Praktische Informatik II SS 2009

Übungsblatt 11

Abgabe: 29.06.2009

Aufgabe 1 Einfache Stringsuche (25%)

Schreibt ein Programm zur Suche eines Worts in einem Text. Der Text und das zu suchende Wort sollen als Kommandozeilenparameter angegeben werden. Der Suchalgorithmus soll folgendermaßen vorgehen:

Das Suchwort wird zuerst mit dem Textanfang verglichen. Der Vergleich der Zeichen im Suchwort geschieht von rechts nach links. Wenn das Suchwort nicht mit dem Text übereinstimmt, wird es um eine Position nach rechts geschoben. Ansonsten wird die Position des Suchworts ausgegeben und die Berechnung endet.

Gebt zum Überprüfen der Lösung den Text, die Position des Suchworts und den ersten nicht übereinstimmenden Buchstaben wie im folgenden Beispiel aus:

```
CBAXBCBBCABA
ABBCBC
   ^
 ABBCBC
      ^
```

Testet Euer Programm und gebt das Fazit Eures Tests an (wählt kurze, übersichtliche Beispiele).

Aufgabe 2 Die Occurence-Heuristik des Boyer-Moore-Algorithmus (30%)

Da der Buchstabe 'X' in dem Suchwort nicht vorkommt, kann man das Suchwort auch direkt um 4 Positionen verschieben:

```
CBAXBCBBCABA
ABBCBC
   ^
    ABBCBC
```

Falls der Buchstabe, an dem sich das Suchwort und der Text zum ersten Mal unterscheiden, im Suchwort vorkommt, kann das Suchwort nur bis zu diesem Buchstaben vorgeschoben werden. Falls der Buchstabe mehrfach vorkommt, wird die kleinere Verschiebung gewählt.

Der Boyer-Moore-Algorithmus benutzt diese sogenannte Occurence-Heuristik (auch bekannt als Bad-Character-Heuristik), um die String-Suche zu beschleunigen.

- Erweitert Euer Programm um diese Heuristik.

Benutzt dazu eine Tabelle 'int[] Last', die zu jedem möglichen Buchstabenwert die letzte Position zurückgibt, an der das Zeichen im Suchwort auftritt. Falls das Zeichen nicht im Suchwort vorhanden ist, soll der Wert -1 enthalten sein.

- Berechnet die Last-Tabelle für die Buchstaben der beiden Suchwörter 'abcabab' und 'abcabcabc' (bevorzugt von Hand).
- Implementiert eine Funktion zur Berechnung der Tabelle.
- Überlegt Euch, welcher (einfache) Zusammenhang zwischen der Last-Tabelle und der Verschiebung des Suchworts bei der String-Suche besteht. Was tut man, wenn die Heuristik eine Verschiebung nach links vorschlägt?
- Integriert die Occurence-Heuristik in Euer Suchprogramm.
- Testet Euer Programm und gebt an, ob die Testergebnisse mit den erwarteten Ergebnissen übereinstimmen.

Aufgabe 3 Die Match-Heuristik des Boyer-Moore-Algorithmus, Verständnisfragen (25%)

Der Boyer-Moore-Algorithmus nutzt zudem die Match-Heuristik (auch bekannt als Good-Suffix-Heuristik). Diese nutzt aus, wenn die Endung des Suchworts mit dem vorderen oder mittleren Teil des Suchworts übereinstimmt. Hier drei Beispiele aus dem (genannten) Buch:

```
BAABBCABCABA
BCAABC
   ^
    BCAABC     (Übereinstimmung BC am Anfang)

CBABBCBBCABA
ABBCBC
   ^
  ABBCBC       (Übereinstimmung BC in der Mitte)

CBABBCBBCABA
ABBABC
   ^
      ABBABC   (keine Übereinstimmung)
```

Um die sichere Verschiebung nach der Match-Heuristik zu berechnen, führen Saake u. Sattler die *Shift-Condition*

$$Cs(i,s): \text{für jedes } k \text{ mit } i < k < m \text{ gilt: } s > k \vee pat[k-s] = pat[k] \qquad (1)$$

ein. Dabei ist pat das Suchwort, $pat[k]$ ist der Buchstabe an Position k, und m ist die Länge des Suchworts.

