fsq – Ein Abfragesystem für syntaktisch annotierte Baumbanken

Stephan Kepser

SFB 441, Universität Tübingen

Syntaktisch annotierte Baumbanken

- Ursprünglich: Morphosyntaktische Tags (POS)
- Anreicherung mit syntaktischen Informationen
- Vollständige syntaktische Analysen
- Komplexere Strukturen als echte Bäume: Unverbundene Teile, sekundäre Relationen

Beispiele für Syntaktisch Annotierte Baumbanken

- Penn Treebank
- French Treebank (Paris)
- British National Corpus
- NEGRA Corpus (Saarbrücken)
- TIGER Corpus (Saarbrücken, Stuttgart)
- DeReKo (Mannheim, Stuttgart, Tübingen)
- Tübingen Treebanks (Verbmobil, TAZ)
- viele weitere . . .

Abfrage von Baumbanken

Beim Stellen einer Abfrage

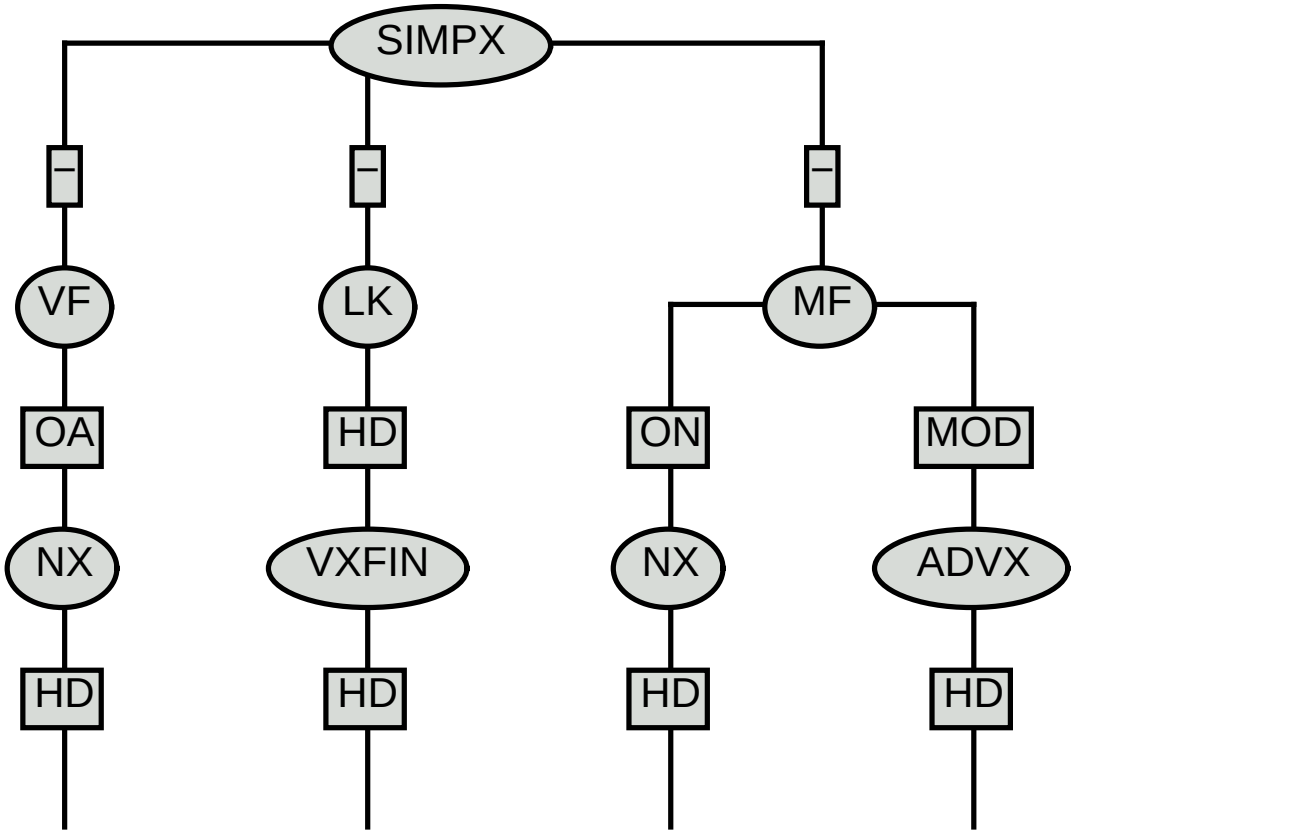
- besteht die Aufgabe nicht darin,
  ein paar Instanzen eines bestimmten linguistischen Phänomens zu finden.
- Die Aufgabe ist vielmehr,
  alles uninteressante herauszufiltern.
- Eine Antwortmenge zu erhalten, die beinhaltet, wonach man sucht, ist trivial.
- Eine Antwortmenge zu erhalten, die beinhaltet, wonach man sucht, aber auch so klein und frei von Müll wie möglich ist, ist sehr schwierig.

