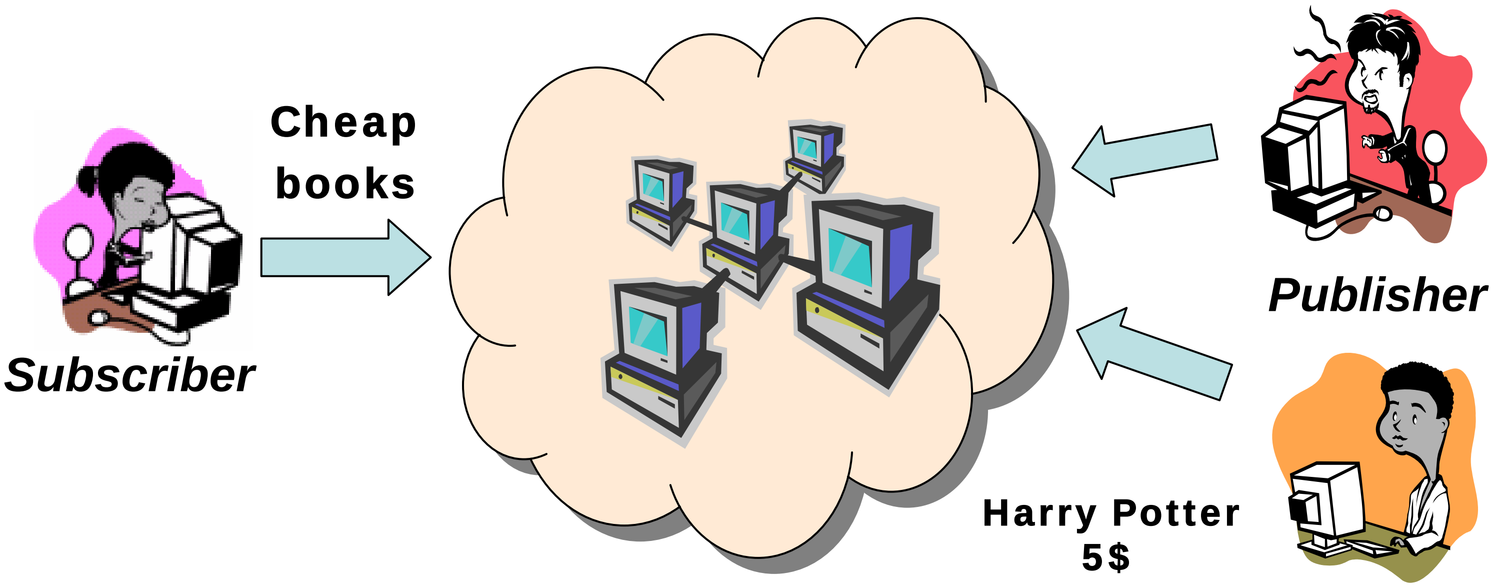
Approximate Covering Detection Among Content-Based Subscriptions Using Space Filling Curves

Zhenhui Shen
Akamai Technologies
(work done while at Iowa State University)

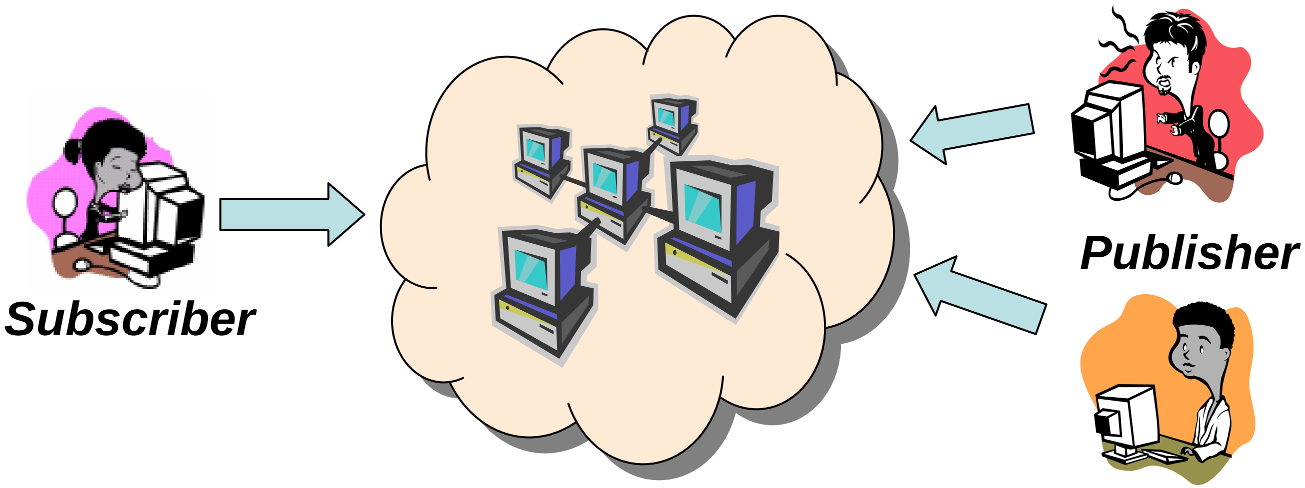
Srikanta Tirthapura
Dept. of Electrical and Computer Engg.
Iowa State University

Publish-Subscribe (Pub-Sub)



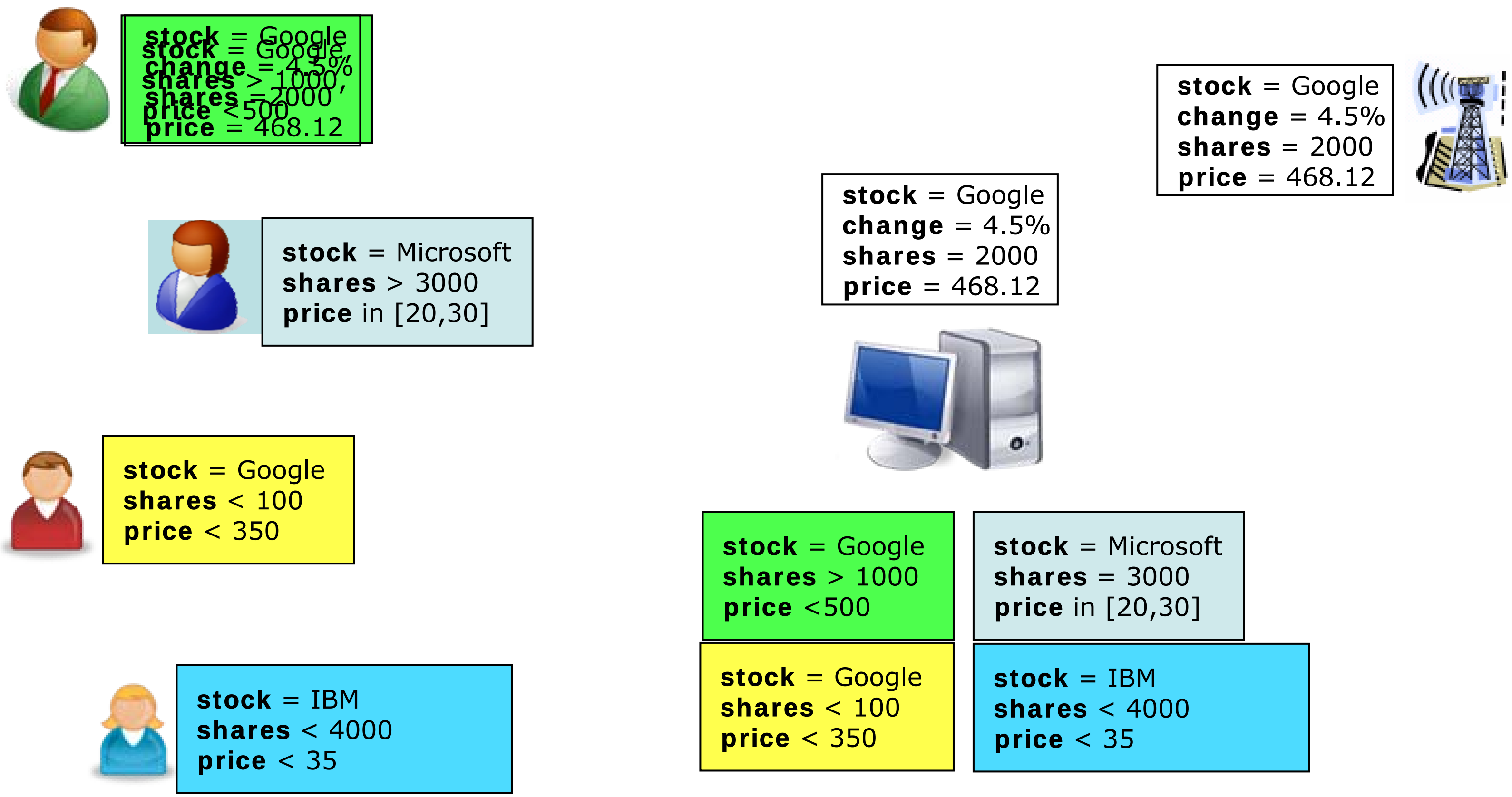
- Publishers generate ***events***
- Subscribers register interests via ***subscriptions***
- Pub-Sub middleware forwards events to interested subscribers

