Eigen 向量化加速，对其导致崩溃问题 1. 解决策略

Eigen 内由于断言失败而终止的打印信息类似如下

```
program: path/to/eigen/Eigen/src/Core/DenseStorage.h:44:
Eigen::internal::matrix_array<T, Size, MatrixOptions,
Align>::internal::matrix_array()
[with T = double, int Size = 2, int MatrixOptions = 2, bool Align
= true]:
Assertion `(reinterpret_cast<size_t>(array) & (sizemask)) == 0 &&
"this assertion
is explained here: http://eigen.tuxfamily.org/dox-
devel/group__TopicUnalignedArrayAssert.html
     READ THIS WEB PAGE !!! ****"' failed.
```

Eigen 官方给出的解决地址：http://eigen.tuxfamily.org/dox/TopicUnalignedArrayAssert.html
国内博客汉语版的翻译地址：https://blog.csdn.net/qq_27806947/article/details/105356401

解决方式根据产生原因不同分为四类

eigen类型是成员变量

```
class Foo
{
  //...
  Eigen::Vector2d v;
  //...
};
//...
Foo *foo = new Foo;
```

解决策略：Structures Having Eigen Members
请注意，此处Eigen :: Vector2d仅用作示例，更一般而言，所有fixed-size vectorizable Eigen types都会出现此问题。

STL容器或手动分配内存(new)

如果您将stl容器（例如std::vector，std::map等）与Eigen对象或包含Eigen对象的类一起使用，

```
std::vector<Eigen::Matrix2f> my_vector;
struct my_class { ... Eigen::Matrix2f m; ... };
std::map<int, my_class> my_map;
```

那么您需要阅读以下单独的页面：STL容器与Eigen结合使用。
请注意，此处Eigen::Matrix2f仅用作示例，更一般而言，对于所有fixed-size vectorizable Eigen types和structures having such Eigen objects as member，都会出现此问题。

值传递eigen对象

如果您代码中的某些函数正在通过值传递Eigen对象，例如这样，

```
void func(Eigen::Vector4d v);
```

那么您需要阅读以下单独的页面：将Eigen对象按值传递给函数。

编译器对堆栈对齐做出错误假设(例如Windows上的GCC)

这是在Windows上使用GCC（例如MinGW或TDM-GCC）的人们的必读内容。If you have this assertion failure in an innocent function declaring a local variable like this:

```
void foo()
{
  Eigen::Quaternionf q;
  //...
}
```

then you need to read this separate page: Compiler making a wrong assumption on stack alignment.
请注意，此处Eigen :: Quaternionf仅用作示例，更一般而言，所有fixed-size vectorizable Eigen types都会出现此问题。

断言的一般解释

fixed-size vectorizable Eigen types必须绝对在16字节对齐的位置创建，否则寻址它们的SIMD指令将崩溃。
Eigen normally takes care of these alignment issues for you, by setting an alignment attribute on them and by overloading their "operator new".However there are a few corner cases where these alignment settings get overridden: they are the possible causes for this assertion.

不在乎最佳矢量化，如何禁止向量优化

三种可能性

1. 使用Matrix，Array，Quaternion等对象的DontAlign选项会给您带来麻烦。这样，Eigen不会尝试对齐它们，因此不会承担任何特殊的对齐。On the down side, you will pay the cost of unaligned loads/stores for them, but on modern CPUs, the overhead is either null or marginal. See here for an example.
2. Define EIGEN_DONT_ALIGN_STATICALLY . That disables all 16-byte (and above) static alignment code, while keeping 16-byte (or above) heap alignment. This has the effect of vectorizing fixed-size objects (like Matrix4d) through unaligned stores (as controlled by EIGEN_UNALIGNED_VECTORIZE), while keeping unchanged the vectorization of dynamic-size objects (like MatrixXd). But do note that this breaks ABI compatibility with the default behavior of static alignment.
3. 或同时定义EIGEN_DONT_VECTORIZE和
   EIGEN_DISABLE_UNALIGNED_ARRAY_ASSERT。这样可以保留16字节的对齐代码，从而保留ABI兼容性，但完全禁用向量化。
   如果您想知道为什么定义EIGEN_DONT_VECTORIZE本身并不能禁用16字节对齐和断言，则说明如下：
   它不会禁用断言，because otherwise code that runs fine without vectorization would suddenly crash when enabling vectorization. 它不会禁用16字节对齐，因为这将意味着矢量化和非矢量化的代码ABI不兼容。即使对于仅开发内部应用程序的人，这种ABI兼容性也非常重要，as for instance one may want to have in the same application a vectorized path and a non-vectorized path.

相关阅读:
rsync 安全复制使用程序
mysql 的sleep线程过多处理方法
test
跨域问题的解决
在VMWare里安装Win11虚机
Nginx 基础入门（收藏）
NMXweb版
ORA12519: TNS:no appropriate service handler found 解决
关于工作中的第一个项目的个人总结[主要是个人学到的东西，细节]
处理警告：No configuration found for the specified action

原文地址：https://www.cnblogs.com/flyinggod/p/13433452.html

最新文章
个人职业规划,你需要注意的8件事
Macro
Guru of the Week 条款01: 变量的初始...
Python标准库参考sched
发牢骚
恐怖庄园的秘密 The Secret of Grisly ...
Guru of the Week 条款03: 使用标准库
the lenght of int
PG,PL,SE,PM都是什么意思,职责划分
iexpress

热门文章
ACE_Task框架
Guru of the Week 条款02: 临时对象
职场上打死都不能够多说的十句话
静态多态与动态多态
ActiveX控件开发总结
boost.timer：一个优秀的计时类库
Enterprise Architect Schema Script.In...
comments valuable
计算机科学中最重要的32个算法
mysql备份