算法宏包 algorithm2e 简单实用样例

LaTeX 排版过程中经常会遇到对算法或伪代码进行排版的问题, 其实对于这方面的处理也是比较方便的. 接下来我们通过几个具体的算法示例对该类问题进行一定的介绍.

首先, 常用到的算法宏包并不多, 例如 algorithm2e、algorithm、algorithmicx 等, 可以说各有各的优势, 有时也可以配合使用, 下面的示例中我们以algorithm2e为主进行讲解.

首先给出几个算法示例的实现效果:

```
1 算法: MYSTERY(n) r := 0 ;
2 for i := 1 to n − 1 do
3     for j := i + 1 to n do
4         for k := 1 to j do
5             r := r + 1;
6 return r;
```

```
1 算法: PERSKY(n) r := 0 ;
2 for i := 1 to n do
3     for j := 1 to i do
4         for k := j to i + j do
5             r := r + 1;
6 return r;
```

```
1 算法: PRESTIFEROUS(n) r := 0 ;
2 for i := 1 to n do
3     for j := 1 to i do
4         for k := j to i + j do
5             for l := 1 to i + j − k do
6                 r := r + 1;
7 return r;
```

```
1 算法: CONUNDRUM(n) r := 0 ;
2 for i := 1 to n do
3     for j := i + 1 to n do
4         for k := i + j − 1 to n do
5             r := r + 1;
6 return r;
```

这几个算法示例中, 主要使用了宏包中的行编号功能, 以及通过竖线形式对环境块进行划分的功能. 为了实现该效果, 我们可以在导言区传递`linesnumbered, ruled, vlined`等参数给所调用宏包, 即可实现相应效果.

```
\usepackage[linesnumbered, ruled, vlined]{algorithm2e}
```

该宏包中包含的可选参数也很多，例如与`vlined`相关的`lined`、`noline`等，可以绘制不带弯角的直线分组或不绘制竖线.

与`linesnumbered`相关的`linesnumberedhidden`、`commentsnumbered`等, 可以隐藏行编号或为注释添加行编号.

该算法中主要是 for 循环的嵌套, 仅有一种形式, 操作和理解都相对简单, 具体的使用方法为`\For{condition \KwTo condition}{for-block}`.

类似的可以, while 循环`\While{condition}{while-block}`, 条件判断语句`\If{condition}{then-block}`等也可以很轻松的实现. 例如:

1 算法: $r := 0$;
2 **while** $n < 1000$ **do**
3 **if** $j < n$ **then**
4 $r := r + 1$;
5 **return** r;

其中常见的关键词在该宏包中也已经有了定义, 例如`\KwIn{input}`、`\KwOut{output}`分别表示输入和输出. `\KwRet{[value]}`, `\Return{[value]}`表示返回.

更加丰富的预定义命令可以参见 algorithm2e 宏包的帮助文档进行查询. 这里我们仅给出部分常用命令

- \Begin{block inside}
- \Begin(*begin comment*){block inside}
- \If{condition}{then block}
- \If(*then comment*){condition}{then block}
- \uIf{condition}{then block without end}
- \uIf(*then comment*){condition}{then block without end}
- \lIf{condition}{then's line text}
- \lIf(*if comment*){condition}{then's line text}
- \ElseIf{elseif block}
- \ElseIf(*elseif comment*){elseif block}
- \uElseIf{elseif block without end}
- \uElseIf(*elseif comment*){elseif block without end}
- \lElseIf{elseif's line text}
- \lElseIf(*elseif comment*){elseif's line text}
- \Else{else block}

- \Else(*else comment*){else block}
- \uElse{else block without end}
- \uElse(*else comment*){else block without end}
- \lElse{else's line text}
- \lElse(*else comment*){else's line text}
- \eIf{condition}{then block}{else block}
- \eIf(*then comment*){condition}{then block}(*else comment*){else block}
- \eIf(*then comment*){condition}{then block}{else block}
- \eIf{condition}{then block}(*else comment*){else block}
- \leIf{condition}{then block}{else block}
- \leIf(*comment*){condition}{then block}{else block}
- \Switch(*switch comment*){condition}{Switch block}
- \Switch{condition}{Switch block}
- \Case{a case}{case block}
- \Case(*case comment*){a case}{case block}
- \uCase{a case}{case block without end}
- \uCase(*case comment*){a case}{case block without end}
- \lCase{a case}{case's line}
- \lCase(*case comment*){a case}{case's line}
- \Other{otherwise block}
- \Other(*other comment*){otherwise block}
- \lOther{otherwise's line}