Content-based Pub-Sub

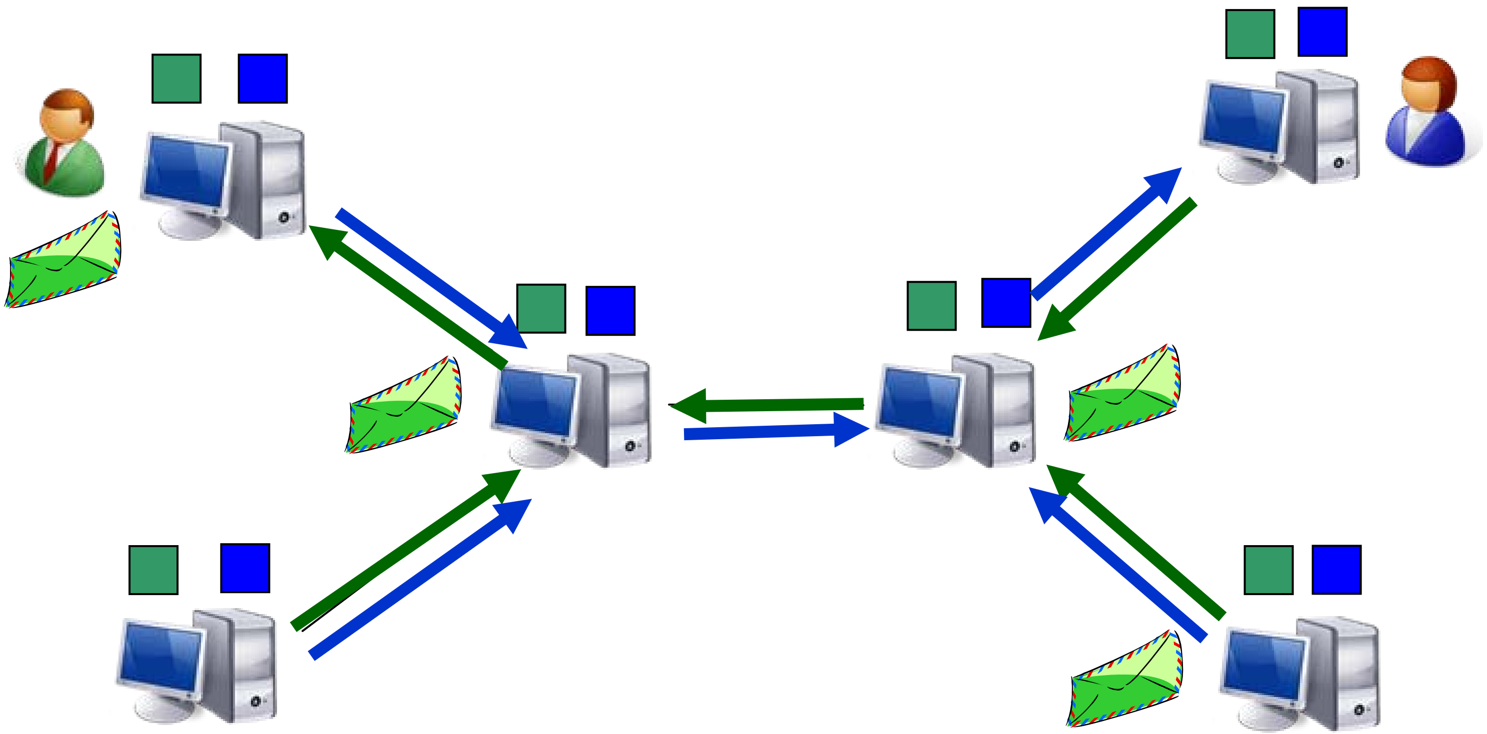


- **Event**: a set of <attribute,value> pairs
  (issue=Microsoft, price=$27, vol = 1000)
- **Subscription**: conjunction of predicates
  (issue=Microsoft, 25 < price < 30, vol > 500)

Content-based Event Filtering



Distributed Pub-Sub

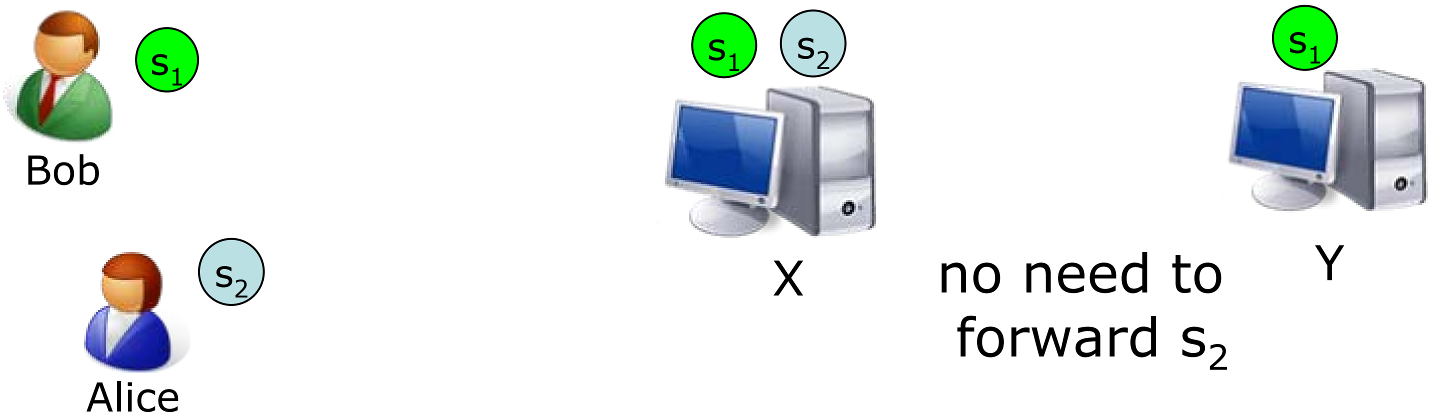


Subscription Propagation can be Costly!

Subscription Covering

$$s_1 = (price \in [0,50],\ year \in [1990,2004])$$

$$s_2 = (price \in [0,30],\ year \in [1998,2000])$$

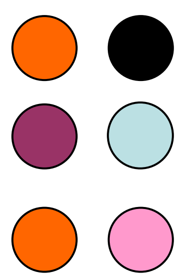


Benefits of Covering

- Reduces number of subscriptions forwarded (less processing overhead)
- Reduces routing table size (faster event filtering)

