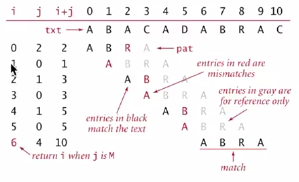
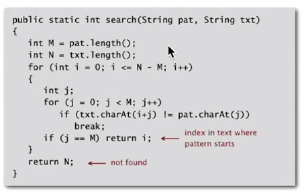
Substring Search

Brute force substring search: check for pattern starting at each text pattern.

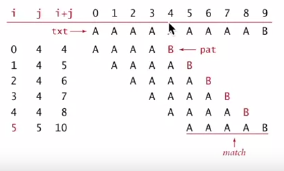


Implementation



Brute-force algorithm:

* Slow if text and pattern are repetitive

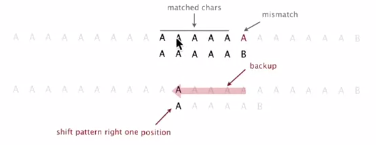


Backup

We want to avoid backup in text stream:

* Treat input as stream of data
* Abstract model: standard input

Brute-force algorithm needs backup every mismatch



Can maintain a small buffer of last M characters as a workaround, but there are better solutions.

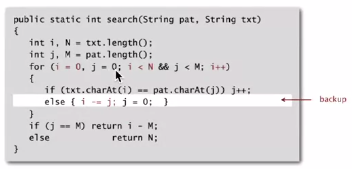
---

Alternate implementation of brute force substring search:

Same sequence of char compares as previous implementation.

* i points to end of sequence of already-matched chars in text
* j stores # of already-matched chars (end of sequence in pattern)

Java implementation



Algorithmic challenges in substring search

Brute-force is not necessarily good enough

**Theoretical challenge:** linear-time guarantee (fundamental algorithmic problem)  
**Practical challenge**: avoid backup in text stream (often no room or time to save text)

Substring search cost summary

(Cost of searching for an M-character pattern in an N-character text)

