I performed the string matching using following strings to prove the efficiency of KMP. Time mentioned is in ms

Example String 1

Haystack : AAA...99999 times...B → (character A for 99999 times and then character B)
Needle: AAA...999 times….B → (character A for 999 times and then character B)

```
found at: 99000
naive search time: 1359
found at: 99000
standard search time: 19
found at: 99000
kmp search time: 3
```

We can see that with the above input, the naive algorithm has to traverse the entire length of the needle before it hits a non matching character. This goes on till the end of the haystack string and hence it performs badly (1359 ms)

KMP on the other hand takes advantage of matching prefix and suffixes and hence performs much better. (3ms)

Standard search as I found uses some a type of dichotomic search (BST) and hence it's performance is relatively better than that of naive approach (http://stackoverflow.com/questions/5383429/what-algorithm-is-behind-stls-find)

Example String 2
Haystack: (AAAA...999 times…. B) pattern followed 100000 times and then followed by AAAA...1000 times
Needle: AAAA….1000 times

```
found at: 100000000
naive search time: 667318
found at: 100000000
standard search time: 11781
found at: 100000000
kmp search time: 5510
```

Using naive algorithm with this input, the needle string has to do 999 comparisons for the first time before hitting the non-matching character and then gradually the no. of comparison characters decrease. But since the input was too long, we can see a significant difference in running time of KMP and standard search when compared with naive method.

Example String 3
Haystack : AAA...99998 times..BA
Needle: BA

```
found at: 99998
naive search time: 2
```

```
found at: 99998
standard search time: 0
found at: 99998
kmp search time: 3
```

However, using a short needle string where comparison between the needle character and haystack character is substantially reduces, we can see the naive algorithm also performs well. Equivalent and sometimes better than KMP. I found that standard search is the indisputable winner in such cases.

Conclusion: With a lot of comparison before hitting the non-matching character, the naive algorithm will show a very bad performance. As the counts of step for comparing characters reduce , naive algorithms shows equivalent performance as KMP. In fact, slightly better performance can be seen with naive algorithms in such cases.