Der sicherlich interessanteste Schritt ist naturgemäß der, einem Muster einen bestimmten ASCII-Wert zuzuordnen. Dieser Vorgang (die eigentliche „Lektüre") ist vom bloßen Abtasten des Dokumentes zu trennen. Erst durch diese Umsetzung in das ASCII-System entsteht such- und umsetzbarer Text, der z. B. in die verschiedensten Textverarbeitungssysteme eingelesen und dort weiterverarbeitet werden kann. Eine Ausgabe auf Bildschirm oder Drucker ist kein Problem mehr. Digitalisieren und Mustererkennen, „Lesen genannt", sind die Grundkomponenten des Eingabesystems für elektronische Texte. Da diese Systeme für die Erkennung von Buchstaben konstruiert worden sind (Optical Character Recognition), werden sie als OCR-Systeme bezeichnet. Manchmal hört man auch die Abkürzung ICR-Systeme. Diese Abkürzung steht für „Intelligent Character Recognition".

Im Bereich der Texterfassungssysteme sind derzeit sowohl Bildverarbeitungssysteme (NCI-Systeme), wie auch Erkennungssysteme für Einzelbuchstaben (OCR- oder ICR-Systeme oder „lernende" OCR-Systeme, OCR-AL-Systeme) anzutreffen. Bei den OCR-Systemen sind (idealtypisch) hardware-orientierte, geschlossene Systeme von software-orientierten zu unterscheiden. Während OCR-Systeme der ersten Kategorie auf eine spezielle Hardware bezogen sind, arbeiten Systeme der zweiten Kategorie auf leistungsfähigen Universalrechnern, mit Eingabe über NCI-Hard- und Software (z. B. PageScanner TM 3119, ImagEdit TM) und Weiterverarbeitung über OCR-Software (z. B. TextReader TM oder MAKROLOG OPTOPUS TM). Eine Sicherung des gesamten Dokumentes ist immer dann erforderlich, wenn es auf das „Bild" des Dokumentes ankommt und nicht so sehr auf seinen Inhalt, der aber natürlich auch auf diese Weise abgespeichert wird. Dies ist z. B. der Fall bei Handschriften, etwa im Bankbereich zur Erkennung von Unterschriften, alten Büchern oder Dokumenten mit Bildanteil. Im kommerziellen Bereich ist diese außerordentlich einfache Technik, einschließlich der Übertragungstechnik Telefax, von besonderem Interesse. OCR-Systeme mit festen, nur teilweise trainierbaren Mustern haben ihr Einsatzfeld im Bereich der kommerziellen Textverarbeitung. OCR-Systeme mit frei trainierbarer Mustererkennung finden im Bereich komplexer Vorlagen (einschließlich Frakturschriften) und bei Fontmischungen Anwendung.

Es wurde betont, daß die Leistungsfähigkeit der heutigen OCR- Systeme überwiegend noch nicht ganz zufriedenstellend sei. Im Schnitt liege die Erkennungsquote bei maximal 99,5%. Bei 2000 Zeichen pro Seite seien 10 Fehler pro Seite immer noch inakzeptabel.

Soweit die Erörterung der Referate. Zum Teil sprechen diese Referate Bereiche an, die dem Arbeitsgebiet (auch des an EDV interessierten) Juristen zunächst fern liegen. Doch bei genauer Betrachtung führen die anderen Fachbereiche dem Juristen in Forschung und Lehre und am anwaltlichen Arbeitsplatz nur vor, was in naher Zukunft auf ihn zukommen wird. Insofern kann es für ihn ein Gewinn sein, diese Trends schon jetzt zur Kenntnis zu nehmen.

Volltext-Retrieval

Einige Beurteilungskriterien für PC-Software

Maximilian Herberger

Vorbemerkung

Seit die Zeichenerkennungstechnologie die Herstellung großer Mengen maschinenlesbaren Materials erlaubt, rückt die Frage der anschließenden Beherrschung dieser Textmengen immer mehr in den Mittelpunkt des Interesses. Es fragt sich, wie man in diesem Anwendungsfeld die vielfältigen Möglichkeiten der Volltext- Retrieval-Programme einzuschätzen hat. Vor diesem Hintergrund soll die folgende Skizze nicht der Rezension einzelner Programme dienen. Vielmehr sollen Beurteilungskriterien beschrieben werden, die die Einordnung von Programmen der hier interessierenden Kategorie erlauben. Wenn trotzdem verschiedentlich einzelne Programme namentlich genannt werden, so dient dies der Veranschaulichung der vorher erläuterten Kriterien. Keineswegs soll dadurch der Eindruck erweckt werden, nur die erwähnten Programme gehörten zu der beschriebenen Kategorie.

