BeyOND – Unleashing BOND

Thomas Bernecker, Franz Graf, Hans-Peter Kriegel,
Christian Moennig and Arthur Zimek

Ludwig-Maximilians-Universität München (LMU)
Munich, Germany
http://www.dbs.ifi.lmu.de
{bernecker, graf, kriegel, zimek}@dbs.ifi.lmu.de
moennig@cip.ifi.lmu.de

Outline

1. **Background**
   - **Motivation: k-nearest neighbor search in high-dimensional databases**
   - BOND revisited
2. **Introducing BeyOND**
   - Filtering objects via distance approximations
   - Sub Cubes, MBRs
3. **Experimental Evaluation**
4. **Conclusions**

Motivation

- Similarity search in high-dimensional space is

  ☺ important in cases of images, e-commerce, etc.

  ☹ slow

- The suitability of index-based solutions depends on the data distribution
- Open question: relevant vs. irrelevant attributes
- Similarity search in subspaces:
  - Fix query attributes beforehand
  - Use multiple pivot points to derive upper and lower bounds
  - Process data vertically to reduce the high-dimensional space

BOND Revisited (1)

- BOND[1]: k-nearest neighbor search on high-dimensional data
  - Resolves feature vectors (FVs) column-wise
  - Ranking of columns w.r.t. relevance
  - Pruning of columns using a branch-and-bound approach
  - Resolved part is known exactly
  - Unresolved part has to be approximated
  - Resolving stops when approximation is „good enough"
  - Support of subspace queries
  - Distance metrics:
    - Histogram intersection (uncorrelated dimensions)
    - Euclidean distance

[1] de Vries, Mamoulis, Nes, Kersten: *Efficient k-NN Search On Vertically Decomposed Data* (SIGMOD'02)

BOND Revisited (2)

- Restrictions of BOND:

1. The approach works only on Zipfian distributed data.

2. The feature values are normalized to [0,1] in each dimension.

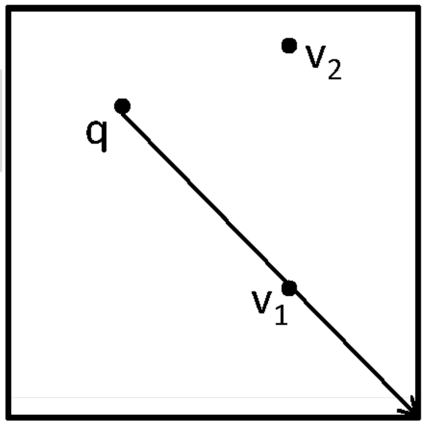
3. The proposed bounds are loose. The validity of stricter bounds (Bond advanced) depends on a certain resolve order of the columns.

BOND Revisited (3)

- Notation:
  - query vector q, database vector v
  - Splitting of v: resolved part v^-, unresolved part v^+ $\Rightarrow$ $v = v^- \cup v^+$
- Approximated distance: $S_{approx}(q,v) = S_1(q^-,v^-) + S_2(q^+,v^+)$
  - Resolved part: $S_1(q^-,v^-) = \sum_i (q_i^- - v_i^-)^2$
  - Unresolved part: $S_2(q^+,v^+) = \sum_i \max\left\{q_i^+, 1-q_i^+\right\}^2 \geq S_1(q^+,v^+)$
- Distance bounds:

$$S_{upper}(q,v) = S_1(q^-,v^-) + S_2(q^+,v^+) \geq S_1(q,v)$$

$$S_{lower}(q,v) = S_1(q^-,v^-) + 0 \leq S_1(q,v)$$



Beyond BOND

- Benefits of BeyOND:
  1. Independence of the data distribution. ☺
  2. No restriction to a normalized data space. ☺
  3. No specific resolve order of the dimensions is needed. ☺

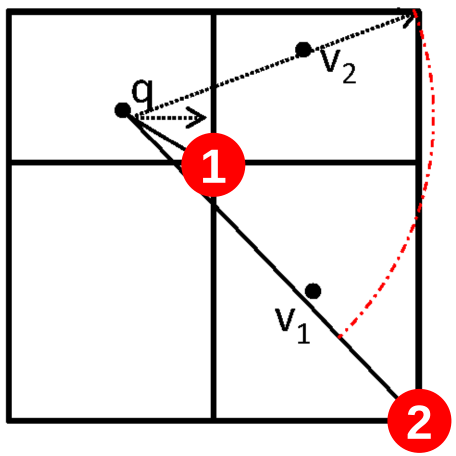
⇒ Price: Distance approximations are no more suitable! ☹

- Solution: Combining the idea of BOND with well-known techniques:
  – VA-file (data space partitioning)
  – MBR (Minimum Bounding Rectangle) approximation (data organizing)

⇒ Remaining restriction: minimum/maximum values for each dimension need to be known ☹

Sub Cubes (1)

