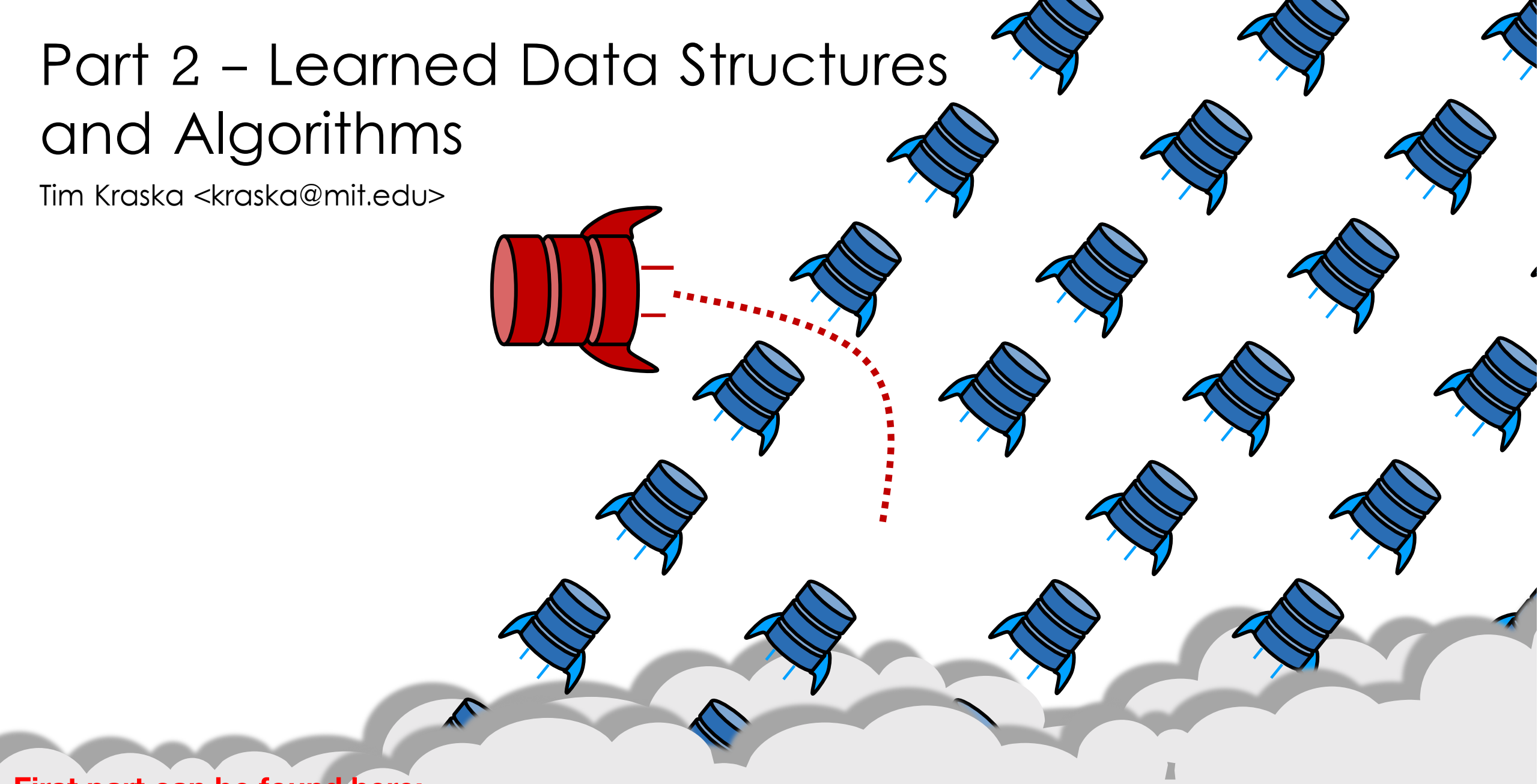
Part 2 – Learned Data Structures and Algorithms

Tim Kraska <kraska@mit.edu>



First part can be found here:
https://stratos.seas.harvard.edu/files/stratos/files/learnedselfdesignedsystemssigmod2019tutorialpart1.pdf

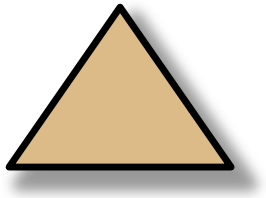
4.5 Ways to Instance Optimized Algorithms and Data Structures

1. **Configure/synthesize** traditional algorithm using a model

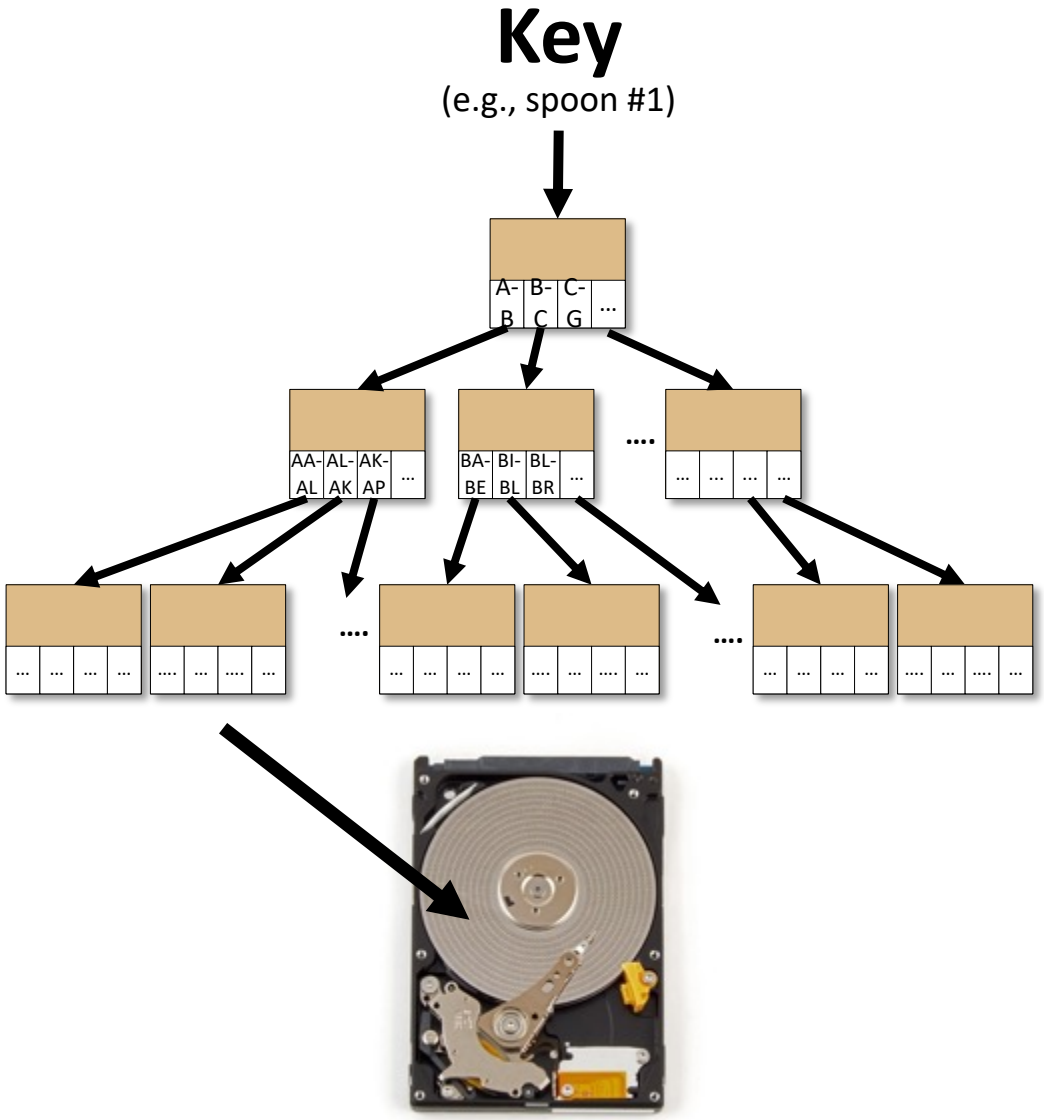
Fundamental Building Blocks

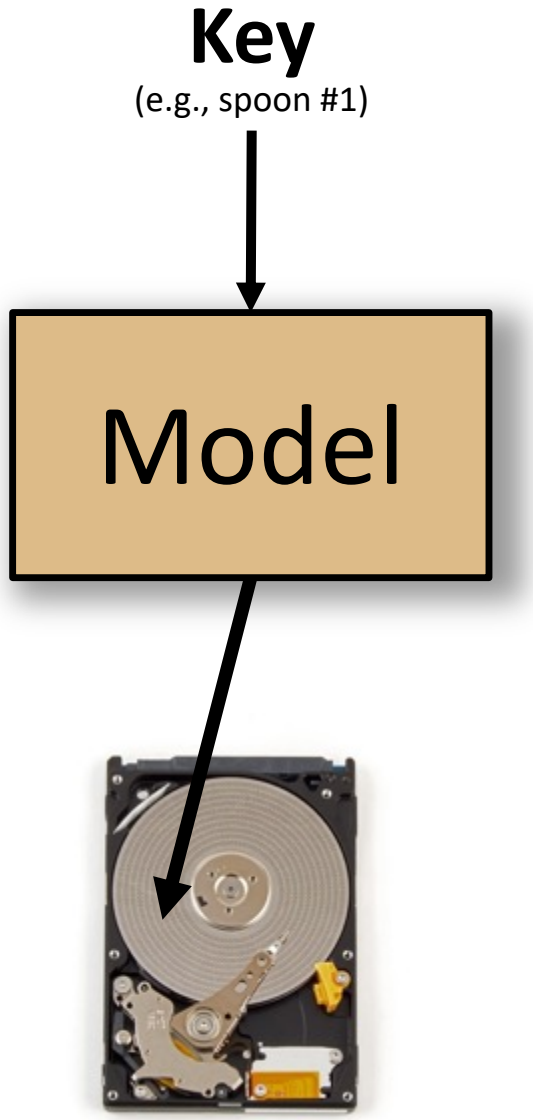
Fundamental Algorithms & Data Structures

B-Tree



Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, Neoklis Polyzotis:
The Case for Learned Index Structures. SIGMOD Conference 2018: 489-504