Zu berücksichtigen ist auch noch, daß die Einordnungskriterien im folgenden idealtypisch vorgestellt werden. Im Einzelfall finden sich u.U. vielfältige Überschneidungen und Mischformen, die sich einer klaren Einordnung entziehen. In diesem Falle muß man je nach Mischungsverhälnis im Einzelfall eine individuelle Bewertungsbilanz aufmachen.

Ausgeklammert bleiben Programme, die eine Volltext-Verwaltung nicht erlauben. Das gilt z.T. für sehr leistungsfähige Datenbank- Systeme wie etwa dBASE IV. dBASE IV hat zwar die Verwaltung der MEMO-Felder durch zusätzliche Funktionen erleichtert, eine Suchmöglichkeit in diesen Textfeldern variabler Länge ist aber standardmäßig immer noch nicht vorgesehen. Hingegen würden Programmiersysteme wie „CLIPPER" (oder als fertige Anwendung das in „CLIPPER" geschriebene „WAMPUM") zur Kategorie (auch) der Volltext-Retrieval-Systeme gehören. Denn „CLIPPER" erlaubt es, den Inhalt der Memo-Felder als String-Variable anzusprechen und entsprechend zu durchsuchen.

Nicht behandelt wird auch die Frage, ob das jeweilige System andere Informationsarten (wie etwa Graphik) zusammen mit Text verwalten kann. Hat man es mit Anwendungen dieser Art zu tun, so verändert sich die Beurteilungssituation erheblich.

Um die Bedeutung der Software-Kategorie „Volltext-Retrieval" zu ermessen, genügt es, sich vor Augen zu führen, daß jede CD-ROM mit einer bestimmten Volltext-Retrieval-Komponente ausgeliefert wird. Will man diese Komponente evaluieren, so benötigt man ebenfalls Einordnungskriterien der Art, von denen jetzt die Rede sein soll.

Einteilungs- und Beurteilungskriterien

- Index-gestützte vs. index-freie Systeme

Systeme, die einen Index verwenden, können von Systemen unterschieden werden, die das nicht tun. Dabei steht „Index" hier für ein Verzeichnis, in dem die Position jedes im Volltext vorkommenden Wortes verzeichnet ist. Es leuchtet ein, daß ein solcher Index den schnelleren Zugriff auf die gesuchten Textstellen erlaubt. Umgekehrt beansprucht das Anlegen des Index Zeit. Hinzu kommt, daß nach jeder Änderung des Ausgangstextes ein neuer Indexierungslauf notwendig ist. Schließlich beansprucht jeder Index zusätzlichen Platz.

Vor- und Nachteile index-gestützter und index-freier Systeme müssen im Einzelfall gegeneinander abgewogen werden. Dabei sind Tests mit einem großen Datenbestand unerläßlich. Es hat sich nämlich gezeigt, daß etwa ab Textgrößen von 2 MB oft drastische Änderungen im Verhalten der jeweiligen Software auftreten. Hinzu kommt, daß die Dauer für das Anlegen des Indexes und die Größe des Indexes von Programm zu Programm erheblich differieren können. Schließlich spielt bei der Beurteilung noch die Organisation des konkreten Projekts eine große Rolle. Wenn etwa Texte selten bis kaum geändert werden und der Indexierungslauf als Batch-Lauf über Nacht durchgeführt werden kann, fallen die beschriebenen Nachteile index-gestützter Systeme kaum noch ins Gewicht.

Da die sequentielle Suche in großen Datenbeständen selbst bei optimaler Programmierung in großen Datenbeständen doch sehr schnell zeitkritisch wird, bieten manche Programme dieses Typs mittlerweile die Möglichkeit einer (partiellen) Indexierung. Als Beispiel sei der INDEX-Befehl in AskSam, das als Programm ansonsten „indexfrei" konzipiert ist.

