多数据源的真值发现与数据融合

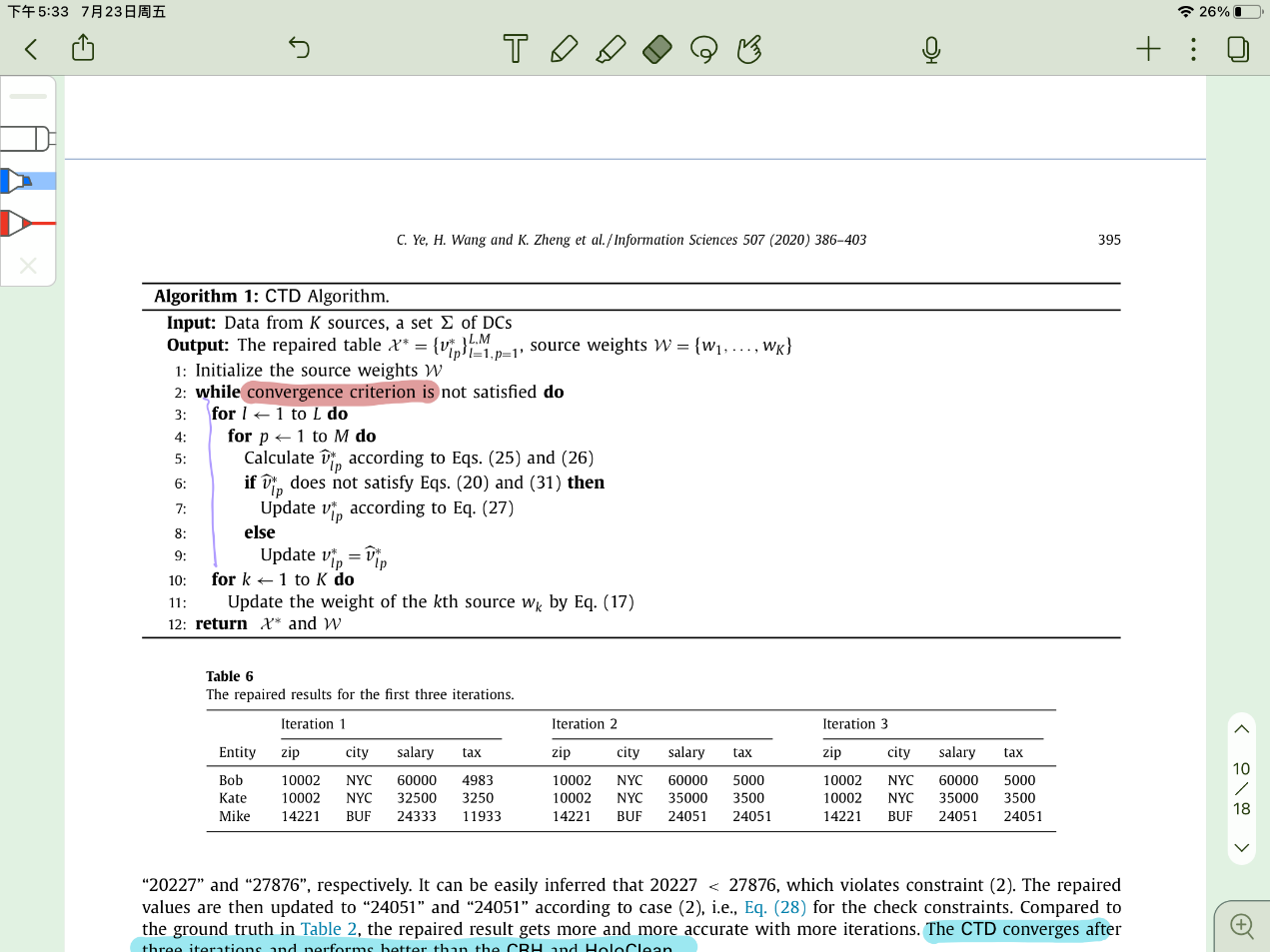
**一．问题难点：**

针对多数据源的真值发现问题的工作难点和重点在于对约束规则的限制 ，目前我们提供了如下规则，进行DC限制：

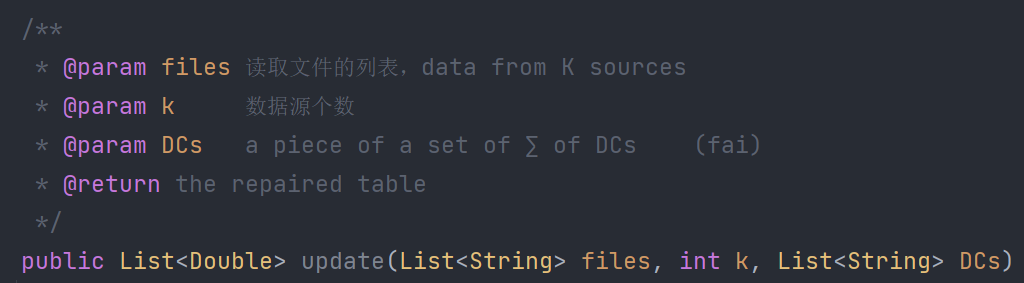
1. 单一实体的值域约束：例如code<10，适用于每一个tuple
2. 单一实体的关系约束：例如，一个人消费支出不可能高于总收入，即output <= input
3. 多实体的DC：例如：Tuple 1’s salary < tuple 4’s wage，这是我们提供的泛型，也就是说，在多实体的DC中，我们允许用户输入tuple的行，指定entity是谁，同时指定我们比较的属性。这里注意，属性可能相同，也可能是不同的。