- First extension: VA-file[2] with one split
  - ⇒ 2^d sub cubes
  - ⇒ Addressing via Z-IDs
  - ⇒ Improved bounds based on the close / far sub cube borders $c_{v_i}^{lower}$ 1 and $c_{v_i}^{upper}$ 2



- Memory-efficient representation (8 bytes → 1 bit)
  - – Sub cube need not be kept in main memory
- Split positions stored in one separate array per dimension
- Dependence on split level:
  - – FV: 8 bytes per dimension
  - – ***s*** splits: ***s / 8*** bytes (***s*** bits) per dimension

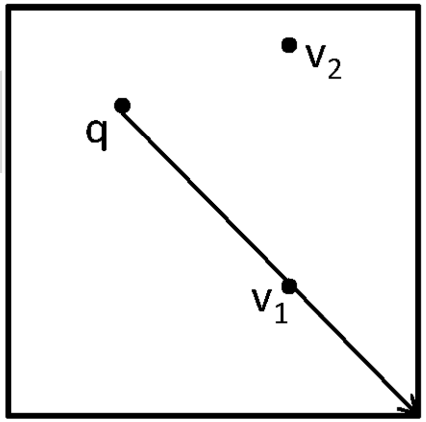
[2] Weber, Schek, Blott. *A Quantitative Analysis and Performance Study for Similarity Search Methods in High-Dimensional Spaces* (VLDB'98)

Sub Cubes (2)

- Old distance bounds:

$$S_{upper}(q,v) = S_1(q^-,v^-) + \sum_i \max\{q_i^+, 1-q_i^+\}^2$$

$$S_{lower}(q,v) = S_1(q^-,v^-) + 0$$



- Approximations of unresolved dimensions:

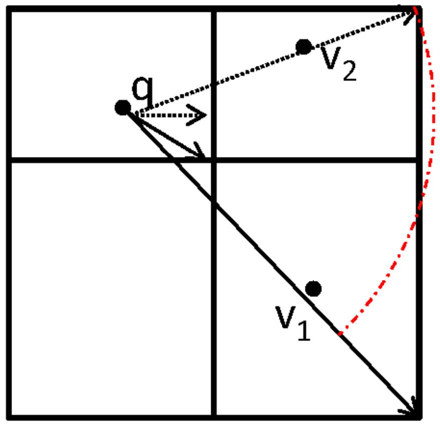
$$S_2'(q^+,v^+) = \sum_i \max\left\{\left|q_i^+ - c_{v_i^+}^{lower}\right|, \left|q_i^+ - c_{v_i^+}^{upper}\right|\right\}^2$$

$$S_2''(q^+,v^+) = \sum_i \begin{cases} 0 & \text{if } q_i^+ \in \left[c_{v_i^+}^{lower}, c_{v_i^+}^{upper}\right] \\ \min\left\{\left|q_i^+ - c_{v_i^+}^{lower}\right|, \left|q_i^+ - c_{v_i^+}^{upper}\right|\right\}^2 & \text{else} \end{cases}$$

- New distance bounds:

$$S'_{upper}(q,v) = S_1(q^-,v^-) + S_2'(q^+,v^+) \geq S_1(q,v)$$

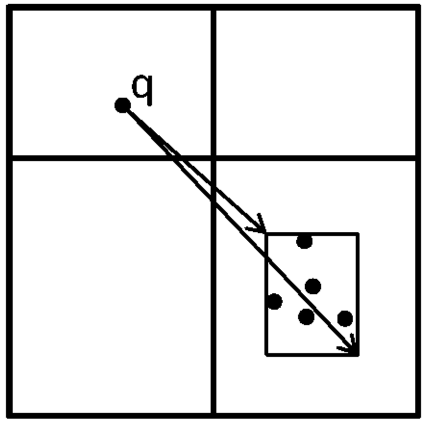
$$S'_{lower}(q,v) = S_1(q^-,v^-) + S_2''(q^+,v^+) \leq S_1(q,v)$$



MBR Caching (1)

- Most sub cubes are (very) sparse, i.e. occupied by at most one FV

- Dense sub cubes allow a tighter approximation via MBRs
  - Restrict the number of MBRs in order to avoid a memory overhead
  - Ranking function for MBRs:



$$f(MBR) = \frac{V_{sub\,cube}}{V_{MBR}} \cdot card(MBR)$$

  - 8 byte coordinates: memory increase is limited to $\frac{d \cdot 16}{card(MBR)}$ bytes per feature vector (+ pointer to Z-ID)

MBR Caching (2)

- Limit the number of MBRs to 1% of the database size
- Threshold as a trade-off between pruning power and additional memory consumption
- Requirements:
  - Either all MBRs can be kept in memory,
  - or the time for loading the MBRs is less than the time for resolving the respective FVs.
- Adaption of the equations for lower and upper bounds

