



LUDWIG-
MAXIMILIANS-
UNIVERSITY
MUNICH


DEPARTMENT
INSTITUTE FOR
INFORMATICS


DATABASE
SYSTEMS
GROUP

Kapitel 2: Ein abstraktes Geo-Datenmodell

Skript zur Vorlesung
Geo-Informationssysteme

Wintersemester 2014/15

Ludwig-Maximilians-Universität München

(c) Matthias Renz 2014, Peer Kröger 2011, basierend auf dem Skript von Christian Böhm
aus dem SoSe 2009



2. Ein abstraktes Geo-Datenmodell

1. Was soll modelliert werden?
2. Spatial Data Types
3. Integration in das relationale Datenmodell

- **Einzelne Objekte**

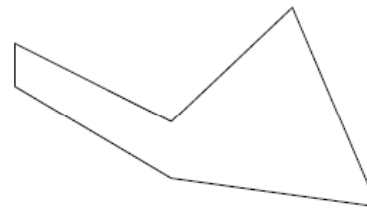


Punkt

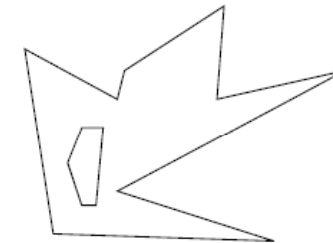
Punkt



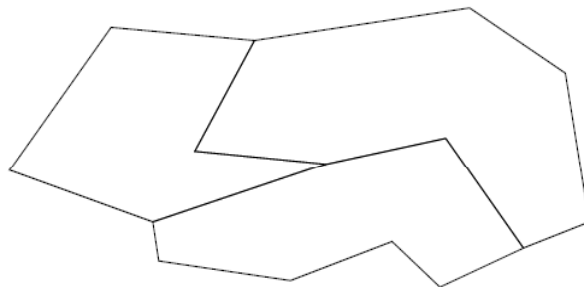
Linie



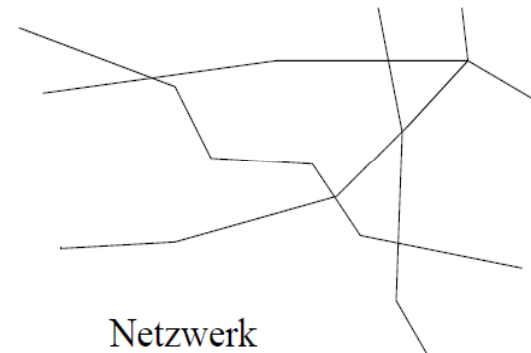
Fläche



- **Mengen räumlich benachbarter Objekte**



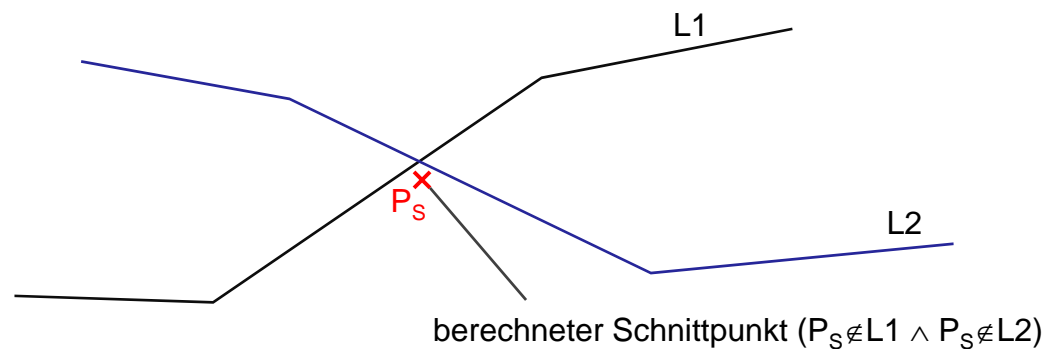
Partition



Netzwerk

Probleme mit dem Euklidischen Raum

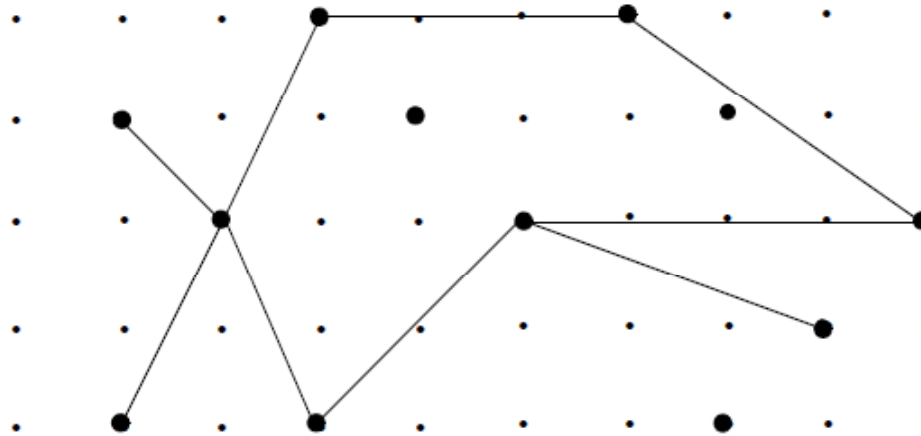
- Punkt = (r_1, r_2)
 - r_1 und r_2 sind theoretisch reelle Zahlen
 - im Rechner aber nur als Fließkommazahlen mit bestimmter Genauigkeit repräsentiert
- Problem mit Schnittpunkten zweier Linien:
 - Runden der Koordinaten des Schnittpunkts zur nächsten Fließkomma-Zahl
 - der Schnittpunkt liegt dann auf keiner der beiden Linien (Inkonsistenz)



2.1 Was soll modelliert werden?

Lösung: Realm-basierter Ansatz [Schneider, Güting 1994]

- Eine *Realm* ist eine Menge von Punkten und nicht-schneidenden Liniensegmenten über einem gegebenen Gitter
- Schnittpunkte zweier Linien müssen auf einem Gitterpunkt liegen



Gegeben sei ein endlicher diskreter Raum $N \times N$ mit $N = \{0, 1, \dots, n-1\}$.

Robuste Geometrische Primitive

- Ein ***N-Punkt*** ist ein Paar $(x, y) \in N \times N$. PN sei die Menge aller *N-Punkte*.
- Ein ***N-Segment*** ist ein Paar verschiedener *N-Punkte* (p, q) , SN die Menge aller *Segmente*.

