DOI:10.19304/j.cnki.issn1000-7180.2013.04.007

KMP算法的理论研究

韩光辉[1]，曾 诚[2,3]

(1 武汉商业服务学院 信息工程系，湖北 武汉 430056；
2 湖北大学 数学与计算机科学学院，湖北 武汉 430062；
3 武汉大学 软件工程国家重点实验室，湖北 武汉 430072)

摘 要：KMP算法是经典的串匹配算法之一.本文首先引入刻划模式串前缀特征的集合K_j及其划分，讨论了其若干性质.然后定义函数f与next，利用f刻划了K_j的构造，由此得到了f的迭代计算方法；证明了next与f之间的关系，从而给出了KMP算法原理的形式表述和数学证明.最后，基于f的迭代计算方法以及next与f之间的关系，给出了算法描述，分析了时间复杂度.

关键词：串匹配；KMP算法；特征集；最大值函数；复杂度分析

中图分类号：TP301.6 **文献标识码**：A **文章编号**：1000－7180(2013)04－0030－04

Theoretical Research of KMP Algorithm

HAN Guang-hui[1], ZENG Cheng[2,3]

(1 Information Engineering Department, Wuhan Commercial Service College, Wuhan 430056, China;
2 College of Mathematics & Computer Science, Hubei University, Wuhan 430062, China;
3 State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China)

Abstract: KMP algorithm is one of classic string matching algorithms. A set K_j, for $1<j\leqslant m$, which characterize a pattern prefix, and its partition are introduced, their some properties are discussed. Then, functions f and next are defined, the structure of K_j is described by f, a iterative computing method of f is proposed, the relationship between functions next and f is proved. Thus, mathematical principles of KMP algorithm is strictly established. Finally, the algorithm is described based on the iterative computing method of f and the relationship between functions next and f, and its complexity is analyzed.

Key words: string matching; KMP algorithm; characteristic set; maximum value function; complexity analysis

1 引言

串匹配是指在文本串中查找模式串的一次或所有出现.串匹配算法在拼写检查、语言翻译、数据压缩、搜索引擎、网络入侵检测、病毒特征码检测、DNA序列匹配等应用中起着关键作用.

设文本串$T=t_1t2\cdots t_n$，模式串$P=p_1p_2\cdots p_m$.根据扫描策略的不同，精确的串匹配算法大致可以分为以下三类：

(1)自前向后匹配模式串前缀[1]；

(2)自后向前匹配模式串后缀[2-3]；

(3)自后向前匹配模式串子串[4].

虽然第一类算法的平均时间复杂度并不理想，但仍是工程应用中的首选算法之一[5-7].

KMP算法[1]是第一类算法中的经典算法，它的原理与方法深刻影响了后来提出的各种串匹配算法.但是，在Knuth D E, Morris J H, Pratt V R的经典文章中，因为缺乏严格的形式表述，尤其是*next*函数缺乏严谨的形式定义和数学证明，使得后来出现了一些不正确的理解和不适当的改进.因此，

收稿日期：2012－07－10；**修回日期**：2012－08－13

基金项目：国家自然科学基金项目(60903034,61100018,61100025,61100026)；湖北省自然科学基金项目(2011CDB069)

建立 KMP 算法严格的理论基础，对于模式匹配领域和有限自动机领域，都是十分必要的.

本文第一次给出了 KMP 算法原理的形式表述和数学证明.

2　特征集 K_j

KMP 算法的基本思想是：自前向后正向逐个扫描文本字符，当

$p_1\cdots p_{j-1}=t_{i-j+1}\cdots t_{i-1}$，$p_j\neq t_i$ 时，

如果存在 $p_1\cdots p_{j-1}$ 的一个前缀 $p_1\cdots p_{k-1}$ 与后缀 $p_{j-k+1}\cdots p_{j-1}$，$1<k<j$，使得

$$p_1\cdots p_{k-1}=p_{j-k+1}\cdots p_{j-1},\ p_k\neq p_j \tag{1}$$

则因

$$p_1\cdots p_{k-1}=p_{j-k+1}\cdots p_{j-1}=t_{i-k+1}\cdots t_{i-1}$$

故下一轮比较从 (p_k,t_i) 开始，显然，这样的 k 应尽可能大. 如果这样的前缀与后缀不存在，则当 $p_1\neq p_j$ 时，下一轮比较从 (t_i,p_1) 开始；当 $p_1=p_j$ 时，下一轮比较从 (t_{i+1},p_1) 开始.

