

Informationsmanagement

Σ Alphabet = Menge von Symbolen/Zeichen

$s = s_1 \dots s_n \in \Sigma^*$ Zeichenkette

Daten - ausgetauschte Nachrichten

Interpretation
Zeichenketten nach Grammatik gebildet

Informationen + Interpretationsbezug

Visionserwart
(Semantik für Empfänger)

Wissen - Gesamtheit Kenntnisse/Fähigkeiten zur Problemlösung

+ Vernetzung zu bekannten Infos

Problem-
Lösung

Strukturierungsgrad

strukturierte Daten

gleichartige Struktur, Datenmodell folgend

z.B. Listen, Diagramme,
Tabellen, Java Objekte

semistrukturierte Daten

ohne festes Datenmodell

aber Daten implizieren Struktur oder erweiterbares Datenmodell

z.B. XML-Dokument

Object Exchange Model

„unstrukturierte Daten“

ohne formalisierte (inhaltliche) Struktur

z.B. natürlichsprachlicher Text

Bild, Musik (schwer für Computer)

(Text Mining, Natural Language Processing)

Inf. Management

Daten organisieren, strukturieren
speichern, abfragen, löschen, ändern
pflegen,
interpretieren, vernetzen
analysieren, verifizieren

Inf. System Anforderungen

Informationsenthalt
nötige Datei ablesbar

Konsistenzherhaltung
nur vernünftige Daten speichern
möglichst Redundanzfrei
Platz sparen, Anomalien vermeiden

Entwurfsprozess

1. Anforderungsanalyse
Fachkenntnisse, Terminologie, Inf. bedarf
 \Rightarrow Informelle Dokumentation z.B. Interviews, Texte

4. Logischer Entwurf impl. spezifisch, beweite Unabhängigkeit
konzeptionelle Modelle \rightarrow Konzepte eingesetztes Inf. System
 \Rightarrow Logische Datenmodelle z.B. relationales Modell
Wie?

2. Konzeptioneller Entwurf / design
Abstrakte Modellierung Domäne
Objekte, Beziehungen?
 \Rightarrow formale Beschreibung z.B. ER-Modell
Welche Daten? UML-Struktur

5. Datendefinition
Deklaration / Programmierung Datenmodell in Inf. System
 \Rightarrow Definitionssprachen z.B. SQL

3. Verteilungsentwurf / distributed system design
Fragmentieren Daten,
Sync & Replica

6. Physischer Entwurf
Zugriffs-, Speicherstrukturen auf disk

7. Implementierung & Wartung

Interne Verarbeitung Anfragen
Anfrageoptimierung

Relationale DB's

Backend Unternehmensdaten

Web-Bereich

Android, Mac OS, SQLite, ...

SQL wichtiger Skill (auch Python, R)

(2) Konzeptionelle Datenmodellierung (Externe Ebene DB)

Anforderungen → konzep. Datenmodell

Konzeptionelle Datenmodellierung

Objekte, Merkmale, Beziehungen, Regeln?

Entity-Relationship (ER) Modelle

statische Eigenschaften (keine Funktionen / Methoden)

beliebt \Rightarrow „Laien“ verständlich
intuitiv

graphische Notation nach Chen

Entitätstyp

Menge Entitäten gleichen Typs
(individuell identifizierbare Objekte
Instanzen, E-Typ)

Entitätstyp Rechteck

Entitätstyp (Attributes, ...)

Beziehungstyp

zwischen ≥ 2 Entitätstypen

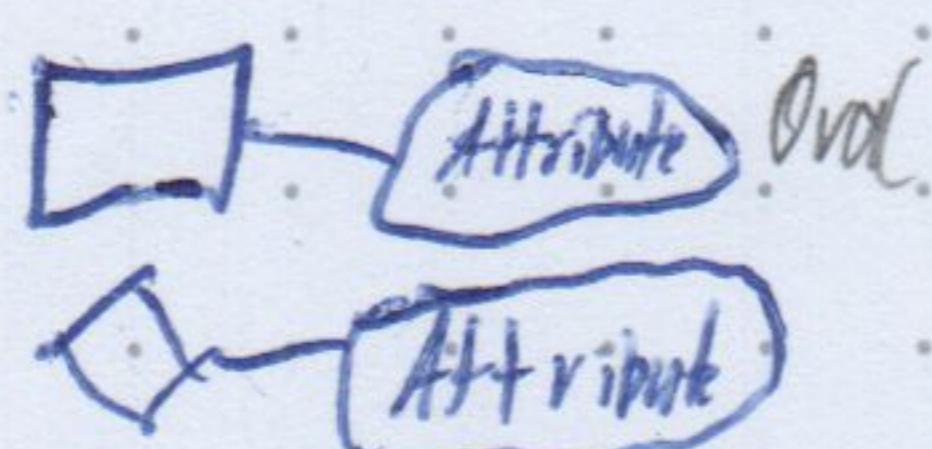
Relation $\subseteq E_1 \times \dots \times E_n$

Beziehung kontaktiert zwischen
Entitäten



Beziehungstyp ($E_1, E_2, \dots; \text{Attributes}, \dots$)

Attribute / Merkmale



Schlüsselattribute

\subseteq Attribute Entitätstyp
identifiziert Entität eindeutig
Modellierungsentscheidung
ggf. künstlich z.B. ID