The Covering Detection Problem

$$s = (price \in [0,50],\ year \in [1990,2004])$$

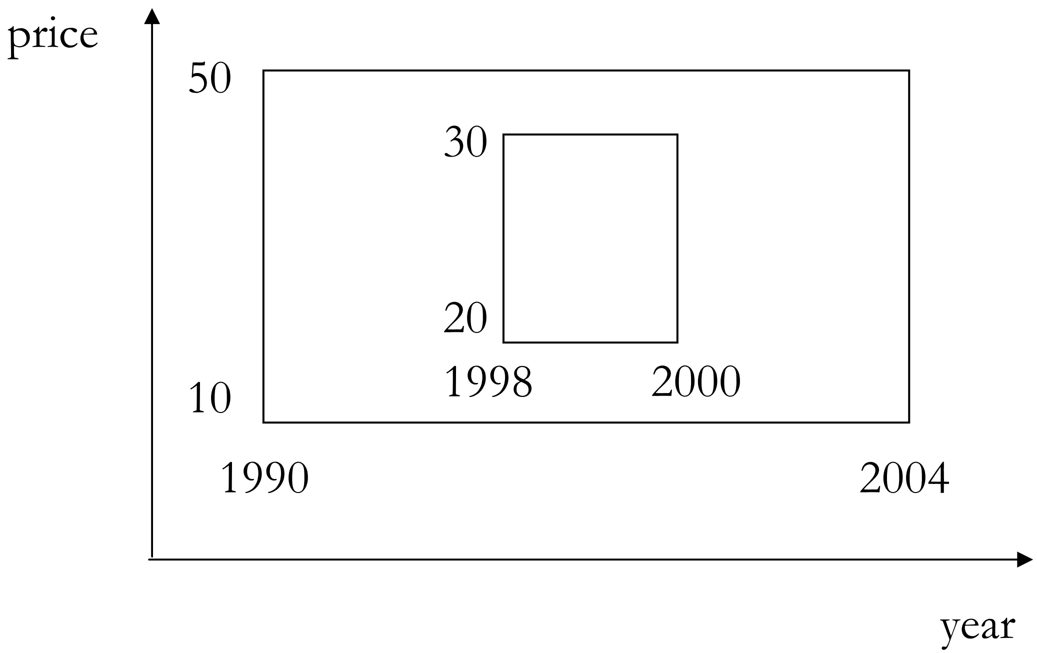


Is S covered by an existing subscription?

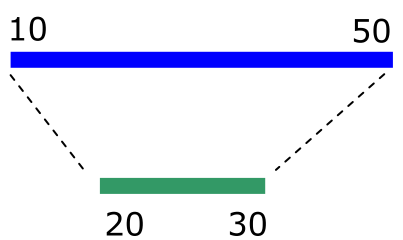
Covering = Rectangle Containment

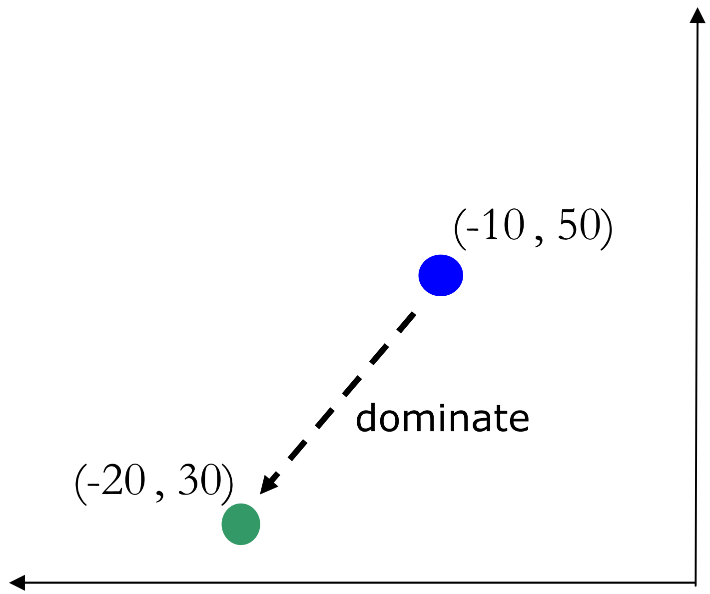
$$s_1 = (price \in [10,50],\ year \in [1990,2004])$$

$$s_2 = (price \in [20,30],\ year \in [1998,2000])$$



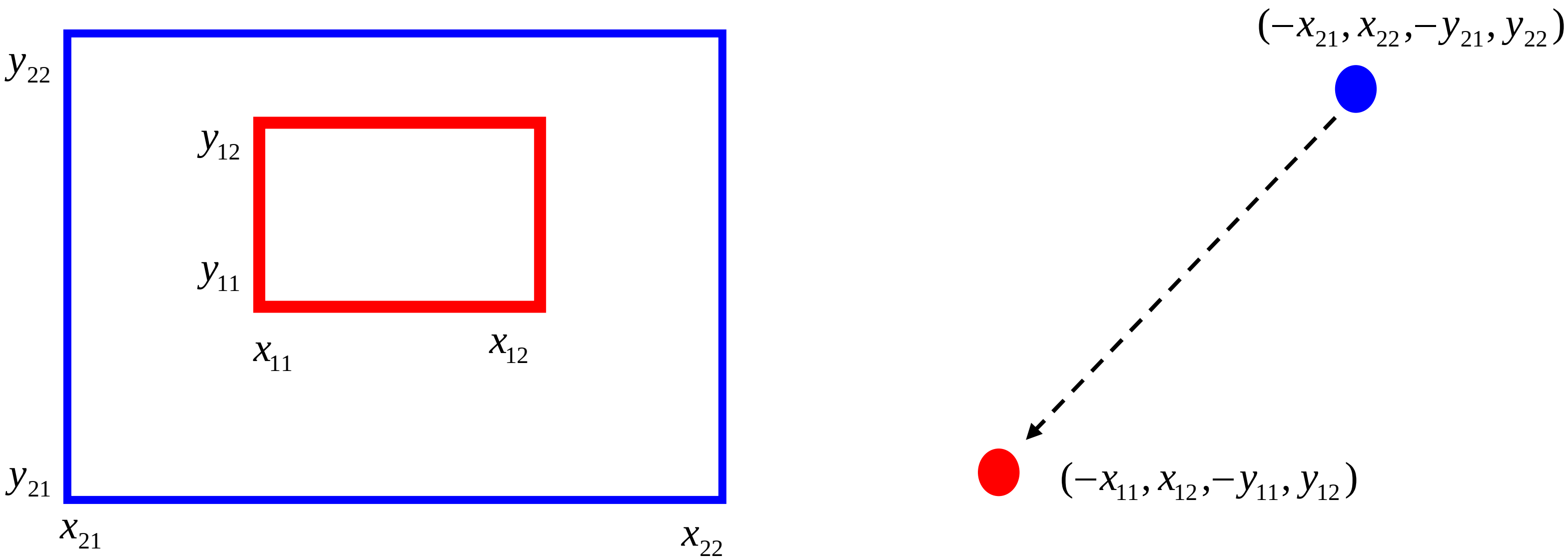
Rectangle Containment Reduces to Point Dominance





Point P_1 **dominates Point** P_2 iff every coordinate of P_1 is **no less than** the corresponding coordinate of P_2

Rectangle Containment Reduces to Point Dominance

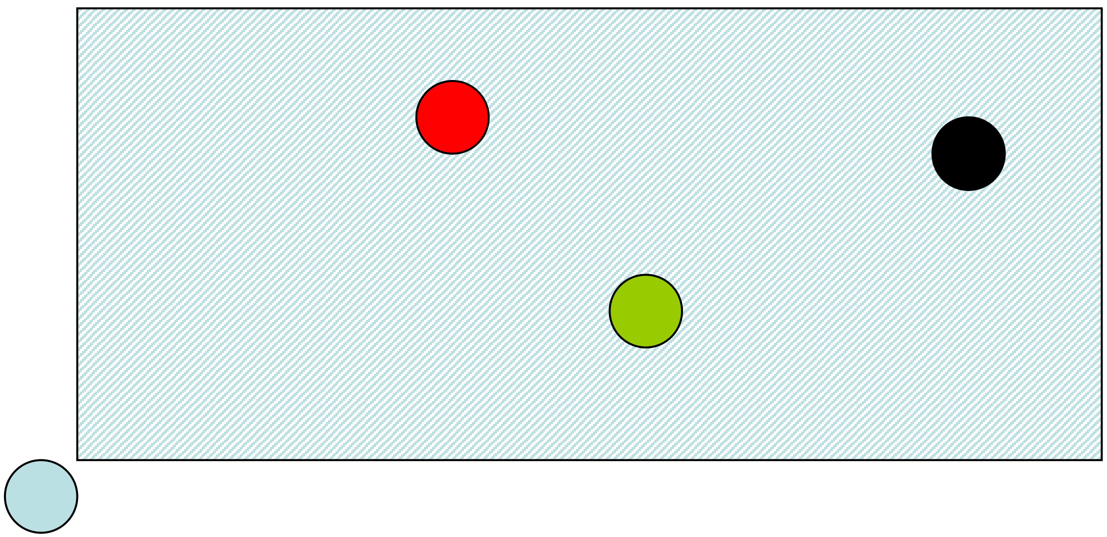


Edelsbrunner and Overmars (1982) show that the reduction goes both ways.

