一种带有通配符和长度约束模式匹配问题的动态剪枝算法

王海平[1] 戴 玮[2] 郭 丹[1]

(合肥工业大学计算机与信息学院 合肥 230009)[1] (陆军军官学院基础部 合肥 230031)[2]

摘 要 近年来,随着生物信息学、信息检索等领域的发展,串模式匹配问题被不断扩展。其中,具有代表性的是在模式中引入可变长度的通配符而形成带有通配符的模式匹配(PMWL)。该问题定义的灵活性给用户提供了方便,却也造成了求解上的困难。因此,如何在多项式时间内得到更好的匹配解成为研究的焦点。提出了一种启发式的小兵算法。小兵算法通过将PMWL问题转化为路径搜索问题,并借鉴动态剪枝思想,在算法搜索的过程中动态地将不可能的匹配位置剪枝,从而提高解的质量。实验在真实DNA序列上进行,并人工生成了196个模式。结果表明,相比于目前最有效的SAIL算法,小兵算法在绝大多数的尾部有重复字符的模式中可以获得更好的匹配解。

关键词 模式匹配,通配符,剪枝,约束

中图法分类号 TP301 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.4.050

Dynamic Pruning Algorithm for Pattern Matching with Wildcards and Length Constraints

WANG Hai-ping[1] DAI Wei[2] GUO Dan[1]

(School of Computer Science and Infomation, Hefei University of Technology, Hefei 230009, China)[1]

(Department of Basic, Chinese Peoples Liberation Army Academy, Hefei 230031, China)[2]

Abstract Recently, with the development of bioinformatics, network security and information retrieval, wildcards are playing an increasingly key role in pattern ma-tching problems. Among these, the challenging problem of pattern matching with wildcards and length constraints (PMWL) further extends the flexibility of pattern matching, thus bringing convenience to the user but producing the expense of higher computational complexity. Therefore, how to get a better matching solution in polynomial time has been well studied and developed. Based on the previous work, we presented a heuristic Pawn algorithm for PMWL. After transforming the PMWL problem into path search problem, a dynamic strategy is adopted with a pruning procedure in an alternating iterative manner. Experiments demonstrate that, comparing with the state-of-the-art algorithm, Pawn algorithm can improve the number of matching occurrences in most cases of 196 artificial patterns with certain characteristic and the DNA sequences.

Keywords Pattern matching, Wildcard, Pruning, Constraints

1 引言

模式串匹配问题作为计算机科学的基本问题之一,是网络安全、信息检索与过滤、计算生物学等重要领域的核心问题[1],同时也是模式挖掘技术的核心与基础[2]。通配符的引入使得匹配问题更加灵活[3],也更加符合实际应用的需要。给定字母表 Σ,通配符 ϕ 可以匹配 Σ 中的任意字符。带有通配符的字符串匹配问题最主要出现在生物信息学[4]、查询XML文档[5]和模式挖掘[6]。目前,在带有可变长度通配符的模式匹配问题的研究中[7-15],Chen 等[9]定义了带有通配符和长度约束的模式匹配问题(Pattern Matching with Wildcards and Length constraints, PMWL),模式 P 中任意两个相邻字符间带有一段通配符序列 ϕ,且序列长度满足约束 $\phi[l_i, u_i]$,可记为 $P=p_1\phi[l_1,u_1]\cdots p_{m-1}\phi[l_{m-1},u_{m-1}]p_m$,约束区间长度 u_i-l_i+1 为通配符跨度 gap_i。此外,同一文本字符至多只能被一个匹配解使用,记为 one-off 约束条件。例如,给定文本 $T=aaggcc$,模式 $P=a\phi[0,2]g\phi[0,2]c$,可得匹配位置为 {{1,3,5},{1,3,6},{1,4,5},{1,4,6},{2,3,5},{2,3,6},{2,4,5},{2,4,6}},则在 one-off 约束下,可得匹配解集为{{1,3,5},{2,4,6}}。