nur Entitätstypen
Unterstreichen

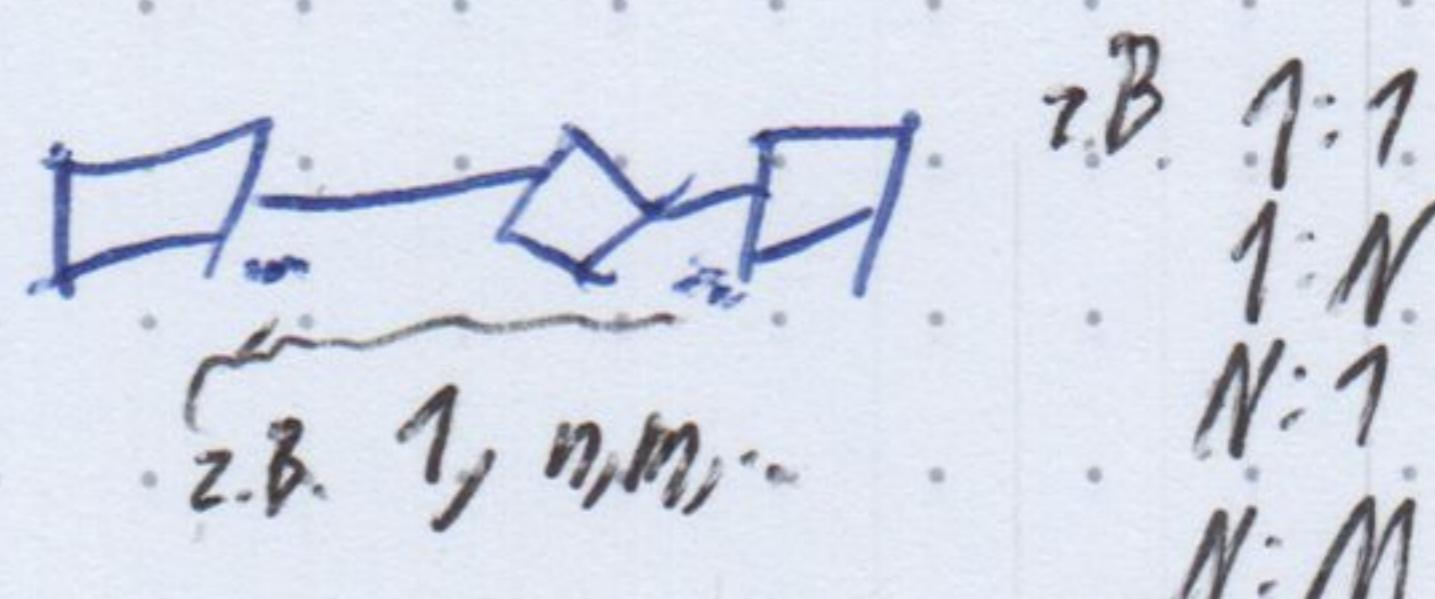
Attribut

Rekursiver Beziehungstyp

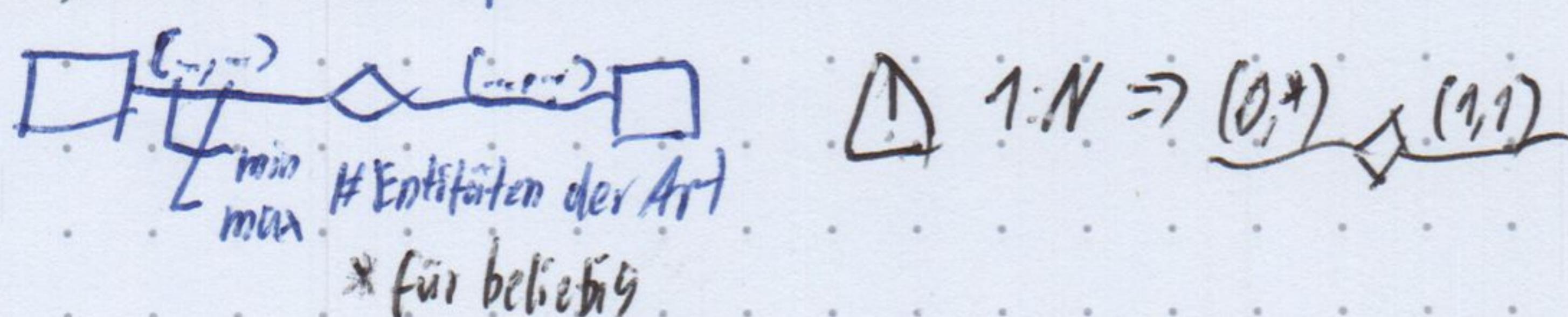


Funktionalitäten / Kardinalitäten für Beziehungstyp

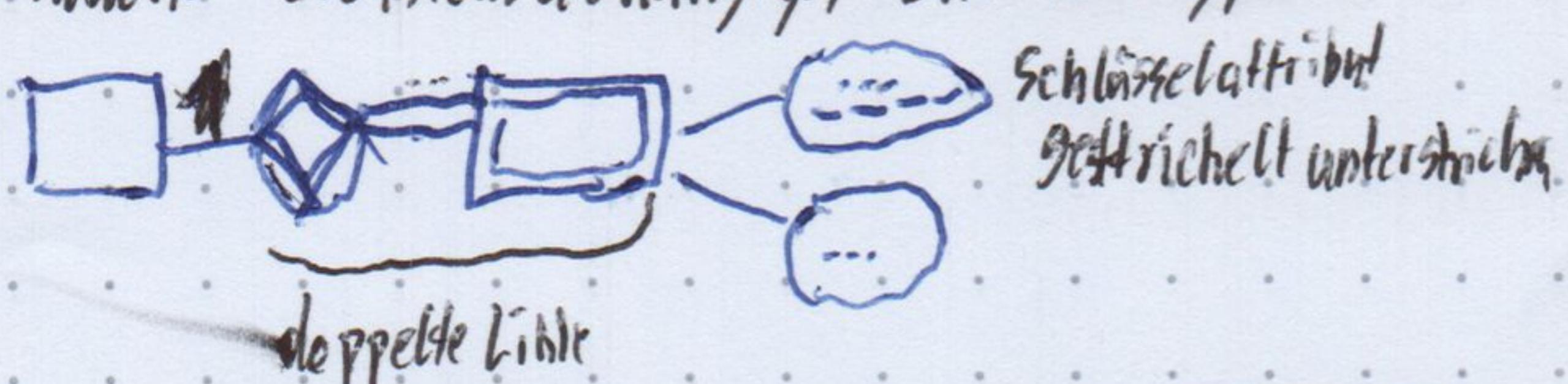
konkrete Beziehungen Entität max. Beteidigt



Δ (min, max) - Notation präziser als Kardinalitäten

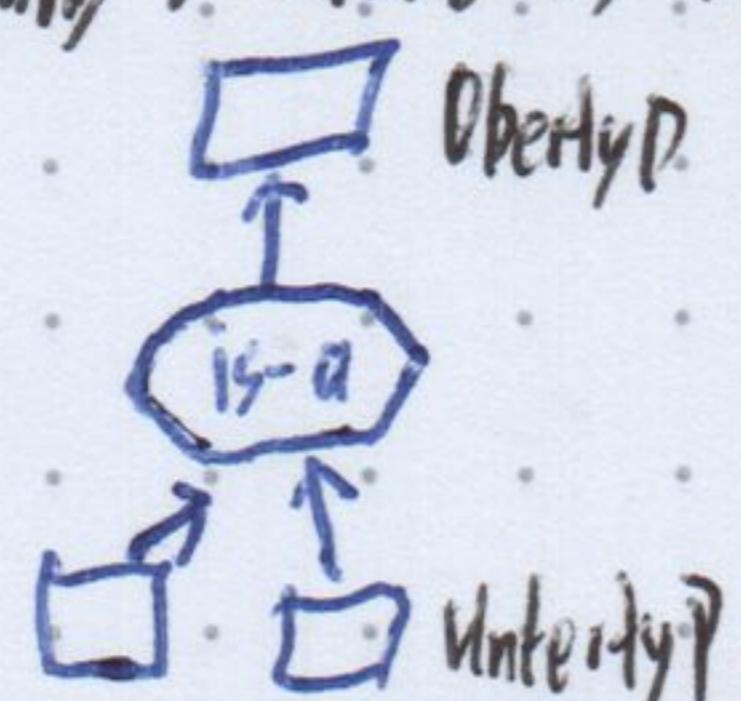
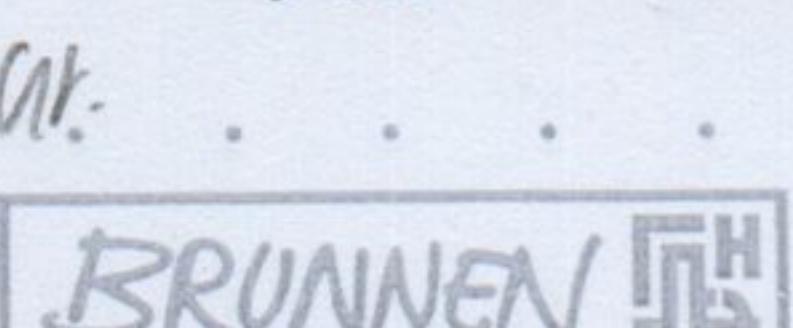


Schwache (Existenzabhängige) Entitätstypen

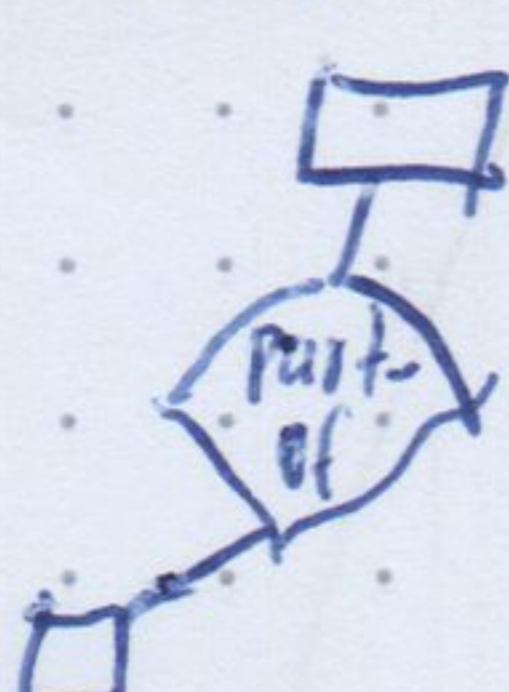


nicht
klassif.
relevant

Generalisierung / Vererbung der Attribute



Aggregation (Teil-Ganzes Beziehung)



Relationale Algebra Abfragesprache

RWK - Relationenwertebereichkalkül

RTK - Relatontupelkalkül

RA - Relationenalgebra

Keine Duplikate

Codd's Algebra

Selection

$\sigma_p(S)$

Konstantenselektion
Attribut D Konstant
Attributselektion
Attribut D Attribut
Logische Verknüpfung \vee, \wedge, \neg