Abfrage von Baumbanken

- (Einige) Baumbanken bieten reiche syntaktische Annotationen.
- Aber Antwortmengen sollen so restingiert wie möglich sein.
- Also sollten Abfragesprachen eine große Ausdrucksmächtigkeit besitzen.
- Wir schlagen volle Prädikatenlogik erster Stufe als Abfragesprache vor.

Beispielbaum

Ein Beispielbaum aus der Tübinger Baumbank des Gesprochenen Deutschen.



Abfragesprache von fsq

- LISP-ähnliche Syntax
  Alle Teilformeln sind geklammert. Der Kopf steht immer links.
- Variablen (x, y, z, etc.) repräsentieren Knoten in einem Baum.
- Atomare Formeln drücken die syntaktische Annotation eines Knotens oder eine Relation zwischen zwei Knoten (wie z.B. Dominanz) aus.
- Komplexe Formeln erlauben boolsche Kombinationen von Formeln und Quantifikation.

Atomare Formeln: Knotenlabel

- `(tok x T)`
  Knoten `x` hat Token (Wort) `T`.
- `(lem x L)`
  Knoten `x` hat Lemma `L`.
- `(cat x C)`
  Knoten `x` ist von der Kategorie (hat POS Tag) `C`.
- `(mor x M)`
  Knoten `x` hat morphologisches Tag `M`.
- `(fct x F)`
  Knoten `x` is von der grammatischen Funktion (hat Kantenlabel) `F`.

Atomare Formeln: Knotenlabel mir Regulären Ausdrücken

- `(tokR x T)`
  Token von Knoten `x` paßt auf RA `T`.
- `(lemR x L)`
  Lemma von Knoten `x` paßt auf RA `L`.
- `(catR x C)`
  Kategorie von Knoten `x` paßt auf RA `C`.
- `(morR x M)`
  Morpholog. Tag von Knoten `x` paßt auf RA `M`.
- `(fctR x F)`
  Funktion von Knoten `x` paßt auf RA `F`.

Atomare Formeln: Knotenrelationen

- `(> x y)`
  Knoten x ist die Mutter von y.
- `(>> x y)`
  Knoten x dominiert y.
  Reflexiv-transitiver Abschluß von >
- `(>+ x y)`
  Knoten x dominiert y echt.
  Transitiver Abschluß von >
- `(. x y)`
  Knoten x steht unmittelbar links von y.
- `(.. x y)`
  Knoten x steht links von y
  Reflexiv-transitiver Abschluß von `.`

Atomare Formeln: Knotenrelationen

- `(ref x y)`
  `ref`-sekundäre Kante von x nach y
- `(= x y)`
  Knoten x und y sind gleich.
- `(!= x y)`
  Knoten x und y sind verschieden.

- `(sent Exp)`
  Satz (Tokensequenz) paßt auf RA Exp.
  (Einzige atomare Formel ohne Variablen)

Komplexe Formeln

- `(! φ)`
  Negation von φ
- `(& φ1 ... φn)`
  Konjunktion
- `(| φ1 ... φn)`
  Disjunktion
- `(-> φ ψ)`
  Implikation
- `(E x φ)`
  Existentielle Quantifikation
- `(A x φ)`
  Universelle Quantifikation

Semantik

- Eine Abfrage ist ein erststufiger Satz (Formel, in der alle Variablen abquantifiziert sind).
- Dieser Satz kann wahr oder falsch sein für eine endliche Struktur, die einen Baum repräsentiert.
- Die Antwort auf eine Abfrage ist die Menge der Strukturen, für die die Abfrage wahr ist.
- Also: Klassische modelltheoretische Semantik.