根据上述算法思想，下面引入刻划(1)式特征的集合.

记

$K_j=\{k\mid 1<k<j\wedge p_1\cdots p_{k-1}=p_{j-k+1}\cdots p_{j-1}\}$，$j=2,\cdots,m$

以及

$K_j^{(1)}=\{k\mid k\in K_j\wedge p_k=p_j\}$

$K_j^{(2)}=\{k\mid k\in K_j\wedge p_k\neq p_j\}$

显然，$K_j=K_j^{(1)}\cup K_j^{(2)}$，$K_j^{(1)}\cap K_j^{(2)}=\phi$.

引理 1　$k\in K_j$，则

① $K_k\subset K_j$；

② 若 $k'\in K_j$，$k'<k$，则 $k'\in K_k$；

③ 若 $k=\max K_j$，则 $K_j=\{k\}\cup K_k$.

证①　不妨设 $K_k\neq\phi$，$k'\in K_k$，则因

$p_1\cdots p_{k'-1}=p_{k-k'+1}\cdots p_{k-1}$

$p_1\cdots p_{k-1}=p_{j-k+1}\cdots p_{j-1}$

故

$p_1\cdots p_{k'-1}=p_{k-k'+1}\cdots p_{k-1}=p_{j-k'+1}\cdots p_{j-1}$

于是 $k'\in K_j$，显然 $K_j-K_k\neq\phi$，所以 $K_k\subset K_j$.

证②　由于

$p_1\cdots p_{k'-1}=p_{j-k'+1}\cdots p_{j-1}$

$p_1\cdots p_{k-1}=p_{j-k+1}\cdots p_{j-1}$

又因 $k'<k$，故

$p_1\cdots p_{k'-1}=p_{j-k'+1}\cdots p_{j-1}=p_{k-k'+1}\cdots p_{k-1}$

于是 $k'\in K_k$.

证③　设 $k'\in K_j$，不妨设 $k'<k$，由②，$k'\in K_k$，故 $K_j\subseteq\{k\}\cup K_k$. 另一方面，由①，$\{k\}\cup K_k\subseteq K_j$，所以 $K_j=\{k\}\cup K_k$. 证毕.

引理 2　设 $k\in K_j^{(1)}$，则

① $K_k^{(1)}\subset K_j^{(1)}$；

② 若 $k'\in K_j^{(1)}$，$k'<k$，则 $k'\in K_k^{(1)}$；

③ 若 $k=\max K_j^{(1)}$，则 $K_j^{(1)}=\{k\}\cup K_k^{(1)}$.

证①　不妨设 $K_k^{(1)}\neq\phi$，$k'\in K_k^{(1)}$，因为 $K_k^{(1)}\subseteq K_k\subset K_j$，$p_{k'}=p_k=p_j$，故 $k'\in K_j^{(1)}$，显然 $K_j^{(1)}-K_k^{(1)}\neq\phi$，从而 $K_k^{(1)}\subset K_j^{(1)}$.

证②　由引理 1，$k'\in K_k$，又因 $p_{k'}=p_j=p_k$，故 $k'\in K_k^{(1)}$.

证③　设 $k'\in K_j^{(1)}$，不妨设 $k'<k$，由②，$k'\in K_k^{(1)}$，故 $K_j^{(1)}\subseteq\{k\}\cup K_k^{(1)}$. 另一方面，由①，$\{k\}\cup K_k^{(1)}\subseteq K_j^{(1)}$，从而 $K_j^{(1)}=\{k\}\cup K_k^{(1)}$. 证毕.

引理 3　$K_{j+1}=\{2\mid p_1=p_j\}\cup\{k+1\mid k\in K_j^{(1)}\}$.