Covering = High Dimensional Range Search

d-dimensional subscription $[x_{11,} x_{12}], [x_{21,} x_{22}], \cdots [x_{d1,} x_{d2}]$

2d-dimensional point $(-x_{11,} x_{12}, -x_{21,} x_{22}, \cdots - x_{d1,} x_{d2})$



High Dimensional Range Searching is Hard

- Fredman (Journal of ACM, 1981)
  a mixed seq. of *n* insertions, deletions and queries requires $\Omega(n \log^d n)$ time in d-dim space

- Chazelle (Journal of ACM, 1990)
  Any structure trying to answer query in $O(\text{polylog}(n) + k)$ time must have size $\Omega(n(\log n / \log\log n)^{d-1})$

- "Curse of Dimensionality" – Indexing cost is exponential in the number of dimensions

To Cover or Not to Cover ?

Approximate Covering
Cheaper than Exact covering
Still useful

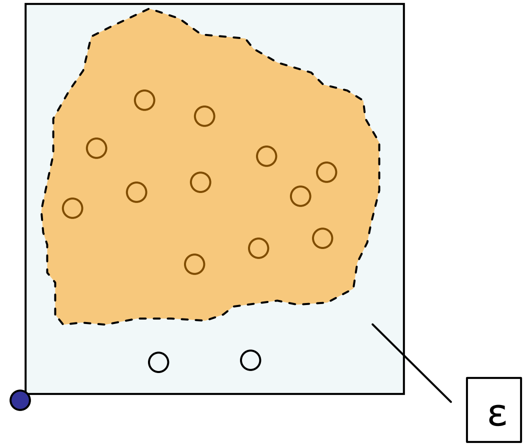
ignore covering
it still works

Opponent

Use covering, makes system
leaner and meaner

Proponent

Approximate Covering



- For a user given ε, search (1- ε) fraction of search space
- Safe Optimization: A subscription will never be held back if it is not covered
- Main Technical Result: Even for small ε, approximate covering can be substantially faster than exact covering

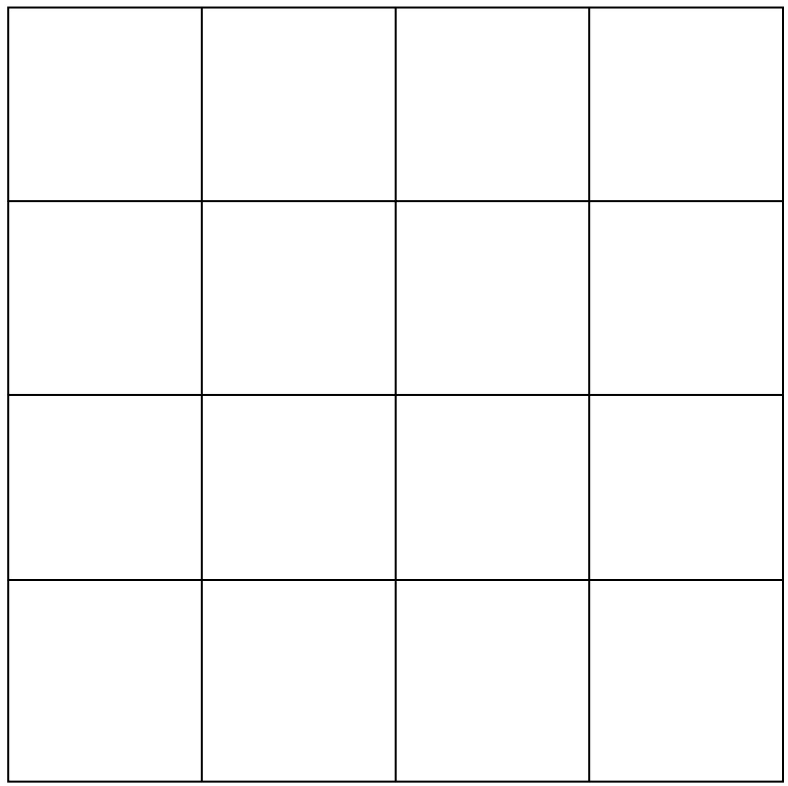
Related Work

- Covering has been implemented in many distributed pub-sub systems, including SIENA, JEDI, REBECA
- Li, Hou and Jacobsen (ICDCS 2005)
  - Exact covering using binary decision diagrams
- Shen, Tirthapura, Aluru (PDCS 2005)
  - Exact Covering for Numeric Subscriptions
- Ouksel, Jurca, Podnar and Aberer (Middleware 2006)
  - Covering by union of subscriptions
  - Monte Carlo algorithms

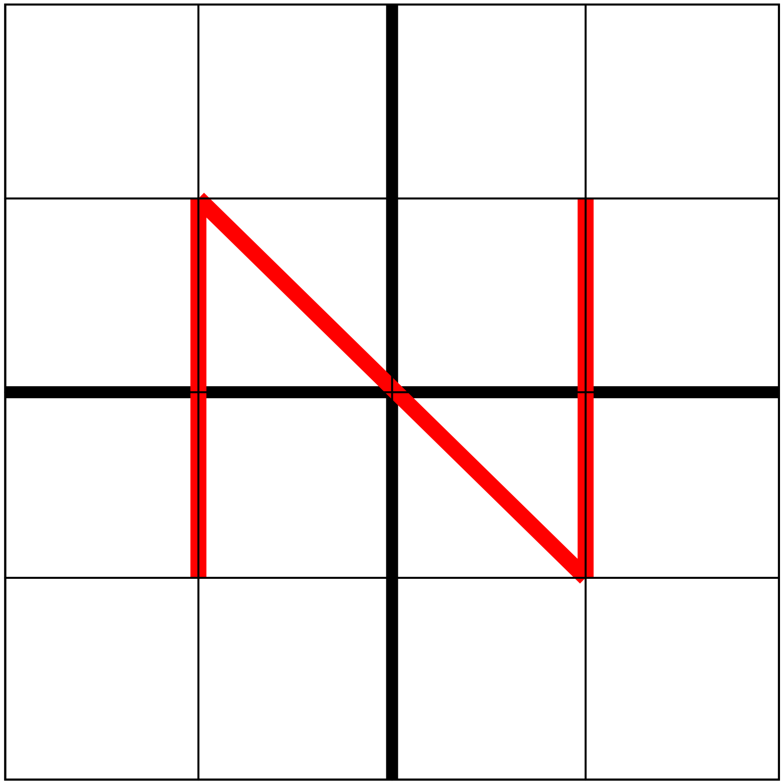
Data Structures for Covering (numeric subscriptions)

- High dimensional spatial data structures
- Space Filling Curves (SFC)
  - Maps points in high dimensions to a single dimension
  - Tried and tested
- Other Alternatives
  - K-d trees
  - Quad Trees