Architektur von fsq

fsq besteht aus vier Komponenten:

- Initialisierungskomponente
- Suchmaschine
- Grafische Benutzeroberfläche
- Baumbankenbrowser

fsq ist implementiert in Java (JDK 1.5).
Es verwendet die Graphikbibliothek yFiles von yWorks.

Plattformunabhängige Implementation, die unter Linux, Solaris, Windows, and Mac OS X läuft.

(Zuletzt getestet unter Linux und Windows.)

Initialisierung

Baumbankformat für fsq: NEGRA Exportformat

- entwickelt an der Universität Saarbrücken
- Datenaustauschformat
- reiner ASCII-Text, für Menschen lesbar
- ungeeignet für schnelles Suchen
- Daher: Präkompilationsschritt notwendig.

Vorkompilierung

- Transformation des NEGRA Exportformats in eine Binärrepresentation für schnelles Suchen.
- Muß pro Baumbank nur einmal durchgeführt werden.
- Jeder Baum wird einzeln transformiert.
- Binäre Relationen (Dominanz, Links-Rechts-Ordnung, sekundäre Relationen) werden in zweidimensionale Arrays übersetzt.
- Unäre Relationen (Kategorien, Funktionen, etc.) werden in eindimensionale Arrays übersetzt.

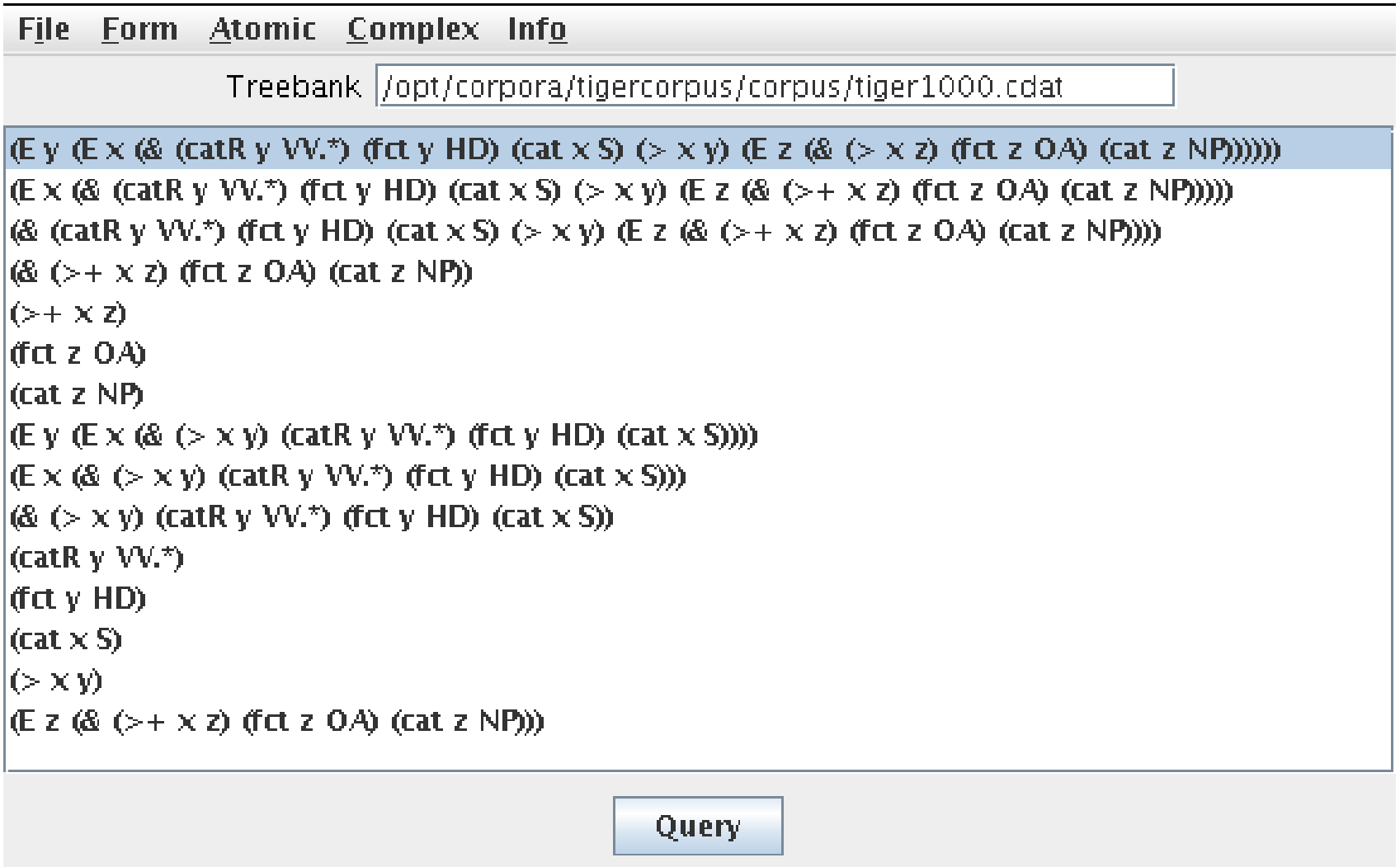
Suchmaschine

- Parst eine eingegebene Abfrage.
- Wertet die geparste Abfrage aus.
- Die Abfrage wird auf jedem Baum einzeln ausgewertet.
- Abfragen sind (als Formeln) rekursiv definiert.
- Die Auswertung einer Formel auf einem Baum erfolgt durch Rekursion über den Aufbau der Abfrage.

Grafische Benutzeroberfläche

- Hauptaufgabe: Unterstützung des Benutzers beim Erstellen von Abfragen.
- Außerdem: Abfragen auswählen, editieren, abschicken, speichern und laden.
- Auswahl einer Baumbank.

