DDT causes a lot of noise in Valgrind output when migrating AMPI ranks. I haven't been able to find the source of its supposed leaks, so I'm not sure if AMPI is doing something to confuse Valgrind or if these are legitimate leaks...

```
$ ./charmrun +p2 /usr/bin/valgrind --tool=memcheck --leak-check=yes -v --log-file=myvalg_txt.%p --trace-children=yes ./jacobi +vp1 1 1 1 6 +balancer RotateLB ++local
```

120 bytes in 1 blocks are definitely lost in loss record 237 of 317
at 0x4C2B0E0: operator new(unsigned long) (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.so)
  by 0x6621D7: CkDDT::CkDDT() (ddt.h:471)
  by 0x630DC8: ampiParent::ampiParent(CkMigrateMessage*) (ampi.C:1256)
  by 0x671E8F: call_migration_constructor<ampiParent, true>::operator() (void*) (charm++.h:281)
  by 0x653E6D: CkIndex_ampiParent::call_ampiParent_CkMigrateMessage(void*, void*) (ampi.def.h:1832)
  by 0x6A07E3: CkDeliverMessageFree (ck.C:593)
  by 0x6BF684: CkLocRec::invokeEntry(CkMigratable*, void*, int, bool) (cklocation.C:2024)
  by 0x6C1085: CkLocMgr::addElementToRec(CkLocRec*, CkArray*, CkMigratable*, int, void*) (cklocation.C:2429)
  by 0x6C31E7: CkLocMgr::pupElementsFor(PUP::er, CkLocRec*, CkElementCreation_t, bool) (cklocation.C:3008)
  by 0x6C3CC: CkLocMgr::immigrate(CkArrayElementMigrateMessage*) (cklocation.C:3212)
  by 0x6C617C: CkIndex_CkLocMgr::call_immigrate_CkArrayElementMigrateMessage(void*, void*) (cklocation.def.h:692)
  by 0x6A07E3: CkDeliverMessageFree (ck.C:593)

120 bytes in 1 blocks are definitely lost in loss record 238 of 317
at 0x4C2B0E0: operator new(unsigned long) (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.so)
  by 0x662220: CkDDT::CkDDT() (ddt.h:473)
  by 0x630DC8: ampiParent::ampiParent(CkMigrateMessage*) (ampi.C:1256)
  by 0x671E8F: call_migration_constructor<ampiParent, true>::operator() (void*) (charm++.h:281)
  by 0x653E6D: CkIndex_ampiParent::call_ampiParent_CkMigrateMessage(void*, void*) (ampi.def.h:1832)
  by 0x6A07E3: CkDeliverMessageFree (ck.C:593)
  by 0x6BF684: CkLocRec::invokeEntry(CkMigratable*, void*, int, bool) (cklocation.C:2024)
  by 0x6C1085: CkLocMgr::addElementToRec(CkLocRec*, CkArray*, CkMigratable*, int, void*) (cklocation.C:2429)
  by 0x6C31E7: CkLocMgr::pupElementsFor(PUP::er, CkLocRec*, CkElementCreation_t, bool) (cklocation.C:3008)
  by 0x6C3CC: CkLocMgr::immigrate(CkArrayElementMigrateMessage*) (cklocation.C:3212)
  by 0x6C617C: CkIndex_CkLocMgr::call_immigrate_CkArrayElementMigrateMessage(void*, void*) (cklocation.def.h:692)
  by 0x6A07E3: CkDeliverMessageFree (ck.C:593)

These same “960 bytes in 8 blocks are definitely lost” from CkDDT's constructor are repeated for every datatype that is in the typeTable... I have verified that we are pup’ing all fields of CkDDT_DataType and that we are properly destructing them as well.

Also the link pointed to by Valgrind in AMPI is ampiParent's migration ctor. The ampiParent owns the DDT object, but it is not
When we migrate only 1 rank from PE 0 to PE 1, only the valgrind output from PE 1 shows the DDT leaks. PE 0 is clean.

I'm using DDT after all dynamic memory allocation inside the CkDDT_DataType classes has been replaced with std::vector and std::string too, so there should be nothing to leak except the CkDDT_DataType objects themselves, which I've verified are being destructed and created anew during PUP. But this makes no difference to Valgrind: https://charm.cs.illinois.edu/gerrit/#/c/2917/

Here's the valgrind output from PE 1 after migrating a single AMPI rank from PE 0 to PE 1.

All 3 of these commits seem to make no difference in Valgrind output, though they make DDT more readable:

Use std::vector instead of raw new/delete in DDT: https://charm.cs.illinois.edu/gerrit/#/c/2219/
Fix/cleanup DDT pup routines: https://charm.cs.illinois.edu/gerrit/#/c/2917/
Simplify ampiParent::myDDT usage: https://charm.cs.illinois.edu/gerrit/#/c/2185/

- Target version changed from 6.8.1 to 6.9.0

Files
myvalg_txt.1990 80.2 KB 08/17/2017 Sam White