Key
(e.g., spoon #1)
A-B
B-C
C-G
...
AA-AL
AL-AK
AK-AP
BA-BE
BI-BL
BL-BR


Key
(e.g., spoon #1)
Model

How To Build Models To Predict the Location

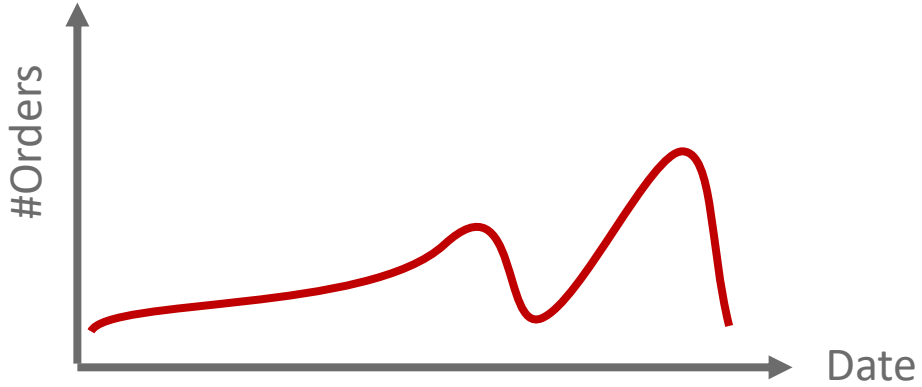
id	date	first_name	last_name	email	address	zip	state	credit_card_nb	amount
1000	2017-01-01	Hobart	Spracklin	hspracklin0@dailymotion.com	20565 High Crossing Plaza	56372	Minnesota	4405-6975-7285-5160	$ 611.00
1001	2017-01-02	Billye	Binnion	bbinnion1@123-reg.co.uk	3698 Upham Point	20260	District of Columbia	3533-7150-7728-9850	$ 244.00
1002	2017-01-02	Johann	Brockley	jbrockley2@bizjournals.com	23844 Artisan Place	98516	Washington	67597-1193-7985-5100	$ 233.00
1003	2017-01-03	Artie	MacMenami	amacmenamin3@hao123.com	6276 Toban Trail	78759	Texas	3537-4829-6134-5000	$ 210.00
1004	2017-01-03	Delilah	O'Currigan	docurrigan4@chron.com	86016 New Castle Avenue	72199	Arkansas	3555-2017-2226-5780	$ 286.00
1005	2017-01-04	Gretta	Will	gwill5@yelp.com	0 Dottie Circle	68524	Nebraska	503844-1984-2085-5000	$ 870.00
1006	2017-01-04	Gordon	Kirsopp	gkirsopp6@utexas.edu	64060 Scott Park	20370	District of Columbia	633332-1895-2414-5000	$ 687.00
1007	2017-01-05	Bendick	Fagg	bfagg7@army.mil	94 Florence Hill	45440	Ohio	3528-9673-1815-8420	$ 733.00
1008	2017-01-05	Dimitry	Boyet	dboyet8@sakura.ne.jp	35886 Golf Plaza	30066	Georgia	3576-6991-4041-3170	$ 382.00
1009	2017-01-06	Ailsun	Beinke	abeinke9@si.edu	1 Badeau Place	46295	Indiana	56022-2011-8072-1400	$ 854.00
1010	2017-01-07	Lou	Hallows	lhallowsa@theguardian.com	1 Twin Pines Junction	91125	California	5602-2364-4079-0250	$ 150.00
1011	2017-01-09	Tiffani	Mathew	tmathewb@seattletimes.com	0456 Meadow Vale Lane	75260	Texas	6387-6943-8910-4580	$ 313.00
1012	2017-01-09	Perl	Bridie	pbridiec@hubpages.com	07 Bluestem Junction	33124	Florida	3539-8662-2397-5880	$ 558.00
1013	2017-01-09	Rosabelle	Blasik	rblasikd@delicious.com	7 Fairfield Pass	79699	Texas	5602-2297-6599-8560	$ 941.00
1014	2017-01-10	Meggi	Belamy	mbelamye@ask.com	0995 Manufacturers Street	10170	New York	3557-5094-7405-8340	$ 875.00
1015	2017-01-10	Tadio	Balderston	tbalderstonf@apache.org	80 Novick Road	75260	Texas	60485-3728-7119-9300	$ 954.00
1016	2017-01-11	Gianina	Oxteby	goxtebyg@google.pl	72674 Fuller Avenue	89505	Nevada	4-0415-9268-2397	$ 239.00
1017	2017-01-12	Brendan	Doody	bdoodyh@craigslist.org	87414 Golden Leaf Street	11480	New York	201-6348-4121-1314	$ 308.00
1018	2017-01-13	Conway	Coombs	ccoombsi@blogger.com	2810 Oakridge Park	32859	Florida	3529-1514-0357-9120	$ 60.00
1019	2017-01-14	Germaine	Bere	gberej@bravesites.com	82802 Oakridge Park	20041	District of Columbia	670961-0240-4054-9000	$ 95.00
1020	2017-01-15	Davide	Tolcharde	dtolchardek@redcross.org	89 Continental Avenue	79165	Texas	5018-7748-4325-9510	$ 137.00
1021	2017-01-16	Nigel	Artharg	narthargl@gizmodo.com	31 Mcbride Point	22301	Virginia	560225-6965-2870-0000	$ 496.00
1022	2017-01-17	Rickard	Trenholm	rtrenholmm@cbslocal.com	93 Hoepker Parkway	70593	Louisiana	3541-5241-5383-9970	$ 760.00
1023	2017-01-18	Juditha	Dwane	jdwanen@vk.com	7914 Eliot Lane	14276	New York	5456-4410-0914-3180	$ 474.00
1024	2017-01-19	Susan	Ilden	sildeno@aol.com	25204 Huxley Road	21684	Maryland	3574-8586-6367-9920	$ 83.00
1025	2017-01-20	Abbey	Triggle	atrigglep@google.com.au	47 Debra Pass	74184	Oklahoma	3538-6047-6315-7710	$ 513.00
1026	2017-01-21	Zsazsa	Dunster	zdunsterq@nature.com	7 Gerald Alley	40576	Kentucky	3562-0325-7709-3490	$ 952.00
1027	2017-01-22	Grantham	Friatt	gfriattr@seattletimes.com	774 Prairieview Circle	29225	South Carolina	3571-1171-9476-8780	$ 942.00
1028	2017-01-22	Ross	Gaudin	rgaudins@samsung.com	3102 Loeprich Trail	68197	Nebraska	5108-7578-4665-2710	$ 572.00
1029	2017-01-22	Aluino	Drover	adrovert@dagondesign.com	2717 Northridge Avenue	72199	Arkansas	670999-3171-8848-0000	$ 318.00
1030	2017-01-23	Shurlock	Braker	sbrakeru@huffingtonpost.com	30783 Jenna Alley	80945	Colorado	6331106-1894-9878-0000	$ 166.00
1031	2017-01-24	Glenda	Goodbody	ggoodbodyv@economist.com	720 Pierstorff Way	7522	New Jersey	36-0593-2719-1684	$ 412.00
1032	2017-01-24	Rollin	Reddie	rreddiew@tinypic.com	09 Gina Park	65810	Missouri	4665-9188-1324-1040	$ 383.00
1033	2017-01-26	Dorry	Jenks	djenksx@virginia.edu	1 Butterfield Road	85210	Arizona	3578-9195-0297-7730	$ 636.00
1034	2017-01-26	Patti	Emby	pembyy@weather.com	26 Hoard Drive	91210	California	3585-8243-7506-2470	$ 957.00

How To Build Models To Predict the Location

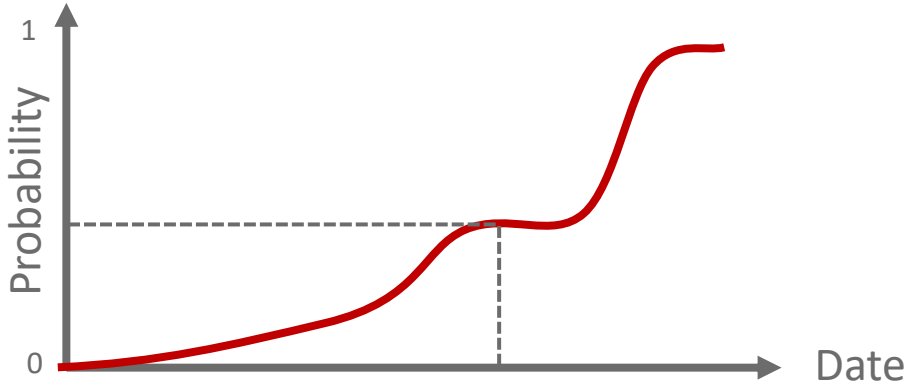
date
2017-01-01
2017-01-02
2017-01-02
2017-01-03
2017-01-03
2017-01-04
2017-01-04
2017-01-05
2017-01-05
2017-01-06
2017-01-07
2017-01-09
2017-01-09
2017-01-09
2017-01-10
2017-01-10
2017-01-11
2017-01-12
2017-01-13
2017-01-14
2017-01-15
2017-01-16
2017-01-17
2017-01-18
2017-01-19
2017-01-20
2017-01-21
2017-01-22
2017-01-22
2017-01-22
2017-01-23
2017-01-24
2017-01-24
2017-01-26
2017-01-26

How To Build Models To Predict the Location

Frequency Distribution



Cumulative Distribution Function (CDF)



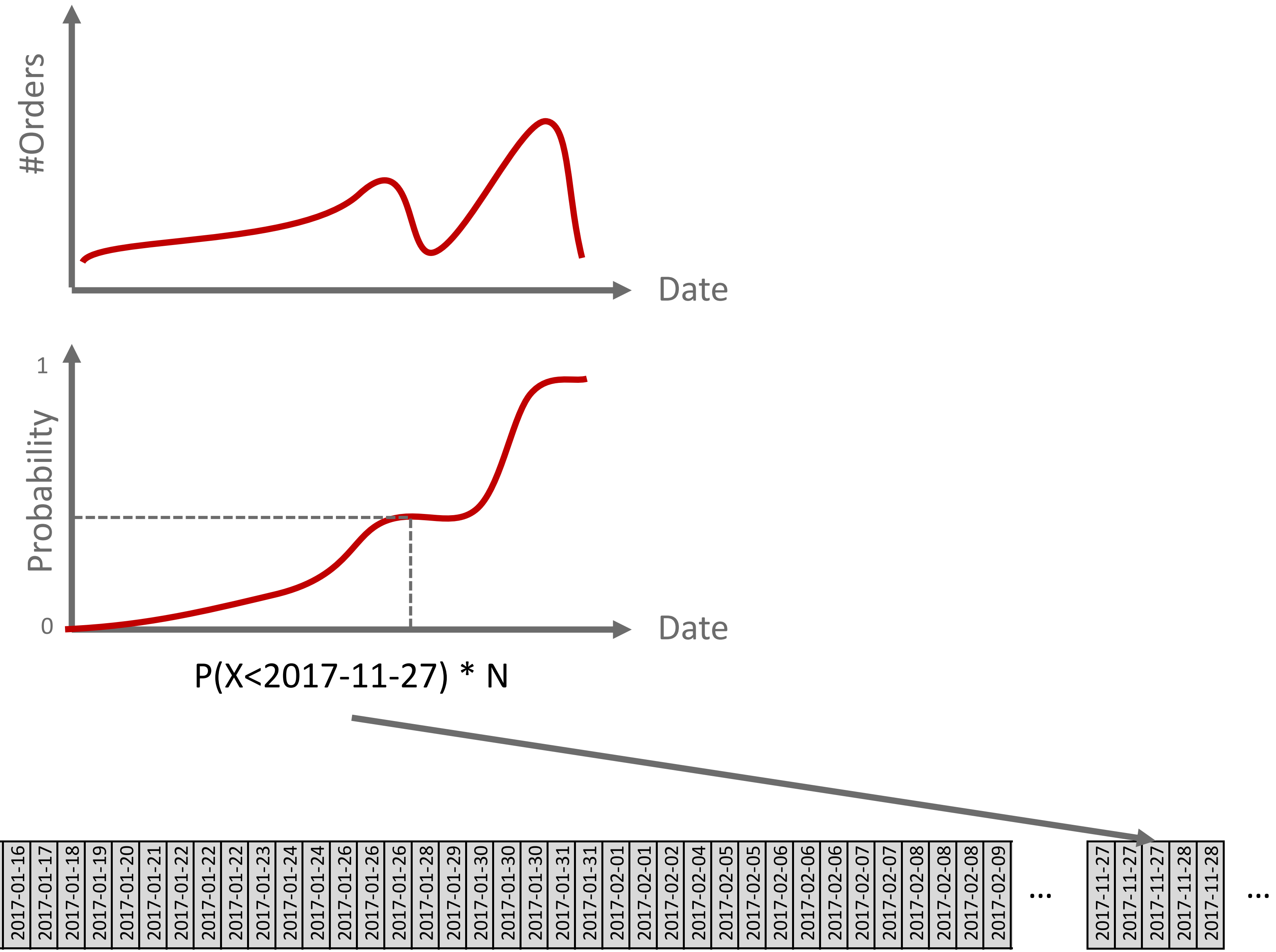
P(X<2017-11-27)

date
2017-01-01
2017-01-02
2017-01-02
2017-01-03
2017-01-03
2017-01-04
2017-01-04
2017-01-05
2017-01-05
2017-01-06
2017-01-07
2017-01-09
2017-01-09
2017-01-09
2017-01-10
2017-01-10
2017-01-11
2017-01-12
2017-01-13
2017-01-14
2017-01-15
2017-01-16
2017-01-17
2017-01-18
2017-01-19
2017-01-20
2017-01-21
2017-01-22
2017-01-22
2017-01-22
2017-01-23
2017-01-24
2017-01-24
2017-01-26
2017-01-26
2017-01-26
2017-01-28
2017-01-29
2017-01-30
2017-01-30
2017-01-30
2017-01-31
2017-01-31
2017-02-01
2017-02-01
2017-02-02
2017-02-04
2017-02-05
2017-02-05
2017-02-06
2017-02-06
2017-02-06
2017-02-07
2017-02-07
2017-02-08
2017-02-08
2017-02-08
2017-02-09
2017-02-10
2017-02-10
2017-02-11
2017-02-12
2017-02-13
2017-02-13
2017-02-14
2017-02-14
2017-02-15

