What's Wrong with High-Dimensional Similarity Search?

Stephen Blott (blott@computing.dcu.ie)
School of Computing, Dublin City University

Roger Weber (roger.weber@credit-suisse.ch)
Credit Suisse, Zurich, Switzerland

Originally:

A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces
Roger Weber, Hans-Jörg Schek and Stephen Blott
VLDB 1998, New York

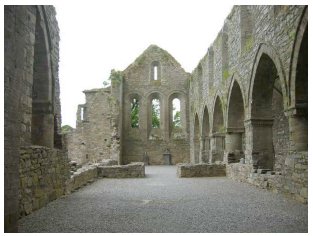
Contents

The Similarity Search Paradigm – 1

?

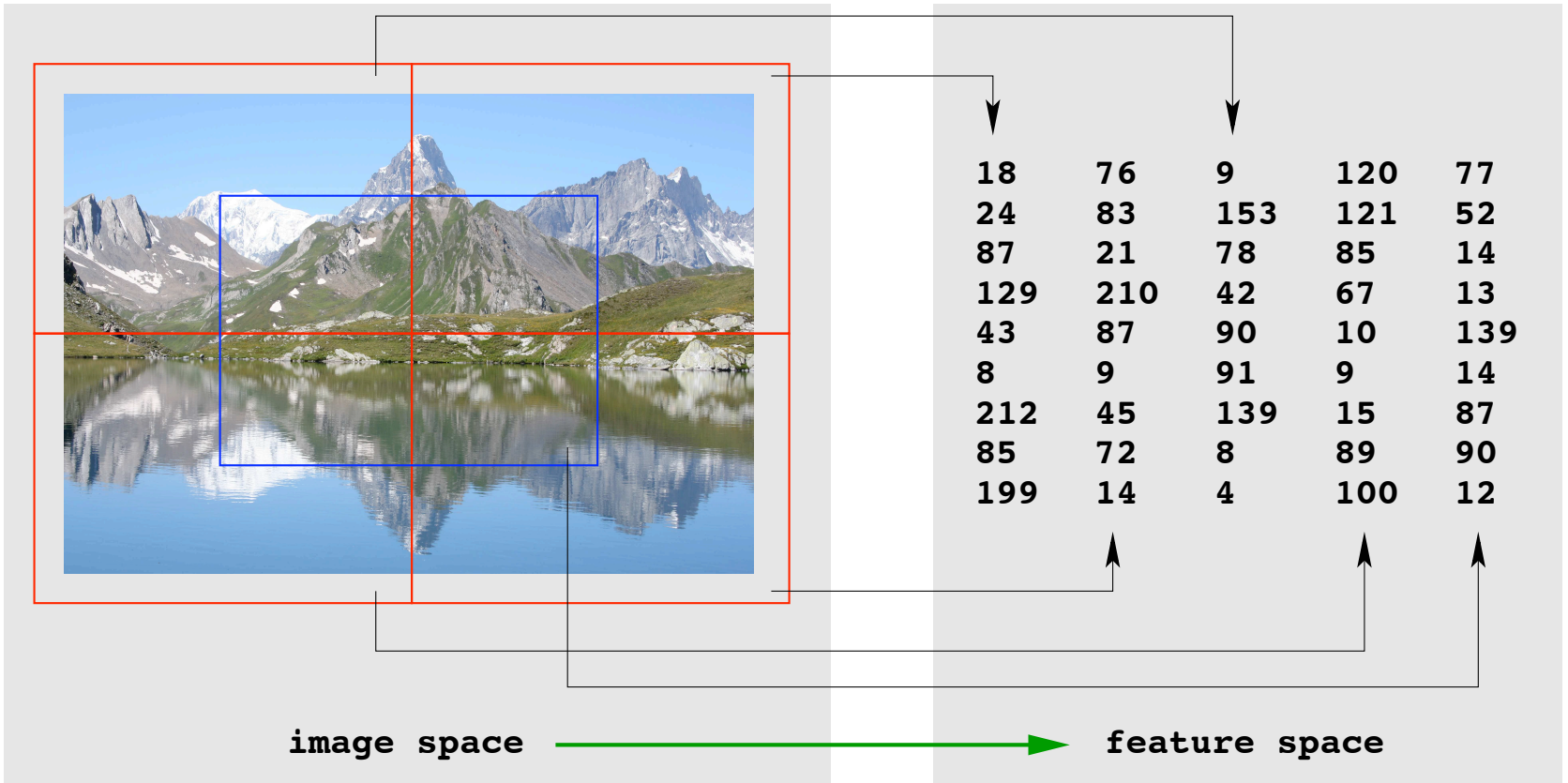
locate similar images in
large image collection

. . .

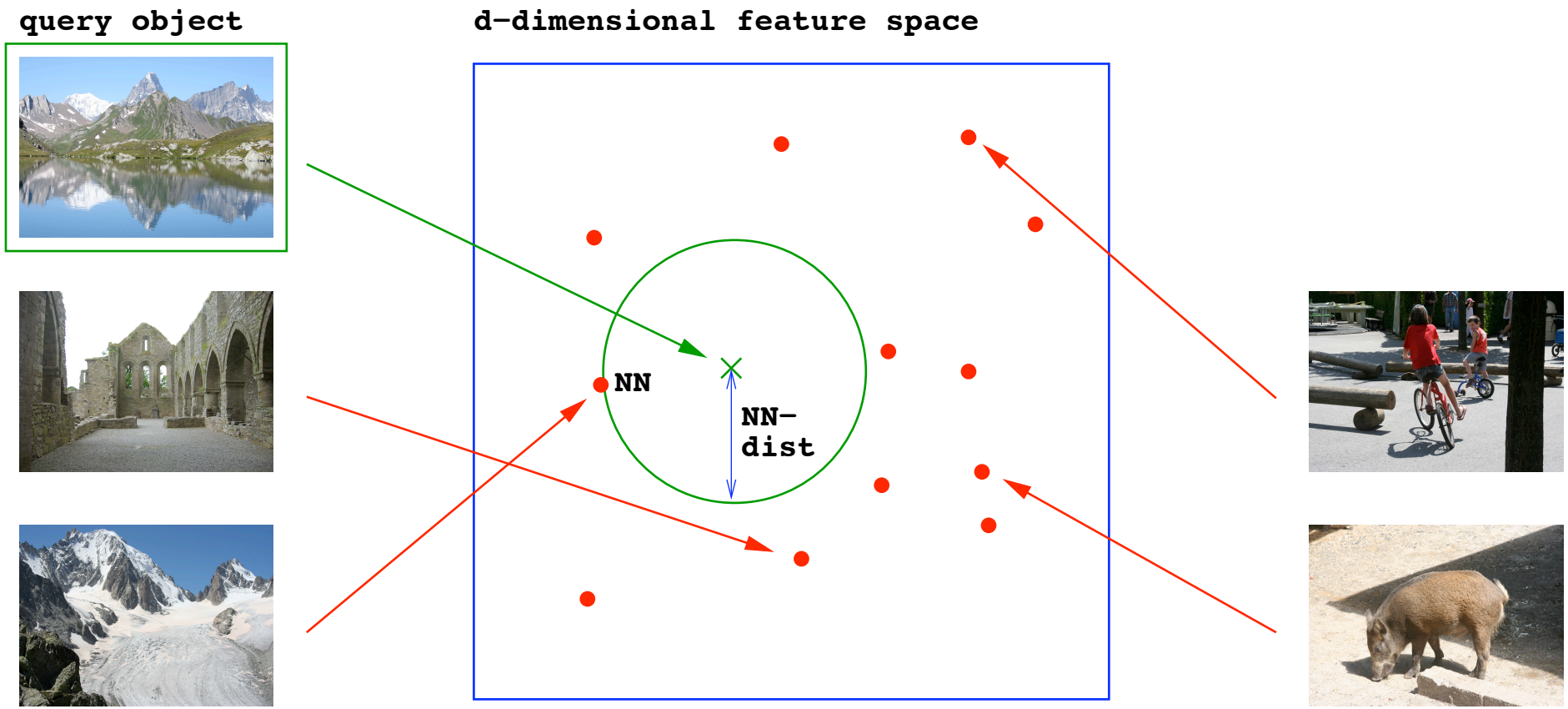



. . .