从求解的方法上,Fischer 和 Paterson 最早使用了 FFT 方法,其在多项式时间内能得到完备解,然而他们问题中的通配符是一个固定的长度[3];Chen 等提出的 SAIL 算法是目前处理 PMWL 问题最具代表性的方法[9],SAIL 算法最早提出用 left-most 贪心匹配策略。该策略的特点是将文本资源优先分配给位置靠前的候选匹配解。在模式中无重复字符时该策略能得到完备解[14],而当模式中出现重复字符特别是尾部出现重复字符时 SAIL 算法尚不完备。

到稿日期:2014-05-20 返修日期:2014-07-22 本文受国家自然科学基金:港澳学者合作研究基金项目(61229301),国家自然科学基金项目(60828005),博士后面上基金项目(2012M511403),安徽省自然科学基金(2013AKZR0082)资助。

王海平(1986—),男,博士生,主要研究方向为模式匹配等,E-mail:hpwang.hfut@gmail.com;戴 玮(1963—),男,副教授,主要研究方向为实验教学等;郭 丹(1961—),女,讲师,主要研究方向为人工智能等,E-mail:guodan@hfut.edu.cn。

本文针对 PMWL 问题中尾部带有重复字符的模式，力求改进匹配解的质量。本文在 SAIL 的基础上引入动态剪枝的思想，提出一种启发式算法——小兵算法。算法思想是将模式匹配问题转化为在棋盘上小兵的路径搜索问题，且在小兵行进过程中，通过不断更新棋盘上的位置元素，对小兵可能行进的路径进行动态剪枝，以减小搜索空间；同时，通过设置 4 条规则，规范小兵行进方向，特别是当不同小兵之间出现位置抢占时，合理地分配资源，以满足 one-off 约束条件。实验在真实的 DNA 序列文本上进行，模式尾部带有重复字符，共分为两组，一组是 16 个有代表性的例子；另一组是 180 个人工随机生成的例子。结果表明，相比于 SAIL 算法，小兵算法能在绝大多数模式中获得更好的匹配解。

本文第 2 节是对 PMWL 问题发展和相关算法的介绍；第 3 节重点介绍 PMWL 问题中最具代表性的 SAIL 算法，并详细阐述它与小兵算法内在的思想联系；第 4 节给出 PMWL 问题的定义和关键概念；第 5 节给出小兵算法中涉及的概念及算法流程；第 6 节为实验设计、结果及探讨；最后是对全文的总结。

2 相关工作

模式串匹配问题是指给定一个文本 T 和模式 P 作为输入，找出 P 在 T 中所有的出现即匹配数或匹配位置作为输出。1974 年 Fischer 和 Paterson 将通配符 don't cares 引入模式匹配问题[3]，之后模式匹配的定义出现了各种各样非标准形式。Muthukrishna 等将各类非标准问题统称为 Non-standard Stringology[16]。这些问题的应用已渗透至各行各业。

经过多年的发展，这种非标准的模式匹配始终围绕一个问题，即如何将通配符与传统的模式匹配有效结合，而通配符出现在模式(或文本)中的位置以及它所能指代的字符个数是问题的关键。文献[3]提出通配符存在于两两字符间，是一个由用户定义的常数。这其中，最具代表性的定义是通配符指代的字符数不仅仅用一个固定的常数表示，而是一个可由用户自定义的区间，即带有上下限约束[17]。如生物序列中的……CAATCTϕ[30,50] TATA……，文献[18,19]将该带有区间的通配符扩展至任意两两相邻的字符之间，而所有的通配符上下限相同，如 Aϕ[1,3]Cϕ[1,3]Gϕ[1,3]C。为了进一步放宽约束，文献[10]的问题定义中通配符彼此独立，因此可以有不同的约束区间。Chen 等将上述定义归纳为如下 3 个条件：首先，任意两两相邻的字符间都存在通配符，且各自带有独立的上下限范围，如 Aϕ[0,1]Tϕ[0,2]Gϕ[1,3]C；其次，在匹配过程中文本串的字符至多只能使用一次。这些条件分别称为长度约束和 one-off 约束[9]。针对带有这两个约束条件的匹配问题，本文将之定义为 PMWL[1] 问题。值得注意的是，one-off 条件在传统的模式匹配问题中并未提及，因为在传统的模式匹配定义下不会发生匹配位置的冲突。这本质上是因为 PMWL 问题中通配符的约束区间造成了匹配解之间可能出现嵌套情况，例如{1,4,7}和{2,5,8}。综上，与以往通配符定义相比，PMWL 将通配符从固定匹配单个字符变为灵活的匹配一段带有长度约束的字符区间，更适合当前应用领域，但也隐含了求解的复杂性。