$\{t = t_1, t_2, \dots, t_n\}$

Projektion

$\pi_L(S)$

$L \subseteq R$

Karte/tisches Produkt

Mengenoperatoren bei Vereinigungsreträglichkeit
auf Tupelmengen
gleicher Grad = # Attribute
gleiche Domänen

Vereinigung

$R \cup S$

Durchschnitt

$R \cap S = R - (R - S)$

Differenz

$R - S$

Erweiterungen

Join \bowtie natürlich \leftrightarrow \bowtie_p allgemein

"vereinigt gleichnamige"

$R_i = \dots$
Attrib. bleiben doppelt

Join

$R \bowtie S =$ in beiden drinnen

Full Outer Join

$R \bowtie L =$ fehlende Tupel Attr=Null

Left "

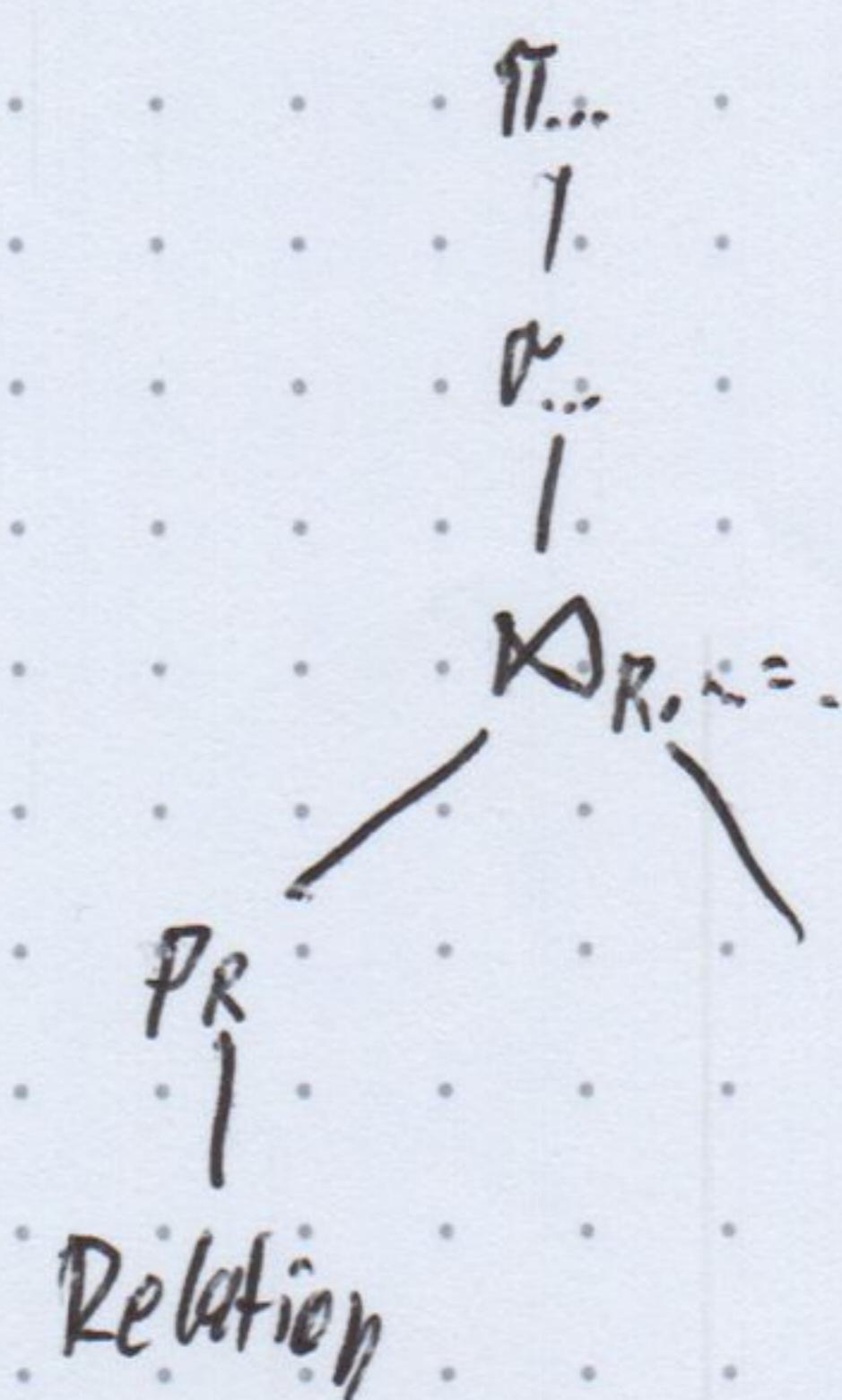
~~$R \bowtie L$~~ $R \bowtie L =$ alle aus R

Right "

$R \bowtie L =$ alle aus L

Left / Right Semi-Join

$R \bowtie L = \pi_R(R \bowtie L)$



Relationale Division

$$R \div S = \{t \mid t \in R \text{ und } \forall s \in S : t \in s \text{ und } \exists r \in R : r \in t \text{ und } r \in s\}$$

Attribut

Projektion

$$= \Pi_{(R-S)}(R) - \Pi_{(R-S)}((\Pi_{(R-S)}(R) \times S) - R)$$

gruppieren nach R-S alle R-S zurückgeben für die Einträge für R-S existieren

Umbenennung

Relation $P_1(S) = V_1$

Attribute $P_1, \dots, P_m, A_1, \dots, A_n$

BRUNNEN

Relationales Datenmodell beim logischen Entwurf

aus Relationen \rightarrow Tabelle

keine Reihenfolge Attribute
Tupel

$D = \{D_1, \dots, D_n\}$ Menge Datentypen/Domänen

$R = \{A_1, \dots, A_n\}$ Schema
dom: $R \rightarrow D$

$r(R) \subseteq \text{dom}(A_1) \times \dots \times \text{dom}(A_n)$
Relation-Ausprägung von R

$t \in r$ Tupel

$t(A_i)$ Wert Attribut

$t(\alpha)$ teiltupel $\alpha \in R$

$R(A_1 : 1:n, A_2 : 1:n, \dots : 1:n)$
 \uparrow
PK

FKs implizit durch Namensgleichheit

null Wert möglich

Schlüssel Attributmenge K (einfach $|K|=1$
zusammengesetzt $|K|>1$
trivial $K=R$)

Superschlüssel

$\forall t_1, t_2 \in r : t_1(K) = t_2(K) \Rightarrow t_1 = t_2$

UI

Schlüsselkandidat / Schlüssel

K minimal $\Leftrightarrow \nexists k' \subset K : k' \text{ schlüsselerindetig}$

UML

$K_R = \{E_1, \dots, E_n\}$

Primärschlüssel (PK)

1 gewählter Schlüsselkandidat

$PK_R = \{E\}$

Fremdschlüssel

$K' \in \text{Schema } S$ bezüglich Schema R $\Leftrightarrow p(K') = \text{PK von } R$ Referenz

Referenz

Referenzialer Wert erlaubt

$FK_R = \{D \rightarrow S.D\}$

$A \rightarrow S.A'$

Referentielle Integrität

$\forall t_S \in S \exists t_R : t_R(K') = p(t_S(K'))$

ER \rightarrow Relationales Modell

Regel 1

EntitätsTyp \rightarrow Schema

\hookrightarrow Schlüssel \Rightarrow PK fulls minimal
sonst neu

Regel 4

Schwache Entitäten

Fremdschlüssel für starke
Teil PK

Regel 2

Beziehungstyp \rightarrow Schema

$R(E_1, \dots, E_n)$ $R(A_1, \dots, A_m)$

PKs aller Attr.

von E. mit Kardinalität
 $N_1 \neq \text{folk}$ keinen einer
mit 1

Fremdschlüssel E_1 Eigene Attr

Benennung

{Name der referenzierten(falls eindeutig)}

{künstlicher Name z.B. EntitätsnamenattrName}

Rollenname falls vorhanden

Regel 3

Zusammenfassen Schemata mit gleichen PKs
Ausnahme: schlecht wenn Ergebnis dünn besetzt

SQL - Structured English Query Language \leftrightarrow Relationale Algebra

auf Multimengen
deklarative Anfragesprache für rel. DB
Imperativ ohne Duplikate

DQL - Data Query Language
 DDL " Definition "
 DML " Manipulation "
 DCL " Control "

DQL

select [distinct] * | Attr₁, Attr₂, ...

from Table₁, ...