The Similarity Search Paradigm – 2



The Similarity Search Paradigm – 3



Locate closest point to query object, i.e. its *nearest neighbour* (NN)

The Similarity Search Paradigm – 4

This search paradigm is not restricted to images

Other examples include:

- music databases, video databases
- medical information systems, genomic databases
- 3D object recognition
- . . .

kNN Search – Problem Statement

Problem statement:

data set: point data in d-dimensional space

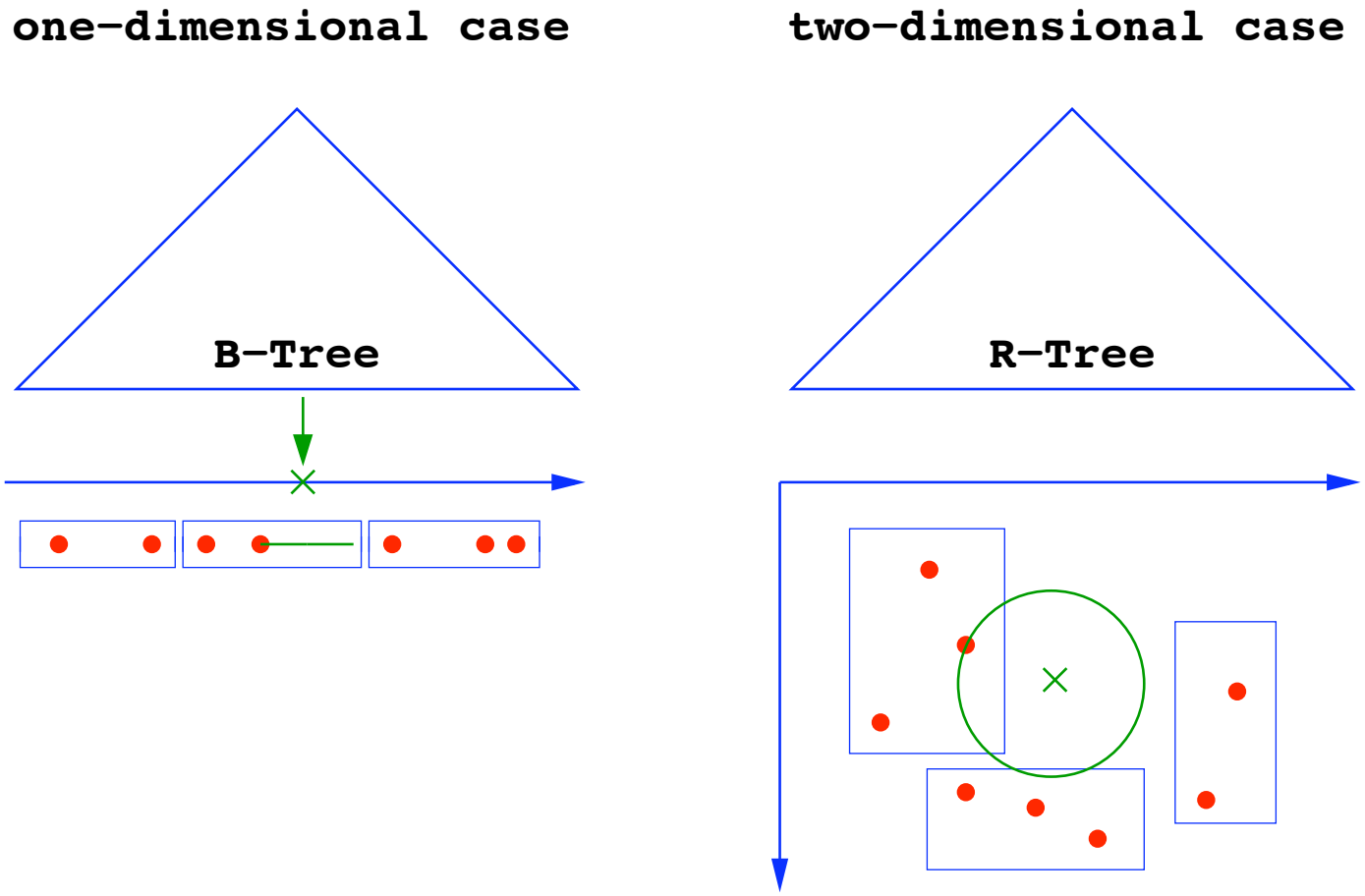
query: a search point in d-dimensional space

task: locate nearest data-set point(s) to the query

metrics: initially disk accesses, but also computational costs

assumptions: initially uniformly-distributed data within unit hypercube with independent dimensions

kNN Search in 1- and 2-Dimensional Spaces



kNN Search in Higher-Dimensional Spaces

Hierarchical methods can similarly be designed for high-dimensional spaces . . .

So many methods . . . it has to be difficult!

Quad trees	[Finkel:1974]	K-d-b-tree	[Robinson:1981]
R-tree	[Guttman:1984]	Gridfile	[Nievergelt:1984]
R^+-tree	[Sellis 1987]	LSD-tree	[Henrich:1989]
R^*-tree	[Beckmann:1990]	hB-tree	[Lomet:1990]
Vp-tree	[Chiueh:1994]	TV-tree	[Lin:1994]
UB-tree	[Evangelidis:1995]	hB-Pi-tree	[Bayer:1996]
SS-tree	[White:1996]	X-tree	[Berchtold:1996]
M-tree	[Ciaccia:1996]	SR-tree	[Katayama:1997]
Pyramid	[Berchtold:1998]	Hybrid-tree	[Chakrabarti:1999]
DABS-tree	[Böhm:1999]	IQ-tree	[Böhm:2000]
Slim-tree	[Faloutsos:2000]	landmark file	[Böhm:2000]
P-Sphere-tree	[Goldstein:2000]	A-Tree	[Sakurai:2000]

Unfortunately,
as dimensionality increases, these methods become ineffective:

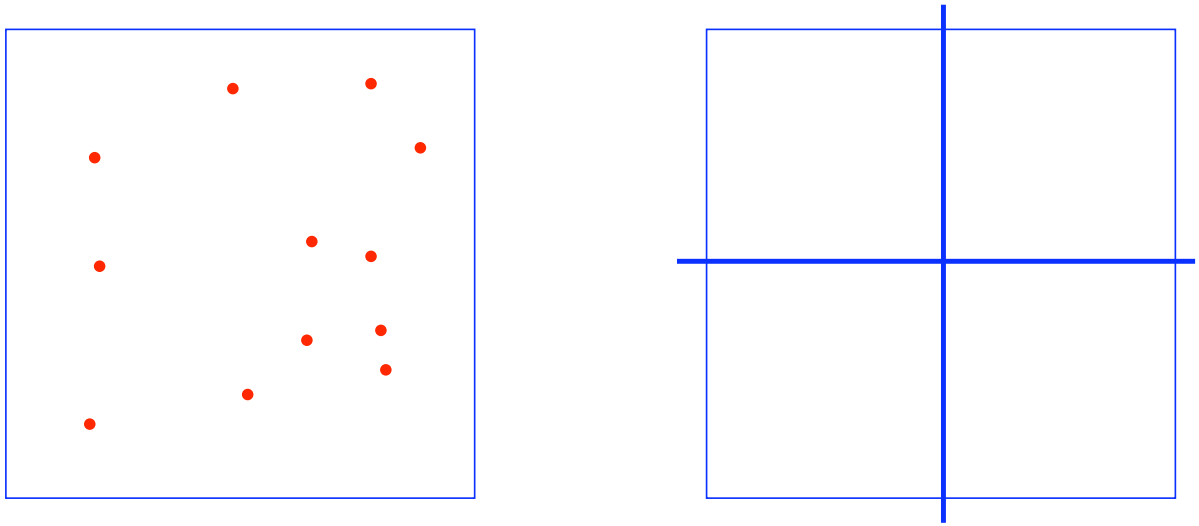
- the so-called *curse of dimensionality* . . . why?

High-dimensional spaces are really odd . . .

Oddity 1

A simple clustering scheme:

cluster into regions created by partitioning *all* dimensions



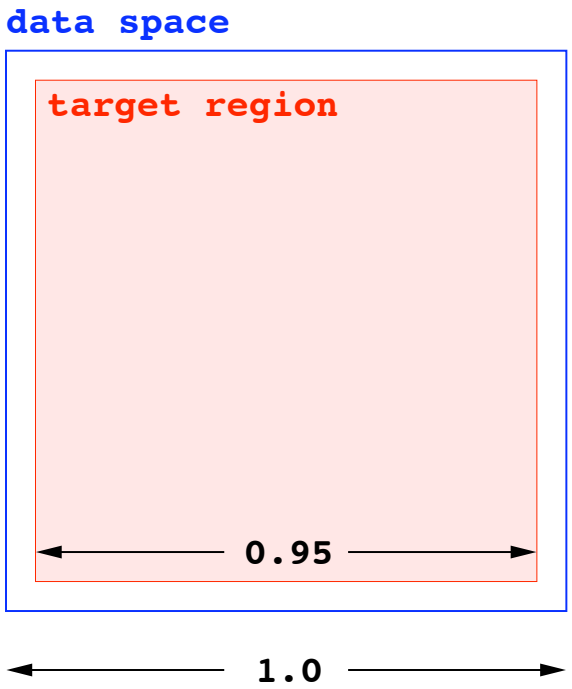
This seems reasonable with two or three dimensions

But with $d = 100$ there are $2^{100} \approx 10^{30}$ regions:

even with billions of points, *almost all* of the regions are empty

Oddity 2

Consider a *really big* square search region of size s, say $s = 0.95$:

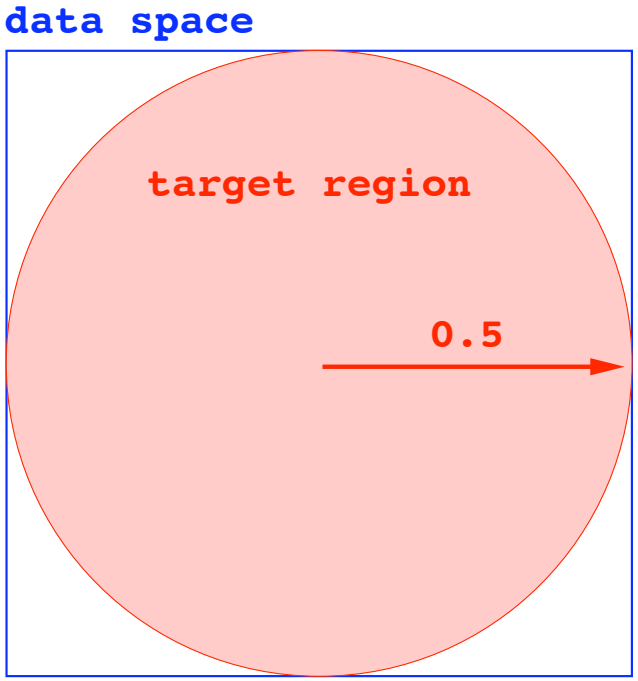


But with $d = 100$:

probability of a point being in this region is $0.95^{100} \approx 0.0059$

Oddity 3

Same game, but with largest possible sphere as the target:



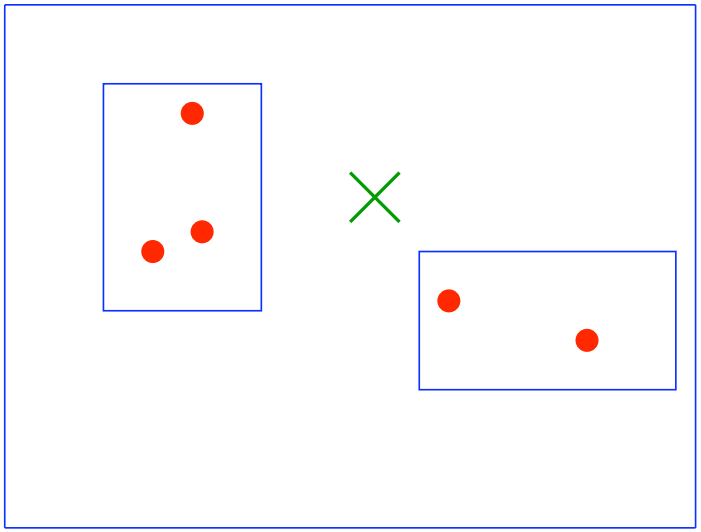
For $d = 40$:

volume of sphere is 3.278×10^{-21}
all the space is in the corners
require 3×10^{20} points to expect, on average, one to be in this sphere

So, high-dimensional spaces are odd . . .

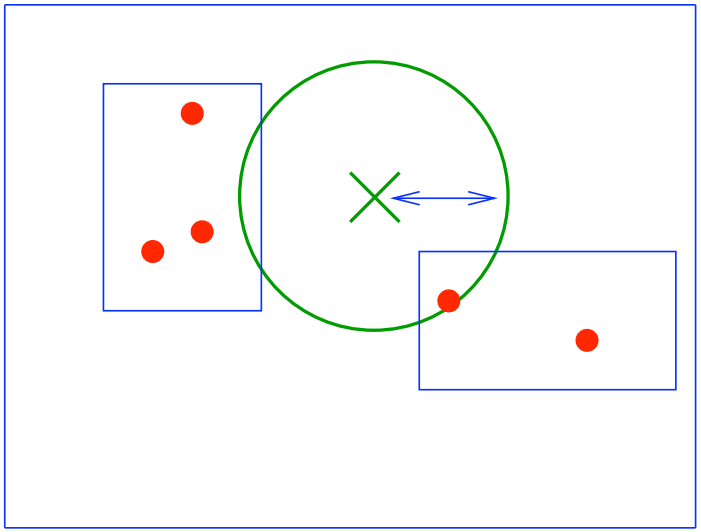
. . . but how does that affect the performance of search structures?

Analysis – Access Probabilities



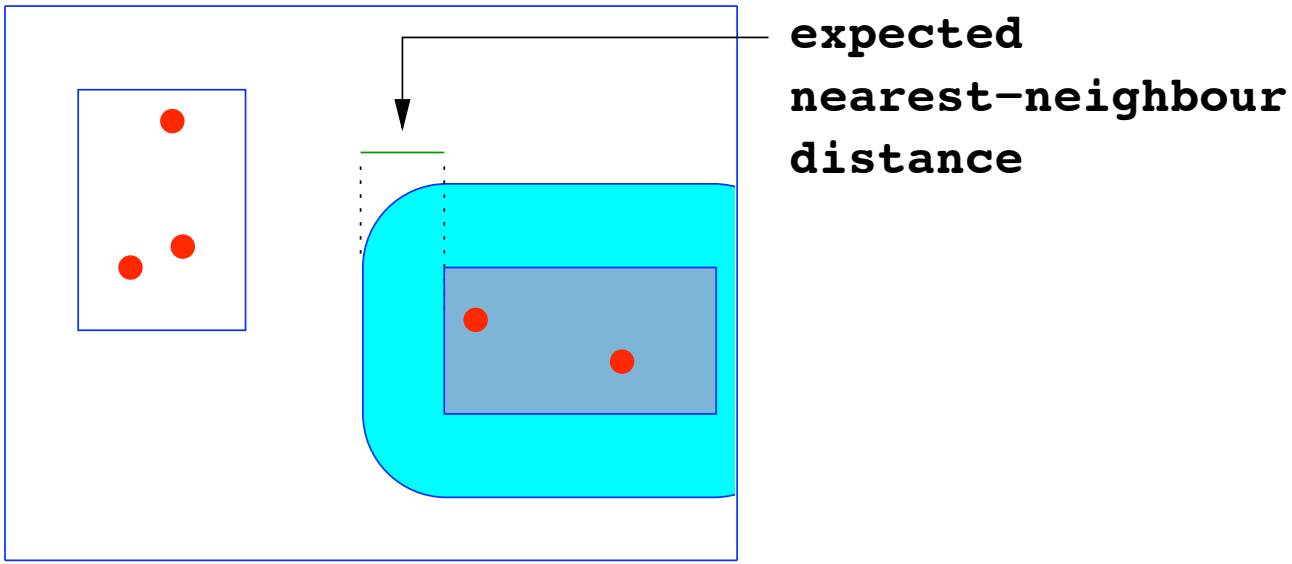
Given some arbitrary query,
what is the probability that a particular region must be accessed?

Analysis – Access Probabilities



Must visit (at least):
every region within the nearest-neighbour distance of the query point

Analysis – Minkowski Sums (MinkSum)



Minkowski sum:

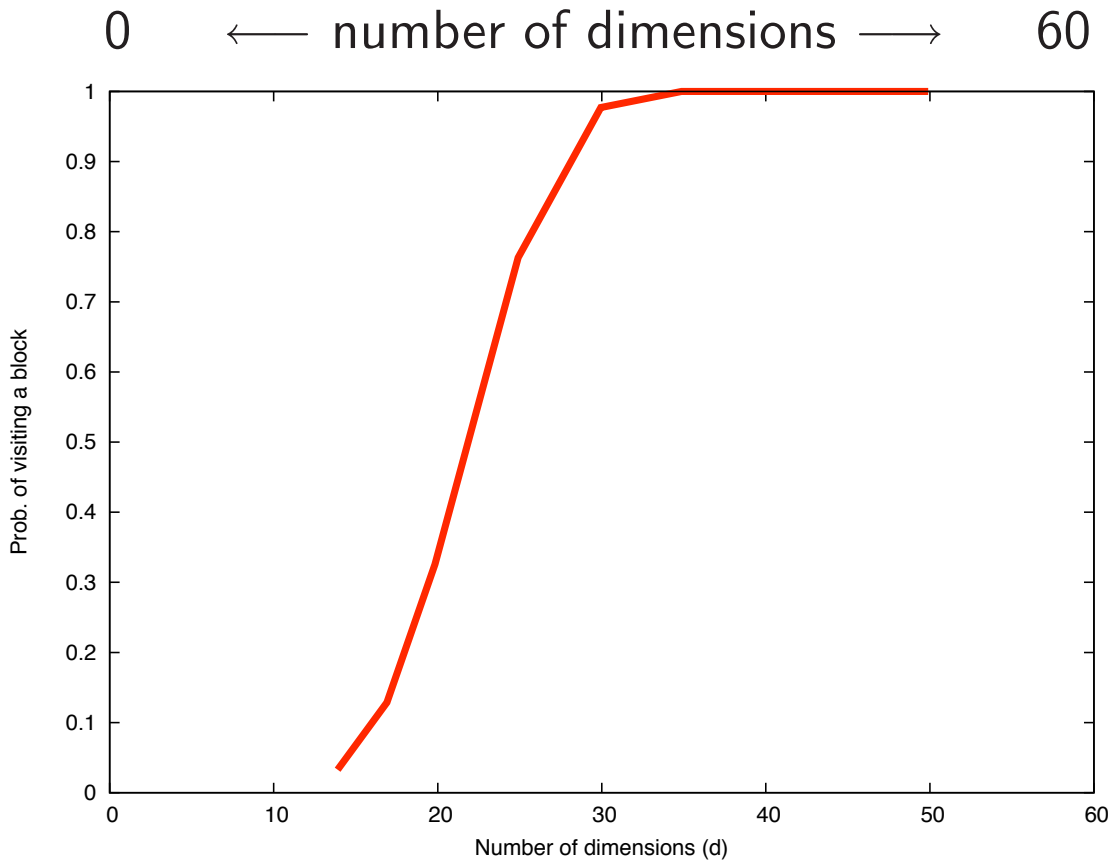
- enlarge the region such that it contains all space within the nearest-neighbour distance of the region
- *the resulting volume is the probability that this region must be accessed for an arbitrary query*