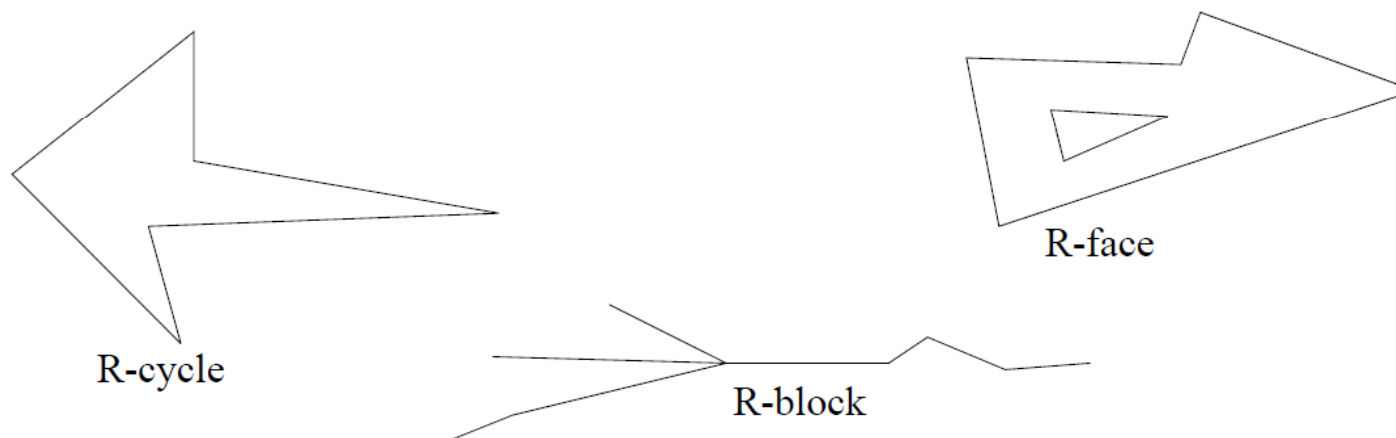
Realms

- Eine ***N-Realm*** ist eine Menge $R = P \cup S$ mit
 - (i) $P \subseteq PN$ (***R-Punkte***), $S \subseteq SN$ (***R-Segmente***)
 - (ii) $\forall s \in S : s = (p, q) \Rightarrow p \in P \wedge q \in P$
 - (iii) $\forall p \in P \forall s \in S : \neg (p \text{ in } s)$
 - (iv) $\forall s, t \in S, s \neq t : \neg (s \text{ intersects } t)$
- Interpretation einer Realm als planarer Graph:
 - Menge der Knoten = P
 - Menge der Kanten = S

Realm Strukturen

Die folgenden Definitionen beruhen auf der Graph-Interpretation einer Realm R:

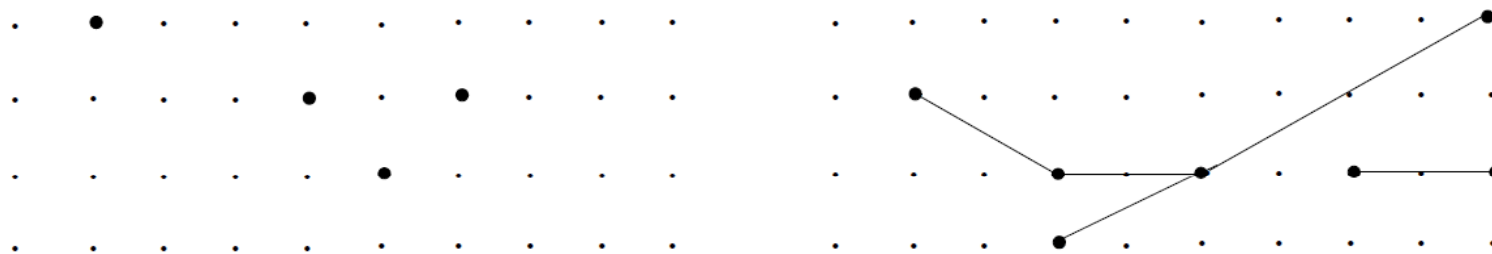
- Ein *R-cycle* ist ein Zyklus im Graphen von R.
- Ein *R-face* ist ein R-cycle der andere disjunkte R-cycles enthalten kann.
- Ein *R-block* ist eine Zusammenhangskomponente im Graphen von R.



Realm-Basierte Spatial Data Types

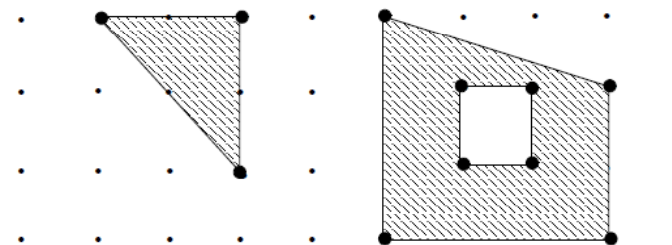
Gegeben sei eine Realm R .