- Möglichkeiten der Suchfragen-Formulierung

Die im folgenden genannten Zusatzaspekte hängen letzten Endes mit der Art des Index und/oder dem Such-Algorithmus zusammen. Es bestehen deswegen meistens Rückwirkungen auf andere Beurteilungskomponenten (wie z.B. Suchgeschwindigkeit, Indexierungszeit, Indexgröße etc.).

-- Suchfrage mit Abstandsangabe

Standardmäßig bieten die meisten Systeme die bekannten Boole'schen Suchoperationen mengentheoretischer Art ("und, "oder", verbunden mit „nicht"). Daneben gibt es oft einen Abstandsoperator ("nearby" oder ähnlich genannt), der bei einer gewünschten Wortkombination die Angabe des Wortabstandes erlaubt. Dieser Abstandsoperator kann unterschiedlich mächtig ausgestaltet sein. Die Bandbreite der Möglichkeiten reicht von der Angabe des Abstandes gemessen in Worten (z.B. „5 Worte entfernt von") über die Angabe des Abstandes in Zeilen bis hin zur Definition des Abstandes durch (in bestimmter Weise qualifizierte) Textabschnitte.

Übrigens bildet die Möglichkeit der Abstandssuche, wirft man einen kurzen Seitenblick auf die Großrechner-Landschaft einen der Hauptunterschiede zwischen dem auf Siemens-Rechnern verfügbaren GOLEM, das diese Suchart nicht vorsieht, und dem für IBM-Rechner angebotenen STAIRS.

- Maskierungen in der Suchfrage

Flexible Suche verlangt, daß Suchworte „maskiert" werden können. Die entsprechenden Grundmöglichkeiten sind fast in allen Retrieval-Programmen enthalten. Man kann mit „VERTRAG*" (oder einer ähnlichen Schreibweise) alle Fundstellen erreichen, an denen die Buchstabenfolge „VERTRAG" am Anfang eines Wortes vorkommt (also z.B. Vertragsbedingungen, Vertragsklausel, Vertragsmuster etc.).

Eine genaue Untersuchung der angebotenen Maskierungsmöglichkeiten ist für eine korrekte Einschätzung der Leistungsfähigkeit eines Retrieval-Programms unverzichtbar. Dabei sollte man besonders darauf achten, daß neben der eben beschriebenen „pauschalen" End- Maskierung auch eine positionsgenaue Einzel-Maskierung möglich ist (z.B. „V?rtrag" für „Vertrag" und „Vortrag").

Von besonderer Bedeutung ist die Frage, ob eine Anfangs-Maskierung vorgenommen werden kann (z.B. „*trag" für „Vortrag", "Vertrag", „Ertrag" etc.). Manche Systeme bieten diese Möglichkeit nicht. Der Grund liegt darin, daß man hier - der Anfangsbuchstabe ist ja unbekannt - entweder den gesamten Index "durchlesen" muß (was zeitkritisch ist) oder andere Indexstrukturen (verbunden mit spezifischen Duchquerungs- Algorithmen) zu entwickeln hat. Es leuchtet jedoch ein, daß die Such-Flexibilität ohne Anfangsmaskierung erheblich eingeschränkt ist. Dies gilt vor allen Dingen in Sprachen, die mit Vorsilben arbeiten. ("*sehen" brächte im Deutschen alle Stellen zu "ansehen", „vorsehen" "nachsehen" etc., was in bestimmten Kontexten sehr erwünscht ist und durch eine „oder"-Suche nur begrenzt ersetzt werden könnte.)

- Zwischen sequentieller Suche und traditionellem Index: „Signature Screening"

Die Geschwindigkeit der sequentiellen Suche konnte in der zurückliegenden Zeit erstaunlich gesteigert werden. Trotzdem stieß man dabei an systembedingte Obergrenzen, so daß sich der Blick auf alternative Such-Strategien richtete. Ein besonders vielversprechender Weg scheint das sogenannte „signature screening" zu sein.