3 问题定义

定义 1 给定字母表 Σ，$t_i \in \Sigma$，其中 $0 \leqslant i < n-1$。称字符序列 $T=t_0 t_1 \cdots t_{n-1}$ 为一段长度为 n 的文本。给定 Σ，$p_0 \in \Sigma$，其中 $0 \leqslant i < m-1$。称二元组 $P=(p,g)$，$p=p_0 p_1 \cdots p_{m-1}$，$g_i=\phi[l_i,u_i]$ 为带有通配符和长度约束的模式。其中，p 为模式的字符，g 为模式的通配符，$[l_i,u_i]$ 为第 i 个通配符的作用区间，$0 \leqslant i < m-1$。称 m 为模式长度，$\max\{u_i-l_i\}$ 为通配符跨度，记为 gap。例如，$P=a\phi[0,2]g\phi[1,4]g$，则 $L=3$，$gap=\max\{2-0,4-1\}=3$。

定义 2 给定模式串 $P=p_0 p_1 \cdots p_{m-1}$，若 $p_i=p_{i+1}=\cdots=p_{m-1}$，$1 \leqslant i \leqslant m-1$，则认为该模式串的尾部带有重复元素。

定义 3 给定文本 T 和模式 P，如果有一组匹配位置 $A=(a_1,a_2,\cdots,a_{m-1})$ 满足 $t[a_i]=p[i]$，其中 $0 \leqslant i \leqslant m-1$，且同时满足长度约束，即 $t[a_{i+1}]-t[a_i] \in [l_i,u_i]$，其中 $0 \leqslant i \leqslant m-2$，则称 A 是模式 P 的一组匹配解。不同匹配解之间同时需要满足 one-off 条件，该条件的定义是文本中的某个位置 $t[j]$($0 \leqslant j \leqslant n-1$)至多只能出现在一组匹配解中。例如，匹配解{0,3,5}和{1,3,6}并不满足 one-off 条件。特别地，多组匹配解 $A_1,A_2,\cdots,A_t$ 构成一个匹配解集 U，其中 t 是匹配数，记为 $N(U)$。

定义 4 给定文本 $T=t_0,t_1,\cdots,t_{n-1}$，其中 t_i,t_j 是 T 的某两个字符，$0 \leqslant i \leqslant j \leqslant n-1$，给定模式 $P=p_0 p_1 \cdots p_{m-1}$，p_r 是 P 的某个字符，$0 \leqslant r \leqslant m-1$。若满足 $t_i=t_j=p_r$，则匹配过程中 P 会优先选取 t_i 作为匹配位置，直观上看就是位置标号较小的，或是最左边的。因此把上述带有贪心思想的匹配方式称为最左最优策略，记为 left-most 策略。

4 SAIL 算法描述及不完备原因分析

4.1 SAIL 算法描述

SAIL 算法是目前处理 PMWL 问题最有效的算法。该算法可描述如下：

输入：文本 $T=t_0 t_1 \cdots t_{n-1}$，模式 $P=p_0 p_1 \cdots p_{m-1}$，长度约束 $g_i=\phi[l_i,u_i]$

输出：P 在 T 中满足约束的匹配解

算法步骤：总体上，SAIL 扫描一遍 T，判断当前文本字符 t[i]是否与模式的尾部字符 p[m−1]匹配，记为 Location 过程。在此阶段若匹配，则进入 Forward 过程，判断是否存在一个潜在的匹配解，若存在，则调用 Backward 过程，找出满足 left-most 策略的一组解。Location 过程结束后，输出所有的匹配解。