Grafische Benutzeroberfläche



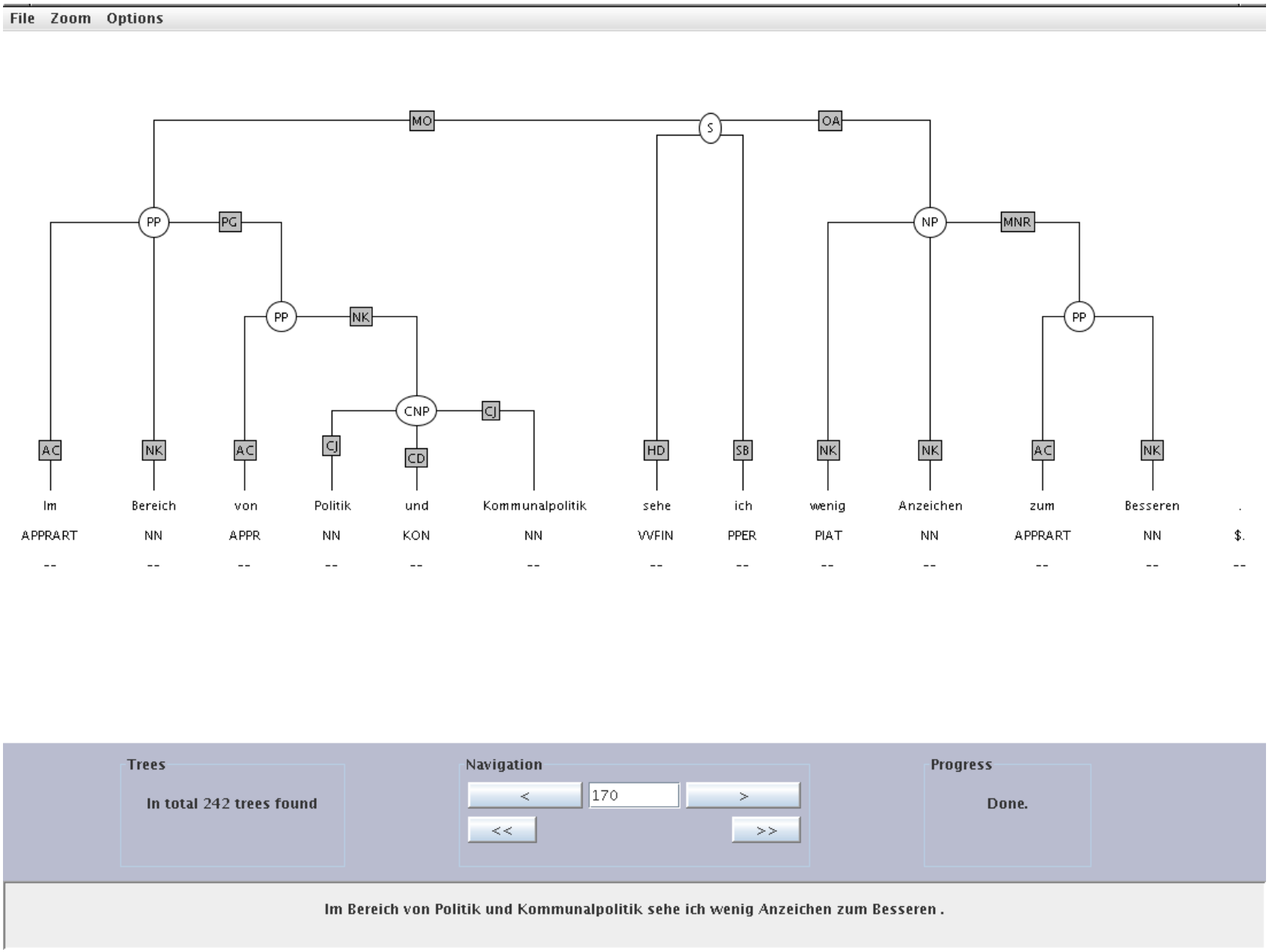
Baumbankbrowser

- Zeigt Treffer einer Abfrage an.
- Sobald die Suchmaschine den ersten Treffer findet,
  öffnet sich der Baumbankenbrowser und zeigt ihn an.
- Man kann die bereits gefundenen Treffer sich sofort ansehen.
- Die Suche läuft im Hintergrund weiter.
  Weitere Treffer werden an den Browser gemeldet.

- Der Baumbankbrowser kann auch die ganze Baumbank anzeigen.

- Optionen: Zoom, Druck, Export

Baumbankbrowser



Ausdrucksstärke der Abfragesprache

- Das Merkmal, das die Abfragesprache von fsq auszeichnet, ist beliebige Quantifikation von Knoten.
- Diese wird gebraucht zur Spezifikation von komplizierteren Beziehungen zwischen Knoten in einem Baum.

Überlegungen zur Komplexität

- Beim implementierten Algorithmus ist die Suchzeit polynomial in der Größe eines Baums.
- Genauer, wenn n die Anzahl der Knoten in einem Baum ist und k die Quantorentiefe der Abfrage,
  dann benötigt die Auswertung der Abfrage auf diesem Baum (maximal) n^k Schritte.
- Bei komplizierten Abfragen kann das zu längeren Suchzeiten führen.
- Aber so komplizierte Abfragen sind wahrscheinlich selten.