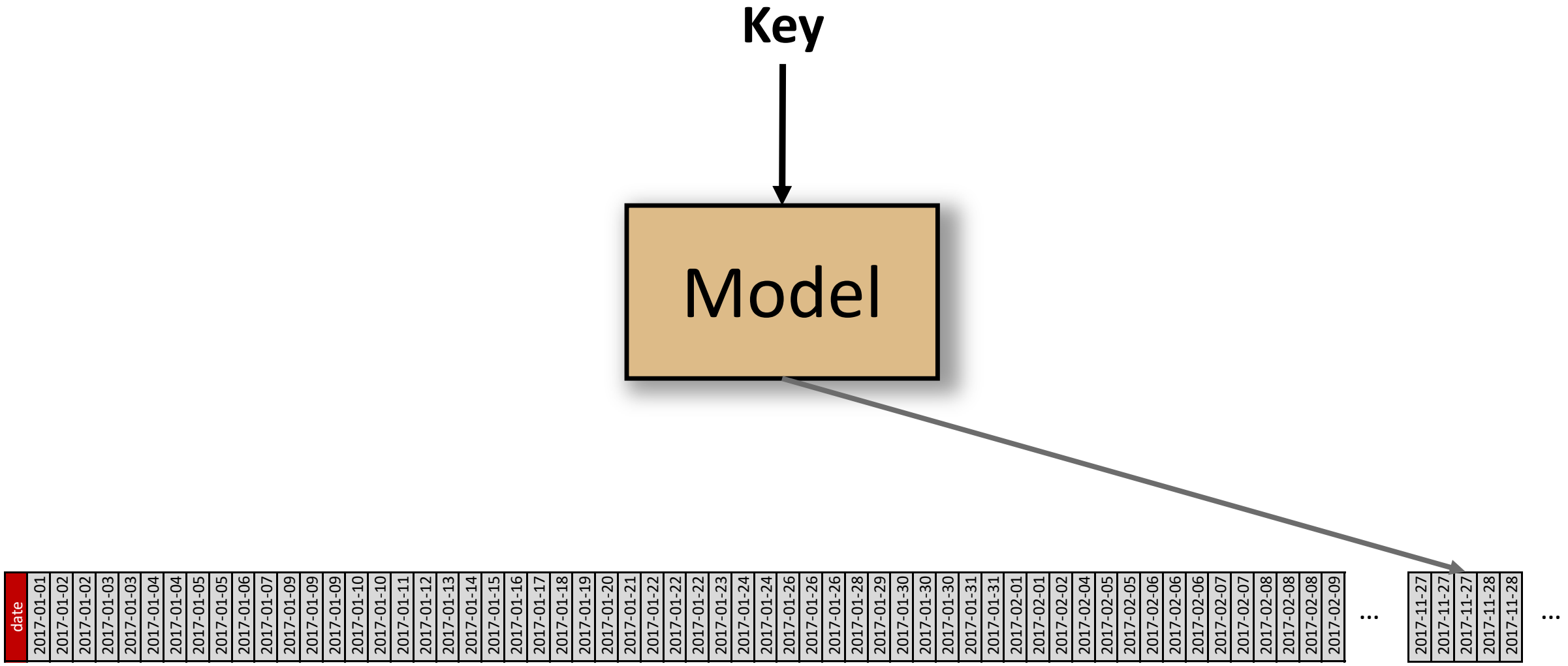
How To Build Models To Predict the Location

Frequency Distribution

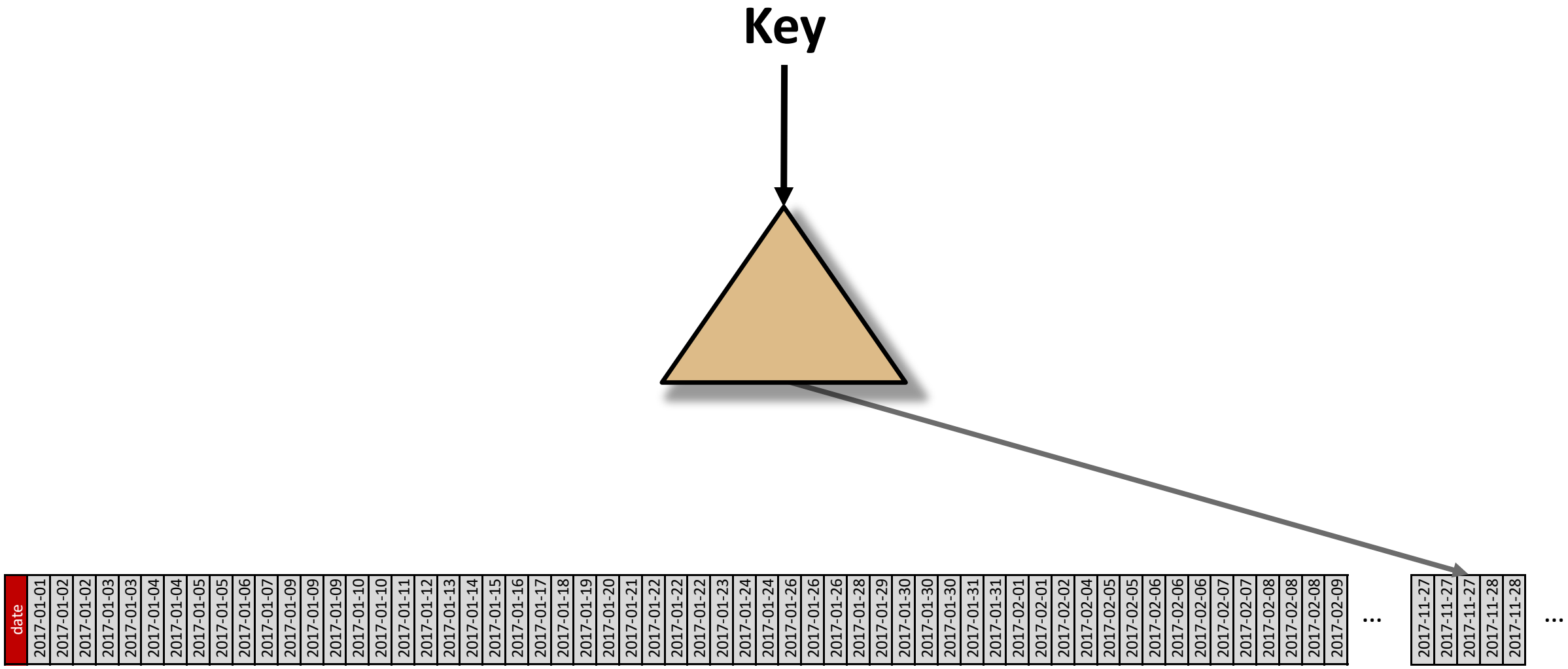
Cumulative Distribution Function (CDF)



CDF Model

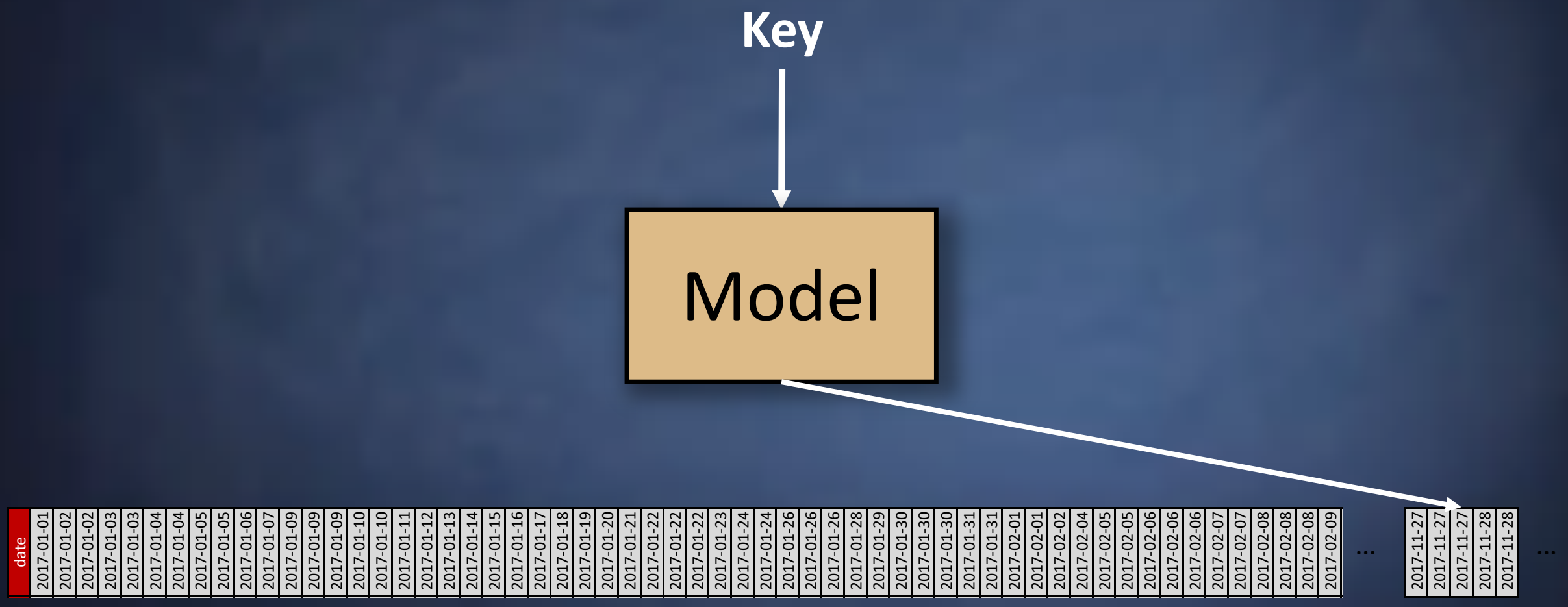

Key
Model
date
2017-01-01
2017-01-02
2017-01-02
2017-01-03
2017-01-03
2017-01-04
2017-01-04
2017-01-05
2017-01-05
2017-01-06
2017-01-07
2017-01-09
2017-01-09
2017-01-09
2017-01-10
2017-01-10
2017-01-11
2017-01-12
2017-01-13
2017-01-14
2017-01-15
2017-01-16
2017-01-17
2017-01-18
2017-01-19
2017-01-20
2017-01-21
2017-01-22
2017-01-22
2017-01-22
2017-01-23
2017-01-24
2017-01-24
2017-01-26
2017-01-26
2017-01-26
2017-01-28
2017-01-29
2017-01-30
2017-01-30
2017-01-30
2017-01-31
2017-01-31
2017-02-01
2017-02-01
2017-02-02
2017-02-04
2017-02-05
2017-02-05
2017-02-06
2017-02-06
2017-02-06
2017-02-07
2017-02-07
2017-02-08
2017-02-08
2017-02-08
2017-02-09
...
2017-11-27
2017-11-27
2017-11-27
2017-11-28
2017-11-28
...

B–Trees are also models (regression tree)



What Does This Mean

Why Is This A Big Deal?