- Der Typ *points* definiert Mengen von R -Punkten.
- Der Typ *lines* definiert Mengen von paarweise disjunkten R -blocks.
- Der Typ *regions* definiert Mengen von paarweise kanten-disjunkten R -faces.



ein Element von points

ein Element von lines



ein Element von regions

Typmengen

- EXT = {*lines, regions* }
- GEO = {*points, lines, regions* }
- OBJ = {*cities, highways, . . .*} (anwendungsspezifisch)
- set(OBJ) modelliert eine Datenbank

⇒ Second Order Signature (Typmengen = kinds)

Arten von Operationen

- Resultat vom Typ bool (Prädikate)
- Resultat vom Typ GEO
- Resultat vom Typ int / real
- Resultat vom Typ set(GEO) (Anfragen)

Prädikate (Resultat vom Typ bool)

$\forall geo \in GEO \forall ext, ext1, ext2 \in EXT$

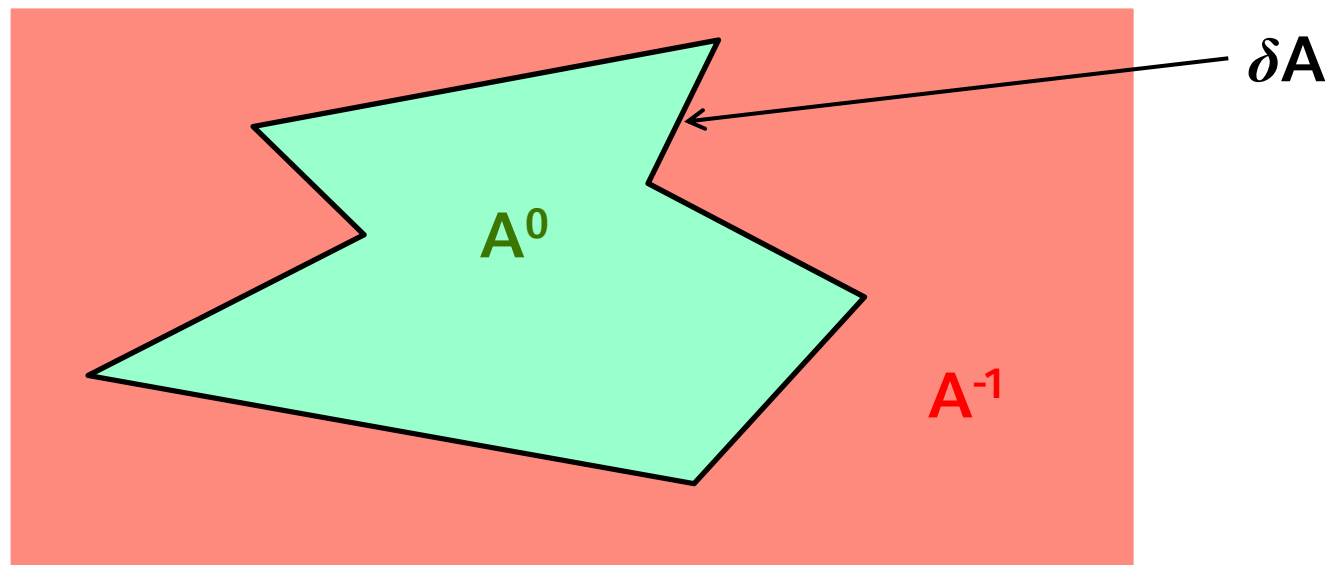
$geo \times geo$	$\rightarrow bool$	$=, \neq, \text{ "nördlich von", "dist} < 100\text{", ...}$
$geo \times regions$	$\rightarrow bool$	inside
$regions \times regions$	$\rightarrow bool$	area_disjoint, edge_disjoint
$points \times ext$	$\rightarrow bool$	on_border_of
$ext1 \times ext2$	$\rightarrow bool$	disjoint, meet, overlaps, covers, contains, covered_by, inside, equal (Topologische Prädikate)

...

\rightarrow vollständige Menge der topologischen Prädikate

9-Schnitt-Modell [Egenhofer 1991]

- Wir betrachten *einfache Polygone ohne Löcher (R-cycles)*. Ein solches Polygon kann als Punktmenge im 2D (N-Punkte) angesehen werden.
- Für eine Punktmenge A bezeichnen
 - A^0 das *Innere* von A ,
 - δA den *Rand* von A ,
 - A^{-1} das *Äußere* (Komplement) von A .