证　事实上，

$k'\in K_{j+1}$

$\Leftrightarrow 1<k'<j+1\wedge p_1\cdots p_{k'-1}=p_{j-k'+2}\cdots p_j$

$\Leftrightarrow(k'=2\wedge p_1=p_j)\vee(2<k'<j+1\wedge p_1\cdots p_{k'-1}=p_{j-k'+2}\cdots p_j)$

$\Leftrightarrow(k'=2\wedge p_1=p_j)\vee(k'=k+1\wedge 1<k<j\wedge p_1\cdots p_{k-1}=p_{j-k+1}\cdots p_{j-1}\wedge p_k=p_j)$

$\Leftrightarrow(k'=2\wedge p_1=p_j)\vee(k'=k+1\wedge k\in K_j\wedge p_k=p_j)$

$\Leftrightarrow(k'=2\wedge p_1=p_j)\vee(k'=k+1\wedge k\in K_j^{(1)})$

$\Leftrightarrow k'\in\{2\mid p_1=p_j\}\cup\{k+1\mid k\in K_j^{(1)}\}$

证毕.

3　最大值函数 f 与 next

由引理 1 不难看出，可以通过 $\max K_j$ 求出 K_j 的全部元素. 为此，定义函数 $f:\{1,\cdots,m\}\to\{0,\cdots,m-1\}$，

$$f(j)=\begin{cases}0, & j=1\\ 1, & K_j=\phi\\ \max K_j, & K_j\neq\phi\end{cases}$$

由 f 的定义，$f(1)=0$，$f(2)=1$，并且由引理 3 易知

$$f(j+1)=\begin{cases}1, & K_j^{(1)}=\phi\wedge p_1\neq p_j\\ 2, & K_j^{(1)}=\phi\wedge p_1=p_j\\ 1+\max K_j^{(1)}, & K_j^{(1)}\neq\phi\end{cases} \tag{2}$$

定理 1 设 $K_j \neq \phi$，则存在 k_r 使得

$f(j) = k_1, f(k_1) = k_2, \cdots, f(k_r) = 1$

并且

$K_j = \{k_1, k_2, \cdots, k_r\}$.

证：因为 $k_1 = f(j) = \max K_j$，由引理 1，$K_j = \{k_1\} \cup K_{k_1}$. 若 $K_{k_1} \neq \phi$，又因 $k_2 = f(k_1) = \max K_{k_1}$，故 $K_{k_1} = \{k_2\} \cup K_{k_2}$，从而 $K_j = \{k_1, k_2\} \cup K_{k_2}$. 如此下去，必有 k_r 使得

$K_j = \{k_1, k_2, \cdots, k_r\} \cup K_{k_r}$, $K_{k_r} = \phi$

由 f 的定义，$f(k_r) = 1$，并且 $K_j = \{k_1, k_2, \cdots, k_r\}$. 证毕.

由定理 1 与式(2)，可以从 $f(1), \cdots, f(j)$ 递推计算 $f(j+1)$. 计算方法是：迭代计算

$f(k_v) = k_{v+1}$, $v = 0, 1, \cdots, r$, $k_0 = j$, $k_{r+1} = 1$

① 若 $p_{k_1} \neq p_j, \cdots, p_{k_{v-1}} \neq p_j, p_{k_v} = p_j$, $v \leqslant r$，则

$f(j+1) = f(k_{v-1}) + 1 = k_v + 1$

② 若 $p_{k_1} \neq p_j, \cdots, p_{k_r} \neq p_j, p_{k_{r+1}} = p_j$，则

$f(j+1) = f(k_r) + 1 = k_{r+1} + 1 = 2$

③ 若 $p_{k_1} \neq p_j, \cdots, p_{k_r} \neq p_j, p_{k_{r+1}} \neq p_j$，则

$f(j+1) = f(k_{r+1}) + 1 = 0 + 1 = 1$

现在，我们定义函数 next：$\{1, \cdots, m\} \to \{0, \cdots, m-1\}$，

$$\text{next}(j) = \begin{cases} 0, & (j=1) \vee (K_j^{(2)} = \phi \wedge p_1 = p_j) \\ 1, & K_j^{(2)} = \phi \wedge p_1 \neq p_j \\ \max K_j^{(2)}, & K_j^{(2)} \neq \phi \end{cases}$$

由 KMP 算法的基本思想看出，当 $\text{next}(j) = \max K_j^{(2)}$ 时，$\text{next}(j)$ 就是(1)式中的 k；当 $\text{next}(j) = 1$ 时，表明下一步比较 (t_i, p_1)；当 $\text{next}(j) = 0$ 时，则表明下一步比较 (t_{i+1}, p_1).