Beim „signature screening" handelt es sich um eine Indexierungstechnik, die auf Grund statistischer Maßzahlen für Buchstaben-Kombinationshäufigkeiten jeder Dokumentationseinheit einen Vektor als singulären „fingerprint" zuweist. Über diese Vektoren, die vergleichsweise wenig Speicherplatz beanspruchen, ist dann ein schneller (und treffsicherer) Zugriff auf die relevanten Dokumentationseinheiten möglich. Bemerkenswert dabei ist, daß das „signature screening" bei geschickter Wahl der statistischen Maßzahlen eine gewisse Fehlertoleranz aufweist und auch der Suchfrage ähnliche Muster mit erfassen kann.

(Wenn man übrigens diese Technik des „signature screening" nach Funktion und Auswirkungen studieren will, ist das Programm als Shareware verteilte „3BY5" dafür geeignet. Allerdings hat man leider bei der „Shareware"-Diskette eine Anzahl von Restriktionen eingebaut, die den Benutzer zum Erwerb von „Upgrades" motivieren sollen. So kann man nur bei einer Menge von bis zu 30 Datenstzen untersuchen, wie man in den (suchfähigen) Volltext Informationen in Feldern mit variabler

Länge einbetten und mit darauf gezielten Suchkommandos erschließen kann. Voll funktionsfähig ist aber - bis auf die Behandlung von Umlauten - die Volltextsuche (ohne „embedded fields") auch für große Datenbestände.)

- Feldorientierte vs. nicht-feldorientierte Systeme

Können im Volltext bestimmte markierte Informationseinheiten als Felder angesprochen werden, so spricht man von feldorientierten Volltext-Retrieval-Systemen. Kann man etwa „Vorname:" vor einen Vornamen setzen und dann im ansonsten nicht weiter bearbeiteten Volltext gezielt nach allen Vornamen mit bestimmten Eigenschaften (und nur nach diesen) suchen, so ist die beschriebene Feldorientierung gegeben. Dabei kann man dann anschließend noch unterscheiden, ob diese Felder eine fixe Länge haben müssen oder von variabler Länge sein können. Die Besonderheit ist dabei nicht die feldorientierte Suche: Diese Möglichkeit bieten die Datenbankprogramme auch. Die Besonderheit besteht darin, daß diese Felder in eine Umgebung „eingestreut" sein können, die ansonsten eine nicht besonders strukturierte Volltext-Umgebung ist.

Auch ohne Feldorientierung kann es u.U. eine gewisse Funktionsäquivalenz geben, die auf einem Ausnutzen des Abstandsoperators beruht. Falls z.B. sicher ist, daß vor dem Vornamen „Vorname:" steht, so genügt es, im Ein-Wort-Abstand zum gesuchten Vornamen nach „Vorname:" zu suchen. Oft genügt das, um die feldorientierte Suche zu simulieren. Sobald aber der Feldinhalt variabel lang sein kann, scheitert oft der Versuch, Teile (des variabel langen) Feldinhalts in der Suchfrage mit dem Feldnamen über den Abstandsoperator zusammenzufassen. In diesem Fall ist dann die Wahl eines Systems mit Feldorientierung unverzichtbar.

- „Präzise" vs. „unscharfe" Suche

Die „präzise" Suche findet nur die Vorkommnisse im Text, die exakt mit der Suchformulierung übereinstimmen. Daneben gibt es Systeme, die die Möglichkeit bieten, verschiedene Abschwächungen einzustellen mit der Folge, daß auch „ungenaue" Übereinstimmungen noch mit herausgefunden werden. (Ein System bietet sogar nur die Möglichkeit, „unscharf" zu suchen.)

Die „Unschärfe" kann einmal den Sinn haben, Schreibfehler zu kompensieren. In diesen Varianten kann bei der Suchfrage "Vertrag" je nach Ausgestaltung u.U. auch „Vertrg" gefunden werden.