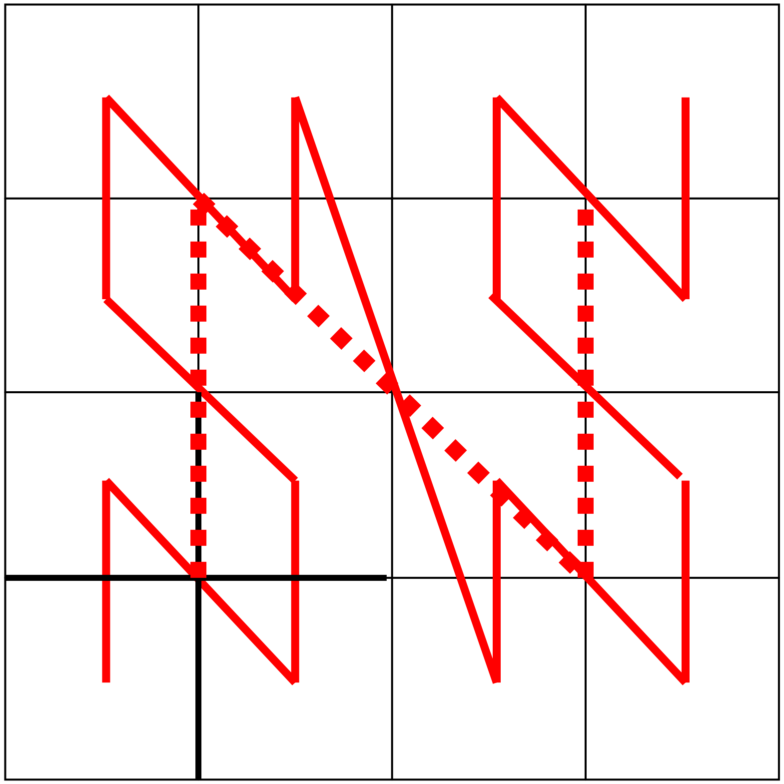
Z Space Filling Curve



Z Space Filling Curve



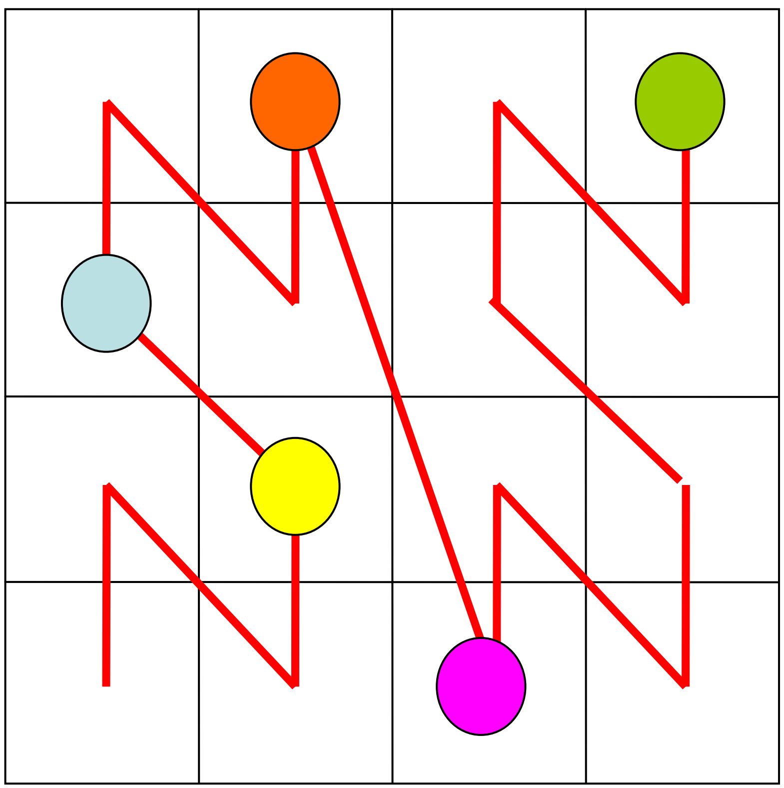
Z Space Filling Curve

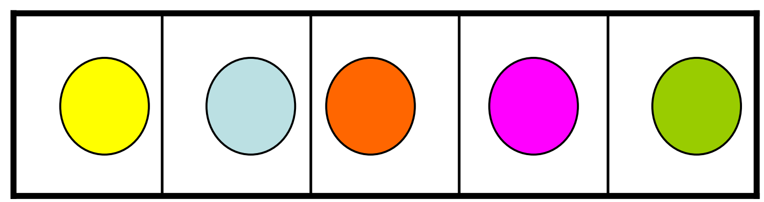


Linear ordering of all cells in the space

SFC Array

An array containing all input points sorted in the SFC order

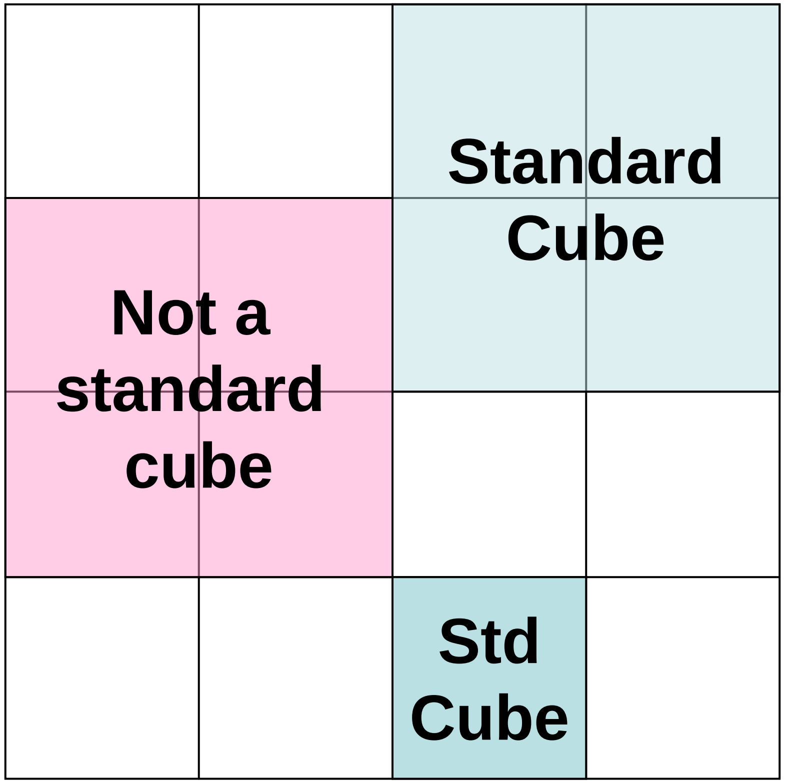




SFC array

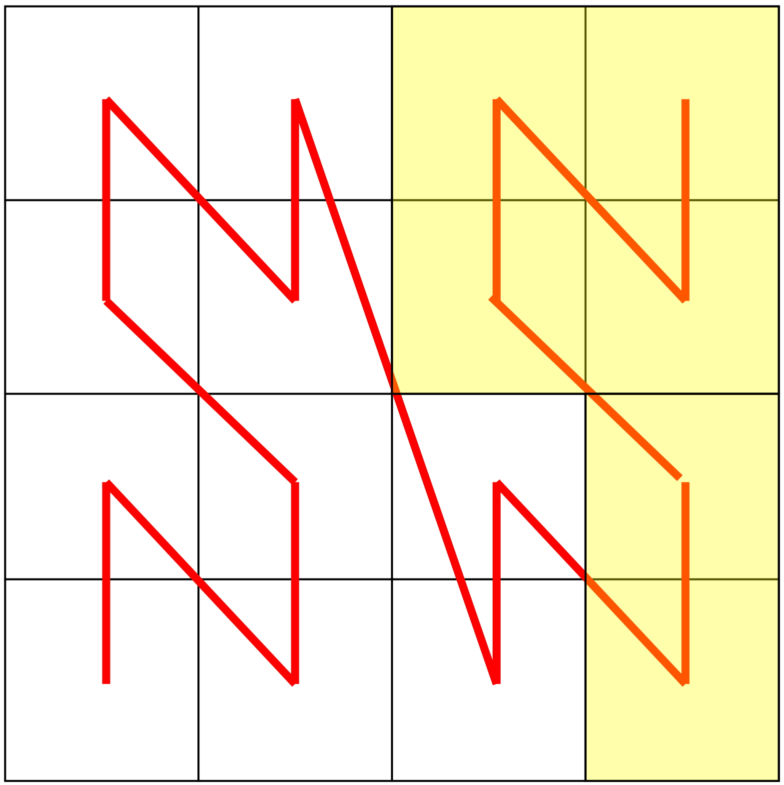
Standard Cube

Any cube arising during a recursive decomposition of the space.