定理 2 $\text{next}(j) =$

$$\begin{cases} f(j), & p_{f(j)} \neq p_j \\ \text{next}(f(j)), & p_{f(j)} = p_j \end{cases}, \quad j = 2, \cdots, m.$$

证：当 $K_j = \phi$ 时结论显然成立. 设 $K_j \neq \phi$，由引理 1，

$K_j = \{f(j)\} \cup K_{f(j)} = \{f(j)\} \cup K_{f(j)}^{(1)} \cup K_{f(j)}^{(2)}$

若 $p_{f(j)} \neq p_j$，则 $f(j) \in K_j^{(2)}$，故

$\text{next}(j) = \max K_j^{(2)} = f(j)$

若 $p_{f(j)} = p_j$，那 $f(j) \in K_j^{(1)}$，由引理 2，$K_j^{(1)} = \{f(j)\} \cup K_{f(j)}^{(1)}$，则必有 $K_j^{(2)} = K_{f(j)}^{(2)}$，从而

$\text{next}(j) = \text{next}(f(j))$

证毕.

4 算法描述

根据 $f(j)$ 的迭代计算方法与定理 2，下面给出 $\text{next}(j)$ 的算法. 其中，步骤(1) ~ (7)计算 f 的函数值，步骤(8) ~ (11)计算 next 的函数值，因为? (j)是中间结果，不需要保留，所以 $f(j)$ 与 $\text{next}(j)$ 共用一个存储单元.

算法 1 计算 next 的函数值

输入：模式串 $P = p_1 p_2 \cdots p_m$.

输出：$\text{next}(j), j = 1, 2, \cdots, m$.

步骤：

(1) $\text{next}(1) \leftarrow 0$；

(2) $j \leftarrow 1$；

(3)若 $j < m$ 则转(4)，否则转(8)；

(4) $k \leftarrow \text{next}(j)$；

(5)若 $k > 0, p_k \neq p_j$，则 $k \leftarrow \text{next}(k)$；重复(5)，直到 $k = 0$ 或 $p_k = p_j$；

(6) $\text{next}(j+1) \leftarrow k + 1$；

(7) $j \leftarrow j + 1$，转(3)；

(8) $j \leftarrow 2$；

(9)若 $j \leqslant m$ 则转(10)，否则结束；

(10)若 $p_{\text{next}(j)} = p_j$，则 $\text{next}(j) \leftarrow \text{next}(\text{next}(j))$；

(11) $j \leftarrow j + 1$，转(9).

上述算法 1 与文献[1]中的算法稍有不同，主要区别在于，算法 1 中 f 函数值的计算与 next 函数值的计算是分开进行的，这样会更清晰一些，并且未增加算法的复杂性.

计算出 next 的函数值以后，就可以利用 next 进行串匹配，算法如下：

算法 2 串匹配

输入：文本串 $T = t_1 t_2 \cdots t_n$，模式串 $P = p_1 p_2 \cdots p_m$，$\text{next}(j)$ $(j = 1, 2, \cdots, m)$.

输出：匹配成功时输出 T 中首个匹配字符的位置，匹配失败时输出 FAILURE.

步骤：

(1) $i \leftarrow 1, j \leftarrow 1$；

(2)若 $j \leqslant m, i \leqslant n$，则转(3)，否则转(5)；

(3)若 $j > 0, t_i \neq p_j$，则 $j \leftarrow \text{next}(j)$；重复(3)，直到 $j = 0$ 或 $t_i = p_j$；

(4) $i \leftarrow i + 1, j \leftarrow j + 1$，转(2)；

(5)若 $j > m$ 则返回 $i - m$，否则返回 FAILURE，结束.