[where Prädikat]

{ kombinieren AND, OR, NOT }

{ Vergleichop. =, <, <=, >, >=, ... }

- between ... and ...

case (when ... then ...) * [else ... end]

[natural join Table]^{*}

[cross join Table on Table.Attr = Table.Attr]^{*} Reihenfolge kann durch Klammern

[inner], [left|right|full] outer join

Pattern Matching Strings ... like '%'

{ % beliebiges Zeichen }

Zeichenkette auch _

[group by Attr₁, ...]

[having ...]

// Aggregationsfunktionen

count, sum, avg, max, min

])

// Relationale Algebra

Π select (#0 having (X group by (o where (from))))

alle in select, having, in group by oder Aggregate

[order by Attr₁ [asc|desc], ...]

'ascending'
default asc.

[limit n] ; MySQL, PostgreSQL; select top n : SQL Server; fetch first n rows only : Oracle

Umbenennung von Tabellen in from [as] newName
Attr. in select [as] newName

Unterabfragen (select...) in Äußerem select, wenn nur ein Aggregat in select Unterabfrage, alt Wert

from als Tabelle

where als Wert

als Tabelle

{ EXISTS T - T nicht leer }

s IN T

s op ALL T

ANY

cast (value/attr as type)

Relationale Division select (R-S) from R group by (R-S) having count(*) = (select count(*) from S)

NULL-Werte

vergleichsoperator unkennen \Leftrightarrow min 1 Operand Null

dreiwertige Logik and, or, not \rightarrow unkennen außer Kurzschlusslogik

Aggregatfunc. ignorieren NULL

Gruppierung NULL eigene Gruppe

zusätzliche Ops

... is NULL

... is not NULL

Recent Directions

Data Luke - Tabellen ohne Schema damit automated Join Detection

NL-to-SQL Natural Language $\xrightarrow{\text{LLM}}$ SQL stat.

Relationale Entwurfstheorie

Redundanzen

- Speicherplatzverschwendum

kann
⇒ potentiell Anomalien ^{kann} ⇒ Inkonsistenz
[Update nicht alle Stellen eines Tupels, stattdessen]
[Einfügen benötigt erl. Dummy daten]
[Löschen z.B. Best. löschen Lieferanten mit,
wenn in keinen anderen Best.]

$\beta \in R$ von $\alpha \in R$ funktional abhängig (α Determinante für β)

will FD $\Leftrightarrow \alpha \rightarrow \beta \wedge \nexists \alpha' (\alpha : \alpha' \rightarrow \beta$
parallel "

β transitiv abhängig von α
 $\Leftrightarrow \exists r: r$ voll FD von α u. β voll FD von r
 $\alpha \rightarrow r \rightarrow \beta$

Attributhülle $\alpha^+ =$ Menge von funk. abhängige Attribute
Algorithmus
 $\alpha^t = \alpha$
while Änderung
 foreach FD A-B
 if $B \subset \alpha^t$ then $\alpha^* = \alpha^t \vee r$

Schlüsselkandidaten aus FDs

1. Attr. nicht auf rechter Seite FDs
⇒ muss in Schlüssel. vorkommen
 2. Attributhüllen der Determinanten bestimmen
durch ⇒ Liste Superschlüssel bestimmen und welche Attr. fehlen
 3. Minimieren Superschlüssel

Schlüsselmenge

$\text{KEYS}(R)$ = Menge Schlüsselkandidaten

alternative Schlüsse (nicht PK Schlüsselkandidaten)

Primattribut $\exists K \in \text{Keys}(R)$: AEK - Teil eines Schlüssels

NPA - Nicht primatibus -

NFAC(R)-Menü dieser

Normalisierung Prozess schlechter Rel. Modell \rightarrow gutes durch Zerlegung

Anforderung

1. redundanzfreie Relationen
2. große Relation verlustfrei ableitbar

Kriterien korrekte Zerlegung

1. Verlustlosigkeit

$$r = r_1 \bowtie r_2$$

2. Abhängigkeitsbewahrung

1 NF - Erste Normalform

$\Leftrightarrow \forall$ Attributwerte a formar

L nicht mehrwertig von Interpretation
L z.B. Listen abhängig

\Rightarrow teilen

VI

2 NF

\Leftrightarrow in 1 NF $\wedge \forall A \in \text{PAIR}(R) : \forall k \in \text{Keys}(R) : A$ null abhängig von K

\Rightarrow Intuitive Zerlegung

VII

3 NF

\Leftrightarrow in 2 NF $\wedge \forall A \in \text{PAIR}(R) : \forall k \in \text{Keys}(R) : \exists Y \in \text{Schema}(R) : k \rightarrow Y \rightarrow A$ (transitive FD)

\Rightarrow Intuitive Zerlegung

Synthesealgorithmus R mit FD's \mapsto 3NF mit Anf.

1. Bestimme „kanonische Überdeckung“ aller FDs in R

Linksreduktion

Rechtsreduktion

Entfernung FDs der Form $x \rightarrow \emptyset$

Zusammenfassen gleicher linker Seiten

2. $\forall \alpha \rightarrow \beta : \text{Erstelle } R := \underbrace{\dots}_{R^*} \alpha \cup \beta$

3. Schlüssel

4. Entferne R wenn $\exists R' : R \subseteq R'$

Anfrageverarbeitung

Anfrage $\text{select } A_{1..n} \text{ from } R_1..n \text{ where } P$

J.A.-Compiler

Nicht-optimierter
Logischer Plan

J.A.-Optimierer

Optimierter
Physischer Plan

Ausführung

Optimierung

Regel-basiert Datenunabhängig
auf algebraischer Ebene

e.g.

1. Selektions-Aufspaltung

$$\text{Konjunktion } \sigma_{c_1, \dots, c_n}(R) = \sigma_{c_1}(\sigma_{c_2}(\dots(R)))$$

weitere Äquivalenzen

X, V, \wedge, \vee kommutativ

A, X, V, \wedge, \vee einzeln assoziativ

σ distributiv mit $V, \wedge, -$

2. Selections-Pushdown

$$\text{Kommutativ } \sigma_{c_1}(\sigma_{c_2}(R)) = \sigma_{c_2}(\sigma_{c_1}(R))$$

$$\pi_{A_{1..n}}(\sigma_c(R)) = \sigma_c(\pi_{A_{1..n}}(R)) \text{ falls } c \text{ sich nur auf } A_{1..n} \text{ bezieht}$$

$$\sigma_c(R M S) = \sigma_c(R) M S$$

3. Join Ersetzung

$$\sigma_c(R \times S) = R M_c S$$

4. Projektions-Pushdown

$$\pi_{L_1}(R) = \pi_{L_1}(\pi_{L_2}(\dots(R))) \text{ falls } L_1 \subseteq L_2 \subseteq \dots$$

$$\pi(A(R)) = \pi(\pi(R)) \text{ falls } A \text{ noch möglich}$$

$$\pi_A(R M S) = \pi_A(R) M \pi_B(S) \text{ falls } A = LNR$$

$$\pi(R \times S) = \pi(R) V \pi(S) \text{ Tidistributiv mit } V$$

5. Zusammenfassen Operatorenfolgen gleicher Op.

Kostenbasierte Optimierung Datenabhängig
Schätzung durch Kostenmodell

1. Auswahl Join-Reihenfolge

Schätzung Zwischenergebnis-Größen

BottomUp | Tabellen-Größen aus DB-Katalog

$$\text{sel}(\sigma_p(R)) = |o_p(R)| / |R|$$

Schätzung \Rightarrow Annahme Gleichverteilte Attrib. Werte

$$\text{sel}(\sigma_{A=\text{konstant}}(R)) = 1 / |A|$$

$$\text{sel}(R M S) = |R| M |S| / |R \times S|$$

$$\text{sel}(\sigma_{A=B}(R)) = 1 / \max(|A|, |B|)$$

2. Auswahl physische Operatoren

1 über Selektivität der Operatoren $\geq 5\%$

Stichprobverfahren

Selektivität für Stichprobe berechnen

Histogramm für Attrib. Werte im Wertebereich

entl. Ausreißer, extra; manuelle Änderungen

multidimensionale Hist. für zusammengest. Attrib.

