问题描述

1. 基本描述：
   1. 输入：PDF文档，输出：HTML文档
   2. 环境：python2.7，依赖pdfMiner (<http://www.unixuser.org/~euske/python/pdfminer/>)
   3. 原定计划：
      1. Step 1: 从可直接解析的PDF文件中提取文字内容、识别并且提取表格、识别并且提取目录信息(包含跳转功能)写入HTML；
      2. Step 2: 解析不可直接解析的PDF文件，基本思路是先转化为图片再进行图像处理
   4. 目前阶段：基本完成可提取内容的PDF的文字、表格、Outline提取。表格部分有些特殊输入会导致出错。
2. 子问题描述
   1. 问题概览：（及子问题编号）
      1. 可直接解析的PDF
   2. 文字内容：内容以及基本的排版布局
   3. 表格内容：位置，结构，内容
   4. 目录内容：目录位置，目录内容（以及层级结构），链接指向位置
      1. 不可直接解析的PDF
   5. 转化成图片
   6. 文字内容：从图片中读取文字的位置和内容
   7. 表格信息：从图片中提取特征分析文档的对应页码中的图表
   8. 目录：想办法根据某些规则处理目录
   9. 已经解决：
      1. 问题1.1
         1. 解决方式：假设一个前提：所有文档的普通文字内容都是单列的，有段首缩进，且一页之内段首缩进只有一种；在此基础上，将每一行的段首位置予以分类，以此为依据区分出段落，并佐以长度判断、位置判断（如：居中、左对齐短于普通文字的一行宜单独处理试做特殊情况而非普通正文）等规则进行限定试图最大限度还原原文状态
         2. 注意事项：目前适用于大部分满足前提假设的文档，但是在特殊情况，比如处理目录的时候容易将之视作同一段落；在目录合乎格式的时候可以按照针对目录的规则进行特殊化的处理，但是如果目录并没有作为PDF中的目录项，就很难排版正确
         3. 测试样例：simple1, simple2, simple3
      2. 问题1.3
         1. 解决方式：使用pdfMiner自带的Outlines提取功能提取Outlines，可以经由分析得到目录的各项标题名称、层级缩进、对应页码目标（抽样观察到的PDF文件中索引大多精确到页）
         2. 注意事项：格式不统一，注意观察pdfMiner解析出的目录相关信息是“个别含有目录的文档有此项属性”还是“只要有目录在就一定有此项属性”；这里光读源码不够，因为解析内容几乎直接来源于pdf输入文件，有时需要参照一下pdf的语法
         3. 测试样例：outline\_example\_1
      3. 问题2.1
         1. 解决方式：已证实ImageMagick可行
            1. <http://www.imagemagick.org/script/index.php>
            2. <http://blog.csdn.net/royal_coffee/article/details/1602933>
         2. 注意事项：注意python版本，操作系统版本……错了可能会有麻烦；此外，也许有别的类似的途径可以转pdf为图片；也许linux或者mac都会比windows方便(因为windows里convert有特殊含义)
         3. 测试样例：aaa.PDF -> aaa-0.png, aaa-1.png, aaa-2.png
      4. 问题1.2 （基础部分——基本表格，加上形态简单的含有虚线的表格）
         1. 解决方式：
            1. Step 1: 遍历当前页码的内容，提取构成表格的图形元素(点、线、长方形)，同时估计一下本页表格的最粗边框宽多少
            2. Step 2: 集点成线，把虚线部分连起来变成实线
            3. Step 3: 表格大致分区(根据x, y范围)，便于后续处理
            4. Step 4: 针对于每一个表区里的表格（一般就一个），统计其元素的x,y值并且稍加规范（合并相近的x,y）
            5. Step 5: 合并一些冗余的线段，去掉一些多余的节点，只保留必要的点和线
            6. Step 6: 将必要的线根据必须保留的点的x, y值进行分割
         2. 注意事项：表格特别复杂，形态堪称乱七八糟，建议多看几家公司的表格；照顾奇葩表格的同时也请记住兼容正常表格……有可能因为调参等原因破坏了正常表格的结果；非到万不得已不用对代码大兴土木，先调整一下bias\_param试试看，很可能问题的根由就是这个参数设置不合理
         3. 测试样例：table\_example\_1, table\_example\_2, … table\_exampe\_8
   10. 正在解决：
       1. 问题1.2（困难部分）- 有些表格并没有边框
          1. 解决方案：（拟）以纵向间隔代表格纵向分隔线；事实上，没有边框的表格最难区分的不是列，而是行，不若暂且不分行，以间隔为准，以竖直间隔区域的两端作为横向分隔线的横坐标……
          2. 注意事项：纵向、横向分隔线不宜太宽，否则影响bias的设置，导致容错率太高……
          3. 测试样例：table\_example\_19
       2. 问题1.2（遗留问题）- 有些表格（主要是虚线表格）会导致出错
          1. 解决方案：暂无，尚未确定错误区间
          2. 注意事项：
          3. 测试样例：table\_example\_18
   11. 尚未着手：
       1. 问题1.3的延伸：如果目录在原PDF文件中并非目录，而是普通的文本，甚至可能不曾插入链接，怎么办？
          1. 解决方式：（计划中）- 根据特征提取
          2. 注意事项：预计会出现很多false positive
          3. 测试样例：outline\_example\_2, outline\_example\_3
       2. 问题2.2：不曾开始
       3. 问题2.3：不曾开始
       4. 问题2.4：不曾开始

参考资料：见note.txt