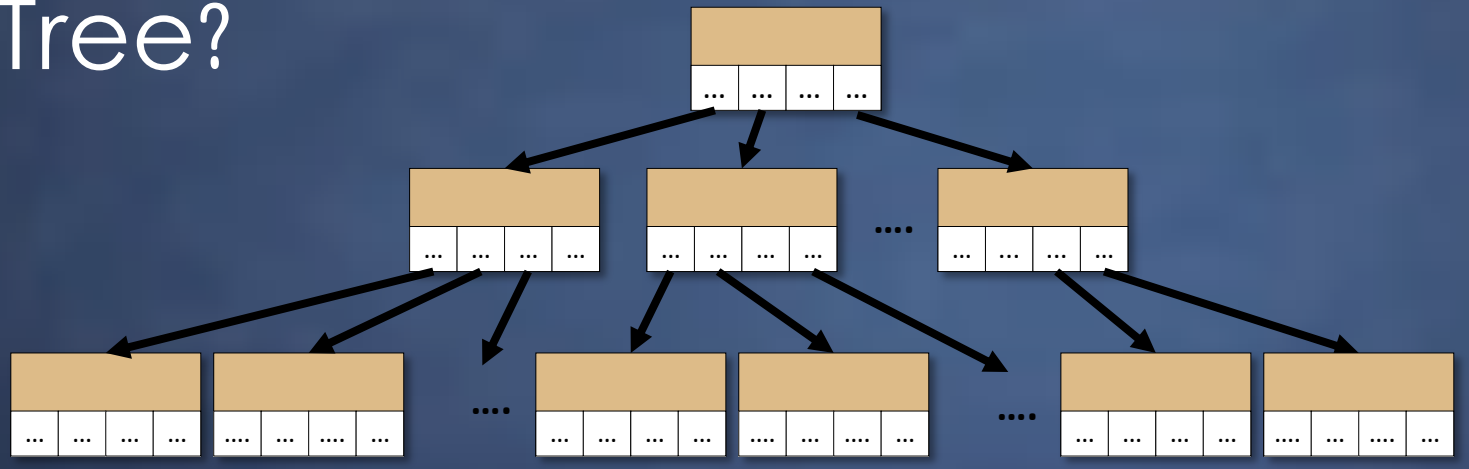


Adaptation To Application Data

id	date	first_name	last_name	email	address	zip	state	credit_card_nb	amount
1000	2017-01-01	Hobart	Spracklin	hspracklin0@dailymotion.com	20565 High Crossing Plaza	56372	Minnesota	4405-6975-7285-5160	$ 611.00
1001	2017-01-02	Billye	Binnion	bbinnion1@123-reg.co.uk	3698 Upham Point	20260	District of Columbia	3533-7150-7728-9850	$ 244.00
1002	2017-01-02	Johann	Brockley	jbrockley2@bizjournals.com	23844 Artisan Place	98516	Washington	67597-1193-7985-5100	$ 233.00
1003	2017-01-03	Artie	MacMenami	amacmenamin3@hao123.com	6276 Toban Trail	78759	Texas	3537-4829-6134-5000	$ 210.00
1004	2017-01-03	Delilah	O'Currigan	docurrigan4@chron.com	86016 New Castle Avenue	72199	Arkansas	3555-2017-2226-5780	$ 286.00
1005	2017-01-04	Gretta	Will	gwill5@yelp.com	0 Dottie Circle	68524	Nebraska	503844-1984-2085-5000	$ 870.00
1006	2017-01-04	Gordon	Kirsopp	gkirsopp6@utexas.edu	64060 Scott Park	20370	District of Columbia	633332-1895-2414-5000	$ 687.00
1007	2017-01-05	Bendick	Fagg	bfagg7@army.mil	94 Florence Hill	45440	Ohio	3528-9673-1815-8420	$ 733.00
1008	2017-01-05	Dimitry	Boyet	dboyet8@sakura.ne.jp	35886 Golf Plaza	30066	Georgia	3576-6991-4041-3170	$ 382.00
1009	2017-01-06	Ailsun	Beinke	abeinke9@si.edu	1 Badeau Place	46295	Indiana	56022-2011-8072-1400	$ 854.00
1010	2017-01-07	Lou	Hallows	lhallowsa@theguardian.com	1 Twin Pines Junction	91125	California	5602-2364-4079-0250	$ 150.00
1011	2017-01-09	Tiffani	Mathew	tmathewb@seattletimes.com	0456 Meadow Vale Lane	75260	Texas	6387-6943-8910-4580	$ 313.00
1012	2017-01-09	Perl	Bridie	pbridiec@hubpages.com	07 Bluestem Junction	33124	Florida	3539-8662-2397-5880	$ 558.00
1013	2017-01-09	Rosabelle	Blasik	rblasikd@delicious.com	7 Fairfield Pass	79699	Texas	5602-2297-6599-8560	$ 941.00
1014	2017-01-10	Meggi	Belamy	mbelamye@ask.com	0995 Manufacturers Street	10170	New York	3557-5094-7405-8340	$ 875.00
1015	2017-01-10	Tadio	Balderston	tbalderstonf@apache.org	80 Novick Road	75260	Texas	60485-3728-7119-9300	$ 954.00
1016	2017-01-11	Gianina	Oxteby	goxtebyg@google.pl	72674 Fuller Avenue	89505	Nevada	4-0415-9268-2397	$ 239.00
1017	2017-01-12	Brendan	Doody	bdoodyh@craigslist.org	87414 Golden Leaf Street	11480	New York	201-6348-4121-1314	$ 308.00
1018	2017-01-13	Conway	Coombs	ccoombsi@blogger.com	2810 Oakridge Park	32859	Florida	3529-1514-0357-9120	$ 60.00
1019	2017-01-14	Germaine	Bere	gberej@bravesites.com	82802 Oakridge Park	20041	District of Columbia	670961-0240-4054-9000	$ 95.00
1020	2017-01-15	Davide	Tolcharde	dtolchardek@redcross.org	89 Continental Avenue	79165	Texas	5018-7748-4325-9510	$ 137.00
1021	2017-01-16	Nigel	Artharg	narthargl@gizmodo.com	31 Mcbride Point	22301	Virginia	560225-6965-2870-0000	$ 496.00
1022	2017-01-17	Rickard	Trenholm	rtrenholmm@cbslocal.com	93 Hoepker Parkway	70593	Louisiana	3541-5241-5383-9970	$ 760.00
1023	2017-01-18	Juditha	Dwane	jdwanen@vk.com	7914 Eliot Lane	14276	New York	5456-4410-0914-3180	$ 474.00
1024	2017-01-19	Susan	Ilden	sildeno@aol.com	25204 Huxley Road	21684	Maryland	3574-8586-6367-9920	$ 83.00
1025	2017-01-20	Abbey	Triggle	atrigglep@google.com.au	47 Debra Pass	74184	Oklahoma	3538-6047-6315-7710	$ 513.00
1026	2017-01-21	Zsazsa	Dunster	zdunsterq@nature.com	7 Gerald Alley	40576	Kentucky	3562-0325-7709-3490	$ 952.00

Adaptation To Application Data

B-Tree?



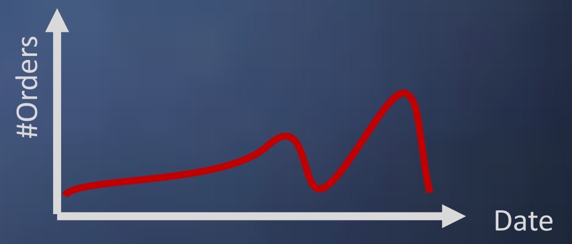
Adaptation To Application Data

id	1000	1001	1002	1003	1004	1005	1006	1007	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	…

```
data_array[id - 1000]
```

date	2017-01-01	2017-01-02	2017-01-02	2017-01-03	2017-01-03	2017-01-04	2017-01-04	2017-01-05	2017-01-05	2017-01-06	2017-01-07	2017-01-09	2017-01-09	2017-01-09	2017-01-10	2017-01-10	2017-01-11	2017-01-12	2017-01-13	2017-01-14	2017-01-15	2017-01-16	2017-01-17	2017-01-18	2017-01-19	2017-01-20	2017-01-21	2017-01-22	2017-01-22	2017-01-22	2017-01-23	2017-01-24	2017-01-24	2017-01-26	2017-01-26	2017-01-26	2017-01-28	2017-01-29	2017-01-30	2017-01-30	2017-01-30	2017-01-31	2017-01-31	2017-02-01	2017-02-01	2017-02-02	2017-02-04	2017-02-05	2017-02-05	2017-02-06	2017-02-06	2017-02-06	2017-02-07	2017-02-07	2017-02-08	2017-02-08	2017-02-08	2017-02-09	2017-02-10	2017-02-10	2017-02-11	2017-02-12	2017-02-13	2017-02-13	2017-02-14	2017-02-14	2017-02-15	…

```
data_array[model(date)]
```



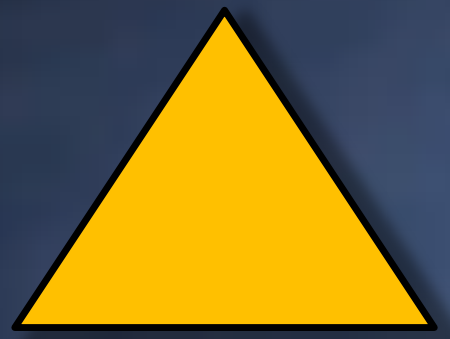
Does It Work? A First Attempt

State–Of–The–Art
B–Tree

260ns

???

Does It Work? A First Attempt



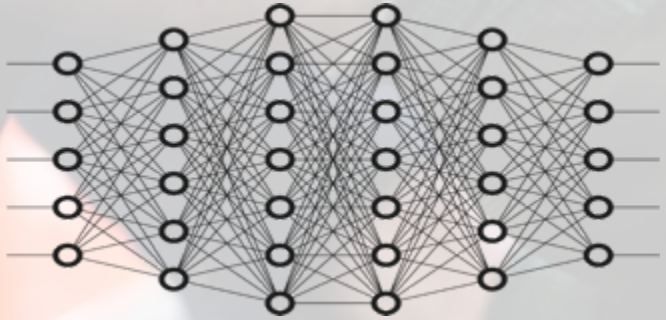
State–Of–The–Art
B–Tree

260ns

>80,000ns

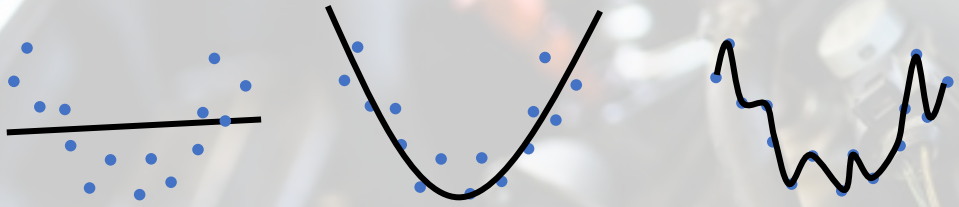
Challenges

Traditional model architectures do not work



Frameworks are not designed for nano-second execution

Overfitting can be good



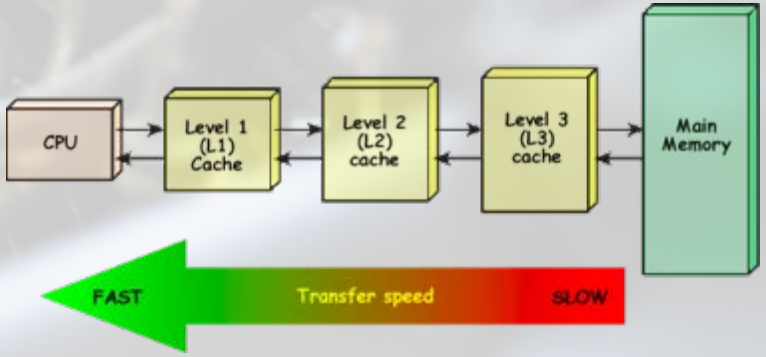
underfitting

desired

~~overfitting~~

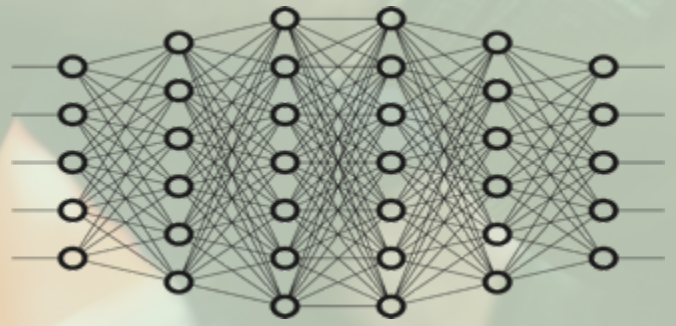
desired

ML+System Co-Design



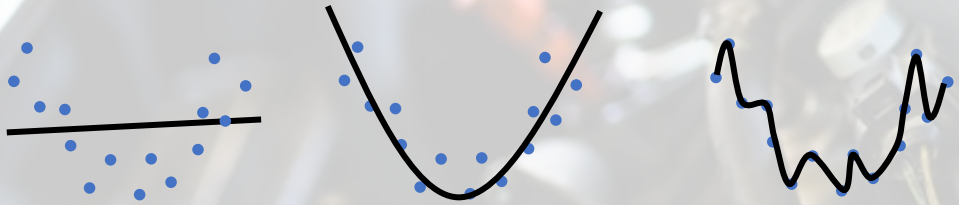
Challenges

Traditional model architectures do not work



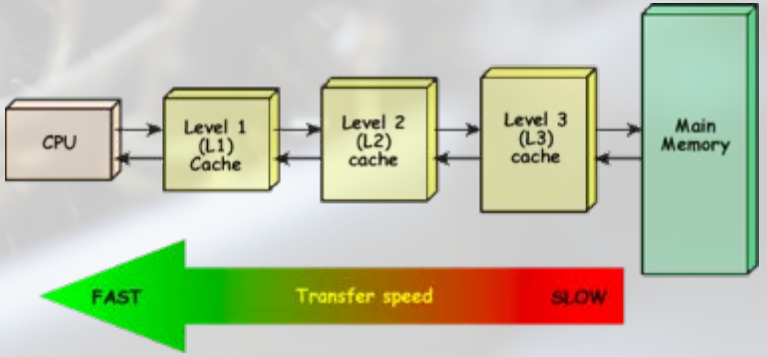
Frameworks are not designed for nano-second execution

Overfitting can be good



underfitting desired ~~overfitting~~ desired

ML+System Co-Design



What model type should I use?