$$\text{sel}(\sigma_{p_1}(R)) = \text{sel}(\sigma_{p_1}(R)) \cdot \text{sel}(\sigma_{p_2}(R))$$

$$\text{or } p_1 + p_2 - (p_1 \cdot p_2)$$

$$\text{sel}(R M_{\text{frenschlüssig}} S) = |S| / |R \times S|$$

$$\text{sel}(R M_{A=B} S) = 1 / \min(|A|, |B|)$$

Suchbaum

$$H \text{ Binärbaum mit } n+1 \text{ Blättern } \approx \text{Catalan-Zahl } (n = (2n)! / ((n+1)! \cdot n!))$$

Unterschiedliche

BRUNNEN

→ klassischer Ansatz: Dynamische Programmierung

BottomUp | Phasen je Optimum für Pläne mit k Relationen

Pruning schlechter

9

Transaktionsverarbeitung

↳ Folge logisch zusammengehörige SQL-Anfragen
änderungsop.

ACID-Garantien

Logging & Recovery	<ul style="list-style-type: none"> Atomicity Consistency - Datenbank vor und nach Tr. ... Isolation unter Nebenläufigkeit Durability Erg. Speicherung nur in Erfolgsfall 	<ul style="list-style-type: none"> - constraint checking - concurrency control
--------------------	--	--

Anomalien durch Nebenläufigkeit

Lost Update - Änderungen in T_1 durch T_2 überschrieben

Dirty Read - T_2 liest partielle nicht komittete Änderung

Non-Repeatable Read - wiederholtes Lesen T_2 's diff. Ergebnisse

Phantom Read - wiederholte Anfragen T_2 's diff. Datensätze (von T_1 eingefügt)

⇒ Concurrency Control

Operationen ~ Read/Write Datenbankobjekte $\theta_1, \theta_2, \dots$
 ↳ typischerweise Datensatz (Zeile)
 ↳ Attribute, ...

Transaktion ~ geordnete Schrittfolge

$$T_i = \langle s_i^1, \dots, s_i^n \rangle$$

$$\left| \begin{array}{l} r_i(\theta_k) \quad w_i(\theta_k) \\ [b_i = \text{begin} \quad c_i = \text{commit}] \quad a_i \text{ abort} \end{array} \right.$$

Schedule S für $T = \{T_1, \dots, T_n\}$ ~ (partielle) Ordnung Schritte
Ein-Prozessor-System: totale Ord.

Reihenfolge innerhalb T_i erhalten

serieller Schedule: s_i^j in T_i konsekutiv + keine Anomalie
serialisierbar \Rightarrow 3 äquivalenter serieller Schedule - Performance
 ↳ gleiche Leseps Err.
 ↳ gleiche Err.

⇒ Konfliktgraph azyklisch

Knoten ~ Transaktionen
Kanten ~ Konflikt $\Rightarrow s_x(\theta_k) \text{ vor } s_y(\theta_k)$
 s_x oder s_y Schreib op.

Implementierung Scheduler

pessimistische Ansätze - Konflikte verhindern vermeiden

Lock-based Concurrency Control

naive - lock während op

! Garantiert nicht Serialisierbarkeit

2PL - Two-Phase Locking - pro Ti: Grow; Release-Phase

! Keine neuen Locks

! Deadlocks => Auflösen durch Timeouts, WaitGraphen

=> Preclaiming

! autom. # Locks vor BOT - begin of transaction

! Cascading Aborts liest T2 Werte von T1 darf T2 nicht vor T1 commit & h

=> Strict 2PL

! autom. # Locks freigeben zum EOT

SQL 92 Isolationsstufen Durchsatz \rightarrow Anomalien

Durchsatz Read Uncommitted V Anomalien möglich nur Lesersps



Read Committed Write-Locks mit Strict 2PL keine Dirty Reads

Read Stability Strict 2PL nur Phantom Reads

keine Anomalien Serializable mit locks auf Wertebereichen keine Anomalien

Optimistische Concurrency Control + ohne Konflikte effizient

Phasen

1. Execution in privatem Bereich
2. Validate konfliktfrei
3. Installation zu Datenbank

Recent Directions

NL-to-SQL

Data Lakes (Tabellen ohne Schema)

Information Management

Data: organise, structure, query
store, create, modify, delete
interpret, link, analyse, verify

Data Growth of unstructured Data

Science
Business
Industry
World Wide Web

containing implicit information
e.g. for Recommender System, best products for supermarket
predict subset of items R.S. interesting for user

Machine Learning - ML
tasks experience (feedback)
⇒ perform well

Document Classification
document of topics \mapsto T
 \Rightarrow T \in T describing of

NLP - Natural Language Processing

Human Language Technology focus
Computational Linguistics tech
analysis

NLU - understanding transform language \rightarrow machine representation
G - generation

Machine Translation - MT

feat in ~~text~~ lang A
 \Rightarrow text in B with same meaning

Information Retrieval - IR

find subset of document collection D
maximally relevant to query

Extraction - IE

D \mapsto structured knowledge

Text Generators

prompt \Rightarrow continuing / fulfilling text

Ambiguity

Homograph \leftrightarrow Homophone
same writing voice / Aussprache
diff meaning

Adverb \leftrightarrow Adjective

Pragmatics

utterance

System Evaluation

Intrinsic - in isolation, compare to predefined expectation (gold standard)

Extrinsic - in use in larger system / experimental setup

Natural Language Text

Script Σ

Symbol/Character $s \in \Sigma$

String $s = s_1 \dots s_n \in \Sigma^*$

Message string, following grammar

Data exchanged messages (not interpreted)

Text (quasi-) written coherent state of language

| connecting ideas
⇒ semantic meaning

Properties

type/genre article, ...

register used lang. variety (dialect, formal, ...)

style how written (funny, polite, ...)

domain/topic

time of writing (changes over time)

Language \leftrightarrow Script

{vocabulary}

„alphabet“

alphabetic Latin, ...

Logographic / syllabic Hanzi, ...

consonant-based (Abugida) Arabic, ...

segment - (Abugida) Devanagari, ...

Writing direction

Electronic Text Formats

Plain text) mostly unstructured

HTML) semi-structured

XML) e.g. paragraphs

.odt, .docx) XML-based formats e.g. ZIPed XML files

.doc, .rtf) binary content

.pdf) mixed text, binary

Storing Text in Relational DBs

with fixed or maximum length

CHAR(n), VARCHAR(n) typically ≤ 4000

arbitrary length

BLOB - Binary Large Object

e.g. for files e.g. images

CLOB - Character Large Object

e.g. text

typically

no sorting/grouping

limited comparison (=, <, >, BETWEEN, ...)