Location：①扫描位置 i，其中 t[i]=p[m−1]。匹配后，定位位置 k，其中 t[k]=p[0]；②从 T 中截取 t[k]到 t[i]的字符，得子串 T′；③构建表格，其中行、列分别对应 T′和 P。

Forward：从前向后扫瞄表格，标记所有满足长度约束的匹配位置作为潜在的匹配解。

Backward：从后向前扫描表格，根据 left-most 策略，为每一行的字符选择对应的匹配位，满足如下条件：①标记过的；②满足长度约束；③最左边的。扫描完成后，将每行选出的匹配位置组成一组匹配解，同时记录这些位置已使用过。

4.2 不完备匹配的引例

Text：*aabbcccc*

Pattern：$a\phi[0,1]b\phi[0,1]c\phi[0,1]c$

[1] Pattern Matching with Wildcards and Length constraint

SAIL 算法在 T 中搜索模式最后一个字符 c，当匹配第一个字符 c 时，调用 forward 过程无法构建表格，则继续往前查找，直至匹配第二个 c 时，构建表格，如表 1 所列。

表 1　SAIL 算法不完备匹配的引例

	a	a	b	b	c	c
A	1	1				
B			1	1		
C					1	1
C						1

此时，SAIL 算法调用 backward 将把{0,2,3,4}作为一组查找的结果，在 one-off 约束条件下，SAIL 的输出仅此一解，而完备解为{{0,2,4,6},{1,3,5,7}}。

4.3 SAIL 不完备的原因

backward 过程采用了 left-most 策略，以保证把更多的字符留给之后的匹配解。该策略在模式中无重复字符的情况下完备[14]。然而当 P 的尾部出现连续字符时，如引例的 $a\complement[0,1]b\complement[0,1]c\complement[0,1]c$，SAIL 可能会丢解。本文认为，SAIL 是一种带有贪心思想的启发式算法，丢解的原因是算法在贪心策略下陷入了局部最优。因此，本文将在匹配中采用动态剪枝技术，以提高匹配解的质量。

5 小兵算法

5.1 小兵算法思想

中国象棋中双方棋手按回合制行走，象棋中的小兵每次只能走一步，且不能后退；敌对的小兵间可以互相吃子。本算法借鉴了象棋的思想，对 SAIL 表格做出修改，加入通配符行，将之看作棋盘，特别是将模式匹配问题转化为棋盘中的路径搜索问题。此外，PMWL 问题中带有 one-off 约束条件，使得文本中的字符至多只能出现在一个匹配解中。因此，在搜索过程中，如何合理分配文本中的字符，是提高匹配数的关键。本文将棋盘上可能匹配解看作小兵，按照回合制的方式轮回访问，并引入优先级，处理文本字符的分配问题。此外，本文将引入 4 条规则，以控制小兵行走的方向和步骤。

5.2 小兵算法中的概念

定义 5　算法中的棋盘，类似 SAIL 算法中的表格，加入了通配符行，棋盘上元素初始值为 0。元素值为 -1 时，代表无法通行，用于标注不可能行进的位置，从而降低搜索空间，是算法剪枝策略的关键；-2 表示该位置“已被某小兵占用”，但可以被优先级更高的小兵抢占。$1,2,\cdots,n$ 表示该位置已被第 $1,2,\cdots,n$ 个小兵使用。

表 2 列出棋盘的初始值。

表 2　棋盘的初始值

	a	a	b	b	c	c
a	0	0	−1	−1	−1	−1
ℂ	0	0	0	0	0	0
b	−1	−1	0	0	−1	−1
ℂ	0	0	0	0	0	0
c	−1	−1	−1	−1	0	0

定义 6　通配符行上的位置可以让任何小兵行走，一个位置也可以让几个小兵同时行走，然而要离开通配符行，需要比较小兵当前的得分与该行所对应通配符的长度约束范围（详见第 5.3.2 节通配符行规则）。

