Cython 是什么？为什么会有 Cython?

转载 Python猫 于 2022-06-12 21:10:50 发布 6627 收藏 15
文章标签： c++ 编程语言 python java 人工智能

Cython 估计很多人都听说过，它是用来对 Python 进行加速的。如果你在使用 Python 编程时，有过如下想法，那么 Cython 非常适合你。

- 1）因为某些需求导致不得不编写一些多重嵌套的循环，而这些循环如果用 C 语言来实现会快上百倍，但是不熟悉 C 或者不知道 Python 如何与 C 进行交互；
- 2）因为 Python 解释器的性能原因，如果将 CPython 解释器换成 PyPy，或者干脆换一门语言，比如 Rust，将会得到明显的性能提升，可是换不得。因为你的项目组规定只能使用 Python 语言，解释器只能是 CPython；
- 3）Python 是一门动态语言，但你希望至少在数字计算方面，能够加入可选的静态类型，这样可以极大地加速运算效果。因为单纯的数字相加不太需要所谓的动态性，尤其是当你的程序中出现了大量的计算逻辑时；
- 4）对于一些计算密集型的部分，你希望能够写出一些媲美 Numpy, Scipy, Pandas 的算法；
- 5）你有一些已经用 C、C++ 实现的库，你想直接在 Python 内部更好地调用它们，并且不使用 ctypes、cffi 等模块；
- 6）也许你听说过 Python 和 C 可以无缝结合，通过 C 来为 Python 编写扩展模块，将 Python 代码中性能关键的部分使用 C 进行重写，来达到提升性能的效果。但是这需要你对 Python 解释器有很深的了解，熟悉底层的 Python/C API，而这是一件非常痛苦的事情；

如果你有过上面的一些想法，那么证明你的 Python 水平是很优秀的，然而这些问题总归是要解决的，于是 Cython 便闪亮登场了。注意：Cython 并不是一个什么实验性的项目，它出现的时间已经不短了，并且在生产环境中久经考验，我们完全是有理由学习它的。

下面让我们开始 Cython 的学习之旅吧，悄悄说一句，我个人非常喜欢 Cython 的语法。

Cython 是什么?

关于 Cython，我们必须要清楚两件事：

1）Cython 是一门编程语言，它将 C 和 C++ 的静态类型系统融合在了 Python 身上。Cython 源文件的后缀是 .pyx，它是 Python 的一个超集，语法是 Python 语法和 C 语法的混血。当然我们说它是 Python 的一个超集，因此你写纯 Python 代码也是可以的。

2）当我们编写完 Cython 代码时，需要先将 Cython 代码翻译成高效的 C 代码，然后再将 C 代码编译成 Python 的扩展模块。

在早期，编写 Python 扩展都是拿 C 去写，但是这对开发者有两个硬性要求：一个是熟悉 C，另一个是要熟悉解释器提供的 C API，这对开发者是一个非常大的挑战。此外，拿 C 编写代码，开发效率也非常低。

而 Cython 的出现则解决了这一点，Cython 和 Python 的语法非常相似，我们只需要编写 Cython 代码，然后再由 Cython 编译器将 Cython 代码翻译成 C 代码即可。所以从这个角度上说，拿 C 写扩展和拿 Cython 写扩展是等价的。

至于如何将 Cython 代码翻译成 C 代码，则依赖于相应的编译器，这个编译器本质上就是 Python 的一个第三方模块。它就相当于是一个翻译官，既然用 C 写扩展是一件痛苦的事情，那就拿 Cython 去写，写完了再帮你翻译成 C。

因此 Cython 的强大之处就在于它将 Python 和 C 结合了起来，可以让你像写 Python 代码一样的同时还可以获得 C 的高效率。所以我们看到 Cython 相当于是高级语言 Python 和低级语言 C 之间的一个融合，因此有人也称 Cython 是 "克里奥尔编程语言"（creole programming language）。

> 克里奥尔人是居住在西印度群岛的欧洲人和非洲人的混血儿，以此来形容 Cython 也类似于是一个（Python 和 C 的）混血儿。

为什么要有 Cython?