Run

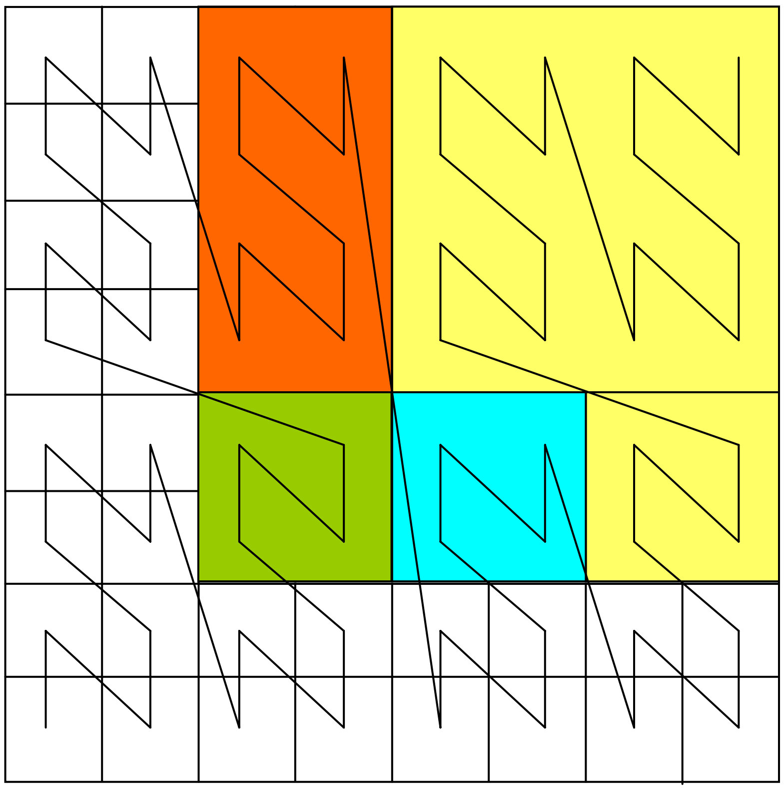
Set of cells that are consecutive in the SFC



Runs are more complicated sets than standard cubes

Fact: Every standard cube is a single run

Covering Detection Using SFC



Subscription is a point

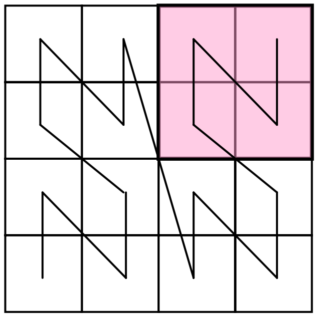
Region to be searched is an extremal rectangle

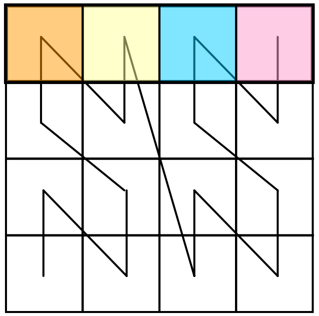
Divide the extremal rectangle into minimum number of runs

Cost of Covering using SFC

Worst-case Cost = Min # of Runs

Lucky



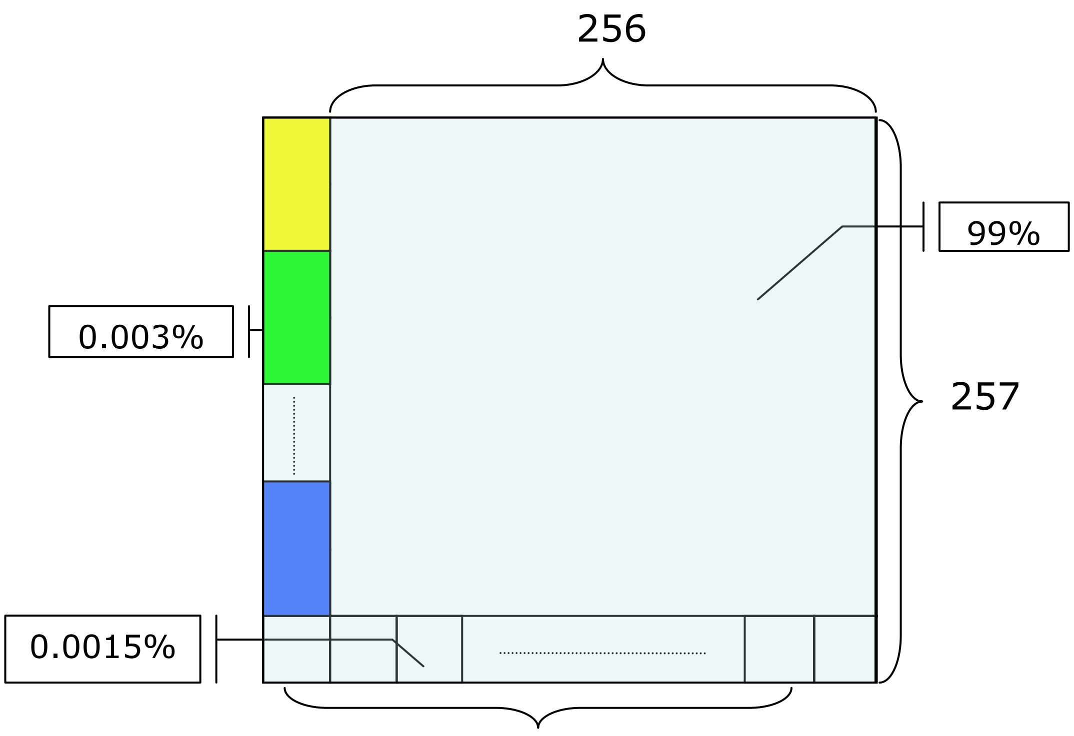


Unlucky

[Moon, Jagadish, Faloutsos, Saltz, IEEE TKDE 2001]

For the Hilbert SFC, the average # of runs in a d-dimensional rectilinear polyhedron is proportional to its surface area

Exact vs. Approximate Covering



385 runs in total, lots of them in marginal area

- Exact covering searches 385 runs
- 99%-covering searches only a single run

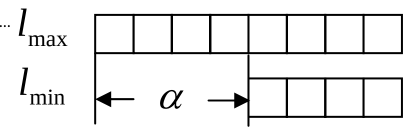
Exact vs. Approximate Covering

- Is approximate covering always much cheaper than exact?
- Not quite..
- However, true for all extremal rectangles with a good “aspect ratio”

Upper Bound on Approximate Covering

Theorem: For a d-dimensional hyper rectangle, let $l_{\max}$ and $l_{\min}$ be the length of the longest (shortest) side of the rectangle, let α be its "aspect ratio", the cost of approximate search using Z SFC, which examines (1-ε) of the solution space, is no greater than

$$O\left(\log\left(\frac{d}{\varepsilon}\right)\cdot\left(2^{\alpha+1}\frac{d}{\varepsilon}\right)^{d-1}\right)$$



$$\alpha \approx \log_2(l_{\max}/l_{\min})$$

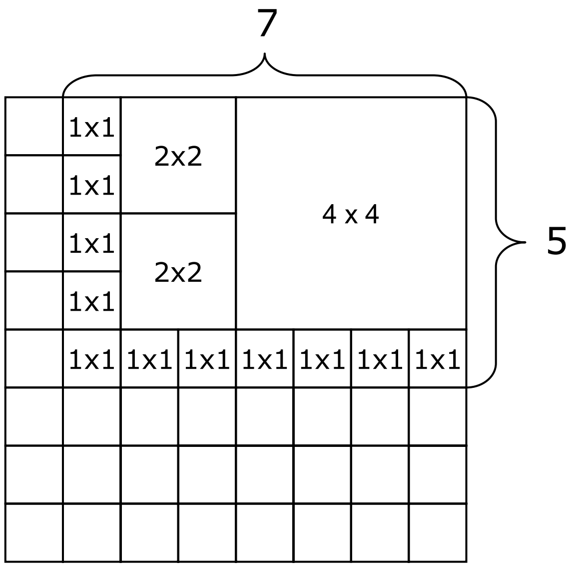
while the cost of exact covering using Z SFC is

$$\Omega\left(\left(2^{\alpha-1}l_{\min}\right)^{d-1}\right)$$

Algorithm for Approximate Covering

Algorithm deals with standard cubes, rather than general runs

Greedy Algorithm: Examine the standard cubes in the extremal rectangle in the decreasing order of size.