- \lOther(*other comment*){otherwise's line}
- \For{condition}{text loop}
- \For(*for comment*){condition}{text loop}
- \lFor{condition}{line text loop}
- \lFor(*for comment*){condition}{line text loop}
- \While{condition}{text loop}
- \While(*while comment*){condition}{text loop}
- \lWhile{condition}{line text loop}
- \lWhile(*while comment*){condition}{line text loop}

接下来看一个更加复杂的示例, 当文件中存在多个算法时, 往往需要对每一个算法示例按顺序编号, 这样也便于我们引用和查找.

算法的环境编号与图、表等形式类似, 仅需要在算法环境中使用`\caption{text.}`即可. 例如算法 2 中我们就对其进行了编号. 也可以通过自定义编号的方式来为算法指定一个编号, 例如通过对 algocf 赋值为 1，则接下来的算法编号则从 2 开始.

```
\setcounter{algocf}{1}
```

为了满足算法对中文输入和输出的支持, 我们可以直接用普通文本的形式输入, 之后进行一定的缩进调整, 另外我们也可以通过对宏包中对该类命令进行重命名以达到相同的目的.

算法 2 中我们就是用了更改输入、输出名的方式

```
\SetKwInput{KwIn}{输入}
\SetKwInput{KwOut}{输出}
```

对于行号的展示我们也做了一定的调整, 有时候我们并不希望对所有的行进行编号, 所以我们可以借助于`\nl`命令进行设置, 仅需要在需要编号的行前面添加该命令即可. 而对于同时存在需要自动编号和需要手动编号的情形时, 我们首先需要使用`\LinesNotNumbered`将自动编号停止，之后方可以使用`\nl`命令来手动设置需要显示编号的哪些行.

如需对编号样式进行调整,可以\SetNlSty{<font>}{<txt before>}{<txt after>}进行调整,在本示例中我们使用了\SetNlSty{}{}{:}命令,通过这种方式在数字后面添加了冒号. 也可以通过该命令,在编号前面添加文字或是更改字体等.

当同一个文档中需要出现不同样式的算法时，例如本文所介绍的算法示例，前面是有竖线进行分组的，而后面我们不希望继续使用这种样式，则可以通过\SetAlgoNoLine的形式进行停止.

算法环境的运用并不一定局限于此,在实际的使用过程中,我们可能会遇到各种各样的问题,这些问题都带有一定的独特性,这就需要我们结合算法宏包中的预定义的命令进行使用或修改,以更好的服务于我们.

算法 2: 经典的二分类 Adaboost 算法

输入:

N 个训练样例 $\langle x_1, y_1\rangle, \langle x_2, y_2\rangle, \cdots, \langle x_N, y_N\rangle$

其中, $x_i \in R, y_i \in -1, +1$ WL $\leftarrow$ 弱学习机

$T \leftarrow$ 迭代次数

$l^t = f(t) \leftarrow$ 第 t 次迭代的损失函数

$D \leftarrow$ 样本的分布

输出: 最终分类器 $G(x) = h_f(x)$

初始化权重向量

$D_1 = (w_1^1, \cdots, w_i^1, \cdots, w_N^1)$

$w_i^1 = 1/N, i = 1, 2, \cdots, N$

输入训练样例集合 $\langle x_1, y_1\rangle, \langle x_2, y_2\rangle, \cdots, \langle x_N, y_N\rangle$

while $t < T$ **do**

设置

$P^t = \frac{w^t}{\sum_{i=1}^{N} w_i^t}$

依据概率 P^t 调用弱学习机 WL

返回假设值 $h_t : X \mapsto [0, 1]$

计算第 h_t 的损失: $\varepsilon_t = \sum_{i=1}^{N} P_i^t |h_t(x_i) - y_i|$

使得 $\beta_t = \varepsilon_t/(1 - \varepsilon_t)$

使得 $\alpha_t = \frac{1}{2} \lg \beta_t$

更新权重分布

$D_{t+1} = (w_1^{t+1}, \cdots, w_i^{t+1}), \cdots, w_N^{t+1}$

$w_i^{t+1} = w_i^t \beta^{1-|h_t(x_i)-y_i|}, i = 1, 2, \cdots, N$

计算最终假设

$h(f) = \text{sign}\{\sum_{i=1}^{T} \alpha_t \text{WL}_t(x)\}$

endwhile