[**AC自动机--尚需努力**](http://www.cnblogs.com/looker_acm/archive/2010/07/21/1782025.html)

　　AC自动机，名字很酷(我就是奔着这名字去的)，其实就比字典树多了一个fail指针，什么是fail指针呢，就是串对字典树匹配失配时字典树节点下一步往哪指。是不是和KMP有点像，呵呵，没错，就是KMP的思想，假设该节点是u，它的fail指针指向v，v就是一个和u的字母一样，并且从根出发到v形成的字符串和从根到u的路径上的某一处到u形成的字符串相同的节点。呵呵，有点麻烦，其实求的时候就是一个递推的过程，root的fail指针是NULL，某一个节点u的fail指针就是它的父节点的fail指针指向的节点(以下简称x节点)的u孩子(u孩子就是指一个节点的孩子中的一个和u的字母一样的孩子)，如果x没有u孩子，按x的fail指针继续往上找。直到找到，如果找到root仍没有，u的fail指针就指向root。代码可描述为：

[view source](http://www.cnblogs.com/looker_acm/archive/2010/07/21/1782025.html#viewSource)



[print](http://www.cnblogs.com/looker_acm/archive/2010/07/21/1782025.html#printSource)[?](http://www.cnblogs.com/looker_acm/archive/2010/07/21/1782025.html#about)

|  |  |
| --- | --- |
| 01 | struct Trie{ |
| 02 | int count; | |

|  |  |
| --- | --- |
| 03 | struct Trie \*fail; |
| 04 | struct Trie \*next[26]; | |

|  |  |  |
| --- | --- | --- |
| 05 | } \*root,\*queue[500005]; | |
| 06 | void get\_fail() |

|  |  |
| --- | --- |
| 07 | { |
| 08 | int start = 0,end = 1,i; | |

|  |  |
| --- | --- |
| 09 | struct Trie \*p = root,\*q; |
| 10 | root->fail = NULL; root->count = -1; | |

|  |  |
| --- | --- |
| 11 | queue[start] = p; |
| 12 | while (start < end){ | |

|  |  |
| --- | --- |
| 13 | p = queue[start]; |
| 14 | for (i = 0; i< 26; i++){ | |

|  |  |
| --- | --- |
| 15 | if (p->next[i] != NULL){ |
| 16 | queue[end++] = p->next[i]; | |

|  |  |
| --- | --- |
| 17 | q = p; |
| 18 | while (q->fail != NULL){ | |

|  |  |  |
| --- | --- | --- |
| 19 | | if (q->fail->next[i] != NULL){ |
| 20 | p->next[i]->fail = q->fail->next[i]; | |

|  |  |
| --- | --- |
| 21 | break; |
| 22 | } | |

|  |  |  |
| --- | --- | --- |
| 23 | else q = q->fail; | |
| 24 | } |

|  |  |  |  |
| --- | --- | --- | --- |
| 25 | if (q->fail == NULL) p->next[i]->fail = root; | | |
| 26 | | } |

|  |  |
| --- | --- |
| 27 | } |
| 28 | start++; | |

|  |  |  |
| --- | --- | --- |
| 29 | } | |
| 30 | } |

至于匹配，和KMP差不多啦，代码为

[view source](http://www.cnblogs.com/looker_acm/archive/2010/07/21/1782025.html#viewSource)



[print](http://www.cnblogs.com/looker_acm/archive/2010/07/21/1782025.html#printSource)[?](http://www.cnblogs.com/looker_acm/archive/2010/07/21/1782025.html#about)

|  |  |  |
| --- | --- | --- |
| 01 | void search(int x,int y,int k) | |
| 02 | { |

|  |  |
| --- | --- |
| 03 | int x1 = x,y1 = y; |
| 04 | struct Trie \*p = root,\*q; | |

|  |  |  |
| --- | --- | --- |
| 05 | while (x1 >= 0 && y1 >= 0 && x1 < n && y1 < m){ | |
| 06 | int id = map[x1][y1] - 'A'; |

|  |  |  |  |
| --- | --- | --- | --- |
| 07 | while (p->next[id] == NULL && p->fail != NULL) p = p->fail; | | |
| 08 | | p = p->next[id]; p = (p == NULL)?root:p; q = p; |

|  |  |  |
| --- | --- | --- |
| 09 | | while (q != root && q->count != -1){ |
| 10 | result[q->count][0] = x1-lenth[q->count]\*dir[k][0]; result[q->count][1] = y1-lenth[q->count]\*dir[k][1]; | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 11 | ans[q->count] = k + 'A'; q->count = -1; q = q->fail; | | |
| 12 | | } |

|  |  |  |
| --- | --- | --- |
| 13 | x1 += dir[k][0]; y1 += dir[k][1]; | |
| 14 | } |

|  |  |
| --- | --- |
| 15 | } |

至于题，下次发吧！下午还有万恶的比赛。。。。