1.3	0.0	98:45.53	hsserver
1	root	20	0	104384	1448	492	S	0.0	0.0	0:01.09	sshd
75668	oracle	20	0	217028	3732	120	S	0.0	0.0	1:17.22	tnslsnr
232050	rpc	20	0	37732	784	392	S	0.0	0.0	0:00.64	rpcbind
241253	root	20	0	124164	1368	720	S	0.0	0.0	0:04.26	crond
267597	algotran	20	0	185972	22996	2116	S	0.0	0.0	2:37.72	hsserver
283270	root	20	0	142112	5488	4128	S	0.0	0.0	0:05.96	sshd

可以看到：虚拟内存降低了1.1G

注意：因为内存表的表数据一直是85M，虚拟内存映射到物理内存的页数量并未变化，因此mt占用的物理内存不会有明显变化

## 待优化点

- mt启动后，占用的虚拟内存过大，达到了26G，这会导致mt一旦崩溃，产生的core也会特别大；
- 其次，虚拟内存分配太大，如果超过物理内存，会将物理内存中暂未使用的内存交换到磁盘（在虚拟机中会产生很大一块交换区，在公共服务器上内存很大，交换区内存为0），降低mt性能。