Experimental Evaluation (1)

- Evaluated approaches:

1. BondAdvanced (stricter bounds, but resolve order dependent)
2. Bond (original bounds)*

---

3. Sequential*

---

4. Beyond-1 (1 split)
5. BeyondMBR-1 (1 split + MBRs)
6. Beyond-2
7. BeyondMBR-2
8. Beyond-3*
9. BeyondMBR-3*

Experimental Evaluation (2)

- Data set descriptions:

Data Set	Dims	Size	Type
ALOI	27	110,250	Color Histograms, Zipfian
CLUSTERED	20	500,000	Synthetic, 50 Clusters, Gaussian
PHOG[3]	110	10,715	CT Histograms, PCA‘ed
SIFT[4]	133	335,583	Image Features

[3] Graf, Kriegel, Schubert, Poelsterl, Cavallaro. *2D Image Registration in CT Images Using Radial Image Descriptors* (MICCAI‘11)

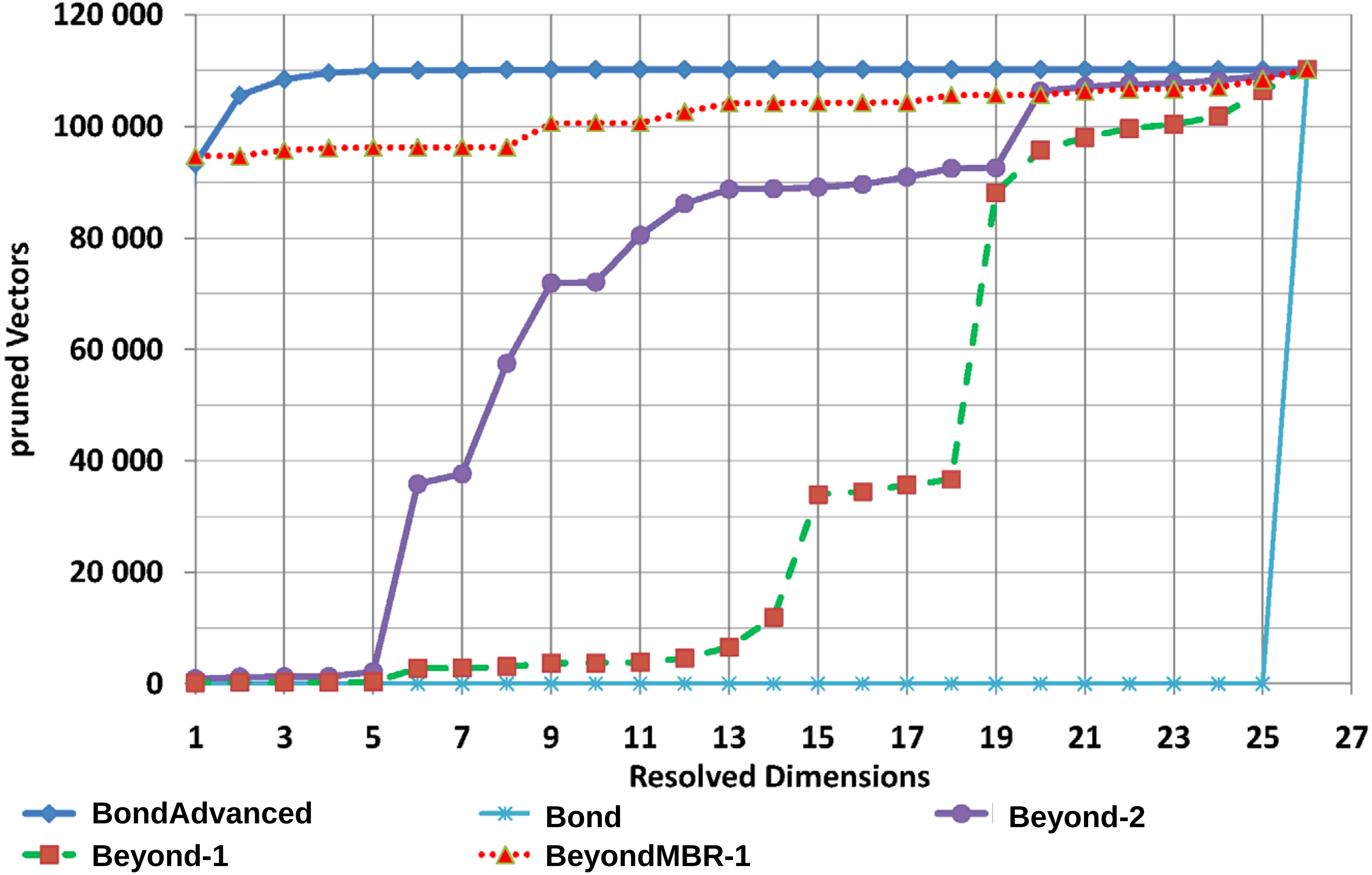
[4] Lowe. *Distinctive Image Features from Scale-Invariant Keypoints* (Int. Journal of Computer Vision, 2004)

