KNUTH-MORRIS PRATT

- problem: exact pattern-matching
  - input: text: $t = t_1 t_2 \dots t_n \in A$ (alphabet)
    pattern: $p = p_1 p_2 \dots p_l \in A$
  - output: first exact occurrence of p in t, if one such exists, and output "no" if no such match exists

t: ——— size(t) = n
p: —— size(p) = l
} KMP is $O(n+l)$ ~ linear ♥♥♥

First, let's construct the skeleton machine ~ a finite automaton with states/edges for each element in p

M_p:

- state j of M_p corresponds to the prefix $p_1 p_2 \dots p_j$ of p
- Think of the input sequence t as being on a tape. The machine reads through the tape, one symbol at a time, and changes states accordingly.
- machine starts in state 0
  - ↳ if $t_i = p_1$, then the machine transitions to state 1, and the head moves 1 spot to the right
  - ↳ if $t_i \neq p_1$, then M_p stays in state 0 and still advances its head to t_2
- suppose after reading $t_1 t_2 \dots t_k$, we have M_p in state j ~ this means that the machine discovered $p_{1:j}$ of pattern p in the text t. In other words, the last j symbols in $t_1 \dots t_k$ are exactly $p_1 p_2 \dots p_j$
  - → if $t_{k+1} = p_{j+1}$, then M_p enters state j+1 and its head advances to t_{k+2}
  - However, if $t_{k+1} \neq p_{j+1}$, then which state should M_p transition to??
  - answer: M_p enters highest state # i such that $p_1 p_2 \dots p_i$ is a suffix of $t_1 t_2 \dots t_k t_{k+1}$
- To determine i, the machine has a failure-function(f) associated with it

Let's define it:

positions in p, so $1 \leq j \leq l$

$f(j)$ is the largest number $\delta < j$ such that:

$p_1 p_2 \dots p_\delta = p_{j-\delta+1} p_{j-\delta+2} \dots p_j$

* In other words, δ is the length of the largest prefix of $p_{1:j}$ that is also a suffix of $p_{1:j}$

Ex) p = aabbaab

Let's apply/calculate the failure function for each position i in p

there is no prefix of "aab" that is also a suffix

"a" is a prefix + suffix of "aabbaa"

p	a	a	b	b	a	a	b
i	1	2	3	4	5	6	7
f(i)	0	1	0	0	1	2	3

by convention

because "a" is a suffix and a prefix of "aa"

aabbaab

prefix "aab" matches suffix "aab", and length("aab") = 3

*this failure function will determine which state of M_p you transition to when $t_{k+1} \neq p_{j+1}$

How does the machine M_p use the failure function?

Define the function $f^{(m)}(j)$ as follows

i) $f^{(1)}(j) = f(j)$

ii) $f^{(m)}(j) = f(f^{(m-1)}(j))$

thus $f^{(m)}(j)$ = f applied m times to j = $f(f(\ldots f(j)\ldots))$

ex. $f^{(2)}(6) = f(f(6)) = f(2) = 1$

▷ Suppose again that M_p is in state j having read $t_1 t_2 \ldots t_k$ and $t_{k+1} \neq p_{j+1}$

▷ At this point, M_p applies f repeatedly to the j position until it finds the smallest value of m for which either:

case 1: $f^{(m)}(j) = u$ and $t_{k+1} = p_{u+1}$

case 2: $f^{(m)}(j) = 0$ and $t_{k+1} \neq p_1$

That is, M_p backs up through states $f^{(1)}(j), f^{(2)}(j), \ldots$ until case 1 or 2 holds for $f^{(m)}(j)$ but not $f^{(m-1)}(j)$

▷ In case 1, M_p enters state $u+1$

▷ In case 2, M_p enters state 0

▷ In either case, the machine head advances to position t_{k+2}

▷ In case 1, it is easy to see that, if $p_1 p_2 \cdots p_j$ was the longest prefix of p that is a suffix of $t_1 t_2 \ldots t_k$, then $p_1 p_2 \cdots p_{f^{(m)}(j)+1}$ is the largest prefix of p that is also a suffix of $t_1 t_2 \ldots t_k t_{k+1}$

▷ In case 2, no prefix of p is a suffix of $t_1 t_2 \ldots t_k t_{k+1}$

⟶ M_p then proceeds, processing the input symbol t_{k+2}. In this way, M_p continues until it enters the final state 1, meaning the pattern p was found in t, or until M_p reaches the last character t_n without entering the final state (1), in which case, we know that the pattern p is not in t.