- Erklärt die Shift-Condition (1 oder höchstens 2 Sätze).

Saake u. Sattler führen als nächstes die *Occurence-Condition*

$$Co(i, s) : \text{wenn } s \leq i \text{ dann gilt: } pat[i - s] \neq pat[i] \tag{2}$$

ein.

- Erklärt diese Bedingung (1 Satz, höchstens 2).

Zur Implementierung der Match-Heuristik soll eine Tabelle 'int[] Shift' eingesetzt werden, die zu jeder Zeichenposition im Suchwort angibt, wie weit das Suchwort verschoben werden kann, wenn an dieser Position die erste Abweichung vom Text auftritt. Der Wert der Tabelle berechnet sich gemäß

$$Shift[i + 1] = min\{s > 0 | Cs(i, s) \text{ und } Co(i, s) \text{ gelten}\}. \tag{3}$$

Der Eintrag $Shift[0]$ wird auf m gesetzt.

- Beschreibt diese Minimum-Bedingung (1 Satz, höchstens 2).
- Veranschaulicht die Bedeutung der Shift- und der Occurence-Condition für das Suchwort 'abcabab' durch eine "Bedingungs-Tabelle" (nicht das Integerarray Shift!). Diese soll für die möglichen Zeichenpositionen des Suchworts (=7 Tabellenspalten) angeben, welche Verschiebungen bei einer Differenz des Suchworts an dieser Stelle möglich sind. Die Verschiebungen 1 bis 7 entsprechen den Zeilen der Tabelle. Die Zellen sollen mit einem 'o' markiert werden, wenn die Occurence-Bedingung nicht gilt, mit einem 's', wenn die Shift-Bedingung nicht erfüllt ist, mit einem 'b' wenn beides nicht erfüllt ist, und mit einem 'v' wenn beide Bedingungen gelten, d.h. die Verschiebung erlaubt ist.

Im günstigsten Fall kann das Suchwort um die volle Länge verschoben werden.

- Welche Tabelleneinträge sind das in der Bedingungs-Tabelle?

Die Berechnung des Shift-Arrays soll effizient mit Hilfe eines Arrays 'int[] Suffix' geschehen. Dieses soll für alle Zeichenpositionen des Suchworts die maximale Länge der Übereinstimmung mit dem Suchwortende angeben. Im Suchwort wird dabei von rechts nach links gelesen (Beispiel: Im Suchwort 'xxabcuuabc' steht das 'c' an den Positionen 4 und 9. An Position 4 stimmen 3 Buchstaben mit dem Ende des Suchworts überein).

- Berechnet das Suffix-Array für das Suchwort 'abcabab' von Hand.

Im zweitgünstigsten Fall stimmt die Suchwortendung mit dem Anfang des Suchworts überein. Dann kann der Suchwortanfang auf die erste Position der Übereinstimmung am Suchwortende verschoben werden (Tipp: Einige Situationen können über Suffix[i]=i+1 erkannt werden).

- Markiert die betroffenen Einträge in der Bedingungs-Tabelle.

Als drittes kann ein mittlerer Teil des Suchworts mit dem Suchwortende übereinstimmen. Da die Übereinstimmung relativ weit hinten auftritt, ergeben sich hier die kleinsten Verschiebungen (Tipp: Aus dem Suffix-Array kann sowohl die Länge des übereinstimmenden Suchwortendes als auch die Position an der das Suchwort vom Text abweicht berechnet werden. Letztere wird maximiert.).

- Markiert die betroffenen Einträge in der Bedingungs-Tabelle.

Aufgabe 4 Die Match-Heuristik des Boyer-Moore-Algorithmus, Implementierung (20%)

Implementiert und testet die Berechnung des Suffix-Arrays. Implementiert die drei Fälle bei der Berechnung des Shift-Arrays. Belegt Euer Verständnis des Codes durch die Wahl anschaulicher Variablennamen und Arrayindices. Testet Eure Implementierung der Match-Heuristik und gebt an, ob die Testergebnisse mit den erwarteten Ergebnissen übereinstimmen.