Experimental Evaluation (3)

- Experimental settings:
  - 50 k-nearest neighbor queries
  - k = 10
  - Averaged cumulative number of pruned FVs after resolving a column
  - AUC: data not resolved
  - AOC: data resolved for refinement

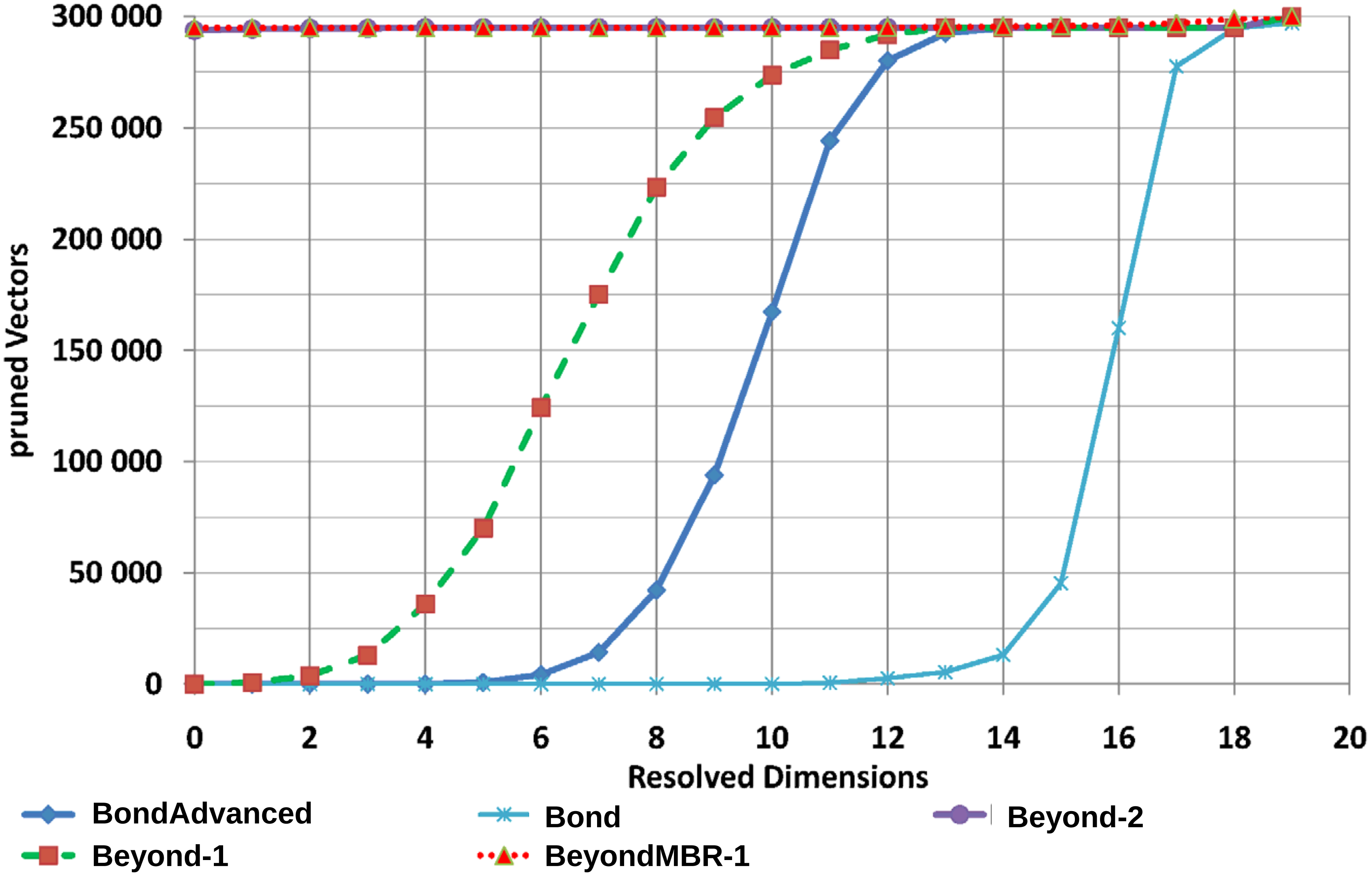
Experimental Evaluation (4)