9-Schnitt-Modell (Forts.)

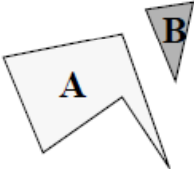
- Um die topologische Beziehung zwischen zwei Polygonen (Punktmengen) A und B zu bestimmen, bilden wir die folgenden 9 Schnitte und notieren die Ergebnisse (\emptyset oder $\neg\emptyset$) in einer Matrix, genannt *Durchschnittsmatrix*:

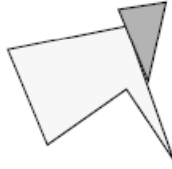
$\delta A \cap \delta B$	$\delta A \cap B^0$	$\delta A \cap B^{-1}$
$A^0 \cap \delta B$	$A^0 \cap B^0$	$A^0 \cap B^{-1}$
$A^{-1} \cap \delta B$	$A^{-1} \cap B^0$	$A^{-1} \cap B^{-1}$

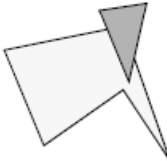
- Von den 512 potentiellen Matrizen können nur 8 auftreten, wenn A und B keine Löcher haben, zusammenhängend sind und nicht 1-dimensional sind.
- Jede dieser 8 Matrizen definiert eine topologische Beziehung zwischen den Polygonen A und B.

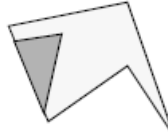
9-Schnitt-Modell (Forts.)

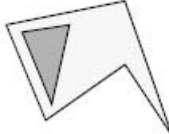
Durchschnittsmatrizen für die 8 topologischen Beziehungen

$$M_{\text{disjoint}}(A, B) = \begin{pmatrix} \emptyset & \emptyset & \neg\emptyset \\ \emptyset & \emptyset & \neg\emptyset \\ \neg\emptyset & \neg\emptyset & \neg\emptyset \end{pmatrix}$$


$$M_{\text{meet}}(A, B) = \begin{pmatrix} \neg\emptyset & \emptyset & \neg\emptyset \\ \emptyset & \emptyset & \neg\emptyset \\ \neg\emptyset & \neg\emptyset & \neg\emptyset \end{pmatrix}$$


$$M_{\text{overlaps}}(A, B) = \begin{pmatrix} \neg\emptyset & \neg\emptyset & \neg\emptyset \\ \neg\emptyset & \neg\emptyset & \neg\emptyset \\ \neg\emptyset & \neg\emptyset & \neg\emptyset \end{pmatrix}$$


$$M_{\text{covers}}(A, B) = \begin{pmatrix} \neg\emptyset & \emptyset & \neg\emptyset \\ \neg\emptyset & \neg\emptyset & \neg\emptyset \\ \emptyset & \emptyset & \neg\emptyset \end{pmatrix}$$


$$M_{\text{contains}}(A, B) = \begin{pmatrix} \emptyset & \emptyset & \neg\emptyset \\ \neg\emptyset & \neg\emptyset & \neg\emptyset \\ \emptyset & \emptyset & \neg\emptyset \end{pmatrix}$$


$$M_{\text{coveredby}}(A, B) = \begin{pmatrix} \neg\emptyset & \neg\emptyset & \emptyset \\ \emptyset & \neg\emptyset & \emptyset \\ \neg\emptyset & \neg\emptyset & \neg\emptyset \end{pmatrix}$$

$$M_{\text{inside}}(A, B) = \begin{pmatrix} \emptyset & \neg\emptyset & \emptyset \\ \emptyset & \neg\emptyset & \emptyset \\ \neg\emptyset & \neg\emptyset & \neg\emptyset \end{pmatrix}$$

$$M_{\text{equal}}(A, B) = \begin{pmatrix} \neg\emptyset & \emptyset & \emptyset \\ \emptyset & \neg\emptyset & \emptyset \\ \emptyset & \emptyset & \neg\emptyset \end{pmatrix}$$