What model type should I use?

Whatever works!

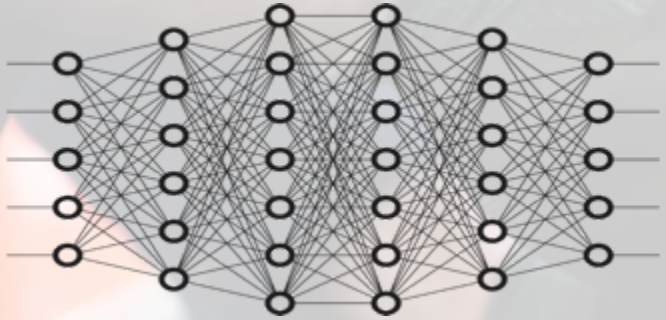
What model type should I use?

Whatever works!

- Often continuous functions
- Possible model types include neural nets, regression models, piece–wise linear functions, among other things
- Model–type can be auto–tuned
- Opens up a complete new toolbox of building data structures and algorithms

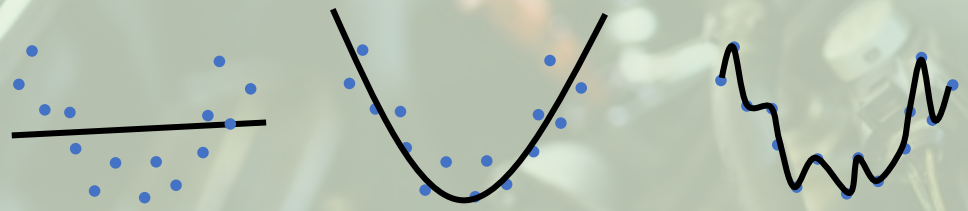
Challenges

Traditional model architectures do not work



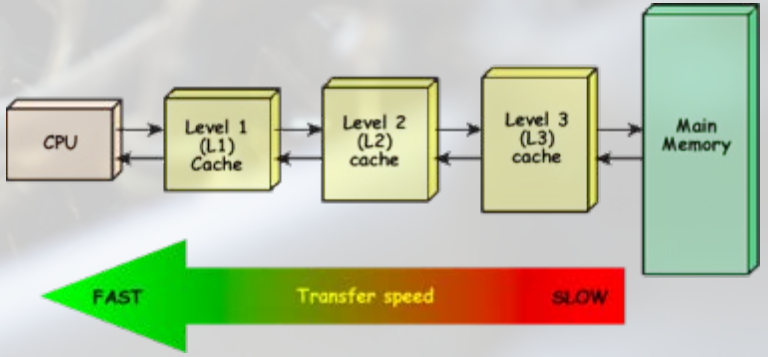
Frameworks are not designed for nano-second execution

Overfitting can be good



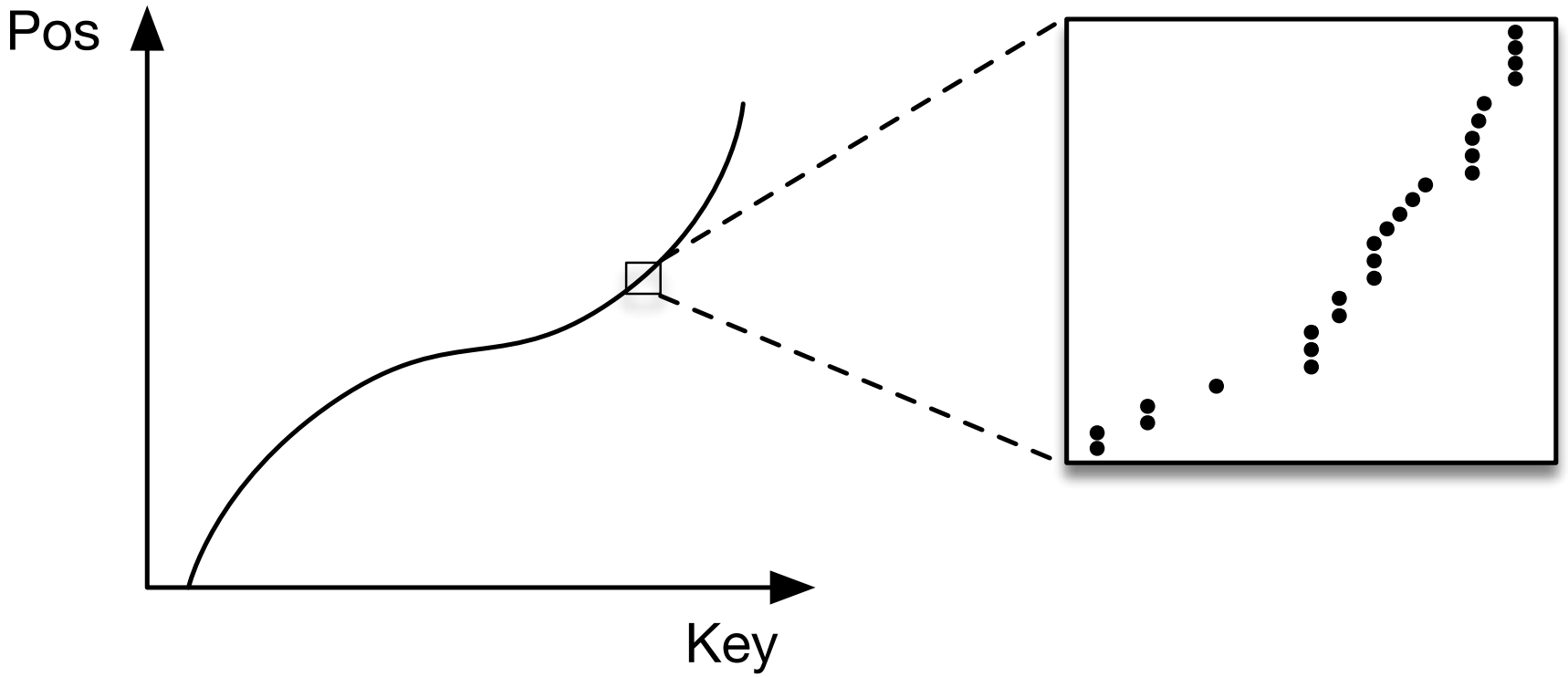
underfitting desired ~~overfitting~~ desired