定义 7　小兵表示可能匹配的模式串，棋盘内可能存在一个或多个，小兵的行走需要符合算法的 4 个规则。它将携带已走路径的信息，当其成功走至最后一行时，被认为成功匹配，并输出匹配位置。

定义 8　小兵每进入一个通配符行时，需要初始化得分，当其在通配符行上每走一步时，得分加 1。离开通配符行时需要判断得分是否满足所对应通配符的长度约束。

定义 9　不同小兵优先级不同，初始位置更小的小兵，优先级更高。如果某一小兵重新游戏、寻找出发点，则其优先级也会相应改变。该定义表明，算法在不遇到文本资源分配的情况时，将采用 left-most 策略。

定义 10　算法执行过程中将采用回合制，轮回访问每个小兵，当满足规则时，当前小兵会行走。

定义 11　小兵在棋盘中有横纵坐标，记为 x、y，则小兵 i 与小兵 j 的距离可定义为：$|x_i+y_i-x_j-y_j|$。

5.3 小兵算法的 4 条规则

5.3.1 方向规则

所有小兵都是从棋盘的“左上”往“右下”前进，所有小兵的目标是搜索一条可达到棋盘最后一行的路径，即为找到一组匹配解。小兵在行进一步时，有 4 种可能情况：(1)向下；(2)向右下；(3)抢占方式向下；(4)向右。小兵行进的方向规则如图 1 所示。

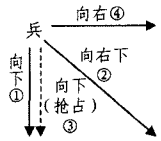

图 1　小兵行进的方向规则

方向优先级的高低为：下＞右下＞下(抢占)＞右。

行走过程中，如果前方单元格为 -1，则不能通过；如果前方为 -2，则通过比较两者的优先级判断能否抢占，若不能则寻找其它方向。

5.3.2 通配符行规则

通配符行的规则是针对 PMWL 问题中的长度约束条件而设计的。当小兵进入通配符行时，*score* 将初始化为 -1。规则如下所示。

1. 该行任意一个位置可以被多个小兵共同使用；

2. 在该行横向移动一次，*score* 的值加 1，直观上说，横向移动就是利用通配符在 T 中往后“跳”了一个字符；

3. 小兵离开该行的判断条件：

设$[l_i,u_i]$为该行的长度约束范围，则

a. 若满足 $l_i\leqslant score\leqslant u_i$，则小兵往下方行走；

b. 若满足 $l_i\leqslant score+1\leqslant u_i$，则小兵往右下方行走。

5.3.3 停等规则

停等规则的引入是为了更合理地分配匹配位置，特别是匹配解中出现的嵌套情况。例如引例中的解集{{0,2,4,6},{1,3,5,7}}。为了便于处理该规则，本文引入缓冲队列，记录等待进入棋盘的小兵。停等发生的条件如下所示。

(1)缓冲队列为空；

(2)与其他小兵距离合适，即在棋盘上构建直角坐标系，左上角设为原点(0,0)，小兵 A 在第 x 行 y 列记为 $S_A(x,y)$。$sup(A)$为小兵 A 的优先级，则满足：

$S_A(x,y) \geqslant S_B(x,y) \Leftrightarrow sup(A) > sup(B)$

$S_A(x,y) - S_B(x,y) \leqslant 1 \Leftrightarrow sup(A) - sup(B) = 1$

即优先级高的小兵距离棋盘坐标原点的距离大于优先级低的小兵，而当两个小兵的优先级相差 1 时，它们之间的距离差距应该保持在 1 以内。该规则约束了小兵在行进过程中的相对位置，避免了有效资源过于集中在某一个匹配解的情况。由于算法采用的是回合制访问，则在停等规则下，需要等待的小兵将停止一回合，停等规则的实例如表 3 所列。

例　T:*accaebddb*　P:$a\complement[1,4]b$

表 3　停等规则的实例

	0	1	2	3	4	5	6	7	8
	a	c	c	a	e	b	d	d	b
a	1			2					
$\complement[1,4]$	1	1	1	1,2	1,2	2	2	2	
b						1			2