ALOI	27	110,250	Color Histograms, Zipfian



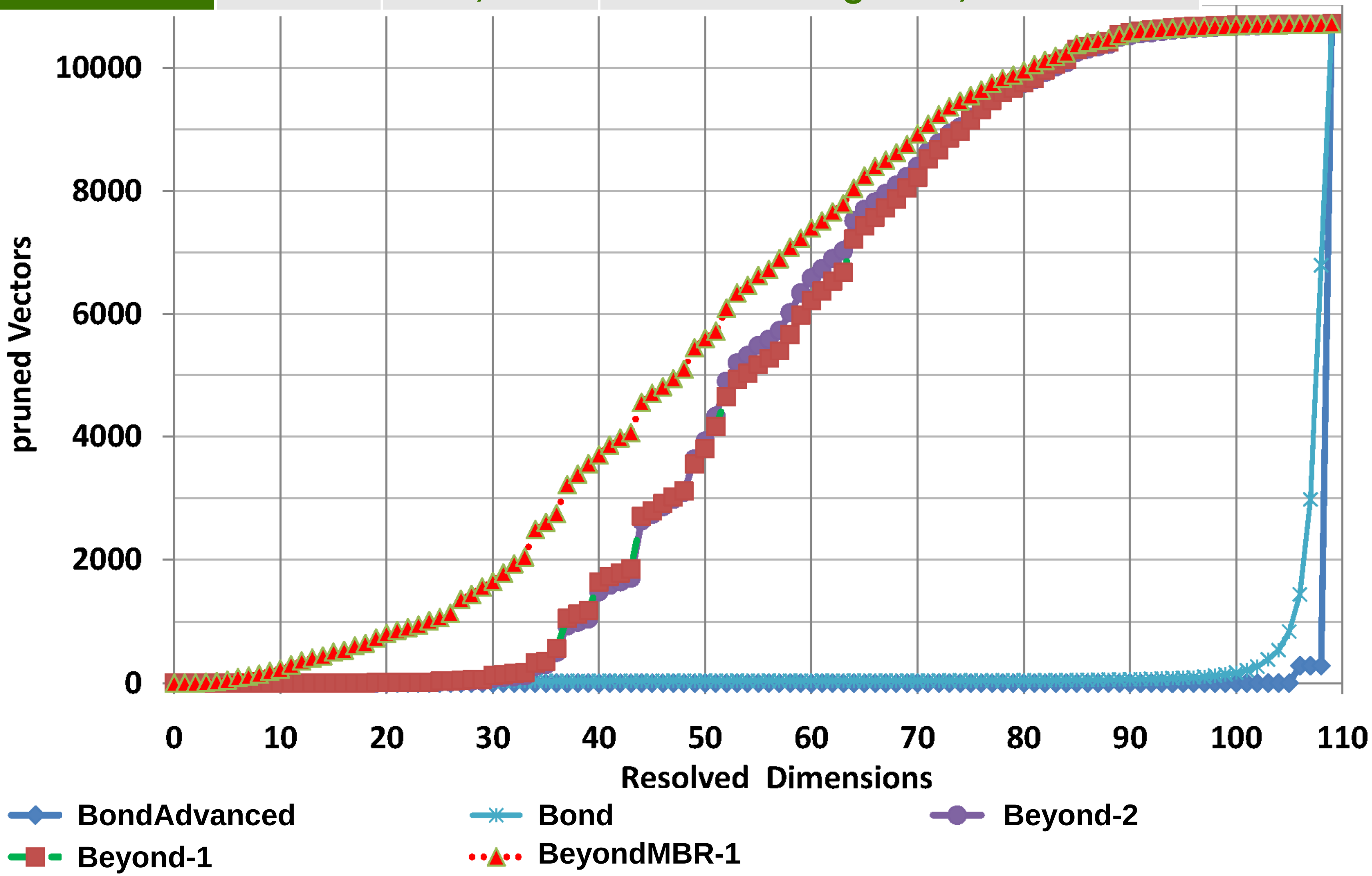
Experimental Evaluation (5)

CLUSTERED	20	500,000	Synthetic, 50 Clusters, Gaussian



Experimental Evaluation (6)

PHOG	110	10,715	CT Histograms, PCA‘ed



Experimental Evaluation (7)

Pruning power (Sub cubes)

Data Set	Splits	25% pruned	50% pruned	90% pruned
ALOI	1	16 (59%)	19 (70%)	23 (85%)
CLUSTERED	1	7 (35%)	8 (40%)	10 (50%)
PHOG	1	45 (41%)	58 (53%)	80 (73%)
ALOI	2	7 (26%)	9 (33%)	21 (75%)
CLUSTERED	2	1 (5%)	1 (5%)	1 (5%)
PHOG	2	45 (41%)	55 (50%)	79 (72%)

Pruning power (Sub cubes + MBRs)

