String Matching - Horspool

Not in Book

Horspool

- Not dynamic programming because there are no subproblems

- Precompute a table to help solve the problem

Problem

- Where does the pattern “Pacific” appear in the string:

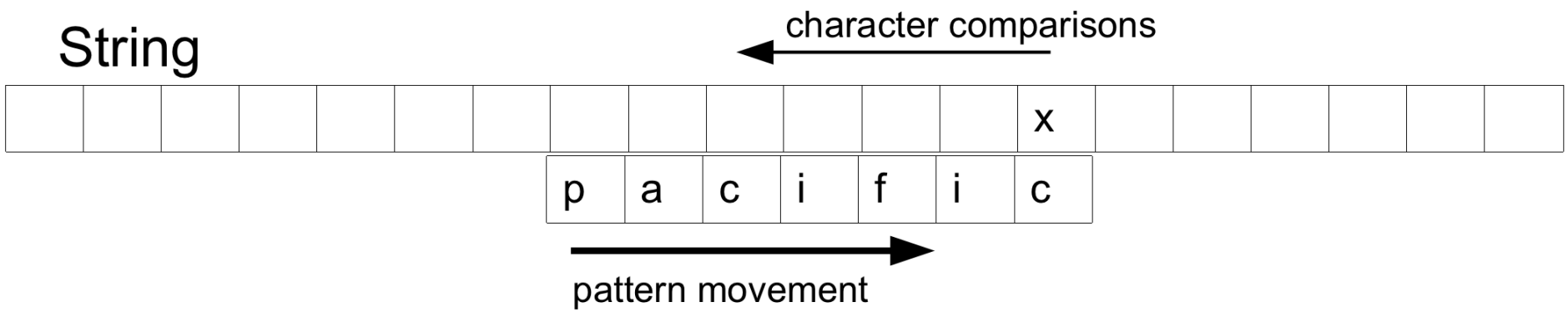
“Lūʻau is a time in our academic year when our 'Ohana comes together to celebrate and share our culture with the extended Pacific community.”

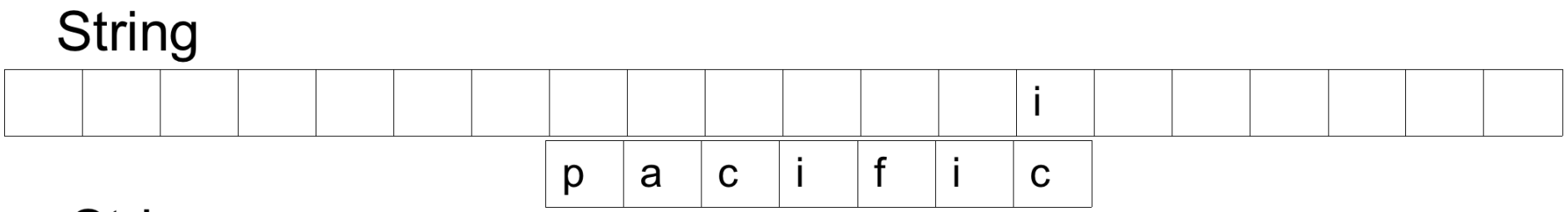
Naïve Solution

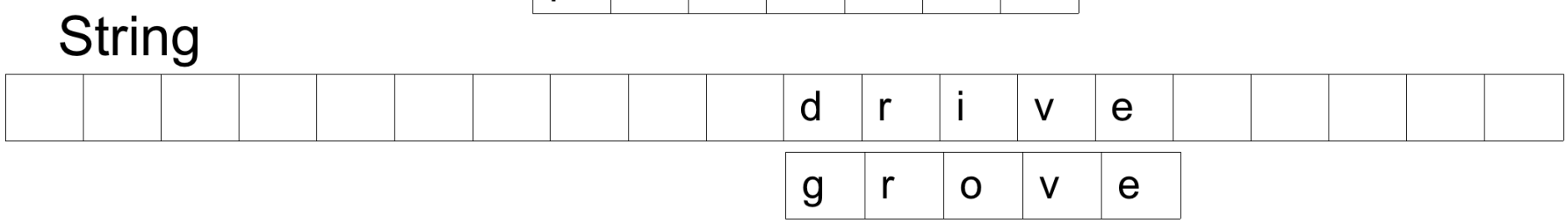
Horspool's Algorithm

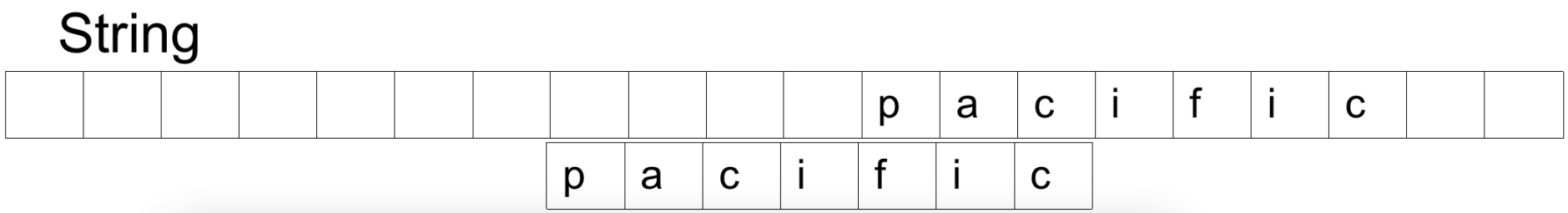
- Match the pattern right to left
- On mismatch, shift the pattern:
  - By +1 character(s)
- Preprocess string to determine shifting
  - Build a table for shifts for each valid character

Four Possibilities









Shifting

- t(c) =
- the pattern's length, m, if c is not among the first m-1 characters of the pattern
- the distance from the rightmost c among the first m-1 characters of the pattern to its last character, otherwise

p	a	c	i	f	i	c

a	b	c	f	i	...	p	...	x	y	z
								7		

Horspool Algorithm

HorspoolMatching$(P[0..m-1], T[0..n-1])$
//Input: Pattern $P[0..m-1]$ *and text* $T[0..n-1]$
//Output: The index of the left end of the first matching substring
// *or −1 if there are not matches*
ShiftTable$(P[0..m-1])$
$i \leftarrow m-1$
while $i \leq n-1$ ***do***
  $k \leftarrow 0$
  while $k \leq m-1$ ***and*** $P[m-1-k]=T[i-k]$ ***do***
    $k \leftarrow k+1$
  if $k=m$
    return $i-m+1$
  else $i \leftarrow i+Table[T[i]]$
return -1

ShiftTable

*ShiftTable (P[*0..*m*−1*])*
//Input: Pattern P[0..*m*−1] *and an alphabet of possible characters*
//Output: Table[0..*size*−1] *indexed by the alphabet's characters and*
// filled with shift sizes
for *i*←0 *to size*−1 ***do*** *Table[i]*←*m*
for *j*←0 *to m*−2 ***do*** *Table[P[j]]*←*m*−1−*j*
return *Table*

Example

- String:
  - G T A C T A G A G G A C G T A T G T A C T G
- Pattern:
  - A T G T A
- Generate the shift table
- Show the steps of the algorithm