小兵 2 根据规则，在位置(0,3)等待优先级更高的小兵 1。匹配结果为：{{0,5},{3,8}}。

5.3.4　抢占规则

在某些情况下，会出现部分匹配位置被优先级较低的小兵提前使用。抢占是为了保证上述位置仍能被更高优先级的小兵使用。抢占条件需满足：

(1)被抢占者优先级低于抢占者；

(2)抢占只能发生在小兵从上往下行进，因为抢占是因为不同的匹配解需要获得同一个文本字符，因此一定发生在棋盘的同一列上。

5.4　总体算法

5.4.1　总体流程

设棋盘为 M，$M[x][y]$表示位置(x,y)的元素值。

1. 根据字符 $P[0]$在文本 T 中的匹配位置，生成 K 个互不相交的棋盘。

2. 对于每个棋盘，初始化棋盘大小及小兵初始位置，并设定各棋盘不可行进的位置。

3. 对当前的小兵，假设位置为(x,y)，在 4 条规则的约束下，按照回合制的方式在棋盘上行走。对于每一回合：

a)若已达最后一行，则记录该小兵的行走路径，作为一组匹配解，同时在棋盘上标记上述位置已无法使用，以保证 one-off 约束条件。退出循环。

b)若满足停等规则条件，则需要停止一回合，则不做任何移动，退出循环。

c)若 $M[x+1][y]=0$

i. 若第 x 行不是通配符行，则向下移动，退出循环。

ii. 若第 x 行是通配符行，且满足长度约束的条件，则向下移动，退出循环。

d)若 $M[x+1][y+1]=0$

i. 若第 x 行不是通配符行，则向右下移动，退出循环。

ii. 若第 x 行是通配符行，且满足长度约束的条件，则向右下移动，退出循环。

e)若 $M[x+1][y] > 0$，即已被其余小兵使用，则按照抢占规则执行。

i. 若满足抢占条件，则向下移动，且将被抢占的小兵放入缓冲队列，退出循环。

ii. 否则，执行 f)步骤。

f)若 $M[x][y+1]=0$ 且第 x 行是通配符行，则向右方移动，退出循环。

g)若不满足上述所有条件，则将该小兵撤离棋盘，且将该小兵所有经过的位置更新为 0。

4. 若缓冲队列不空，则队首的小兵出队，执行 3；若为空，则执行 5。

5. 汇总所有的匹配解作为最终输出。

5.4.2　回到引例

T:*aabbcccc*　P:$a\complement[0,1]b\complement[0,1]c\complement[0,1]c$

(1)先根据 T、P 生成 8 * 9 的棋盘，而后初始化，用−1 标记出不可行进的位置；

(2)确定小兵数目为 2；

(3)选定出发位置，小兵 1 在位置(0,0)，小兵 2 在(0,1)；

(4)执行算法，当小兵 1,2 的路径搜索完毕时算法结束；

(5)如图 4 所示，获取的匹配位置为{{0,2,4,6},{1,3,5,7}}。

表 4　小兵算法对引例的求解

	0	1	2	3	4	5	6	7
	a	a	b	b	c	c	c	c
a	1	2	−1	−1	−1	−1	−1	−1
$\complement[0,1]$	1	1,2	2					
b	−1	−1	1	2	−1	−1	−1	−1
$\complement[0,1]$			1	1,2	2			
c	−1	−1	−1	−1	1	2		
$\complement[0,1]$					1	1,2	2	
c	−1	−1	−1	−1			1	2

6　实验设计

实验部分将在更多实例和真实文本下比较小兵算法与 SAIL 的完备性和时间效率。实验中文本源自 NCBI 中的 AX829174 的 DNA 数据集，模式包括两组：第一组是 16 个有代表性的模式；第二组是随机生成 180 个尾部有重复字符的模式。实验运行的软硬件环境为：奔腾双核 E5200、主频2.49 GHz、内存 2.0G、WINDOWS 7 操作系统。