e.g. $\varepsilon = 10\ \%$

$\gamma = 45.7\%$

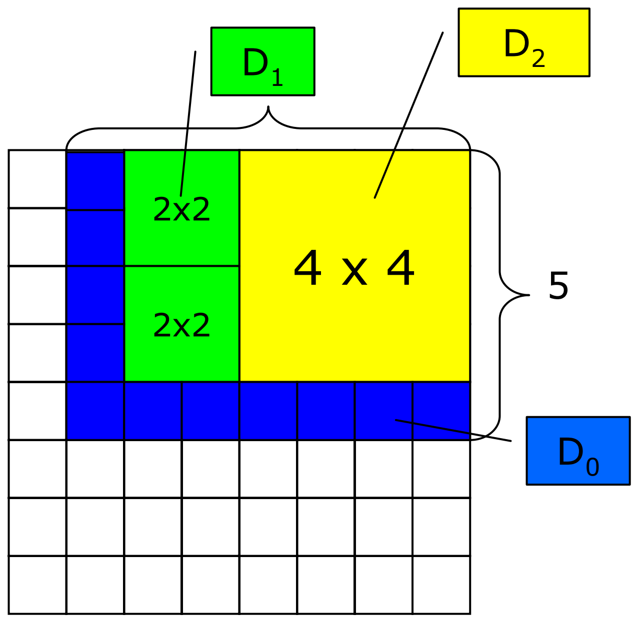
$\gamma = 57.1\%$

$\gamma = 68.6\%$

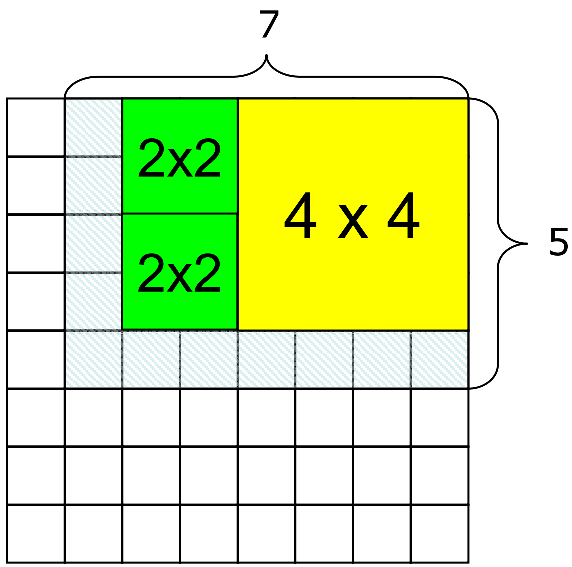
$\gamma = 91.4\%$

Track γ, the fraction of volume examined

Complexity of Approximate Covering or, "How many runs?"



Complexity of Approximate Covering



2^2	2^1	2^0
1	1	1
1	0	1

2^2	2^1	2^0
1	0	0
1	0	0

D_2

2^2	2^1	2^0
1	1	0
1	0	0

$D_2 \cup D_1$

This characterization used to find the number of standard cubes of each size.

Complexity of Approximate Covering

Side Lengths

1	1	0	1
0	1	1	0
2^3	2^2	2^1	2^0

8 x 8 standard cubes

1	0	0	0
0	0	0	0

None!

4 x 4 standard cubes

1	1	0	0
0	1	0	0

12 x 4 / 16
= 3

2 x 2 standard cubes

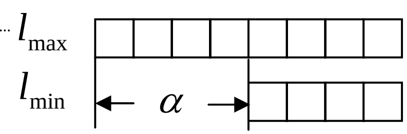
1	1	0	0
0	1	1	0

(12 x 6 - 48)/4
= 6

Upper Bound on Approximate Covering

Theorem: For a d-dimensional hyper rectangle, let $l_{\max}$ and $l_{\min}$ be the length of the longest (shortest) side of the rectangle, let α be its "aspect ratio", the cost of approximate search using Z SFC, which examines (1-ε) of the solution space, is no greater than

$$O\left(\log\left(\frac{d}{\varepsilon}\right)\cdot\left(2^{\alpha+1}\frac{d}{\varepsilon}\right)^{d-1}\right)$$



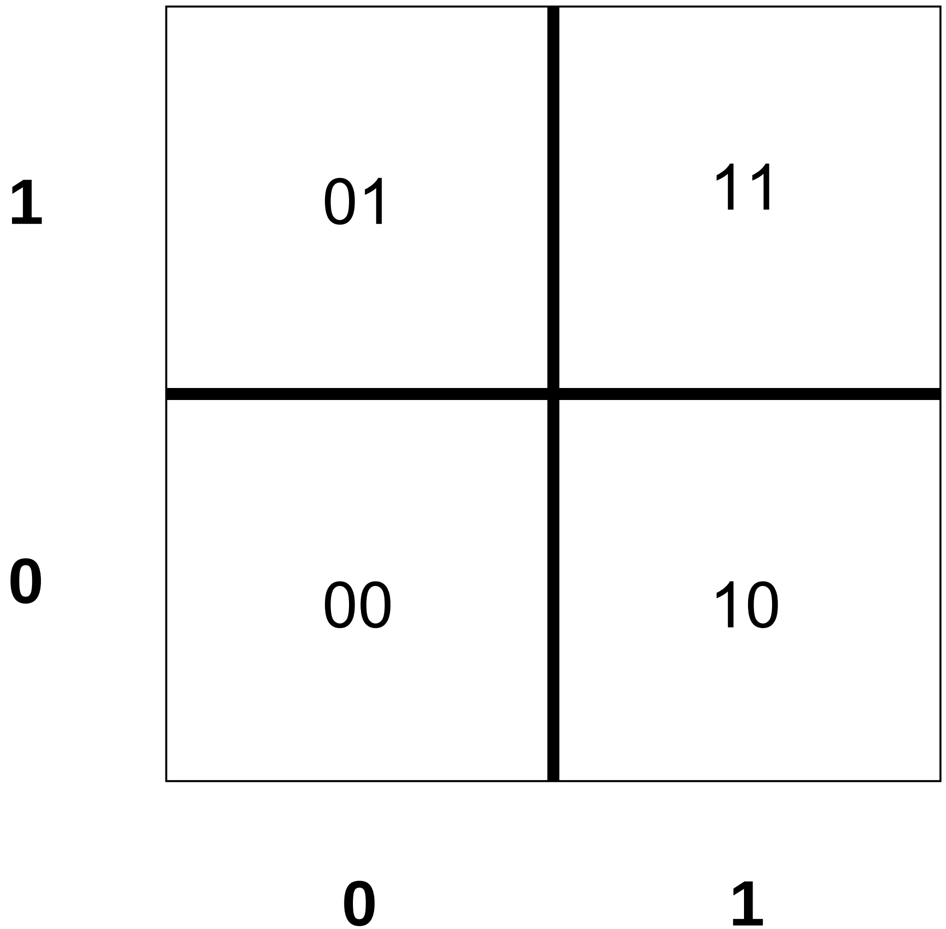
$$\alpha \approx \log_2(l_{\max}/l_{\min})$$

Lower Bound on Exact Covering

Lower bound on the number of runs, rather than the number of standard cubes

Idea: Construct an extremal rectangle with a set of standard cubes S such that no two standard cubes in S can belong to the same run