Eine andere Form unscharfer Suche (die auch manche Schreibfehler irrelevant macht) ist die phonetische Suche. Diese Suchart orientiert sich nicht an der Schreibweise des Suchworts, sondern an dessen Aussprache. Hier würde man z.B. „malen" genauso wie "mahlen" finden. Toleriert würde auf die Frage „zivil" hin auch die Schreibweise „ziviel" usw.
"Unscharfe" Such-Prozeduren sind in aller Regel sprachabhängig. Deshalb empfiehlt sich bei amerikanischer Software mit derartigen Algorithmen vorab ein genauer Test, wenn es um nichtenglischsprachige Texte geht. Vielfach sind nämlich die Ergebnisse bei deutschen Texten so sinnlos, daß die unscharfe Suche nicht nur keine Hilfe bietet, sondern zusätzlich stört. Es gibt aber auch für die deutsche Sprache beeindruckend effektive Algorithmen in am Englischen orientierten Programmen. Wer dazu Versuche anstellen will, dem sei die Demo-Version von „Friendly Finder" empfohlen, die den patentierten Suchalgorithmus "Proximity-Scan" implementiert. Dort kann man mit der Liste der Mitglieder des 99. amerikanischen Kongresses arbeiten. Dabei führt z.B. die Suchfrage „Müller" zu „Miller", „Mählcher" zu „Melcher", „Hattscher" zu „Hatcher".

- Volltextsuche mit/ohne Import

Es können Programme unterschieden werden, die den Ausgangstext in unveränderter Form akzeptieren und Indexierungsfunktionen (gegebenenfalls) wie Suchoperationen über diesen unveränderten Ausgangstext ausführen. Zu einer noch feineren Klassifizierung kommt man dabei, wenn man die Programme, die heterogene Ausgangsformate akzeptieren (im günstigsten Fall sogar selbst erkennen), von denjenigen Programmen unterscheidet, die nur ein einheitliches Ausgangsformat verarbeiten. Dieser ersten Gruppe stehen die Programme gegenüber, die einen Import des Ausgangsmaterials in ein internes eigenes Format verlangen. Wie bereits bei der Indexierung angemerkt, ist auch dieser Vorgang zeit- und platzkritisch.

Die eben beschriebene Einteilung wird nicht überall starr gehandhabt. Verschiedentlich stellen Programme es auch dem Benutzer frei, ob er mit unverändertem Ausgangsmaterial oder mit transformiertem Material arbeiten will. Dabei ist dann meistens die Arbeit mit dem importierten Material schneller. Dies ist vor allen Dingen dann der Fall, wenn bei dem Import die Text-Dateien komprimiert werden. Findet dann (wie etwa bei dem nützlichen kleinen speicherresidenten Suchprogramm GOFER) die Suche in den komprimierten Dateien statt, wird zusätzlich zur Zeit auch Platz gespart, da man die (unkomprimierten) Originaldateien nicht mehr benötigt.

- Verfügbarkeit von Run-Time-Versionen („Retrieve only")

Zum Schluß sei ein für die Distribution fertiger Produkte besonders wichtiges Kriterium genannt: Für das Retrieval-Programm muß eine kostengünstige Möglichkeit gegeben sein, selbst erstellte Textmengen mit einer „Retrieve only"-Komponente des zur Aufbereitung und Erschließung verwandten Programms zu verteilen. Hier gibt es außerordentlich große Preisdifferenzen. Da CD-ROM- Applikationen alle eine derartige Such-Komponente benötigen, stellt sich hier die Frage mit besonderer Dringlichkeit.

Schlußbemerkung

Die reine Suchphilosophie wird dann überschritten, wenn man nicht mehr nur auf „Retrieval" abstellt, sondern darüberhinaus auf das "Navigieren" im Text Wert legt. In diese Richtung geht es, wenn (etwa wie bei „AskSam") sog. „Hypertext"-Komponenten in das Programm eingebaut werden. „Hypertext" bedeutet dabei, daß von beliebigen Teil-Informationseinheiten durch Setzen eines „Links" zu anderen Teil-Informationseinheiten verzweigt werden kann. Entlang dieser „Links" kann der Benutzer dann „navigieren".

Da jedoch Hypertext-Relationen definiert werden müssen, erfordert der Einsatz derartiger Instrumente Aufbereitungsarbeit am Ausgangsmaterial. Das geht über die den Aspekt der automatischen Volltexterschließung vorliegenden Materials hinaus und soll deswegen hier nicht vertieft erörtert werden. Es liegt aber auf der Hand, daß Volltext-Retrieval-Programme, die zusätzlich eine Hypertext-Komponente enthalten, im Vergleich zur "reinen" Retrieval-Software bemerkenswerte zusätzliche Aufbereitungs- und Erschließungsschritte erlauben.