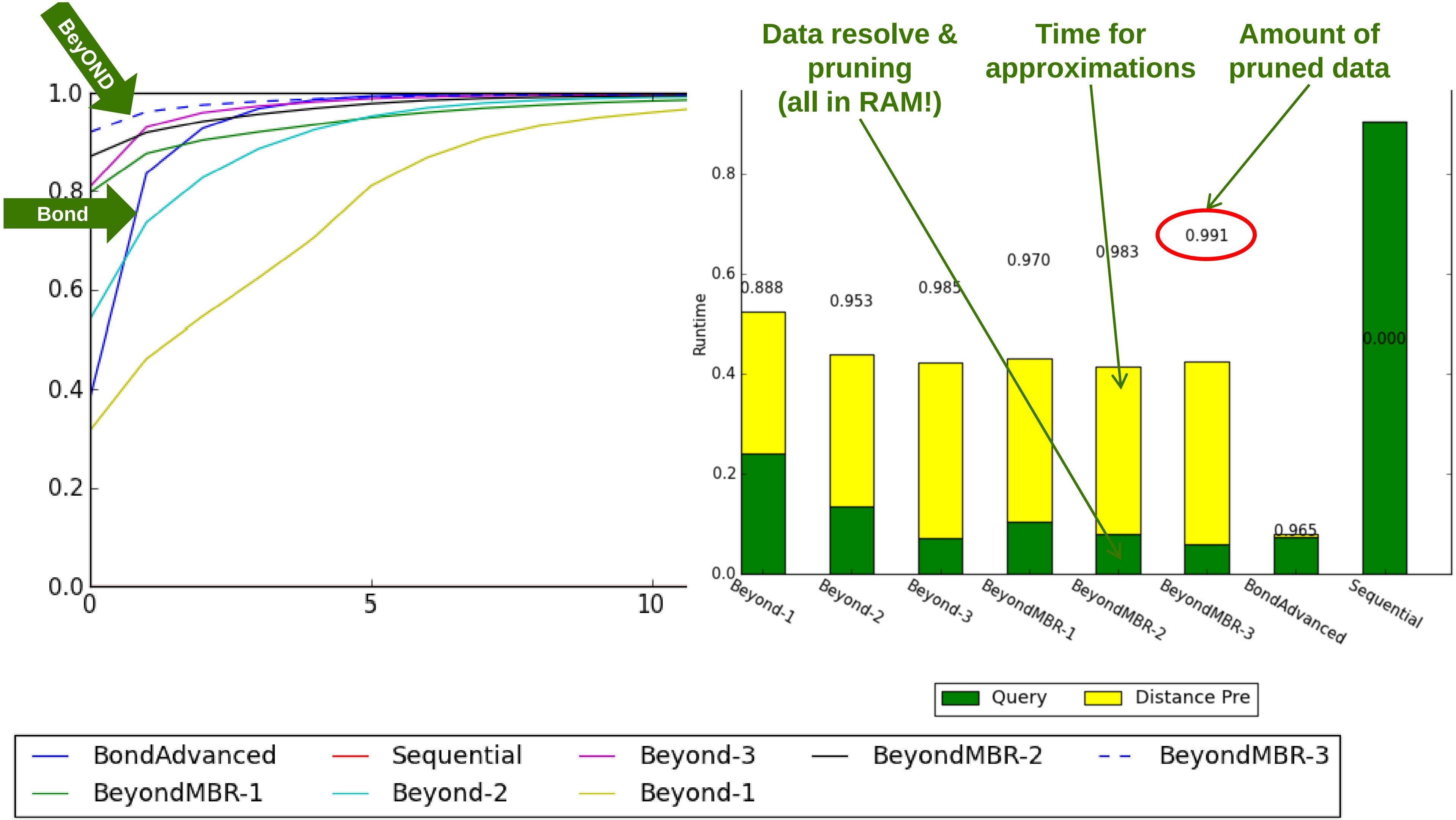
Data Set	Splits	25% pruned	50% pruned	90% pruned
ALOI	1	1 (4%)	1 (4%)	10 (37%)
CLUSTERED	1	1 (5%)	1 (5%)	1 (5%)
PHOG	1	37 (34%)	50 (45%)	77 (70%)

 # Accessed columns

Data Set	1 split	2 splits	1 split + MBR
ALOI	66.9%	38.3%	7.7%
CLUSTERED	34.1%	1.6%	1.4%
PHOG	52.6%	52.3%	45.4%