Encoding Issues

default of source files EIDE

String types e.g. UTF-16 in JAVA

files OS

table

database connection

Variable length $\Rightarrow n^{\text{th}}$ char

Search

Sort - lang. specific SET

Character Encoding

functions

$$\text{enc}: \Sigma^* \rightarrow \{0,1\}^*$$

$$\text{dec}: \{0,1\}^* \rightarrow \Sigma^*$$

Typically ~~Zentrale~~ character set

$$\Psi: \Sigma \rightarrow \{0,1\}^*$$

$$\text{enc}(c_1 \dots c_n) = \Psi(c_1) \dots \Psi(c_n)$$

Types

$$\text{Single Byte } \Psi: \Sigma \rightarrow \{0,1\}^8$$

e.g. ASCII, ISO 8859-1, CP-1252

$$\text{Multi Byte } \Psi: \Sigma \rightarrow \{0,1\}^n$$

e.g. UCS-2, UTF-32

Variable Length depending on char and
e.g. UTF-8, UTF-16 predefined rules

Escape-code-based switching between
e.g. ISO 2022 character sets

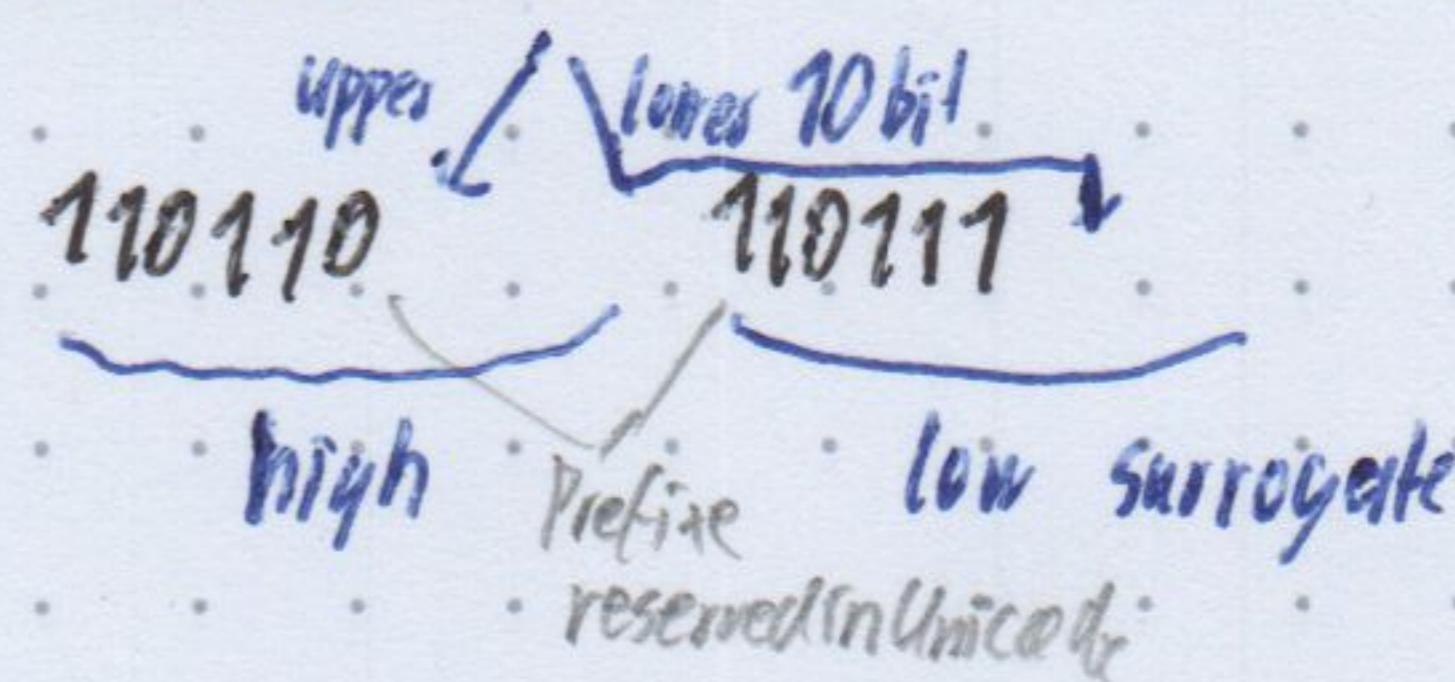
UTF-32 4 byte direct Unicode correspondence - slower, bigger than ISO 8859-1

UCS-2 2 byte only BMP deprecated

UTF-16 2-4 byte

| BMP \Rightarrow direct correspondence

| supplementary Plane \Rightarrow subtract 0x10000



UTF-8 1-4 byte direct ASCII correspondence

leading 1s followed by 0 = N bytes (0 statt 10)

continuation bytes start with 10

(theoretically allows up to 6 bytes)

ASCII 7 significant bits, fahrende 0
control codes

Sonderzeichen

Zahlen ab 30₁₆

Buchstaben groß 41₁₆
klein 61₁₆

Single
Byte

ISO 8859-x Single Byte

(lower 128 symbols \Rightarrow ASCII)

upper \Rightarrow language specific
e.g. x=1 \Rightarrow Western Europe
basically one per major lang.

Unicode idea: 1 charset for all incl. logographic, fantasy lang., currency symbols, emojis

4 Byte

21 significant bit

written U+ppxxxx

17 Planes 0x00 to 0x10

pp=0 - BMP - Basic Multilingual Plane

- Latin, Arabic, most Chinese, -

pp=1 - SMP - Subsupplementary Multilingual Plane

- Historic Lang., emojis, -

2 - SIP - Supplementary Ideographic Plane

- Extension for Chinese, -

3-13 empty

14 - SSP - Supplementary Special Purpose
control chars

15-16 - PUA-A/B - Supplementary Private Use Area

- private chars of certain font

Linguistic Preprocessing

Segmentation

Tokenization input stream → ordered seq. of tokens

segmented into

split at whitespaces?

A cents. \$1.25

Ambiguities

Periods

| sentence termination → separate token

| abbreviations, ordinal numbers, urls

commas, Whitespace

|

| in number

Single quote

| enclosing quote

| contractions, elisions

Dash

| part of token

| ranges

Colon

| sentence delimiter

| time expr.

↳ inflected word forms

, e.g. Chinese: no spaces

need context info

← Morphology - study of word forms / formation

Morphemes - smallest meaning-bearing units.

| Free " / Stamm - useable in isolation

| open class words e.g. nouns, a cat =

| closed conjunctions

contracted forms

| Bound " / Affix - to words, often for inflection

| Suffix

| Prefix e.g. -s für Plural

| Infix fan + bloody + taste e.g. fügenlaute

| Circumfix yet sagtt.

Word Formation / Bildung

Derivation

Stamm + derivation affix

Conversion / zero derivation

change part of speech of stamm. without affix

Composition / compounding

Linking, Stamms

Decomposition useful for "productive" lang.
e.g. German

Agglutinative lang. combine affixes
e.g. Turkish

Orthographical Normalization

Stemming - strip word endings

same family → similar stemmed repr.

Errors

Understemming related words → diff. stamm

Over diff. words → same stamm

Ambiguity

e.g. Homographs

Lemmatization - undo inflection / base form

typically requires part of speech

deals with irregular forms

Syntax - regularities & constraints, word & phrase structure.

Part of Speech Tagging (POS Tagging)

Tagset of word classes

e.g.

Noun	Proposition of by, to
Verb	Pronoun
Adjective	Determiner the, a, that
Adverb	

L = lexical class

⇒ valuable info for

word formation, lemmatization

possible neighbors [Lm]

prenunciation, speech synthesis

semantic word sense disambiguation

shallow parsing

Ambiguities

Approaches

Rule-based

dictionary: word form → pos tag ~ maybe 90% correct

if unknown to most common (noun)