Analysis – VLDB 1998 Approach

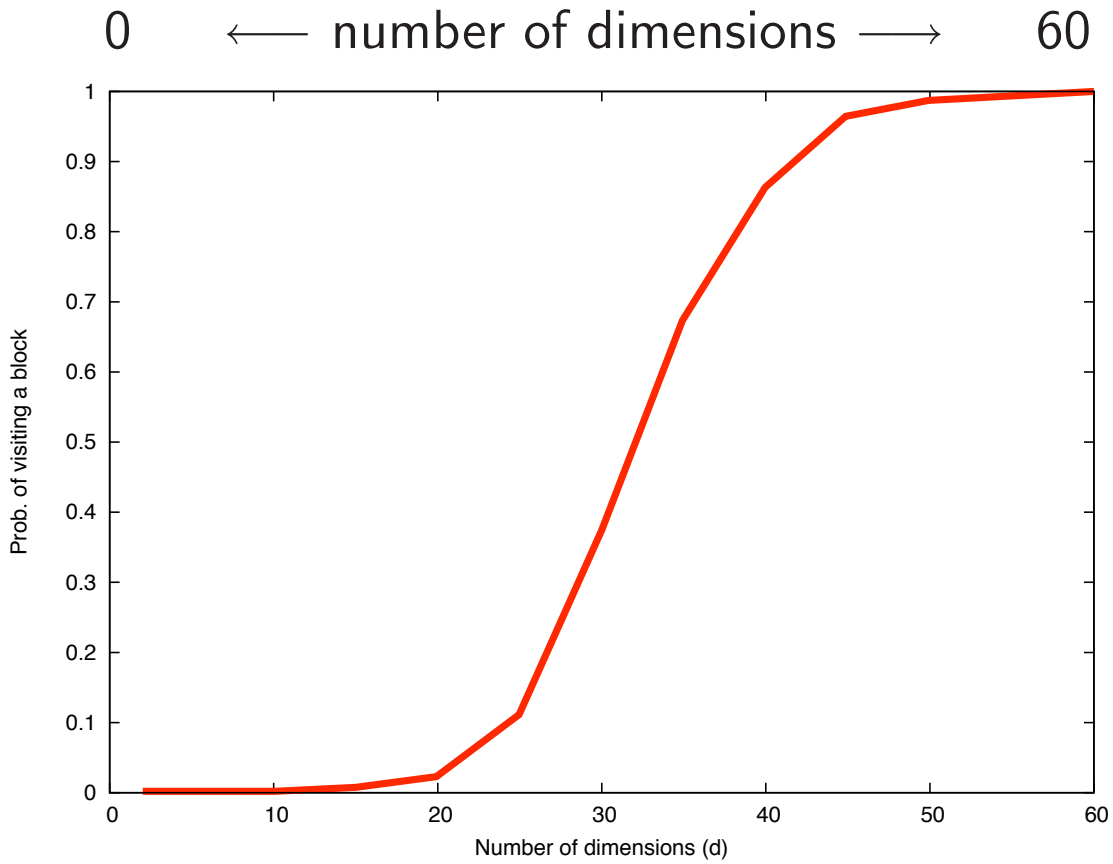
Approach:

- calculate expected nearest-neighbour distance
  (Monte-Carlo method)
- consider various possible clustering/partitioning schemes
  (initially hyper-rectangles, hyper-spheres)
- for each scheme and as dimensionality increases:
  calculate the probability that a region is accessed for an arbitrary query

Analysis – Probability of Visiting a Region – Hyper-Rectangles



Analysis – Probability of Visiting a Region – Hyper-Spheres

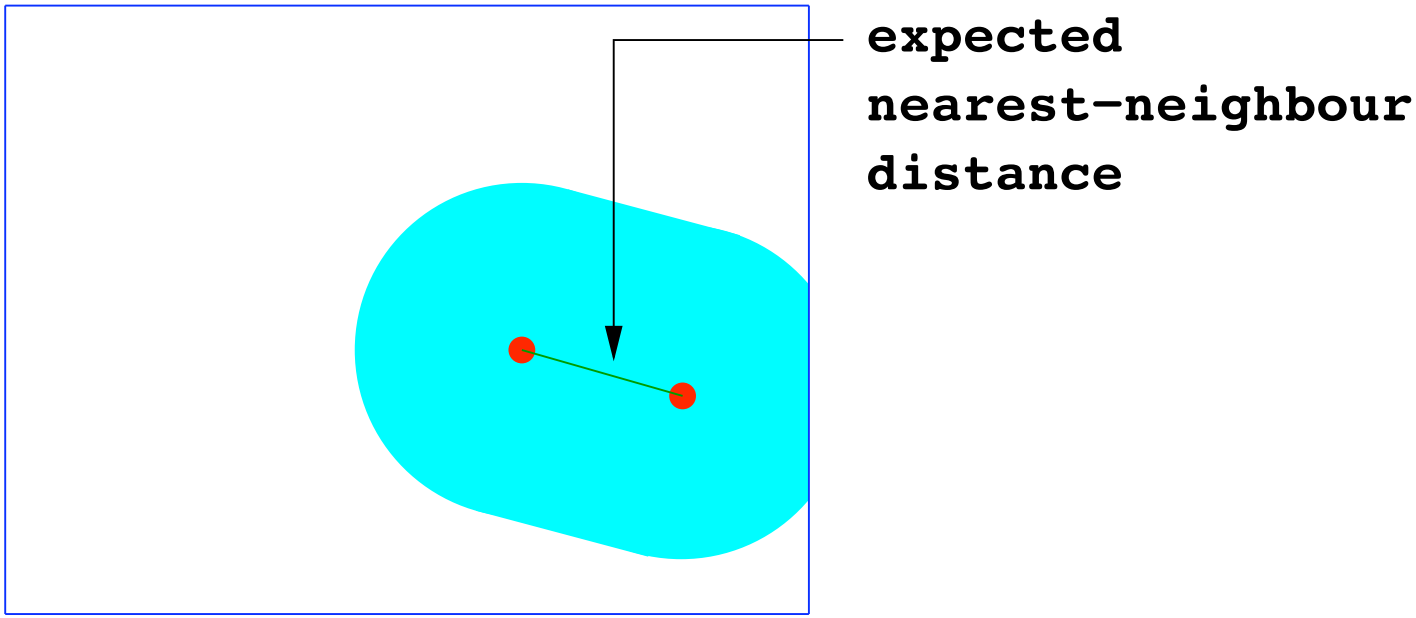


VLDB 1998 – Results

From a theoretical perspective:

- above some threshold, the access probability of clusters approaches 1
  - *all regions are accessed*
  - *NN search becomes linear*
- for spherical and rectangular regions,
  selectivity degenerate from 20 to 40 dimensions

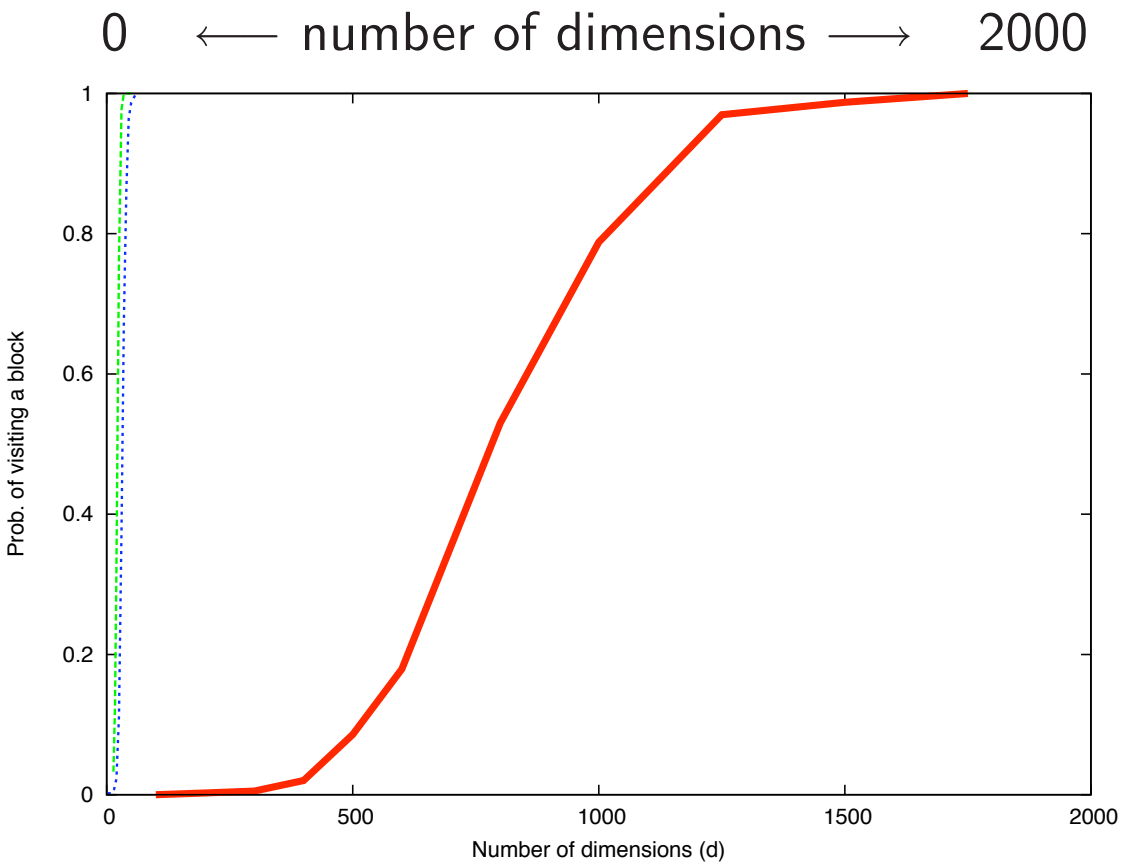
Analysis – Extreme Case



In the extreme case,
a region might be a zero-volume line between just two points