6.1　实验结果

实验 1　在 16 个有代表性的模式上的匹配数比较，其中，匹配时间单位为毫秒。

实验 2　在 180 个人工随机模式上的匹配数比较，其中模式特征满足$|P|=3, gap \in [0,4]$。

实验 2 定义：

$$\gamma = \frac{N(\text{小兵算法}) - N(\text{SAIL 算法})}{N(\text{SAIL 算法})}$$

由定义 3，N 表示匹配数。

6.2　结果分析

针对尾部带有重复字符的模式串，总体上，小兵算法可以在绝大多数模式中得到更好的匹配解。在实验 1 中，由表 5 可知，平均情况下小兵算法能比 SAIL 算法多找出 6 个匹配解；最好情况下，小兵算法能找到 29 个 SAIL 无法定位的匹配解，占匹配总数的 4.76%。在实验 2 中，本文随机生成了 180 个模式，以区分在选择代表性实例中可能产生的偏向性。由图 2 可知，在 54 个模式中，小兵算法比 SAIL 算法获得了更多的匹配解，而仅有 3 个模式的匹配解数目比 SAIL 少，其余模式中两算法匹配数相等。由此可知，小兵算法有效地弥补了 SAIL 算法处理尾部带有重复字符模式串时的不足。

表 5　小兵算法与 SAIL 算法在 16 个模式上的匹配结果

	匹配数（小兵）	匹配数（SAIL）	匹配数之差	时间消耗（小兵）	时间消耗（SAIL）
a¢[0,1]c¢[0,1]c	268	267	1	94	62
a¢[0,1]c¢[0,1]c¢[0,1]c	101	98	3	125	94
a¢[0,1]c[0,1]c¢[0,1]c¢[0,1]c	40	39	1	156	93
a¢[0,3]a¢[0,3]g¢[0,4]g	490	480	10	625	93
a¢[0,3]c¢[0,3]t¢[0,4]a¢[0,4]a	383	382	1	609	172
a¢[1,2]t¢[1,2]t¢[1,2]t¢[1,2]t	167	160	7	297	160
a¢[2,3]t¢[0,2]c¢[0,2]c	264	264	0	219	94
c¢[1,3]t¢[0,2]a¢[0,1]a[1,4]a	230	223	7	297	156
t¢[0,2]a¢[0,1]c¢[4,4]c	84	84	0	172	94
t¢[0,0]a¢[0,5]c¢[0,5]c	252	249	3	203	94
c¢[0,1]a¢[0,2]a¢[0,3]a¢[0,4]a	309	308	1	344	157
a¢[0,1]t¢[0,1]t	562	555	7	109	94
t¢[2,5]a¢[2,5]a¢[2,5]a	609	580	29	1219	141
c¢[1,3]t¢[1,3]t	623	602	21	141	94
a¢[0,2]c¢[0,2]a¢[0,2]a	330	343	−13	266	125
t¢[0,5]c¢[1,3]a¢[0,3]a	581	554	27	593	141
平均	331	324.25	6.56	341.8	116.5