iterative (learned) ordered transformation rules

Probabilistic

estimate $P(\text{word}, \text{context}, \text{pos tag})$

⇒ most probable

Learned from manually labeled training data

Parsing: determine grammatical structure of sentence

Phrase structure grammar

decompose sentence into constituents

L = wordgroup behaving as single unit

generally closed

Substitution e.g. with it

Movement as group in sentence

Coordination group and sm

Question yielding constituent

mostly Phrases - word seq., yielding syntactic VUnit

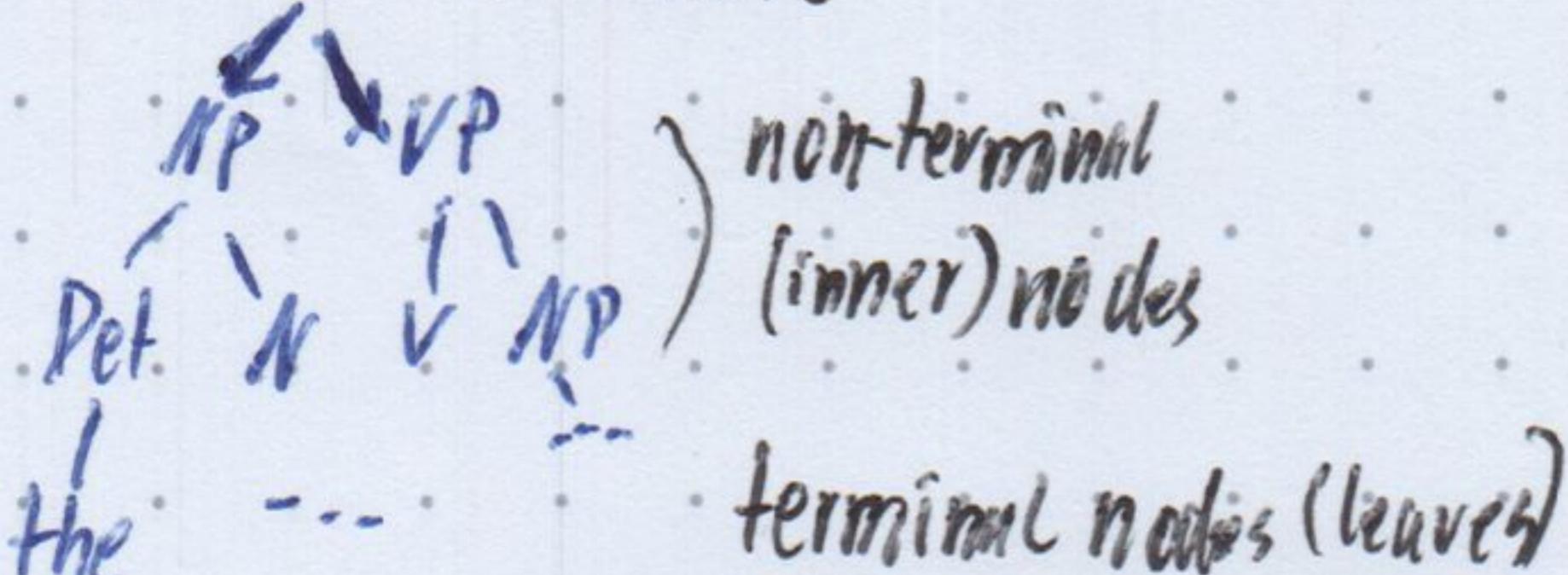
head / "key" word determines syntactic type

Grammatical modifier optional, "modify"

{ Pre modifier before head meaning

Post " after head

Parse tree S root node



Bracketed Notation [S [NP ...] ...]

Parenthesized notation (S (NP ...))

↓

Syntactic Ambiguity ≥ 1 several sync. structures

Attachment " - constituent can be added to tree at diff places

Coordination " - varying conjunction scope

Garden Path sentence ⇒ obvious parse doesn't work

e.g. The old man the boat

BRUNNEN

e.g. CFG - context free grammar

$G = (T, N, S, R)$

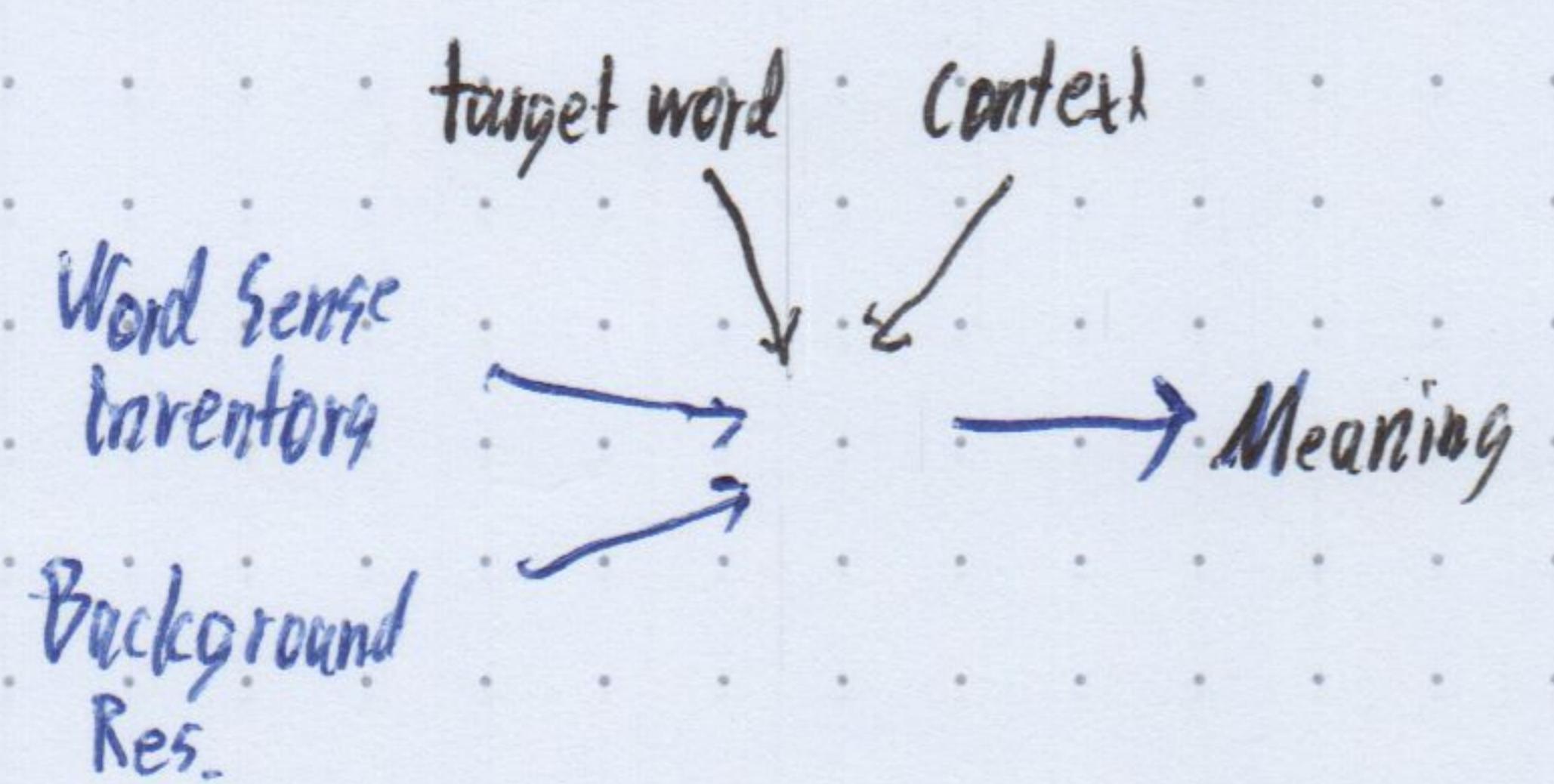
terminals, non-t., startsymbol, Production Rules, $N \rightarrow (T \cup N)^*$

16

semantics - meaning /
Lexical " - " of lexical items (words)
Structural " - relationships meaning of lexical items in context

Lexical Ambiguity
syntactic

WSD - Word Sense Disambiguation



Pragmatics purpose/intention of utterance

Text Corpus - Text collection

real word + frequency analysis
+ basis for experimental res.

e.g. Brown Corpus - 1961 500 texts POS-tagged

Common Crawl - including copyrighted web

Treebanks - POS, parse tree

Parameters

Language

Monolingual

Multi-

Parallel - translated texts
ideally sentence aligned

Genre / Text type

Domain / topic

Time of compilation

Size

Static ↔ Dynamic

Communication

Written

Spoken Mix

Annotation level

- Word - POS, lemma, sense - expensive
- Phrase entities, multi-word expr - time consuming
- Sentence - syntactic tree, boundaries
- Discourse - co-referential chain
discourse segments

Storing Annotations

Inline - changes original

e.g. likes / POS-tag

Stand-off - separate file

e.g. at position ... starts ... of length ...

LOB - Annotation Scheme for spans

```

  | L Begin
  | - Outside
  | - Inside & continue
  e.g. B O B I

```

Corpora in Relational Databases

Words (Word-ID, Word, Frequency)

Word-Index (Word-ID, Sent-ID, Position)

Source-Index (Source-ID, Sent-ID)

Sentences (Sent-ID, Sentence)

Sources (Source-ID, Source)

Bigrams (Word1, Word2, Frequency, Significance)

Co-Occurrence within sentence (Word1, Word2, Frequency, Significance)

Workflow

- 1) retrieve, store original documents
- 2) convert to plaintext
- 3) segment
- 4) annotations
- 5) format
- 6) analyse/use

Lexical Resources derived from corpora

Lemma occurs once

Rare & frequent phenomena equal

meta info

describes word senses

e.g. dictionary, Encyclopedia

Thesauri, Wordnets, ...