Operationen mit Resultat vom Typ GEO

$\forall geo \in GEO \ \forall ext, ext1, ext2 \in EXT$

$geo \times geo$	$\rightarrow geo$	<i>plus, minus</i>
ext	$\rightarrow points$	vertices
<i>regions</i>	$\rightarrow lines$	contour
$points \times points$	$\rightarrow ?$	intersection
$lines \times lines$	$\rightarrow ?$	intersection
$regions \times regions$	$\rightarrow ?$	intersection
$regions \times lines$	$\rightarrow ?$	intersection

...

Semantik basierend auf Realm-Operationen

Seien $R \in regions$ und $L \in lines$:

$intersection(R,L) := blocks(\{s \in Segments(L) \mid \exists r \in R: s \text{ inside } r\})$

siehe Übung!

Operationen mit Resultat vom Typ *int* oder *real* (skalar)

$\forall geo, geo1, geo2 \in GEO$

geo $\rightarrow int$ *no_of_components*

geo1 \times *geo2* $\rightarrow real$ *dist*

geo $\rightarrow real$ *diameter*

lines $\rightarrow real$ *length*

regions $\rightarrow real$ *area*

...

Semantik basierend auf Realm-Operationen

Seien $G \in GEO$ und $L \in \mathbf{lines}$:

$diameter(G) := \max\{dist(p,q) \mid p, q \in vertices(G)\}$

$length(L) := \sum dist(p,q)$

$(p,q) \in Segments(L)$

Anfragen

$\forall obj \in OBJ, \forall geo1, geo2 \in GEO$

$set(obj) \times (obj \rightarrow geo1) \times geo2 \rightarrow set(obj)$ nearest_neighbor_query

Beispiel:

$set(\mathbf{cities}) \times (\mathbf{cities} \rightarrow \mathbf{regions}) \times \mathbf{points} \rightarrow set(\mathbf{cities})$

$set(obj) \times (obj \rightarrow geo1) \times \mathbf{regions} \rightarrow set(obj)$ region_query

Beispiel:

$set(\mathbf{cities}) \times (\mathbf{cities} \rightarrow \mathbf{regions}) \times \mathbf{regions} \rightarrow set(\mathbf{cities})$

Idee

- Geo-Objekte einer Anwendung werden als Tupel bzw. Objekte mit mindestens einem Attribut eines Spatial Data Types modelliert.
- Das Datenmodell soll neben den atomaren Datentypen wie *int* und *string* auch *Spatial Data Types* anbieten.

Beispiel: Relationales Schema

- **relation** states (sname: string; area: *regions*; population: int)
- **relation** cities (cname: string; center: *points*; ext: *regions*; population: *int*)
- **relation** rivers (rname: string; route: *lines*)

Beispiel: Anfragen in Geo-Relationaler Algebra

(1) cities select [center inside Bavaria]

“Bavaria” sei eine Konstante des Typs *regions*

(2) rivers select [route intersects Window]

(3) cities select [dist(center,Hagen) < 100 and population > 500.000]

(4) cities states join [center inside area]

(5) cities rivers join [dist(center,route) < 50]

(6) rivers select [route intersects Bavaria]

extend [intersection(route,Bavaria) {part}]

extend [length(part) {plength}]

project [rname,part,plength]

Integration der algebraischen Operationen in konkrete Anfragesprache

z.B. SQL:

(4') SELECT cname, sname FROM cities, states WHERE center inside
area

Input und Output von Konstanten

- Input von Konstanten von Spatial Data Types:
 - *points, lines, regions* benennen (z.B. über Mapping *obj* → *geo*)
 - oder graphischer Input der Konstanten
- Output von Konstanten von Spatial Data Types:
 - Graphischer Output
 - mit Legende (Namen, Typ, . . . von Objekten)