ML+System Co-Design



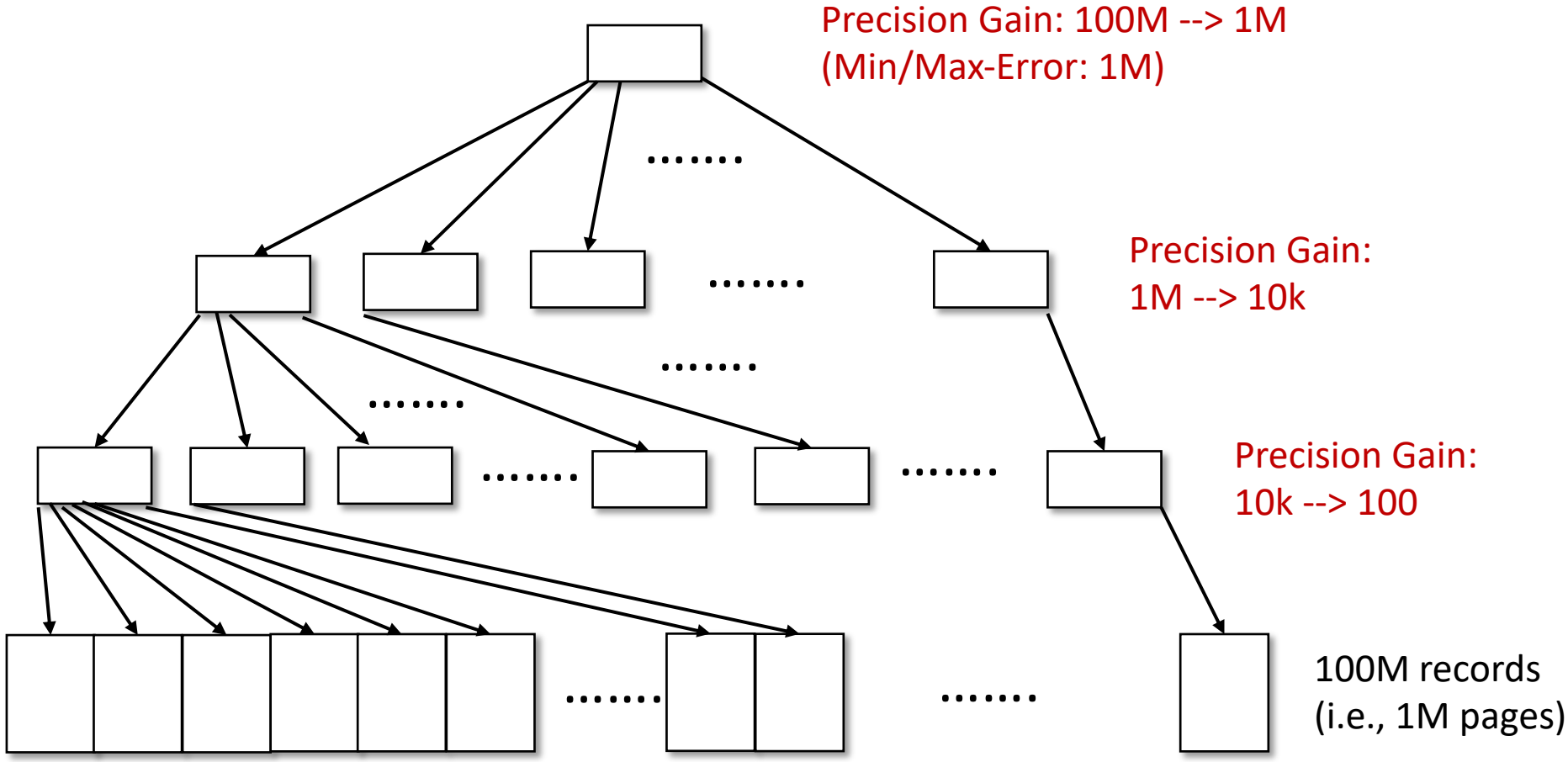
Overfitting is a Good Thing

The Last Mile Problem



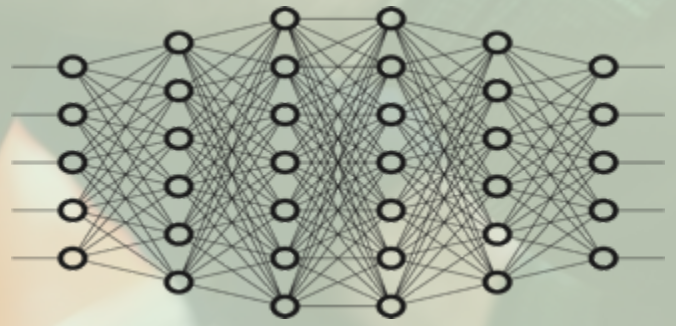
Overfitting is a Good Thing

Index over 100M records. Page-size: 100



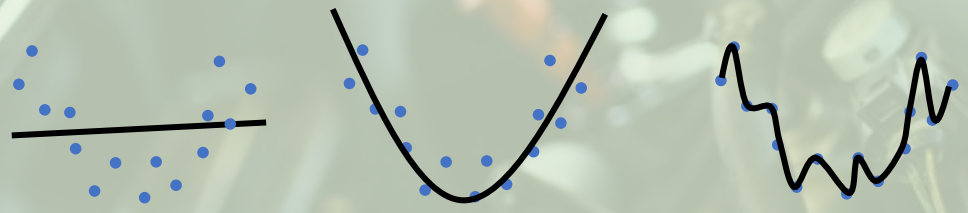
Solution

Traditional model architectures do not work



Frameworks are not designed for nano-second execution

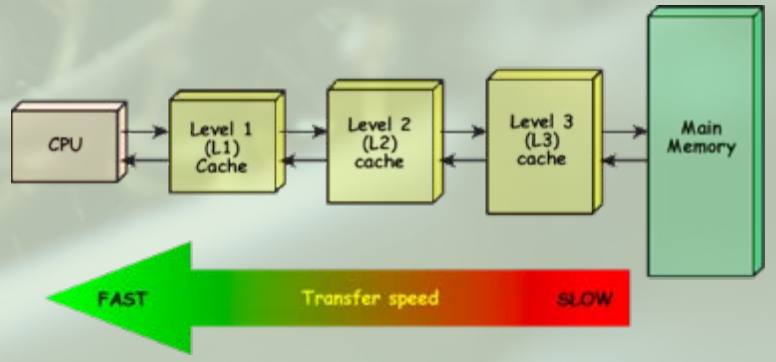
Overfitting can be good



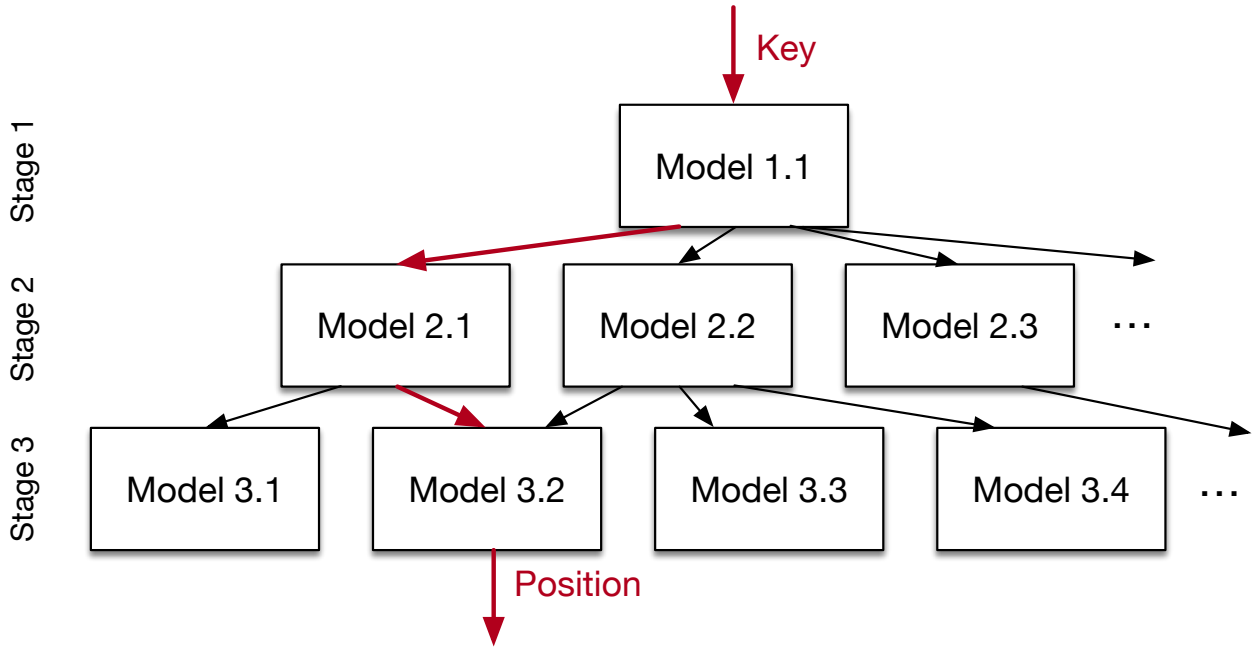
underfitting desired overfitting

~~overfitting~~ desired

ML+System Co-Design



Recursive-Model Index (RMI)



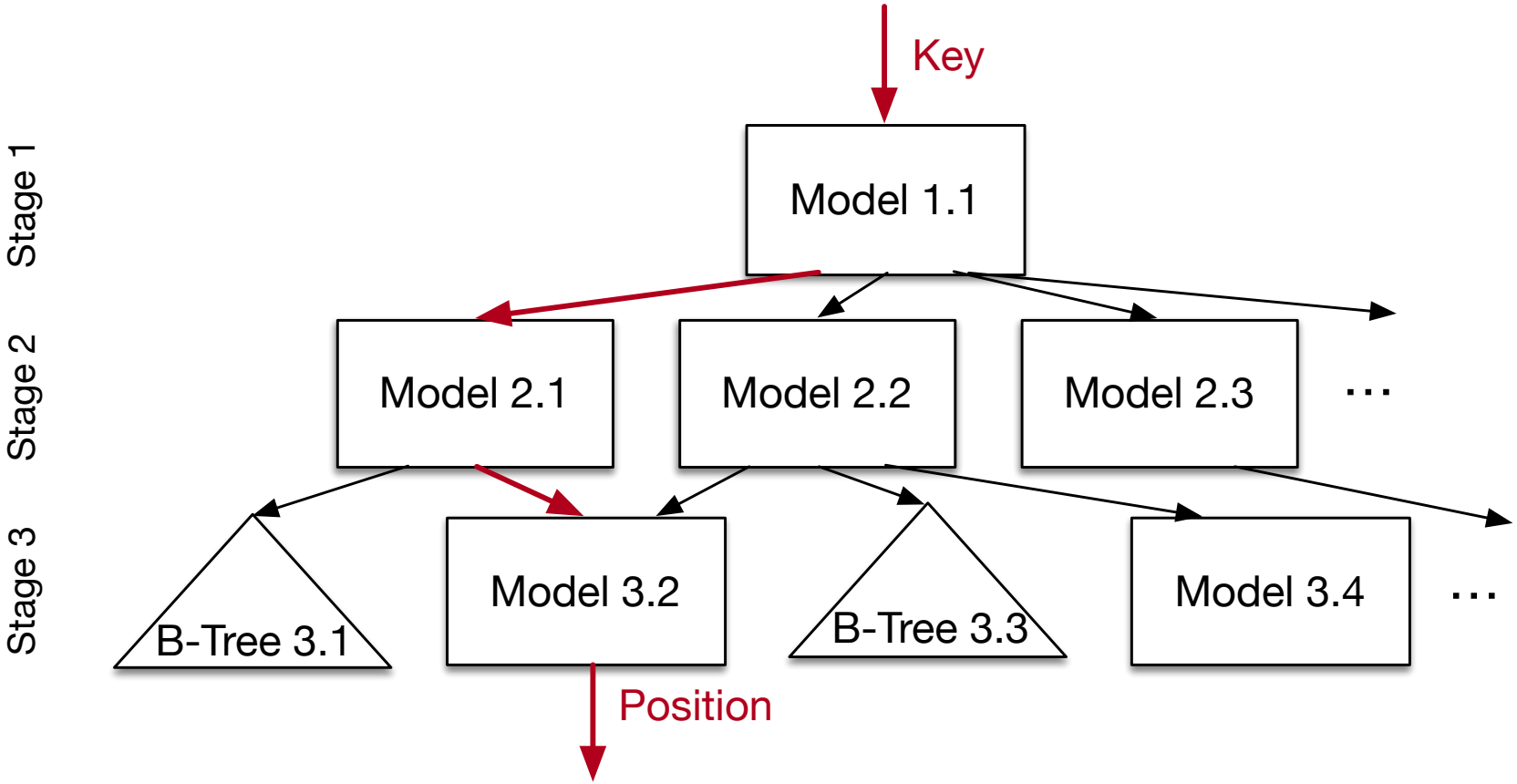
2-Stage RMI with Linear Model

$pos_0 = a_0 + b_0 * key$

$pos_1 = m_1[pos_0].a + m_1[pos_0].b * key$

$record = local\text{-}search(key, pos_1)$

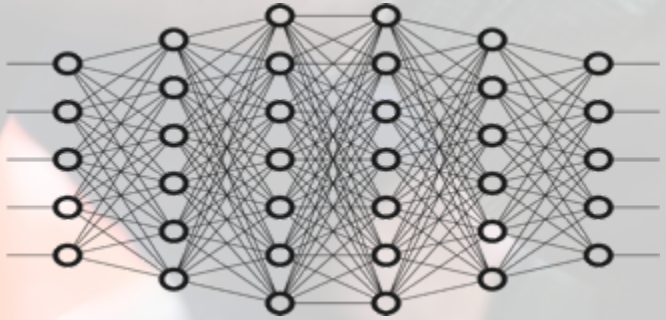
Hybrid RMI



Worst-Case Performance is the one of a B-Tree

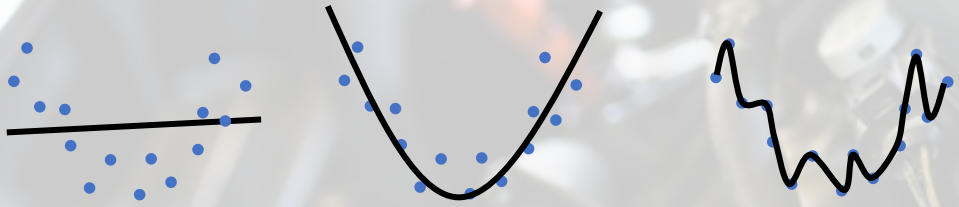
Solution

Traditional model architectures do not work



Frameworks are not designed for nano-second execution

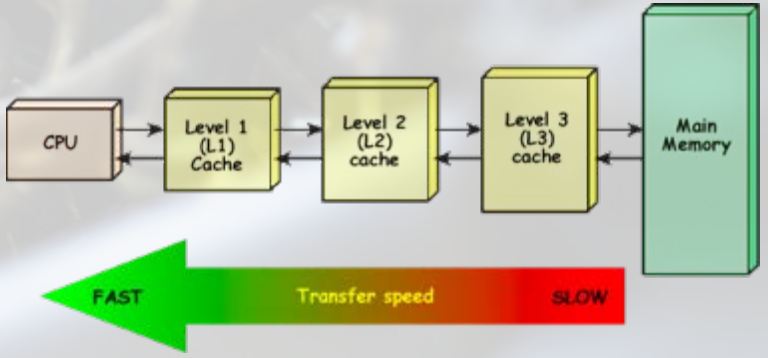
Overfitting can be good



underfitting | desired | overfitting

desired

ML+System Co-Design



The Learning Index Framework (LIF)

- An index synthesis system
- Given an index configuration generate the best possible code
- Uses ideas from Tupleware [VLDB15]
- Simple models are trained “on-the-fly”, whereas for complex models we use Tensorflow and extract weights afterwards (i.e., no Tensorflow during inference time)
- Best index configuration is found using auto-tuning (e.g., see TuPAQ [SOCC15]

