13-D4

串

BM算法：BC策略：性能分析

要不犯十四次，甚至一百四十次错误，就不会得到任何一个真理

邓俊辉
deng@tsinghua.edu.cn

最好情况

❖ $\mathcal{O}(n/m)$ —— 除法？没错！比如：

$$T = \boxed{\texttt{x x x x}\ \boxed{\texttt{1}}}\ \boxed{\texttt{x x x x}\ \boxed{\texttt{1}}}\ \boxed{\texttt{x x x x}\ \boxed{\texttt{1}}}\ \boxed{\texttt{x x x x}\ \boxed{\texttt{1}}}\ \boxed{\texttt{x x x x}\ \boxed{\texttt{1}}}$$

$$P = \boxed{\texttt{0 0 0 0}\ \boxed{\texttt{0}}}$$

❖ 一般地： 只要P不含T[i+j]，即可直接移动m个字符

仅需单次比较，即可排除m个对齐位置

❖ 单次匹配概率越小，性能优势越明显 //大字母表：ASCII、UniCode

❖ P越长，这类移动的效果越明显

最差情况

- $\mathcal{O}(n \times m)$ —— 退化为蛮力算法？是的！比如：

$$T = \boxed{0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ \ldots}$$

$$P = \boxed{1\ 0\ 0\ 0\ 0}$$

- 每轮迭代，都要在扫过**整个**P之后，方能确定右移**一个**字符

  此时，须经**m次**比较，方能排除**单个**对齐位置

- 单次匹配概率**越大**的场合，性能**越接近于**蛮力算法 //小字母表Bitmap + DNA

- 反思：借助以上bc[]表，仅仅利用了**失配**比对提供的信息（**教训**）！

  类比：可否仿照KMP，同时利用起**匹配**比对提供的信息（**经验**）？