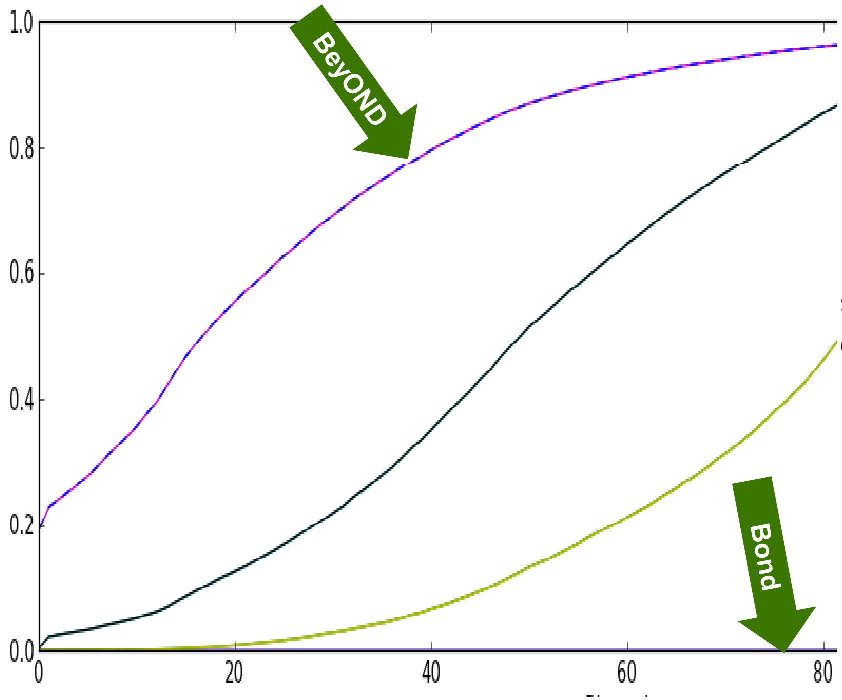
Experimental Evaluation (8)

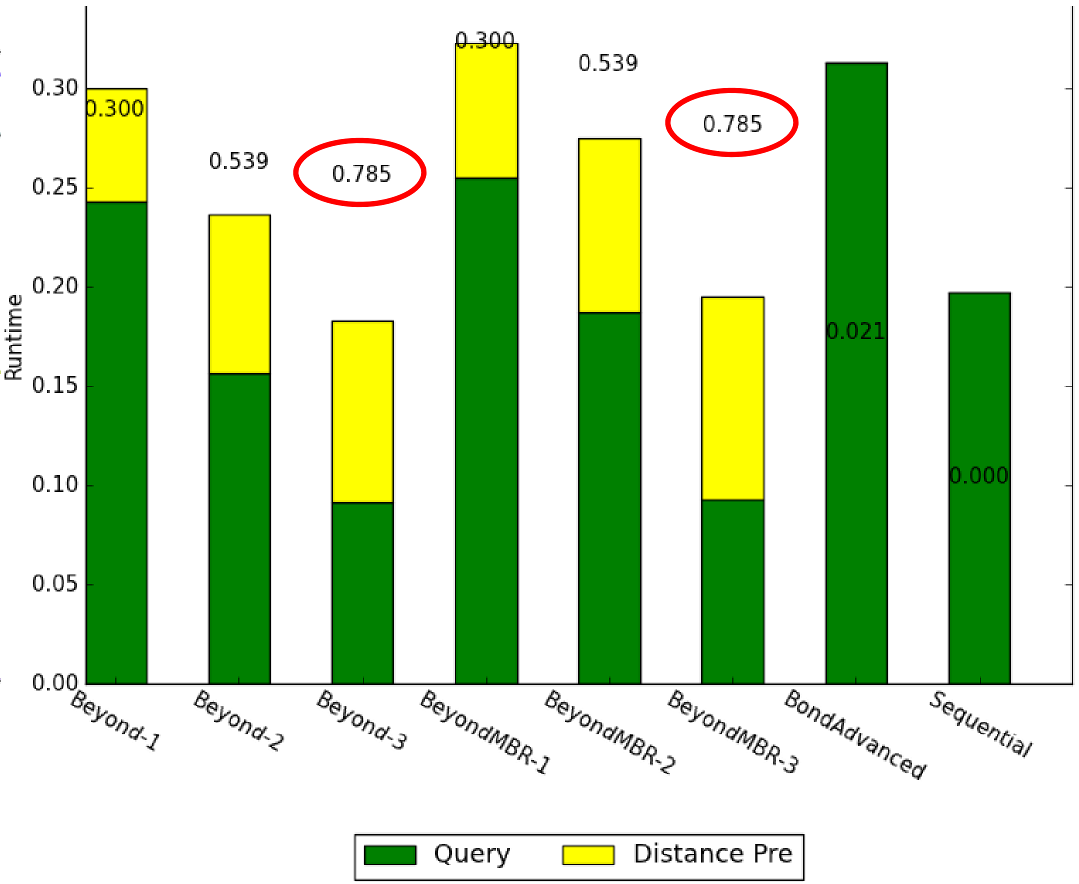
ALOI	27	110,250	Color Histograms, Zipfian

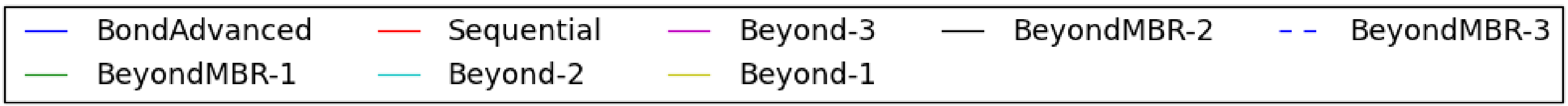


Experimental Evaluation (9)