Initial Results

>80,000ns

State-Of-The-Art
B-Tree

265ns

13MB

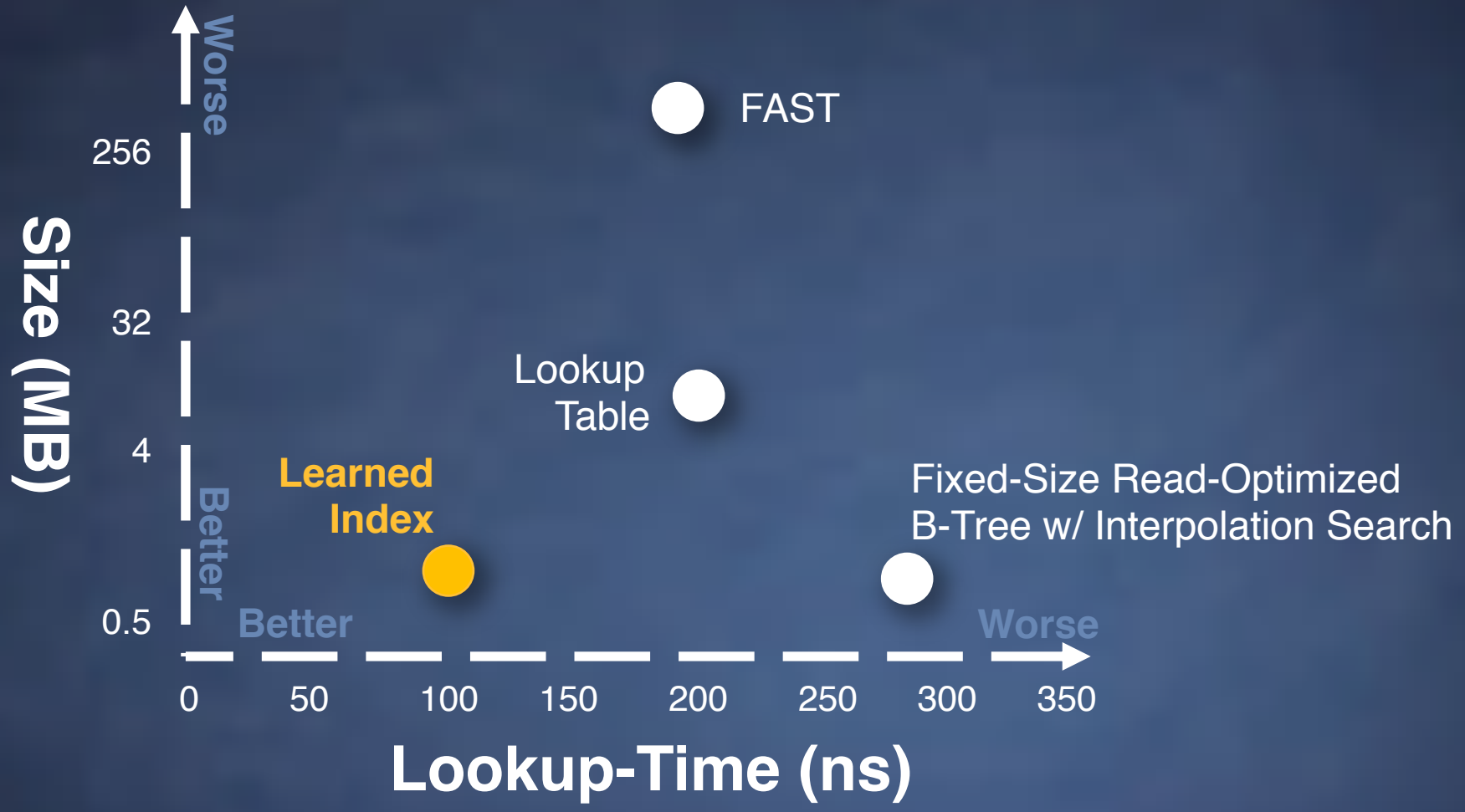
Learned Index

85ns

0.7MB


You Might
Have Seen
Certain
Blog Posts

Initial Results




My Own
Comparison

A Comparison To ARTful Indexes (Radix-Tree)

Viktor Leis, Alfons Kemper, Thomas Neumann: **The Adaptive Radix Tree: ARTful Indexing for Main-Memory Databases.** ICDE 2013

Experimental setup:

- Dense: continuous keys from 0 to 256M
- Sparse: 256M keys where each bit is equally likely 0 or 1.

A Comparison To ARTful Indexes (Radix-Tree)

Viktor Leis, Alfons Kemper, Thomas Neumann: **The Adaptive Radix Tree: ARTful Indexing for Main-Memory Databases.** ICDE 2013

Experimental setup: **continuous keys from 0 to 256M**

Reported lookup throughput: 10M/s ≈ 100ns[1]

Size: not measured, but paper says overhead of ≈8 Bytes per key (dense, best case): 256M * 8 Byte ≈ 1953MB

[1]Numbers from the paper

Learned Index

Generate Code:

```
Record  lookup(key) {
    return data[0 + 1 * key];
}
```

Learned Index

Generate Code:

```
Record  lookup(key) {
    return data[key];
}
```

Learned Index

Generate Code:

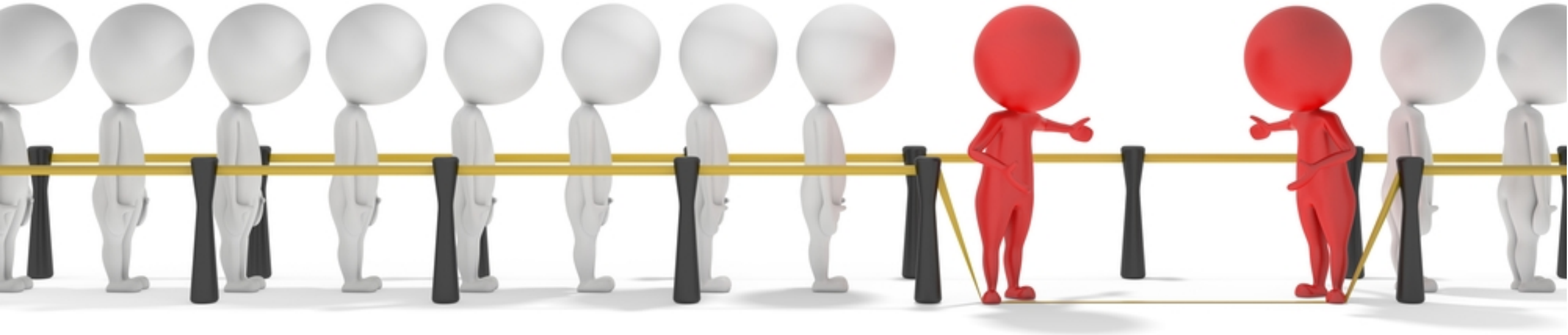
```
Record  lookup(key) {
    return data[key];
}
```

Lookup Latency: 10ns (learned index) vs 100ns* (ARTfull)
or one-order-of-magnitude better

Space: 0MB vs 1953MB
Infinitely better :)


UNFAiR?

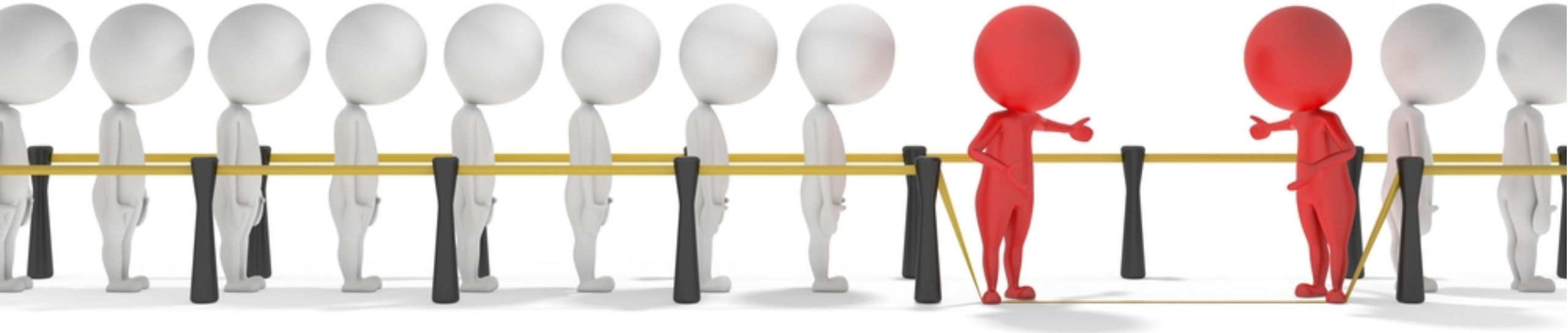
What about Updates and Inserts?



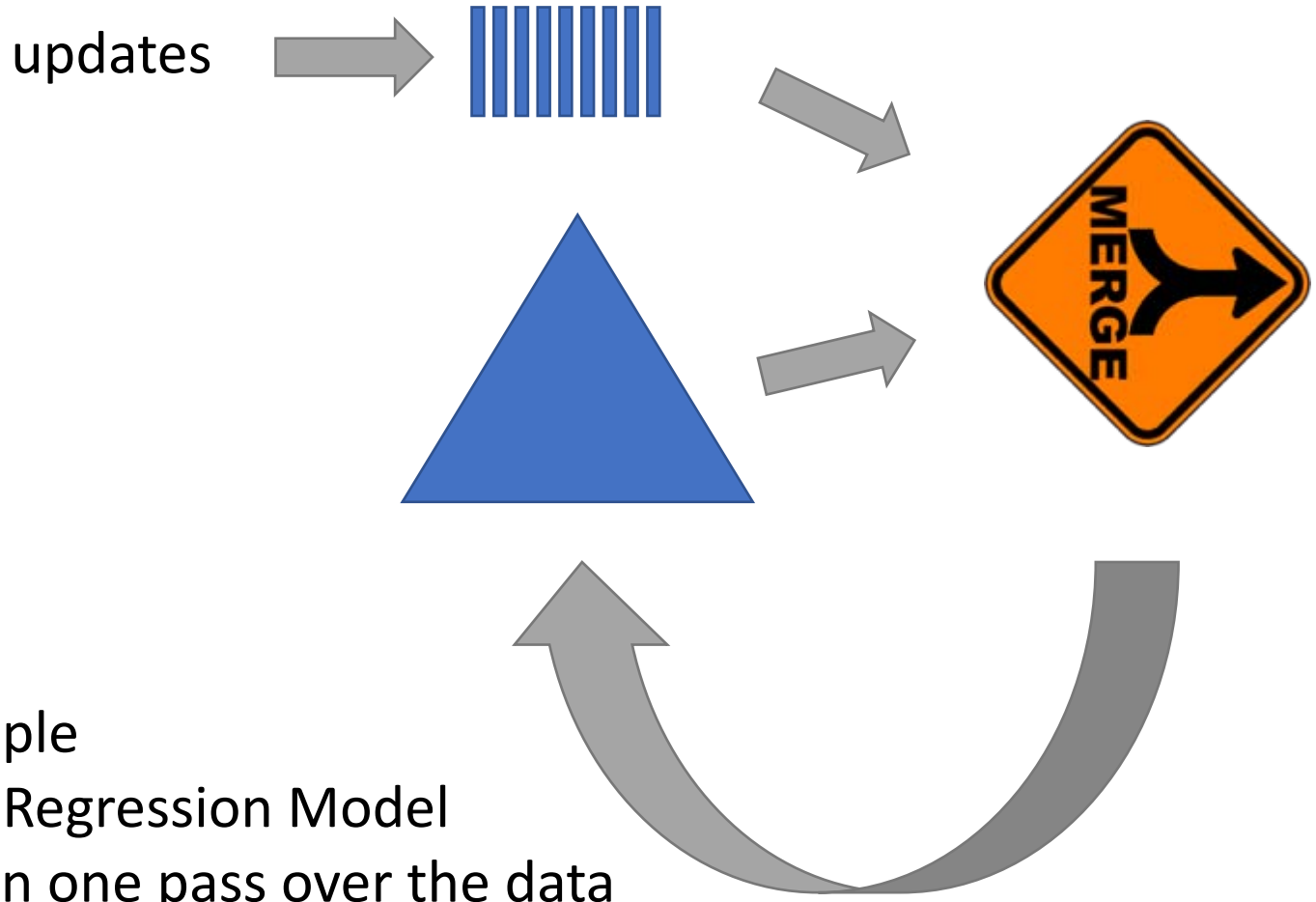
What about Updates and Inserts?