Python 和 C 语言大相径庭，为什么要将它们融合在一起呢？答案是：因为这两者并不是对立的，而是互补的。

Python 是高阶语言、动态、易于学习，并且灵活。但这些优秀的特性是需要付出代价的，因为 Python 的动态性、以及它是解释型语言，导致其运行效率比静态编译型语言慢了好几个数量级。

而 C 语言是最古老的静态编译型语言之一，并且至今也被广泛使用。从时间来算的话，其编译器已有将近半个世纪的历史，在性能上做了足够的优化，因此 C 语言是非常低级、同时又非常强大的。然而不同于 Python 的是，C 语言没有提供保护措施（没有 GC、容易内存泄露），以及使用起来很不方便。

所以两个语言都是主流语言，只是特性不同使得它们被应用在了不同的领域。而 Cython 的美丽之处就在于：它将 Python 语言丰富的表达能力、动态机制和 C 语言的高性能汇聚在了一起，并且代码写起来仍然像写 Python 一样。

注意：除了极少数的例外，Python 代码（2.x和3.x版本）已经是有效的 Cython 代码，因为 Cython 可以看成是 Python 的超集。并且 Cython 在 Python 语言的基础上添加了一些少量的关键字来更好地开发 C 的类型系统，从而允许 Cython 编译器生成高效的 C 代码。如果你已经知道 Python 并且对 C 或 C++ 有一定的基础了解，那么你可以直接学习 Cython，无需再学习其它的接口语言。

另外，我们其实可以将 Cython 当成两个身份来看待：

1）如果将 Cython 翻译成 C，那么可以看成 Cython 的 '阴'；

2）如果将 Python 作为胶水连接 C 或者 C++，那么可以看成是 Cython 的 '阳'。

我们可以从需要高性能的 Python 代码开始，也可以从需要一个优化 Python 接口的 C、C++ 开始，而我们这里是为了学习 Cython，因此显然是选择前者。为了加速 Python 代码，Cython 将使用可选的静态类型声明并通过算法来实现大量的性能提升，尤其是静态类型系统，这是实现高性能的关键。

Cython 和 CPython 的区别?

关于 Cython，最让人困惑的就是它和 CPython 之间的关系，但是需要强调的是这两者是完全不同的。

首先 Python 是一门语言，它有自己的语法规则，我们按照 Python 语言规定的语法规则编写的代码就是 Python 源代码。但是源代码只是一个或多个普通的文本文件，我们需要使用 Python 语言对应的解释器来执行它。

而 Python 解释器也会按照同样的语法规则来对我们编写的 Python 源代码进行分词、语法解析等等，如果我们编写的代码不符合 Python 的语法规则，那么会报出语法错误，也就是 SyntaxError。如果符合语法规范的话，那么会顺利地生成抽象语法树（Abstract Syntax Tree，简称 AST），然后将 AST 编译成指令集合，也就是所谓的字节码（bytes code），最后再执行字节码。

所以 Python 源代码是需要 Python 解释器来操作的，如果我们想做一些事情的话，光写成源代码是不行的，必须要由 Python 解释器将我们的代码解释成机器可以识别的指令进行执行才可以。而 CPython 正是 Python 语言对应的解释器，并且它也是官方实现的标准解释器，同时还是使用最广泛的一种解释器。基本上我们使用的解释器都是 CPython，也就是从官网下载、然后安装之后所得到的。

标准解释器 CPython 是由 C 语言实现的，除了 CPython 之外还有 Jython（java实现的 Python 解释器）、PyPy（Python 语言实现的 Python 解释器）等等。总之设计出一门语言，还要有相应的解释器才可以；至于编译型语言，则是对应的编译器。

最后重点来了，我们说 CPython 解释器是由 C 实现的，它给 Python 语言提供了 C 级别的接口，也就是熟知的 Python/C API。比如：Python 的列表，底层对应的是 PyListObject；字典则对应 PyDictObject，等等等等。

所以当我们在 Python 中创建一个列表，那么 CPython 在执行的时候，就会在底层创建一个 PyListObject。因为 CPython 是用 C 来实现的，最终肯定是将 Python 代码翻译成 C 级别的代码，然后再变成机器码交给 CPU 执行。

而 Cython 也是如此，Cython 代码也要被翻译成 C 代码，然后 C 代码再变成扩展（本质上也是机器码），导入之后直接执行，而无需动态解释。因此 Cython 是一门语言，它并不是Python 解释器的另一种实现，它的地位和 CPython 不是等价的，不过和 Python 是平级的。

总结：Cython 是一门语言，可以通过 Cython 源代码生成高效的 C 代码，再将 C 代码编译成扩展模块，同样需要 CPython 来进行调用。

以上我们就解释了什么是 Cython，以及为什么需要 Cython。下一篇文章我们来比较一下，Cython, Python, C 扩展, 还有原生的 C 语言之间的效率差异。

所以这又是一个新系列，通过一点点地深入了解，你会发现 Cython 的魅力。