PHOG	110	10,715	CT Histograms, PCA'ed





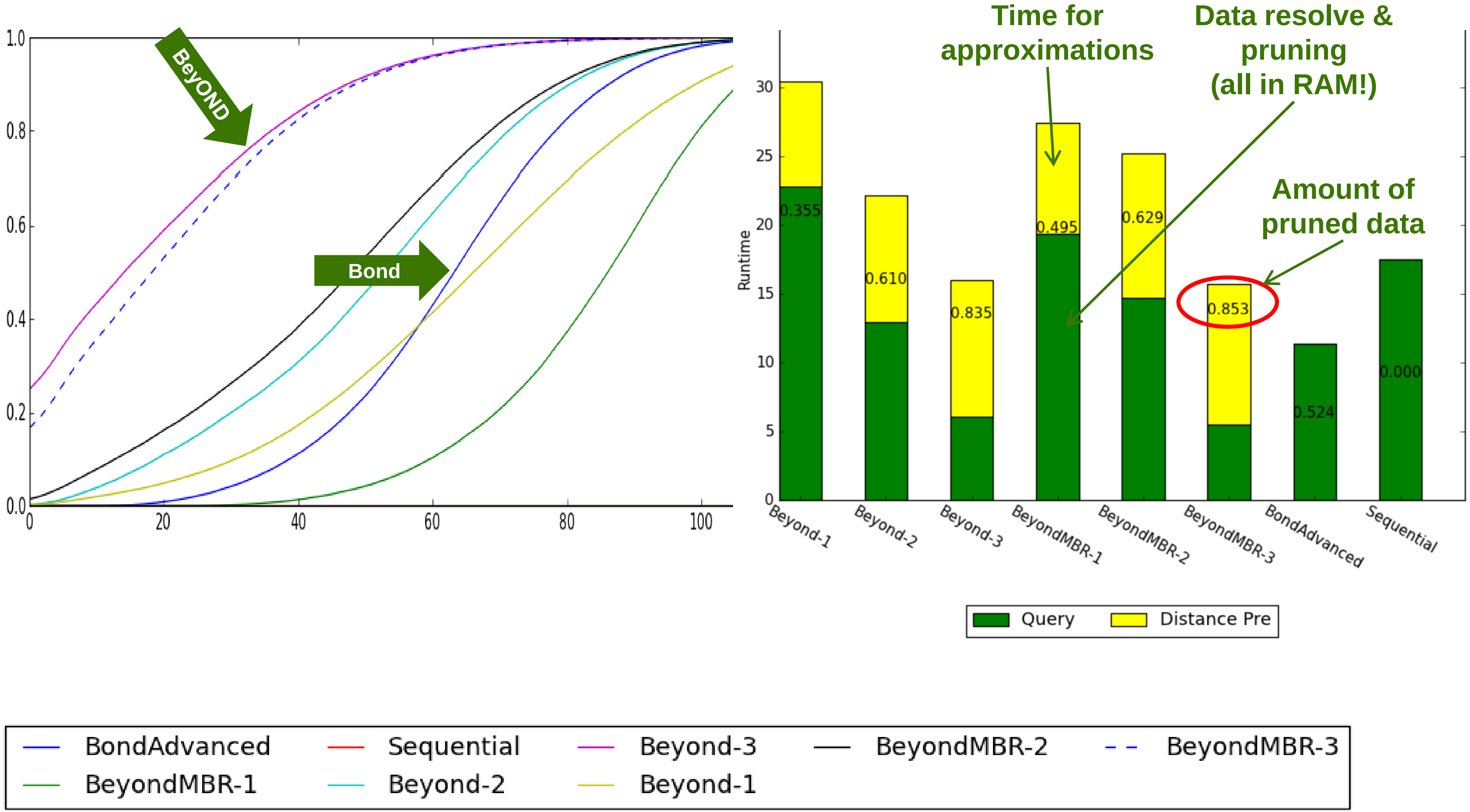


DATABASE SYSTEMS GROUP

Experimental Evaluation (10)

LMU

SIFT	133	335,583	Image Features



Conclusions

- Removed restrictions…
  1. Independence of the data distribution.
  2. No restriction to a normalized data space.
  3. No specific resolve order of the dimensions is needed.

- Combination of relevant techniques…
  – VA-file-based partitioning of the data space
  – MBR caching

- Still open issues…
  – Trade-off: split level vs. pruning power
  – Trade-off: MBR memory consumption vs. pruning power
  – Sophisticated techniques for the creation of the MBRs
  – Overcome the restriction that the vector lengths have to be known

Thank you for listening!

Any questions?

http://www.dbs.ifi.lmu.de/cms/Publications/BeyOND_-_Unleashing_BOND