Space-Time Tradeoffs

Use of auxiliary space may help to solve a problem faster

Example Sorting in linear time

Task: sort records \( r_1, \ldots, r_n \) by their keys \( r_i.key \in \{1, \ldots, m\} \)

\[
\text{for } i := 1 \text{ to } m \text{ do } L[i] := \text{new List}(); \\
\text{for } i := 1 \text{ to } n \text{ do } \\
\quad \text{Add } r_i \text{ at the end of list } L[r_i.key]; \\
\text{for } i := 1 \text{ to } m \text{ do } \\
\quad \text{Output contents of list } L[i];
\]

If the range of keys is unrestricted, sorting requires super-linear time

Boyer-Moore-Horspool algorithm

BMH matches the pattern \( M[1..m] \) starting from its end. The pattern is shifted forward according to a pre-computed table

\[
\text{shift}[x] = m \text{ if } x \notin M[1..m], \text{ else } \\
\min\{j \geq 1 \mid M[m - j] = x\}
\]

\[
\text{procedure } \text{BMH}(M, T : \text{string}) \text{ returns index} \\
\text{for } x \in V \text{ do } \text{shift}[x] := m; \\
\text{for } j := 1 \text{ to } \text{shift}[x] \text{ do } \text{shift}[M[j]] := m - j; \\
i := m; \\
\text{while } i \leq n \text{ do} \\
\quad \text{if } T[i] = M[m] \text{ then } \\
\quad \quad \text{if } T[i - m + 1..i] = M[1..m] \text{ then} \\
\quad \quad \quad \text{return (or print) } i - m + 1; \\
\quad \quad i := i + \text{shift}[T[i]]; \\
\]

Example Pattern \( M = "sana" \) and text \( T \):

\[
\begin{align*}
1234567890 \\
onko \text{ osana}
\end{align*}
\]

\[
\begin{array}{c|c}
\times & \text{shift}[x] \\
\hline
\text{s} & \text{a} \\
\text{n} & \text{otherwise}
\end{array}
\]

Worst-case time is still \( \Theta(mn) \), but usually the algorithm is fast, \( O(n/m) \) at best.
Also \( O(m + n) \) versions exist.