Keys of Standard Cubes



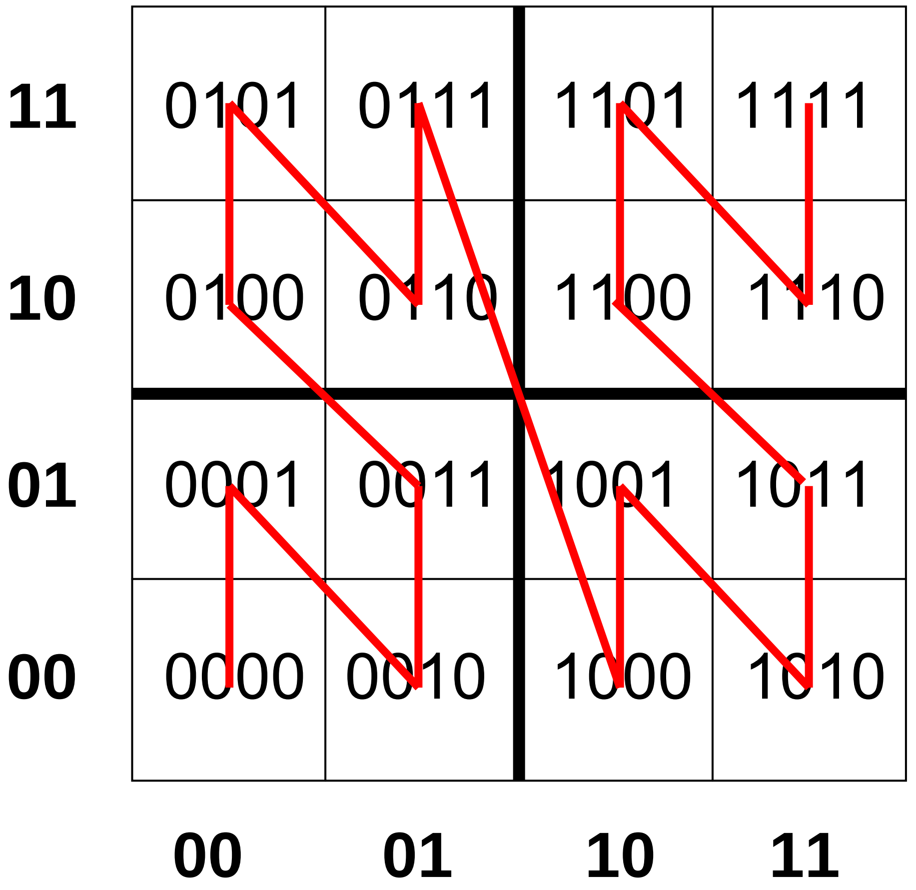
Key is the position of the cube in the SFC

Keys of Standard Cubes

11	0101	0111	1101	1111
10	0100	0110	1100	1110
01	0001	0011	1001	1011
00	0000	0010	1000	1010
	00	**01**	**10**	**11**

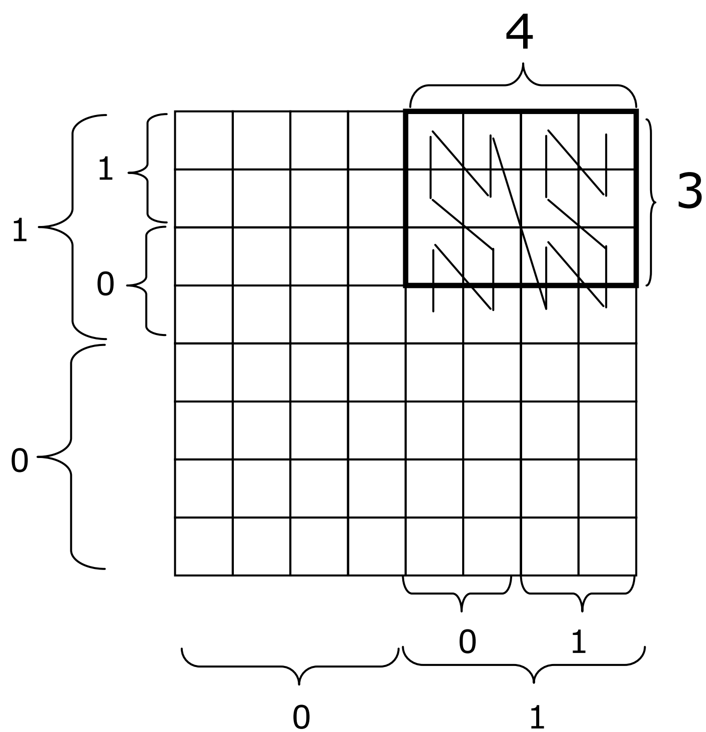
Key is the position of the cube in the SFC

Keys of Standard Cubes



Key is the position of the cube in the SFC

Lower Bound for Exact Covering



side length

1	0	0

0	1	1

key pattern

1	x	x

1	0	1

after interleaving

1	1	x	0	x	1

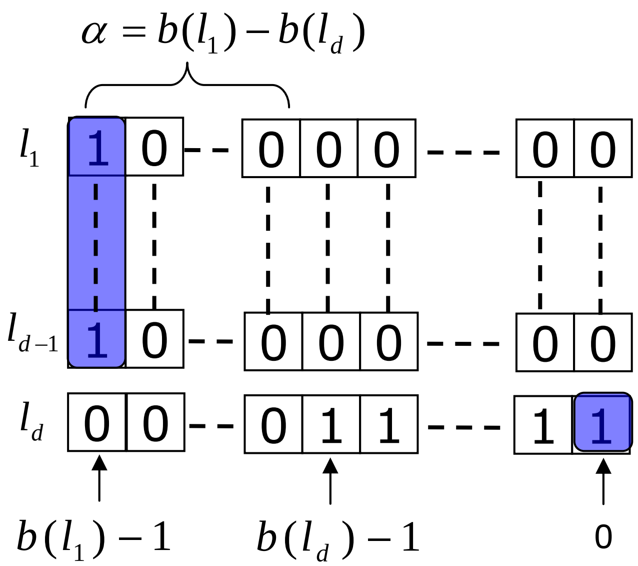
(1) no two cubes are adjacent

(2) can't be connected by other cubes which are in the same query region

Predecessor

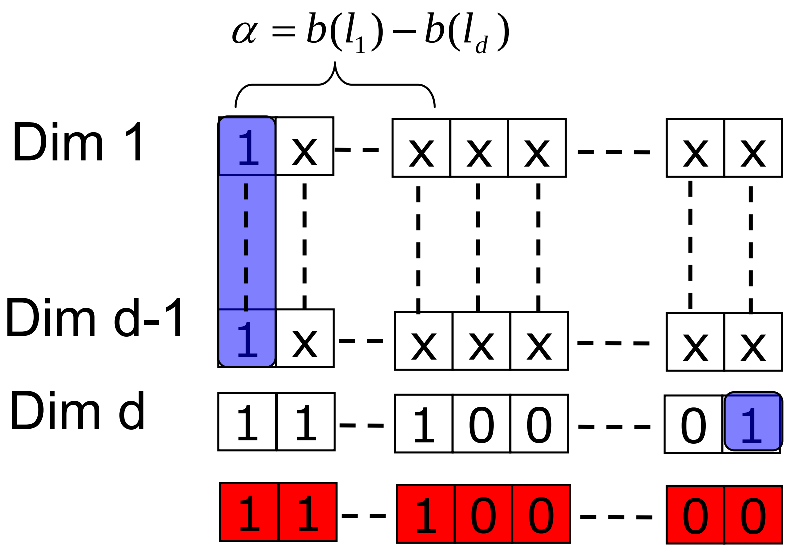
1	1	x	0	x	0

Generalization to *d*-Dimensional Extremal Rectangle



Aspect Ratio = α

Generalization to *d*-Dimensional Extremal Rectangle



after interleaving

$$\overbrace{1\cdots1}^{d}\underbrace{\overbrace{\times\cdots\times1}^{d}\cdots\overbrace{\times\cdots\times1}^{d}}_{\alpha-1}\underbrace{\overbrace{\times\cdots\times0}^{d}\cdots\overbrace{\times\cdots\times0}^{d}}_{b(l_d)-1}\overbrace{\times\cdots\times1}^{d}$$

(1) no two cubes are adjacent

(2) can't be connected by other cubes which are in the same query region

predecessor

$$\overbrace{1\cdots1}^{d}\underbrace{\overbrace{\times\cdots\times1}^{d}\cdots\overbrace{\times\cdots\times1}^{d}}_{\alpha-1}\underbrace{\overbrace{\times\cdots\times0}^{d}\cdots\overbrace{\times\cdots\times0}^{d}}_{b(l_d)-1}\overbrace{\times\cdots\times0}^{d}$$

$$runs(R) = cubes(R) = vol(R) = \prod_{j=1}^{j=d-1} 2^{b(l_j)-1} = \left(\frac{2^{b(l_d)}\cdot 2^{\alpha}}{2}\right)^{d-1} = \left(2^{\alpha-1}\cdot l_{\min}\right)^{d-1}$$

Summary

- **Approximate Covering may be much cheaper than exact covering, while providing many of the advantages**
- “Safe” Optimization, does not affect correctness
- Algorithm for numeric subscriptions using space filling curves
  - Provably fast for subscriptions with small aspect ratio
  - Performance better than lower bound for exact covering
  - Very practical and uses simple data structures

Future Directions

- Does approximation help other numeric covering data structures such as k-d trees?

- Covering Data structures for non-numeric (string) subscriptions?