- Erfahrungswert: Suchzeiten für Abfragen bis zur Quantorentiefe 4 sind noch akzeptable.

Intelligente Reformulierungen von Abfragen

- Große Quantorentiefe kann häufig durch geschickte Umformulierung der Abfrage vermieden werden.
- Ein Beispiel: Suche nach den Token *heute*, *Treffen*, *in*, *Hannover*.
- Naive Abfrage:
  $\exists x \exists y \exists z \exists w (\ (tok\ x\ heute) \wedge (tok\ y\ Treffen) \wedge (tok\ z\ in) \wedge (tok\ w\ Hannover))$
  Quantorentiefe: 4
- Bessere Abfrage:
  $(\exists x (tok\ x\ heute)) \wedge (\exists y (tok\ y\ Treffen)) \wedge (\exists z (tok\ z\ in)) \wedge (\exists w (tok\ w\ Hannover))$
  Quantorentiefe: 1

Schluß

- fsq ist ein mächtiges Abfragewerkzeug.
  - Es kann zur Abfrage beliebiger endlicher Strukturen verwendet werden, nicht nur für echte Bäume.
  - Die Abfragesprache – erststufige Logik – ist recht ausdrucksstark.
  - Sie hat eine etablierte, gut verstandene Semantik.
- Nach Kenntnis des Autors ist fsq eines der mächtigsten Abfragewerkzeuge.
- Die Suchzeiten sind im allgemeinen kurz.

Verfügbarkeit

- fsq ist für akademische Zwecke frei verfügbar.
- fsq – Webseite: `http://tcl.sfs.uni-tuebingen.de/fsq`