The Minkowski Sum then contains all space within the expected nearest-neighbour distance of the that line

Analysis – Probability of Visiting a Region – Lines



VLDB 1998 – Results

So, from a theoretical perspective:

- it appears that *any* clustering scheme degenerates at or before around 1000 dimensions

This would difficult to achieve in practice

VLDB 1998 – From a Practical Perspective

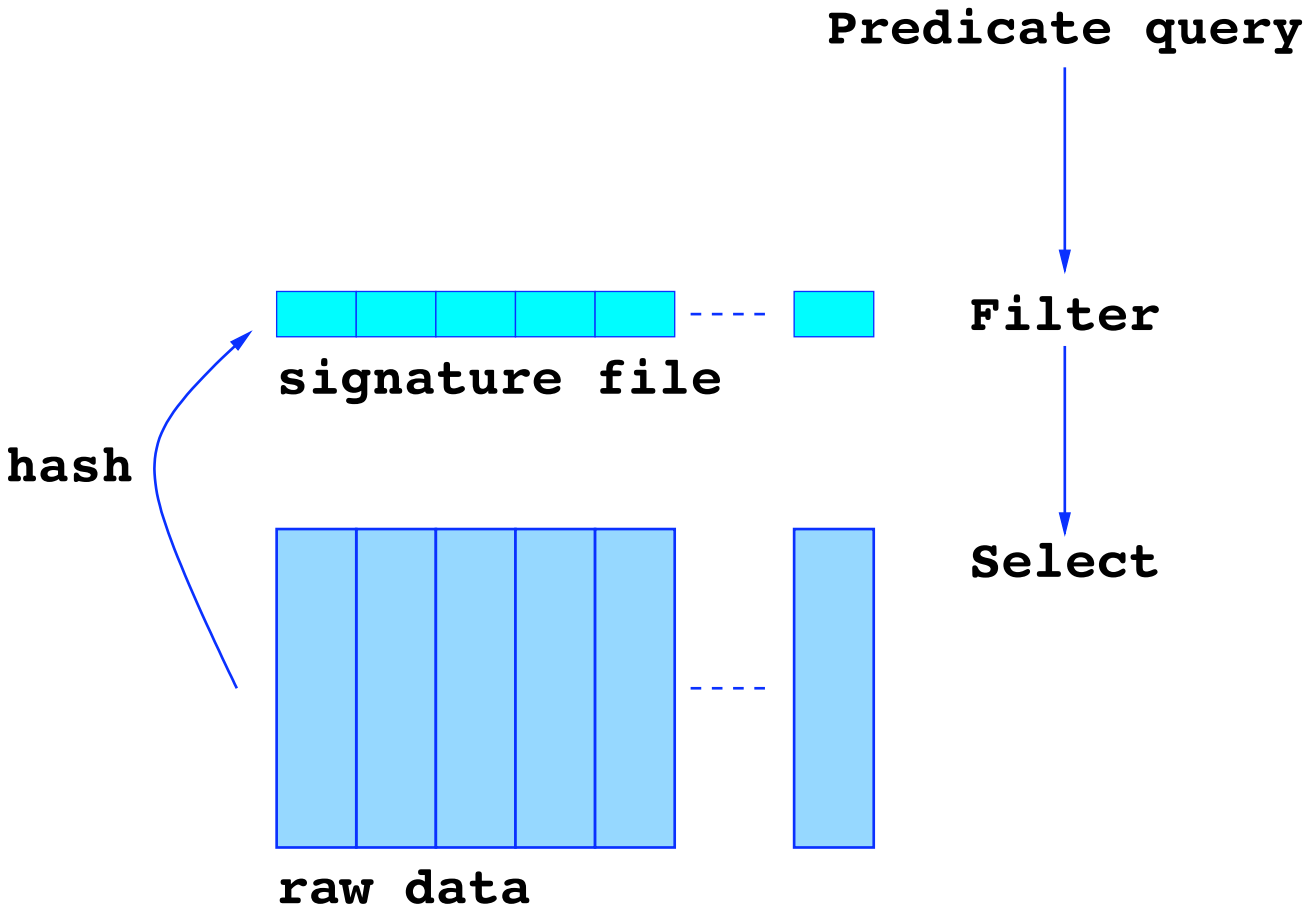
From a practical perspective:

- because of the properties of sequential scan, hierarchical methods are effective only when accessing less than around 5-10% of regions

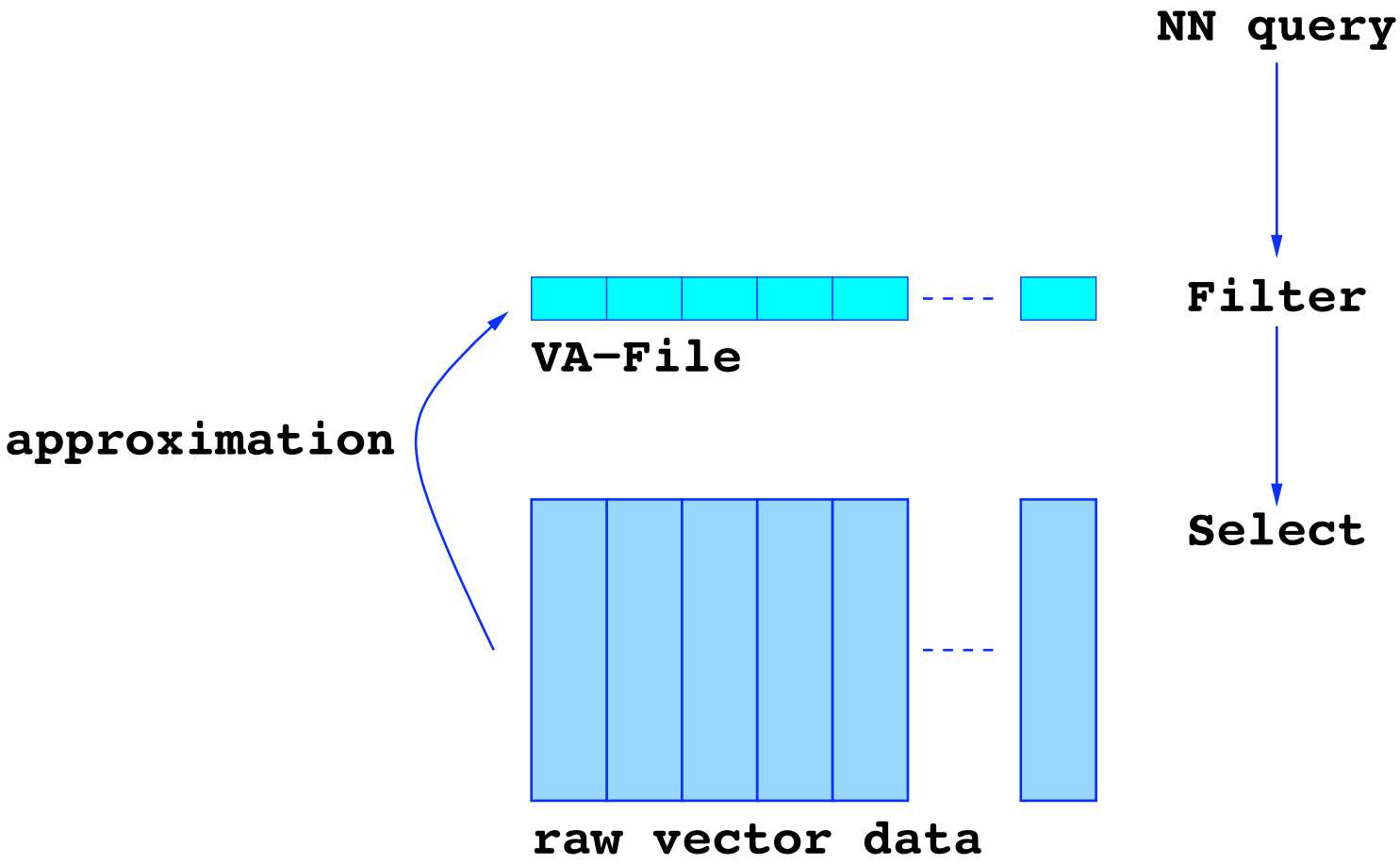
- therefore:
  kNN search becomes *linear* at surprisingly low dimensionality

So, why not look more carefully at sequential methods?