Syntactic Res.
Semantic

synonym ↔ antonym
same opposite meaning

hypernym ↔ hyponym
broader narrower meaning

co-hyponym - same hyponym

troponym - for verbs hyponym

holonym ↔ meronym
X is whole part of Y

homonym

Transfer Learning | general/simila
r data
on other model ~ human learning
Fine-tuned pretrained model + less data hungry

Distributional Hypothesis

words in same context ~ similar meaning word space models
word/text similarity

LM - Language Model

- probability text sequence
- text completion/generation

Bi-gram LM

Word1 Word2 Count $P(W_2|W_1)$

$\langle \text{BOS} \rangle$
 $\langle \text{EOS} \rangle$

- Only predecessor context
- Unknown words
- poor semantic representation

Information Retrieval (IR) on unstructured

Information Need / intend
state requiring info
to solve problem

→ retrieve relevant documents
RE - Retrieval Engine

↔ DBMS on structured data

Query

Repr. info need & interpretable

Relevance of document

to info need

minimal: about topic

useful, interesting, satisfying

user depending

⇒ no clear cut-off

Core Challenges

H results e.g. Millions

Relevance?

Intend? suboptimal query?

Lexical gap! synonyme phrasen

Ambiguity! Homographie

Activities

Indexing - document repr. enhancing search

Searching - interpret query, relevance score, ranking,
improve of user feedback

Boolean Retrieval

document ~ set of words

Query word - document contains word $\sum_{i=1}^n r_i$

Index word \mapsto Vektor of documents $\{ \vec{v}_i \text{ in } d \text{ in } \vec{v}_i \text{ somit in } \vec{v}_i \}$
Term Document Matrix

△ large, many 0s

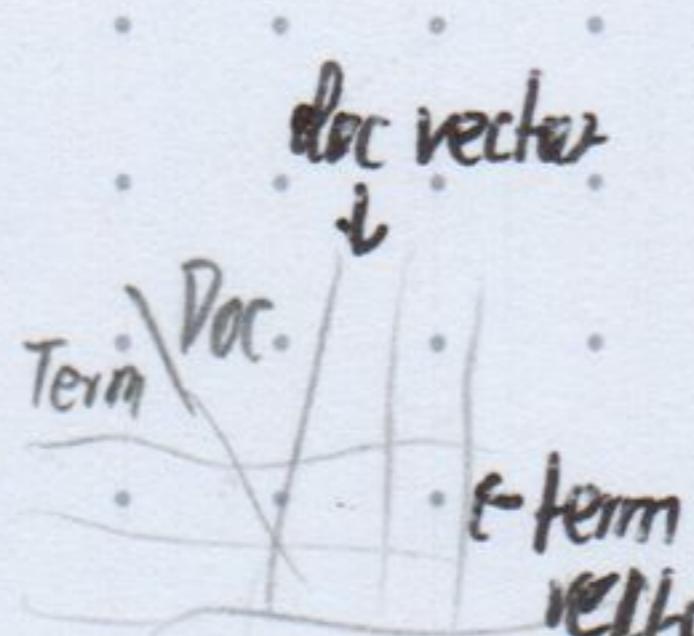
⇒ Inverted Index

word \mapsto List of documents

Dictionary Postings

- user needs syntax

- "feast or famine" - often too few or many results



Search

bitweise Ops res 1 d relevant

often too few or many results

Vector Space Model (VSM)

document ~ Bag of Words

Multimenge

TF - Term Frequency - #occurrences in document $f_{df}(t)$

(1) DF - (Inverse) Document Frequency - #docs containing word

query vector in doc vector space

euclidean distance? - depends on vec length

⇒ cosine similarity $\frac{q \cdot v}{\|q\| \cdot \|v\|}$

Vector weights

$$w_t = \begin{cases} 0 & \text{if } t \notin d \\ \frac{1}{\text{Binary}} & \text{if } t \in d \\ \frac{f_{df}(t)}{\text{Normalized TF}} & \text{if } t \in d \text{ and } t \neq \text{Binary} \end{cases}$$

e.g. 2^{-100} : mächtig linear hilfreicher?

$$\frac{\log(f_{df}(t)) + 1}{\text{Normalized TF}}$$

still considers useless determiners (a, the)

$$\text{TF-IDF} = \frac{\log(\frac{N}{df}) + 1}{\log(\frac{N}{df})}$$

from df to tf

Ranked Retrieval

solves feast or famine
user decides H results
/ all ≥ threshold

Invers Doc. fr.

IR Evaluation

$$P = \text{Precision} = \frac{\text{retrieved} \cap \text{relevant}}{\text{retrieved}} = \frac{\#TP}{\#TP + \#FP}$$

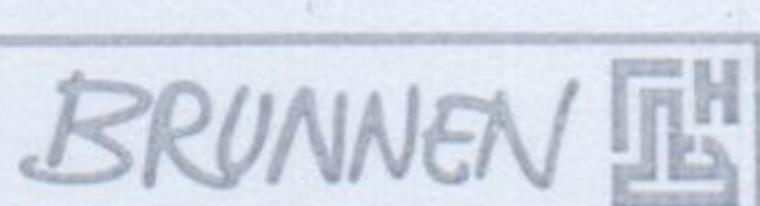
		Relevant	Not Relevant	
		True Positve	False Pos.	E
		False Neg.	True Neg.	
Precision		#TP	#FP	

achieving one is easy

⇒ F₁ score / measure

$$F_1 = \frac{2PR}{P+R} \quad F_B = \frac{(1+\beta^2)PR}{(\beta^2 P) + R}$$

harmonic mean



$$\text{Precision at Rank } P@R = \frac{\# \text{relevant docs before } R}{R}$$

IE - Information Extraction

Given: D document collection

⇒ structured knowledge

(set of structured facts)

entities, events

relationships, facts, ...

often dependent

Predefined e.g. SQL

+ for supported
query types
directly answer
for user

↔ IR

easier

Domain-independent

Query types usually unconstrained

faster

- less effective for user goal (the answer)

Entity Recognition

Challenges

Entity ↔ No.Entity
mobile phone ↔ Mobil

Coverage issue

complete list of possible Entitys?

Variation e.g. titles, abbreviations
Mr., Mr. Dr., Dr. First, last name
A, ...

Ambiguity

Darmstadt (German ↔ US city)

time dependency

Deutsche Bundeskanzler?

Multi-Word expressions, boundaries?

"... biersch-Stiftung an der TU Darmstadt"

Metonymy figure of speech, Entity by associated Thing

Deutschland für → Fußball Mannschaft

List Lookup Approach

+ simple, fast

+ domain adaptable

- collects lists

- No Variations

Ambiguity

Gazetteers - lists of locations

Rule-based

hand crafted set of regular expressions

+ fairly good performance
- Labour intensive

mid 1990s

⇒ Supervised Machine Learning from (hand-labeled) database

Corpus-based probabilistic models

probability language structures from large corpora

After IE \Rightarrow store in Knowledge Base - database for knowledge management
collect, organize, share, info
search, utilize

Knowledge Graphs

fact \sim binary Relationship

e.g. SPO-Triple subject $\xrightarrow{\text{predicate}}$ object

entities \sim nodes

Science Fiction

Genre

Star Trek

Automatic Classification

data points = instances annotated with class labels
features \sim vectorspace coordinates

binary \leftrightarrow multi-class

decomposable in binary

single \leftrightarrow multi-label per instance

\Rightarrow learn, generalise
instances \rightarrow class labels

sequence classification

instance sequence jointly classified

Split the data

supervised - trained on labeled instances

Dev-Set

Training-Set

Dev-Test Set - analyse errors
optimize param

Test-Set - no optimization with it
of model

Training

Neural network

label

input

Prediction

input

label

label