**二．算法定义与介绍**

具体实现方面，我们参考如下算法进行简介：



我们对于主干的CTD算法进行规约如下所示：



算法第一行，对于数据源的权重进行初始化，本方法采取最普通的平摊方式，对于k个数据源，每个初始化为1/k，然后读取文件列表中的数据，也就是k个数据源，并由此获取数据的L行tuple以及p个属性，注意，这里weights，L，p都是成员变量，可供后续使用。

算法第二行，对于收敛准则的可满足性进行了讨论，这里的收敛，指的是我们最终经过更新，修复得到的，k个数据源进行数据融合后的表的变化情况。我们用result数组表示这个最终呈现的表。采用UDF技术，对于表中数据变化情况进行了分析衡量。在MyUDF类中，实现了对于字符串的比较匹配，调用evaluate方法就可以看到匹配程度（0-1），在这里我们保留上次的result存入preresult中，然后和现有的result，每个元素进行比较，输出总体的error偏差，当误差缩减到一定程度时，认定该result已经收敛，停止更新。

Result中每个vlp的求法：按照公式给出，分为连续型（数字）和离散型。两种类型的区分利用正则表达式，匹配所有类型的数字，支持以下类型：1，1.2，1e2，1e-2，1.2e3等等，以及对应的各种负数也都支持。如果匹配，说明是连续类型。这种连续和离散的区分在后面的update中有要求，其中公式28，29要求是连续的，对于单一实体而言；而公式35要求是离散的，并且仅针对操作符为=进行更新。

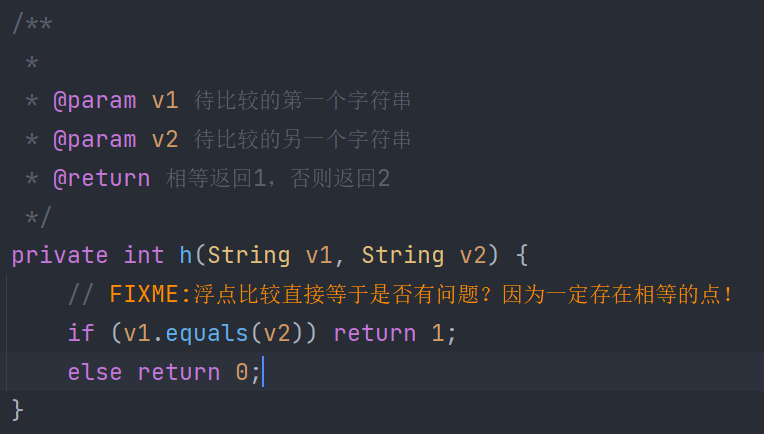
将数据计算存入result之后，查看是否满足公式20，31的要求。首先按照公式20进行计算，如果发现，开头不是属性（因为多属性开头是tuple），说明是多属性，利用两层的break，跳转到multi\_entity函数进行处理，同时传入result和value，以及处理过的dc约束。处理过的是指将普通dc字符串，去掉空格分割。

我们利用20中的约束进行分析，看约束是否满足至少有一个为真。然后按照值域约束和关系约束进行分别处理。核心思想就是尽量少的修改，实际操作时候看这两个列那个列先被访问，就改它。

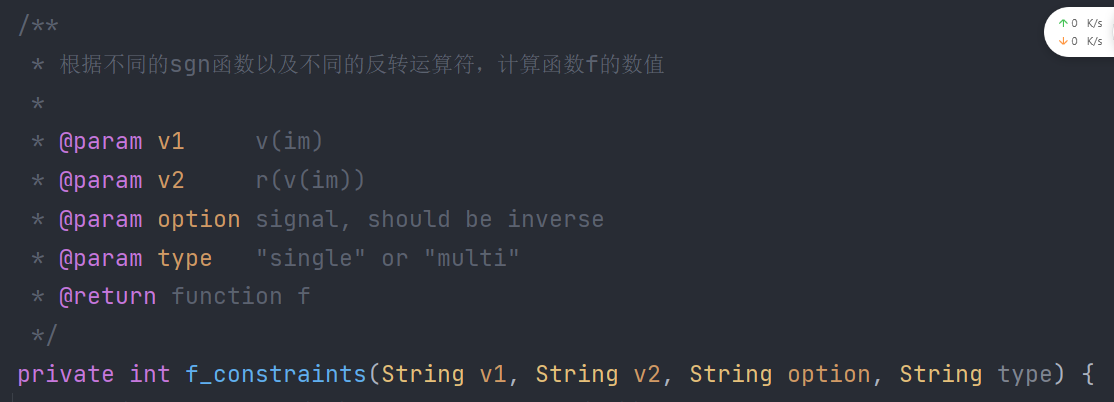
最后的权重更新计算按照公式即可，注意这里有d函数，计算距离，需要和embedding相连接。

**三．辅助计算函数介绍**

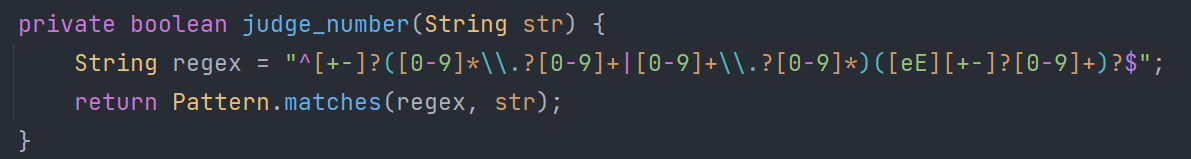
1. 其中FIXME是可能存在的问题



1. 这里的type参数没有用到，设计用于区分多实体和单实体，但是这样需要吗（？



1. 普通的sgn函数，和论文中一样。
2. Judge\_number，利用正则表达式判断是否是数字：



1. 计算距离的distance函数，待实现。
2. 多实体的更新操作