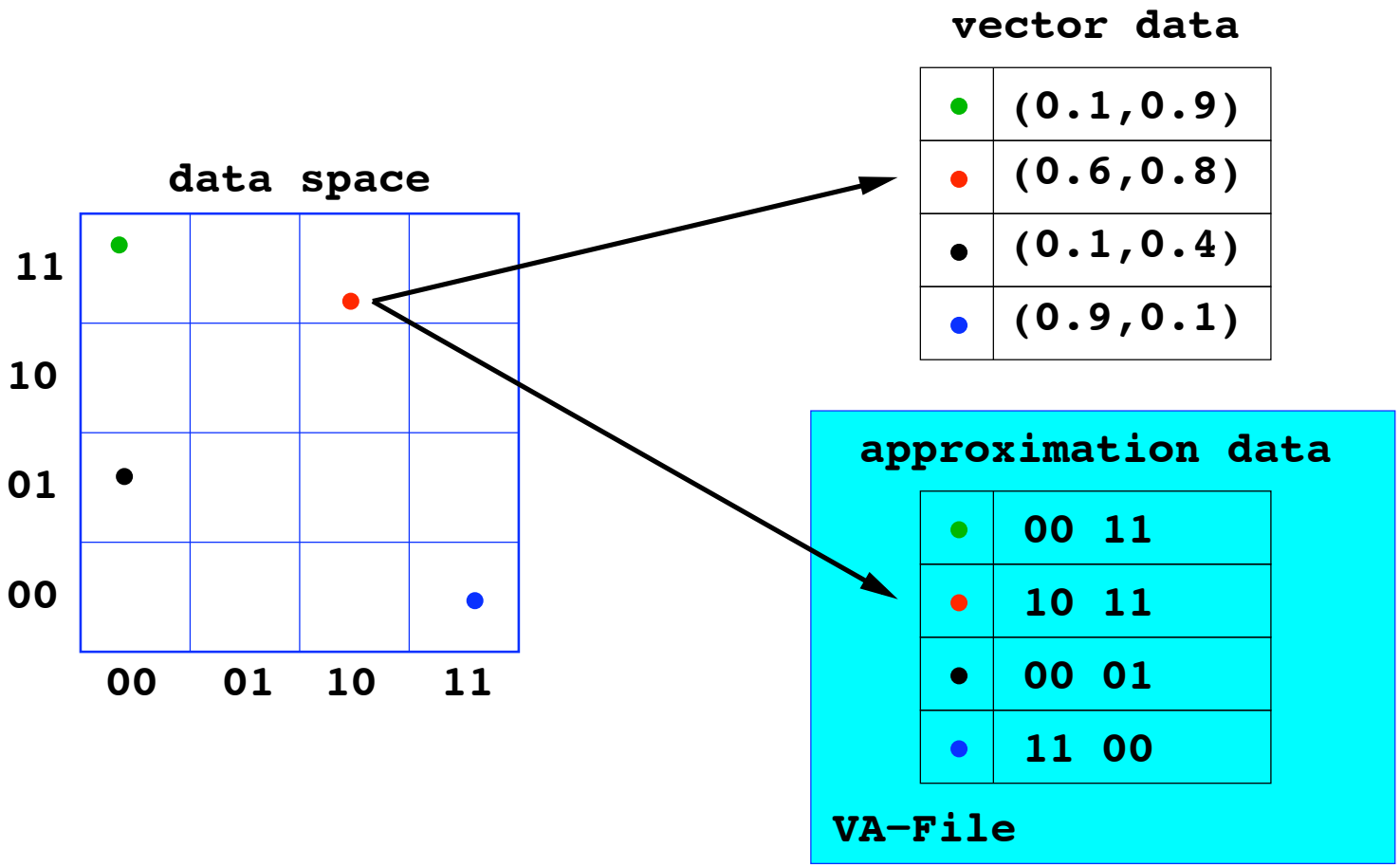
Signature Files



The Vector-Approximation File (VA-File)



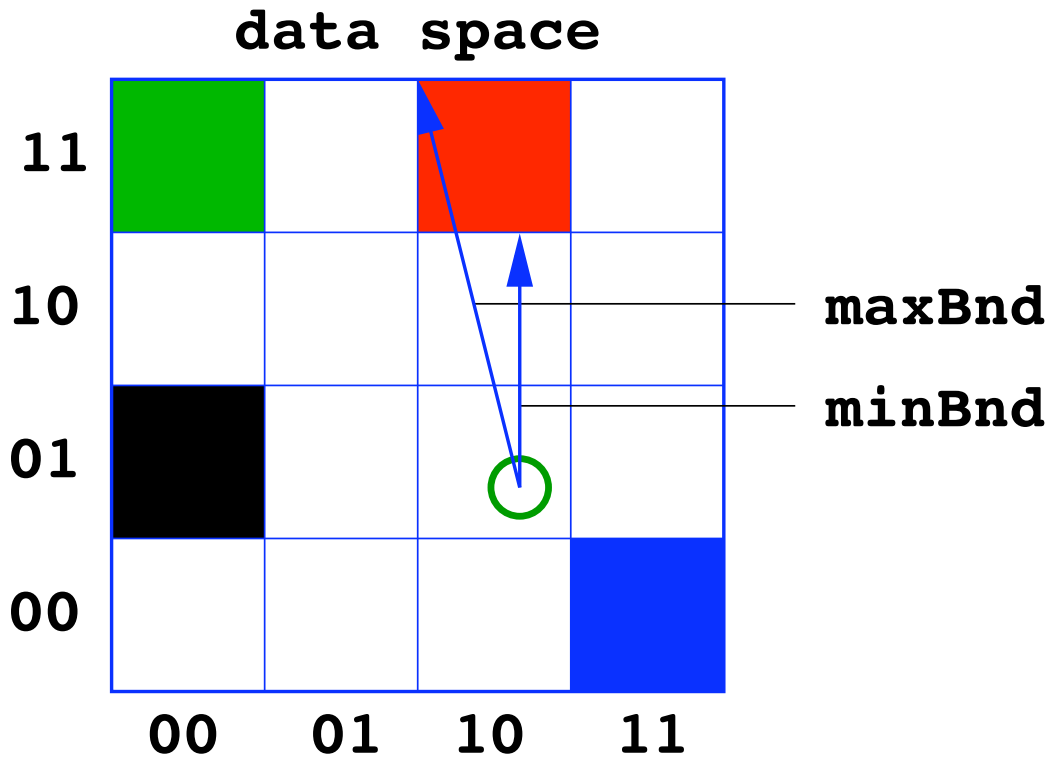
VLDB 1998: The Vector-Approximation File (VA-File)



VA-File – Minimum and Maximum Bounds

query:
(0.67,0.31)

approximation:
10 11

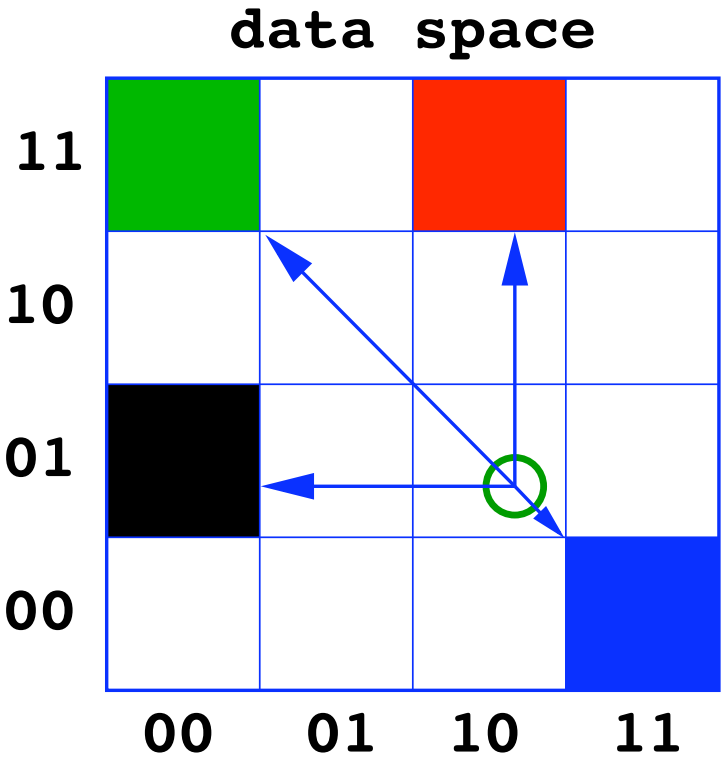


Distance contributions from grid boundaries can be precomputed

VA-File – Search Phase 1 – Filtering

Phase 1:

- calculate minBnd and maxBnd for each point
- eliminate those points that *cannot possibly* be the nearest neighbour

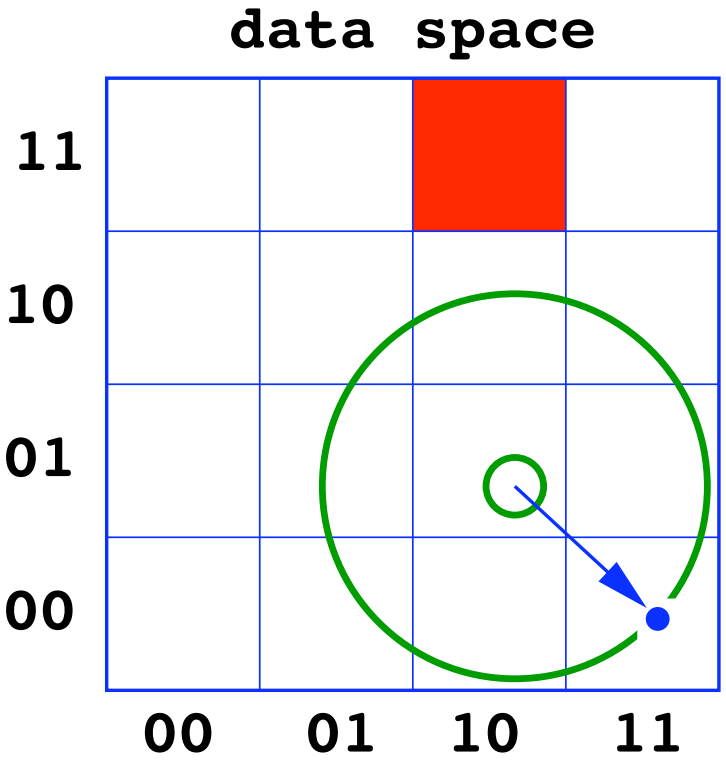


approx.	minBnd	maxBnd	rank
10 11	0.39	0.68	2
11 00	0.11	0.45	1
00 11	0.58		×
00 01	0.47		×

VA-File – Search Phase 2 – Vector Search

Phase 2:

- visit vectors in *increasing order* of `minBnd` calculating actual distance
- stop when `minBnd` exceeds nearest neighbour encountered so far



Phase 1 output:

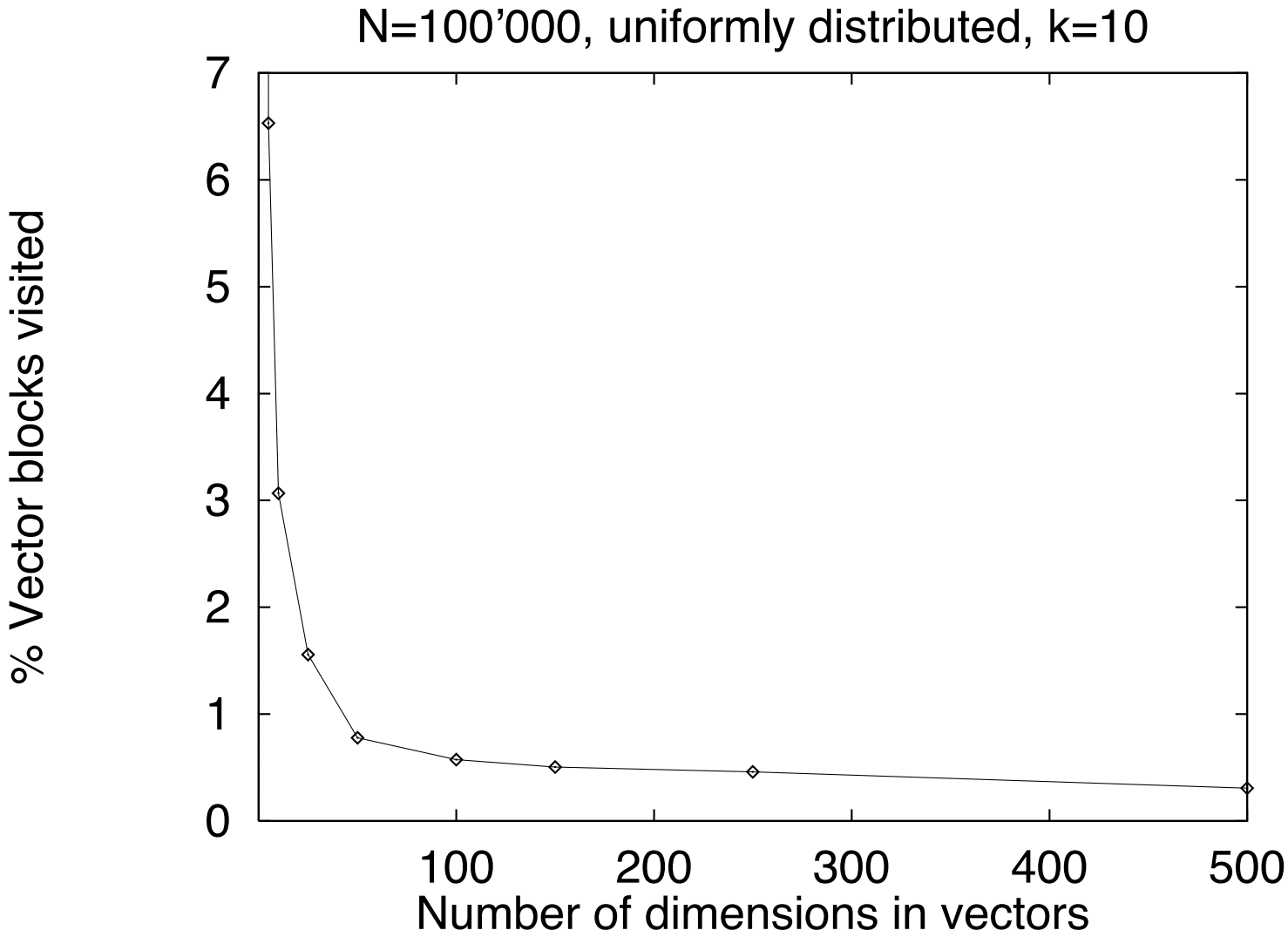
approx.	minBnd	maxBnd	rank
11 00	0.11	0.45	1
10 11	0.39	0.68	×

Phase 2:

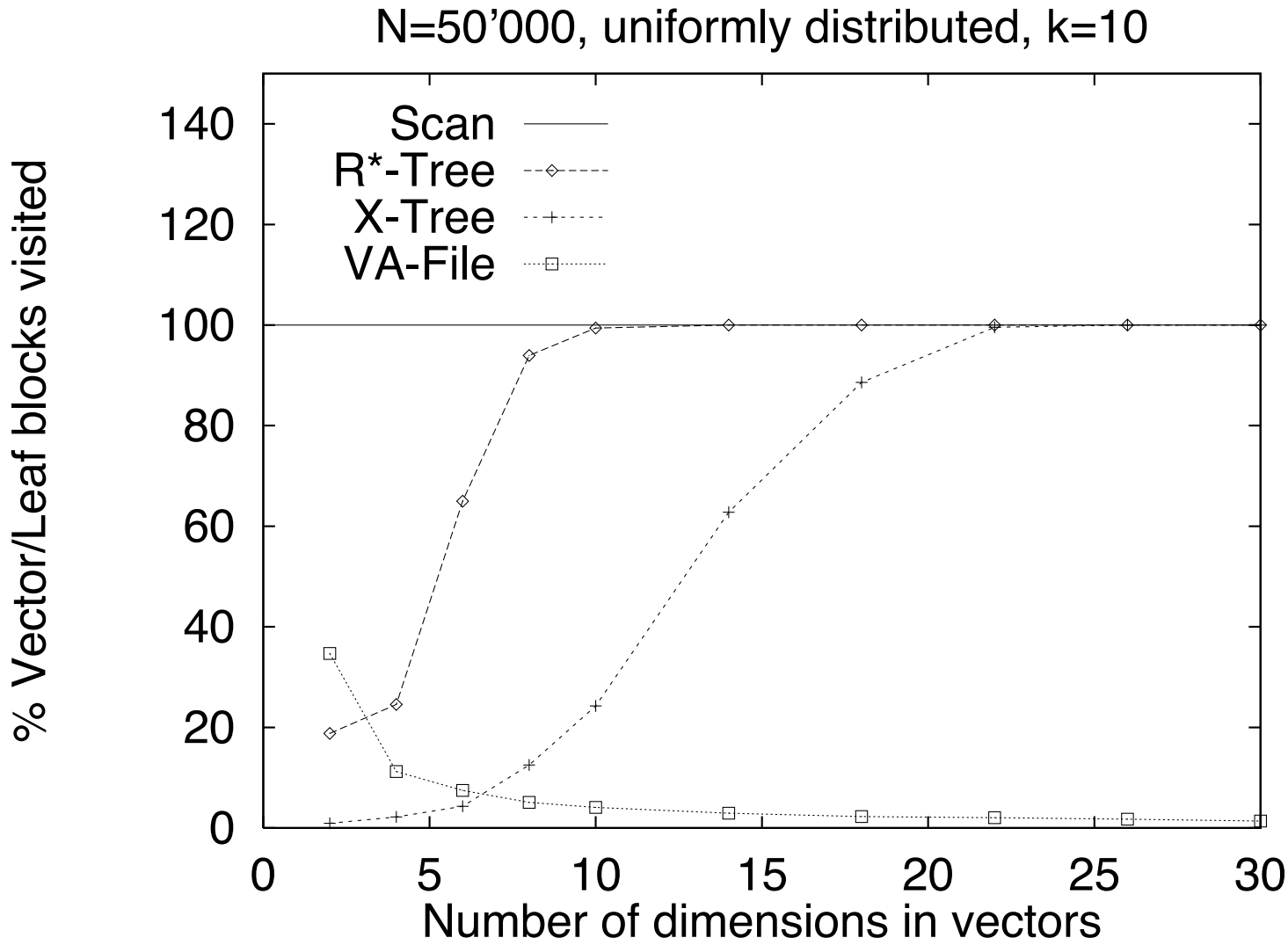
vector data	distance
0.1 0.9	0.29 (< 0.39)

Selectivity experiments . . .