5　复杂度分析

分析预处理算法 1 的时间复杂度. 考虑循环步骤(2) ～ (7)，其循环次数为 $m-1$. 显然，结果为 $p_k = p_j$ 的比较次数$\leqslant m-1$. 对于结果为 $p_k \neq p_j$ 的比较，观察变量 k：$p_k \neq p_j$ 的每次比较导致 k 减小；而步骤(4)的第一次执行使得 k 的初值为 0，其后的每次执行导致 k 增加 1，步骤(4)共执行了 $m-1$ 次，注意到 $k \geqslant 0$，故 $p_k \neq p_j$ 的比较次数$\leqslant m-1$，从而步骤(2) ～ (7)中字符比较的次数$\leqslant 2(m-1)$. 循环步骤(8) ～ (11)中字符比较的次数显然为 $m-1$，所以算法 1 中字符比较的次数$\leqslant 3(m-1)$，即算法 1 的最差时间复杂度为 $O(m)$.

分析扫描算法 2 的时间复杂度，不妨设 $m < n$. 显然，结果为 $t_i = p_j$ 的比较次数$\leqslant n$；对于结果为 $t_i \neq p_j$ 的比较，观察变量 j：$t_i \neq p_j$ 的每次比较导致 j 减小；而步骤(4)的每次执行导致 j 增加 1，步骤(4)最多执行了 n 次，注意到 $j \geqslant 0$，故 $t_i \neq p_j$ 的比较次数$\leqslant n$，于是算法 2 中字符比较的次数$\leqslant 2n$，即算法 2 的最差时间复杂度为 $O(n)$.

以上分析只是粗略的，利用 Fibonacci 串，可以证明算法 2 中字符比较的次数$\leqslant n + 1 + \log_{\varphi} m$[1]，其中$\varphi = (1+\sqrt{5})/2$.

6　结束语

本文引入刻划模式串前缀特征的集合 K_j 及其划分 $K_j^{(1)}$ 与 $K_j^{(2)}$，讨论了它们的若干性质. 然后，定义函数 f 与 next，给出了两个重要结果；定理 1 利用 f 刻划了 K_j 的构造，由此得到了 f 的迭代计算方法；定理 2 揭示了 next 与 f 之间的关系，从而严格建立了 KMP 算法的数学原理. 最后，基于 f 的迭代计算方法与定理 2，给出了 KMP 算法描述，证明了其最差时间复杂度为 $O(m+n)$.

本文的研究思想与基本方法可以运用到串匹配的其它两类算法的理论研究. 例如，BM 算法中的函数 rpr(j)[2]就可以采用与 K_j 以及 f 完全类似的方法给出形式定义并加以研究.

参考文献：

[1] Knuth D E, Morris J H, Pratt V R. Fast pattern matching in strings[J]. SIAM Journal of Computing, 1977, 6(2): 323－350.

[2] Boyer R S, Moore J S. A fast string searching algorithm[J]. Communications of the ACM, 1977, 20(10): 762－772.

[3] Sunday D M. A very fast substring search algorithm [J]. Communications of the ACM, 1990, 33(8): 132－142.

[4] Raffinot M. On the multi backward dawg matching algorithm(multibdm)[M]. the 4th South American Workshop on String Processing, 1997: 149 - 165.

[5] 刘萍，刘燕兵，郭莉，等. 串匹配算法中模式串与文本之间关系的研究[J]. 软件学报，2010，21(7)：1503－1514.

[6] 曾诚，李兵，何克清. KMP 算法在 Web 服务语义标注中的应用[J]. 微电子学与计算机，2010，27(8)：1－3.

[7] 戈晓斐，黄竞伟，胡磊. 改进的 KMP 算法在生物序列模式自动识别中的应用[J]. 计算机工程，2004，30(10)：140－142.

作者简介：

韩光辉　(1956－)，硕士，副教授. 研究方向为计算理论、软件形式化.

曾　诚　(1976－)，博士，副教授. 研究方向为网络化软件工程.