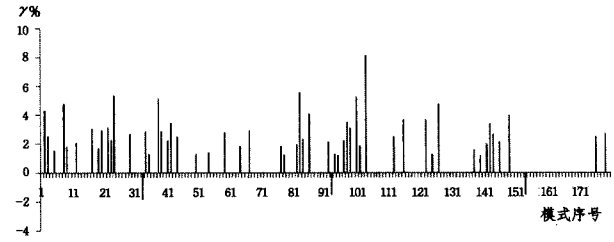

图 2　小兵算法与 SAIL 算法在 180 个模式上的匹配结果

结束语　针对 PMWL 问题，本文提出一种基于动态剪枝的小兵算法。小兵算法借鉴了最具代表性的 SAIL 算法，在其数据结构中引入通配符行，以此构建棋盘，并将模式匹配过程假想成小兵在棋盘上行走，从而将匹配问题转化成路径搜索问题。此外，小兵算法中引入了通配符行、行走方向、停等、抢占 4 条规则，并以回合制的方式规范小兵行走的时机和方向。同时，在每次访问时，将棋盘上位置分配给优先级较高的小兵，从而将文本资源更合理地分配给多个匹配解。当小兵到达最后一行后，将返回一组匹配解，同时将更新棋盘，并标记该小兵已访问的位置，以缩小搜索空间。实验在真实 DNA 序列文本和 196 个模式上进行，结果表明，小兵算法相比于 SAIL 算法更适合处理尾部带有重复字符的模式。下一步我们将进一步分析模式特征在 PMWL 问题的分析和算法设计中的作用。

参 考 文 献

[1] Navarro G , Raffinot M. Flexible pattern matching in strings: Practical on-line search algorithms for texts and biological sequences[M]. Cambridge, UK: Cambridge University Press, 2002

[2] Han J, Cheng H, Xin D, et al. Frequent pattern mining: Current Status and Future Directions[J]. Data Mining and Knowledge Discovery, 2007, 15(1): 55-86

[3] Fischer M J, Paterson M S. String matching and other products [R]. Complexity of computation, 1974: 113-125

[4] Cole J R, Chai B, Marsh T L, et al. The ribosomal database project (RDP-11): Sequences and tools for high-throughput rRNA analysis[J]. Nucleic Acids Research, 2005, 33 (1): 294-296

[5] 徐小双，冯玉才，王锋，等. 路径分区编码优化小枝查询[J]. 计算机科学，2010，37(3)：182-204

[6] Xie F, Wu X D, Hu X G, et al. Sequential Pattern Mining with Wildcards[C]// 22nd IEEE International Conference on Tools with Artificial Intelligence (ICTAI). 2010: 241-247

[7] Guo D, Hu X G, Xie F, et al. Pattern matching with wildcards and gap-length constraints based on a centrality-degree graph [J]. Applied Intelligence, 2013, 39: 57-74

[8] Bille P, Gortz I, Vildhoj H, et al. String matching with variable length gaps[J]. Theoretical Computer Science, 2012, 443: 25-34

[9] Chen G, Wu X D, Zhu X Q, et al. Efficient String Matching with Wildcards and Length Constraints[J]. Knowledge and Information Systems, 2006, 10(4): 399-419

[10] Min F, Wu X D, Lu Z. Pattern matching with independent wildcard gaps[C]// Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC-2009). 2009: 194-199

[11] Zhang M, Kao B, Cheung D, et al. Mining periodic patterns with gap requirement from sequences[C]// Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data. 2005: 622-633

[12] Zhu X Q, Wu X D. Mining complex patterns across sequences with gap requirements[C]// Proceedings of 2007 International Joint Conference on Artificial Intelligence. 2007: 2934-2940

[13] Guo D, Hong X L, Hu X G, et al. A Bit-Parallel Algorithm for Sequential Pattern Matching with Wildcards[J]. Cybernetics and Systems, 2011, 42(6): 382-401

[14] 王海平，胡学钢，谢飞，等. 模式特征对带有通配符和长度约束的模式匹配问题的影响[J]. 模式识别与人工智能，2012，25(6)：1013-1021

[15] 运正佳，李铁男，杨晓春. 支持带有通配符的字符串匹配算法[J]. 计算机科学与探索，2010，4(11)：984-995

[16] Muthukrishnan S, Palem K. Non-standard stringology: algorithms and complexity[C]// Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing. 1994: 770-779

[17] Manber U, Baeza-Yates R. An algorithm for string matching with a sequence of don't cares[J]. Information Processing Letters, 1991, 37(3): 133-136

[18] Zhu X Q, Wu X D. Discovering relational patterns across multiple databases[C]// Proceedings of IEEE 23rd International Conference on Data Engineering. 2007: 726-735

[19] Zhang M, Kao B, Cheung D W, et al. Mining periodic patterns with gap requirement from sequences[C]// Proceedings of ACM SIGMOD. 2005: 623-633