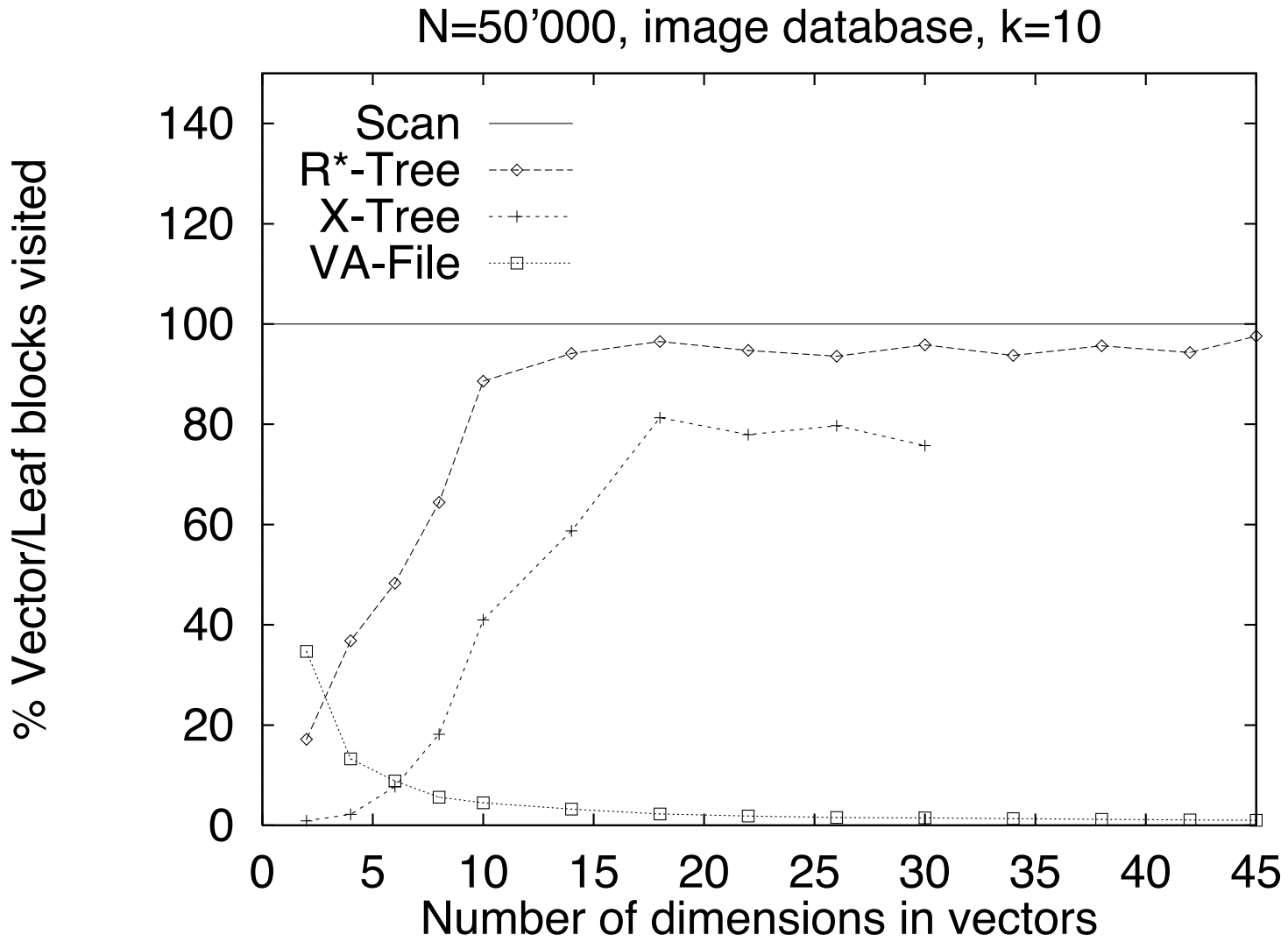
VA-File – Block selectivity with dimension



VA-File – Block selectivity with dimension – Synthetic data

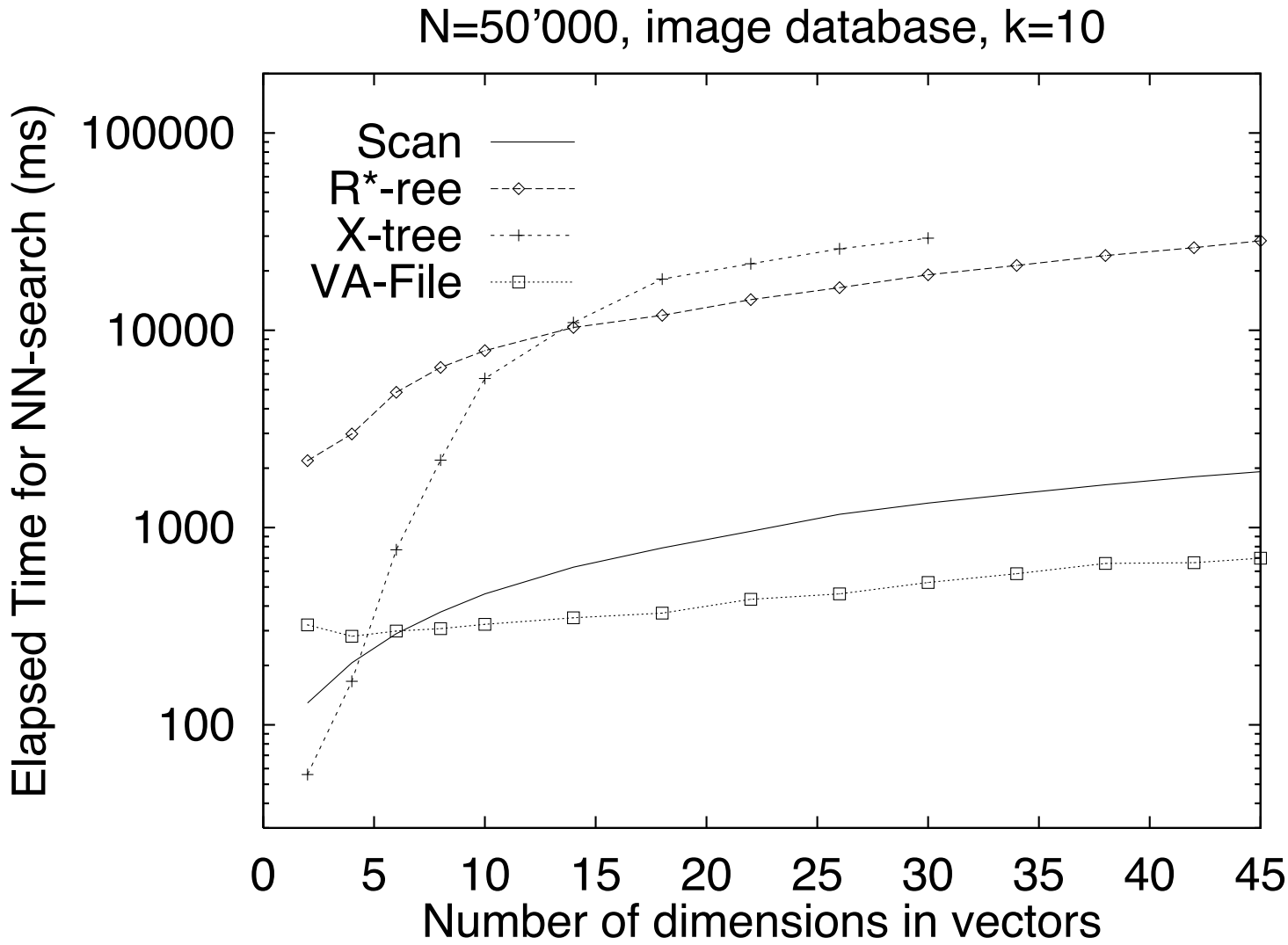


VA-File – Block selectivity with dimension – Image data



Wall-clock time experiments . . .

VA-File – Elapsed time – Image data



VLDB 1998: VA-File Commentary 1 – Dimensionality

Tree structures and clustering methods degenerate with increasing dimensionality

Sequential methods can work better the *more* dimensions there are

(and must be considered for validation of new approaches)

Even if all data is memory resident:

VA-File superior due to computational optimisations

VLDB 1998: VA-File Commentary 2 – Practicalities

Updates, concurrency and parallelisation are straightforward

(while these issues can be tricky for hierarchical methods)

Integration into the query engine:

- integration with text retrieval, predicate constraints and signature files is straightforward
- likewise search over multiple features, approximate answers
- . . .

Contents, again

VLDB 1998: Impact

Impact – shifted focus of research:

- no more trees!
  (almost)

- what does similarity *mean*?
  how should queries be expressed?
  how to interact with query engine?
  how do we extract better features from images?
  how do we incorporate segmentation?
  how do we integrate segmentation into the query engine?

Hardware Parameters

CPUs	10-30 times faster
main memory	2-4 times faster cheap, 64GB server not unusual
disks	2-4 times faster sequential access 10-30% faster random access
solid-state drives	200 times faster for random access

Effect of changed hardware parameters?

Theoretical results:

- access probability becomes linear regardless of underlying hardware
- hierarchical structures suffer from their random access patterns regardless of hardware improvements
- there remain substantial overheads for hierarchical methods:

  . . . all to no advantage

Effect of changed hardware parameters?

Performance:

- performance of trees boosted by factor of 200?
  - in principle, yes

    but scan bandwidth increased too
  - still have to read majority of tree:

    hence still linear, and still suffer increased complexity of tree structure without any substantial performance gain
  - computational costs are an important factor too

VA-File – Subsequent Work

Extensions:

- parallelisation, approximate answers, complex queries (GeVAS), region-based search, relevance feedback

Projects:

- Chariot, HERMES, ETH World, DELOS DMS (Diligent), QBS (SNF project)

Local outputs:

- 3 PhDs, 20^+ student projects, 10^+ publications

A Better Understanding of the Problem

When is "Nearest Neighbor" Meaningful?
Beyer, Goldstein, Ramakrishnan, Shaft
ICDT 1999

What is the Nearest Neighbor in High Dimensional Spaces?
Hinneburg, Aggarwal, Keim
VLDB 2000

On the Surprising Behavior of Distance Metrics in High Dimensional Spaces
Aggarwal, Hinneburg, Keim
ICDT 2001

More Oddities of High-Dimensional Spaces

When is "Nearest Neighbor" Meaningful?
Beyer, Goldstein, Ramakrishnan, Shaft
ICDT 1999

Conclusion

So, what have we learned?

- high-dimensional spaces are odd
- for hierarchical schemes:
  selectivity degrades rapidly as dimensionality increases
- sequential methods (including VA-File):
  outperform hierarchical schemes as dimensionality increases

  . . . and they're a lot easier to integrate within a database engine

If you're dealing with high-dimensional data . . .

be careful