**系统代码说明**

1. 系统整体流程

这个系统的代码围绕着四个类展开：

BioNERDocument

BioNERSentence

BioNERToken

BioNEREntity

从名字上就可以看出，它们分别代表着一个文档，一个句子，一个token，一个识别出来的名字实体。它们的关系如图所示：

Document

Abstract

Full text

Sentence

Sentence

Sentence

Sentence

Sentence

Sentence

token

token

token

entity

entity

系统的流程是，首先从输入文件中读入数据，建立起上面图中的数据结构，也就是得到了一串BioNERDocument实例，然后将每一个BioNERDocument依次通过一串BioNERProcess接口的实例，在这个过程中，结构上的一些信息将会被增添、修改，得到我们最终要的结果，即BioNEREntity，最后就可以输出了。如下图所示：

BioNERProcess

BioNERDocument

BioNERProcess

BioNERProcess

BioNERDocument

……

输入

输出

请看包bioner.application.api，这里体现了上述的主流程。BioNERApplication实现了上述流程。它需要其他三个接口：

BioNERDocumentBuilder：负责从输入构建前面所说的BioNERDocument结构。

BioNERProcessFactory：负责选择使用哪些BioNERProcess实例，并将它们按顺序串好。

BioNERDocumentOutput：负责将BioNERDocument结构输出。

下面通过一个例子来具体讲解这个流程，同时也演示一下如何在这个系统中添加一个具体的应用。

这个例子是BioCreative II的Gene Metion任务。目标是要对5000个句子进行基因蛋白质的NER。这部分代码在包bioner.application.bc2gm

第一步，实现BioNERDocumentBuilder接口。

请看BC2GMDocumentBuillder.java。这里的任务是把5000个句子读进来，组成一组BioNERDocument。这里遇到一个问题：数据是单独的句子，并不是一个真正意义上的Document。我们就把每一个句子当成一个文档，组装成5000个“文档”。在这一步我们只需要把句子这一级做好，token先不用管，entity更是还没有。这里由于数据已经是句子了，所以不需要分句。如果需要分句的话，可以用bioner.tools.nlptools包里提供的一些NLP工具。

第二步，实现BioNERProcessFactory接口。

请看BC2GMProcessFactory.java。这里要组成一个BioNERProcess的生产线（pipeline），其实就是返回一个BioNERProcess的数组。为了完成GM任务，我们需要下面这几道工序：

pipeline[0] = new ProcessImpPreprocess();

预处理一下。从句子中拆出token，并且标准化。

pipeline[1] = new ProcessImpProteinNER();

使用字典匹配的方法进行蛋白质识别。通过这一步，我们的句子上终于有了一些entity。

pipeline[2] = new ProcessImpEntityFilter();

对entity过滤一下。字典匹配的结果毕竟有些不太靠谱。

pipeline[3] = new ProcessImpSetLable();

将字典匹配的结果变成BIO标签，标在每一个token上。这是为CRF做准备。

pipeline[4] = new ProcessImpSetLabelByCRF();

使用CRF，对每个token抽取feature，进行BIO标签的标注。

pipeline[5] = new ProcessImpRightBoundAdjust();

对CRF得到的标签结果做一些右边界的调整，像cyclin a,b and c这样的。

pipeline[6] = new ProcessImpGetEntityFromLabel();

从CRF的结果标签中得到entity。这会扔掉之前已有的entity，只从BIO标签上获得。

pipeline[7] = new ProcessImpPostProcessEntityFilter();

对结果进行一些后处理，过滤一些常见错误等等。

通过这些工序以后，我们的一组BioNERDocument上就存有了我们想要的结果，也就是那些BioNEREntity。

第三步，实现BioNERDocumentOutput。

请看BC2GMDocumentOutput.java。这里我们要实现三个函数。Init()和close()分别会在程序开始和结束时被运行一次。由于我们要把结果打印在文件里，所以我们在这两个函数里分别写上打开和关闭文件句柄的内容。如果你不需要这两个函数，让它们空着就行了。

在outputDocument函数中，我们从Document里得到它所有的entity，按照BioCreative II的格式要求打印到文件里面。由于这个方法在每个线程对每一个Document处理结束后立即被调用，所以我们用synchronized关键字把这个函数锁上，避免多线程中出错。

第四步，写运行的主类。

请看BC2GMTaskRun.java。在main中，第一行，让程序读一下配置文件，主要是要线程数。下面三行，将我们上面写的三个类实例化。然后把它们交给BioNERApplication，再运行run()，就完成了。

在BioNERApplication中，它会用BioNERDocumentBuilder得到所以文档类的实例，用BioNERProcessFactory得到处理的流水线，然后根据配置文件中的线程数，开启多个线程，每次为一个线程分配一个文档，这样多个线程平行处理。每个线程里，会按照流水线将文档过一遍，完成后用BioNERDocumentOutput输出。

可以看出，最重要的一个步骤是第二步，配置流水线。这里可以用一些已经写好的Process实例，如果你需要的工序目前程序里还没有，那么只能自己写了，实现BioNERProcess接口。具体哪些工序已经有了，还有哪些工具可以用，请看后面的详细说明。

1. 各部分详细说明