Alex Galakatos, Michael Markovitch, Carsten Binnig, Rodrigo Fonseca, Tim Kraska:
A-Tree: A Bounded Approximate Index Structure
SIGMOD 2019 (talk was yesterday)

Jialin Ding, Umar Farooq Minhas, Hantian Zhang, Yinan Li, Chi Wang, Badrish Chandramouli, Johannes Gehrke, Donald Kossmann, David B. Lomet: **ALEX: An Updatable Adaptive Learned** Index. CoRR abs/1905.08898 (2019)

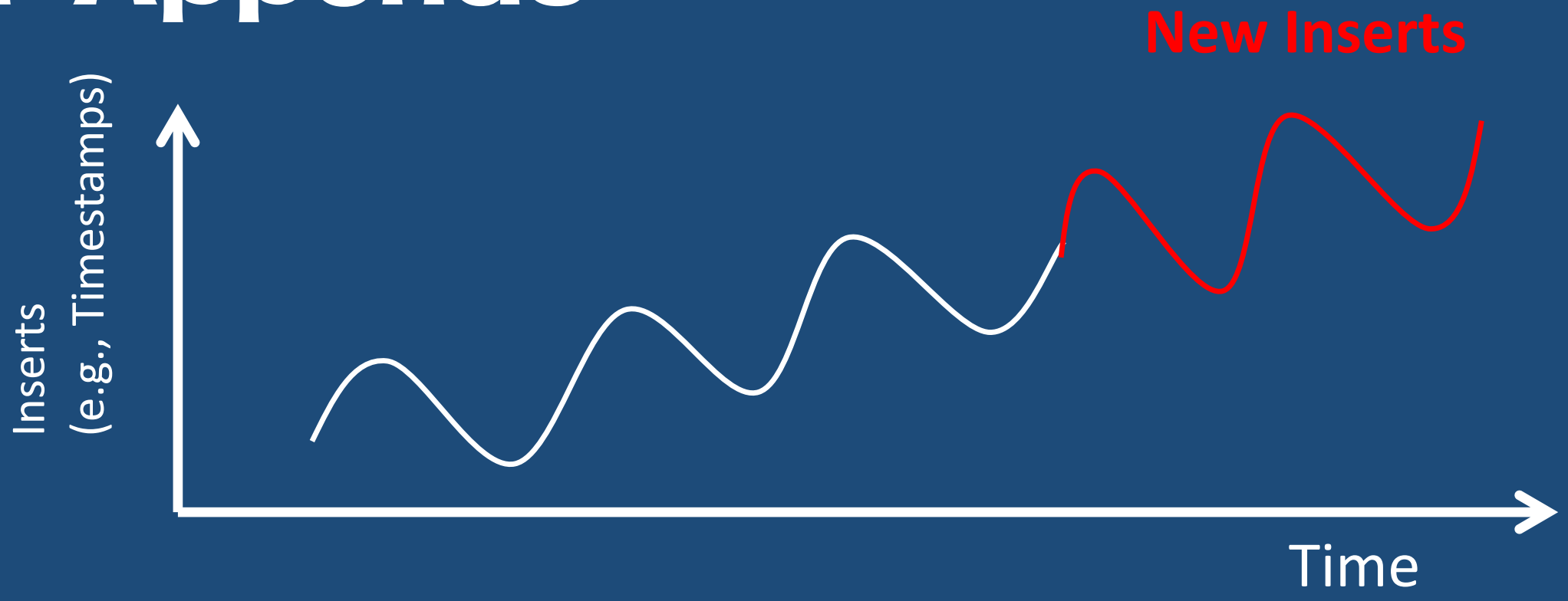


The Simple Approach: Delta Indexing



Training a simple
Multi-Variate Regression Model
Can be done in one pass over the data

Leverage the Distribution for Appends



If the Learned Model Can Generalize to Inserts
Insert complexity is O(1) not O(Log N)

Ali Hadian, Thomas Heinis: Considerations for handling updates in learned index structures. aiDM@SIGMOD 2019: 3:1-3:4

Updates/Inserts

- Less beneficial as the data still has to be stored sorted
- Idea: Leave space in the array where more updates/inserts are expected
- Can also be done with traditional trees.
- But, the error of learned indexes should increase with

  $\sqrt{N}$ per node in RMI whereas traditional indexes with N

Still at the Beginning!

- Can we provide bounds for inserts?
- When to retrain?
- How to retrain models on the fly?
- …

A lot of follow up work (all from 2019)

Jialin Ding, Umar Farooq Minhas, Hantian Zhang, Yinan Li, Chi Wang, Badrish Chandramouli, Johannes Gehrke, Donald Kossmann, David B. Lomet: **ALEX: An Updatable Adaptive Learned Index.** CoRR abs/1905.08898, 2019

Ali Hadian, Thomas Heinis: **Considerations for handling updates in learned index structures**. aiDM@SIGMOD 2019

Alex Galakatos, Michael Markovitch, Carsten Binnig, Rodrigo Fonseca, Tim Kraska**: A-Tree: A Bounded Approximate Index Structure**, SIGMOD 2019

Yingjun Wu, Jia Yu, Yuanyuan Tian, Richard Sidle, Ronald Barber: **Designing Succinct Secondary Indexing Mechanism by Exploiting Column Correlations.** SIGMOD Conference 2019: 1223-1240

Ali Hadian, Thomas Heinis: **Interpolation-friendly B-trees: Bridging the Gap Between Algorithmic and Learned Indexes**. EDBT 2019: 710-713

P Van Sandt, Y Chronis, JM Patel: **Efficiently Searching In-Memory Sorted Arrays: Revenge of the Interpolation Search?,** SIGMOD 2019

Wenkun Xiang, Hao Zhang, Rui Cui, Xing Chu, Keqin Li, Wei Zhou: **Pavo: A RNN-Based Learned Inverted Index, Supervised or Unsupervised?** IEEE Access 7: 293-303 (2019)

Giorgio Vinciguerra, Paolo Ferragina, Michele Miccinesi: **Superseding traditional indexes by orchestrating learning and geometry**, CoRR abs/1903.00507 (2019)

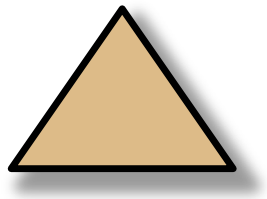
Yingjun Wu, Jia Yu, Yuanyuan Tian, Richard Sidle, Ronald Barber: **Designing Succinct Secondary Indexing Mechanism by Exploiting Column Correlations**, CoRR abs/1903.11203, 2019

Chuzhe Tang, Zhiyuan Dong, Minjie Wang, Zhaoguo Wang, Haibo Chen: **Learned Indexes for Dynamic Workloads**. CoRR abs/1902.00655, 2019

….

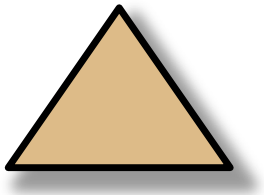
Fundamental Algorithms & Data Structures

Tree

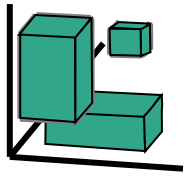


Fundamental Algorithms & Data Structures

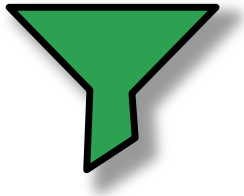
Tree



Multi-Dim Index

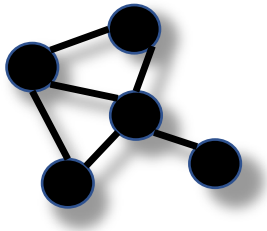


Bloom-Filter

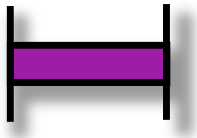


Sorting

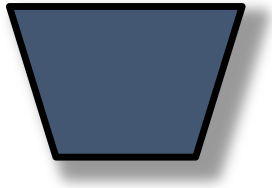
Scheduling



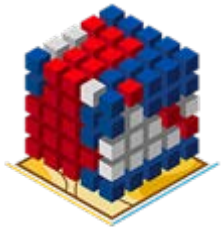
Range-Filter



Hash-Map



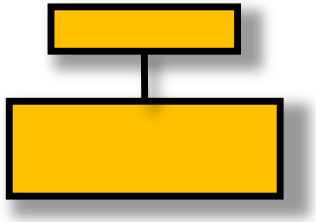
Data Cubes



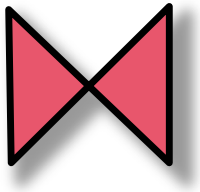
DNA-Search

SQL Query Optimizer

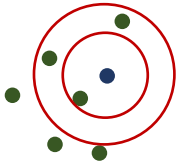
Cache Policy



Join



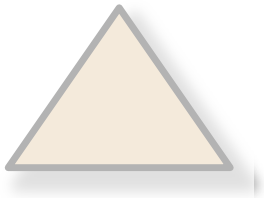
Nearest Neighbor



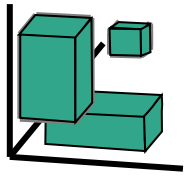
.....

Fundamental Algorithms & Data Structures

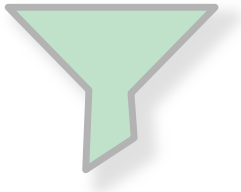
Tree



Multi-Dim Index

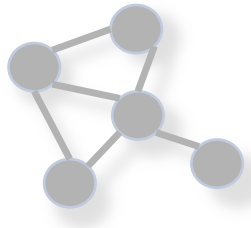


Bloom-Filter



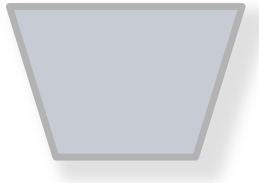
Sorting

Scheduling

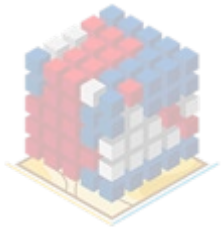


Range-Filter

Hash-Map



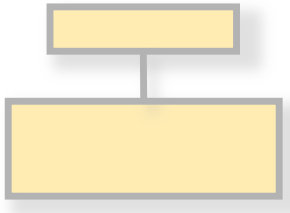
Data Cubes



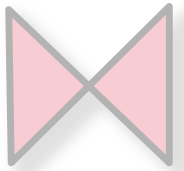
DNA-Search

SQL Query Optimizer

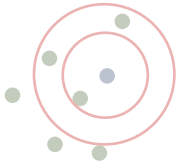
Cache Policy



Join



Nearest Neighbor

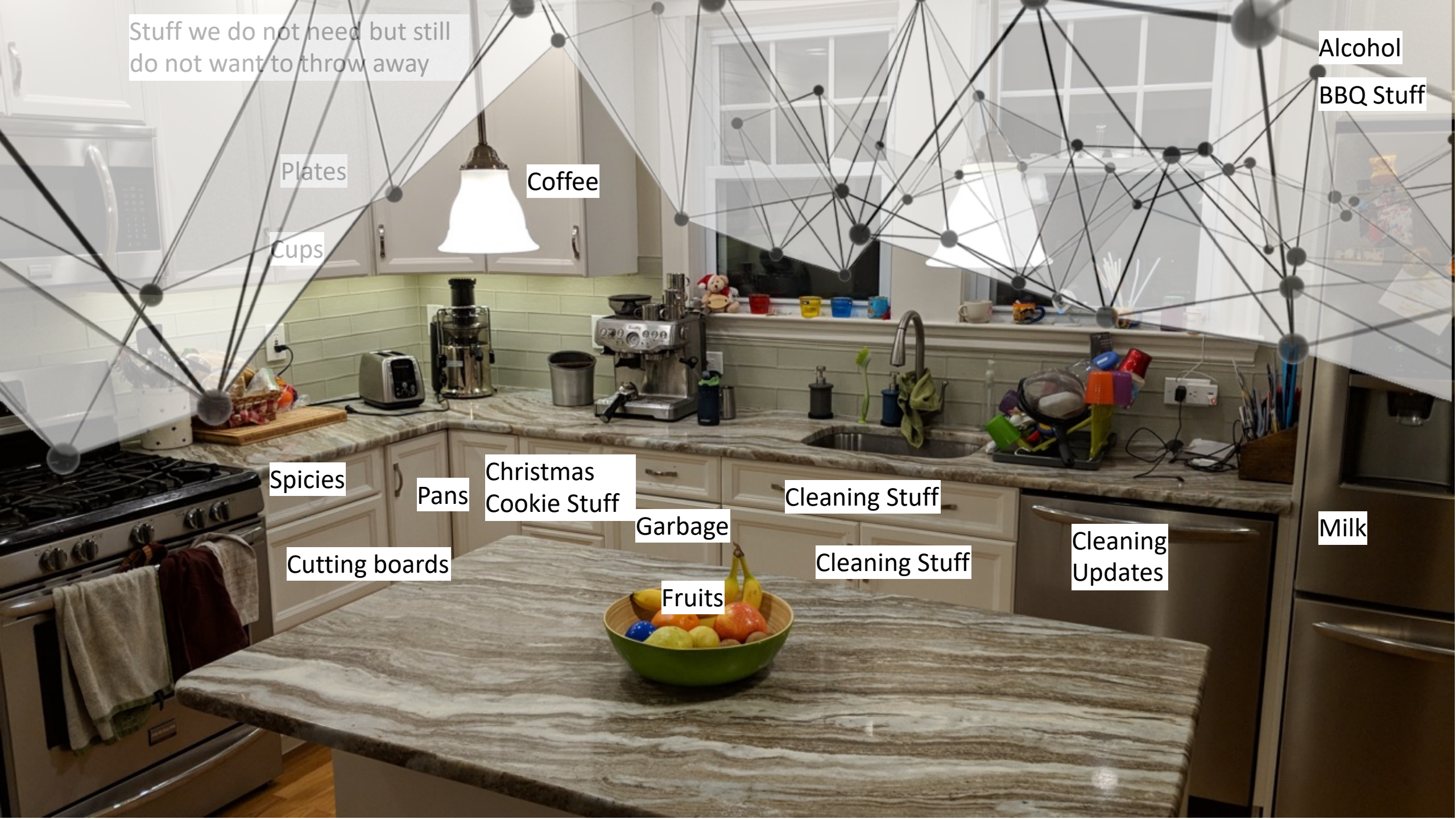


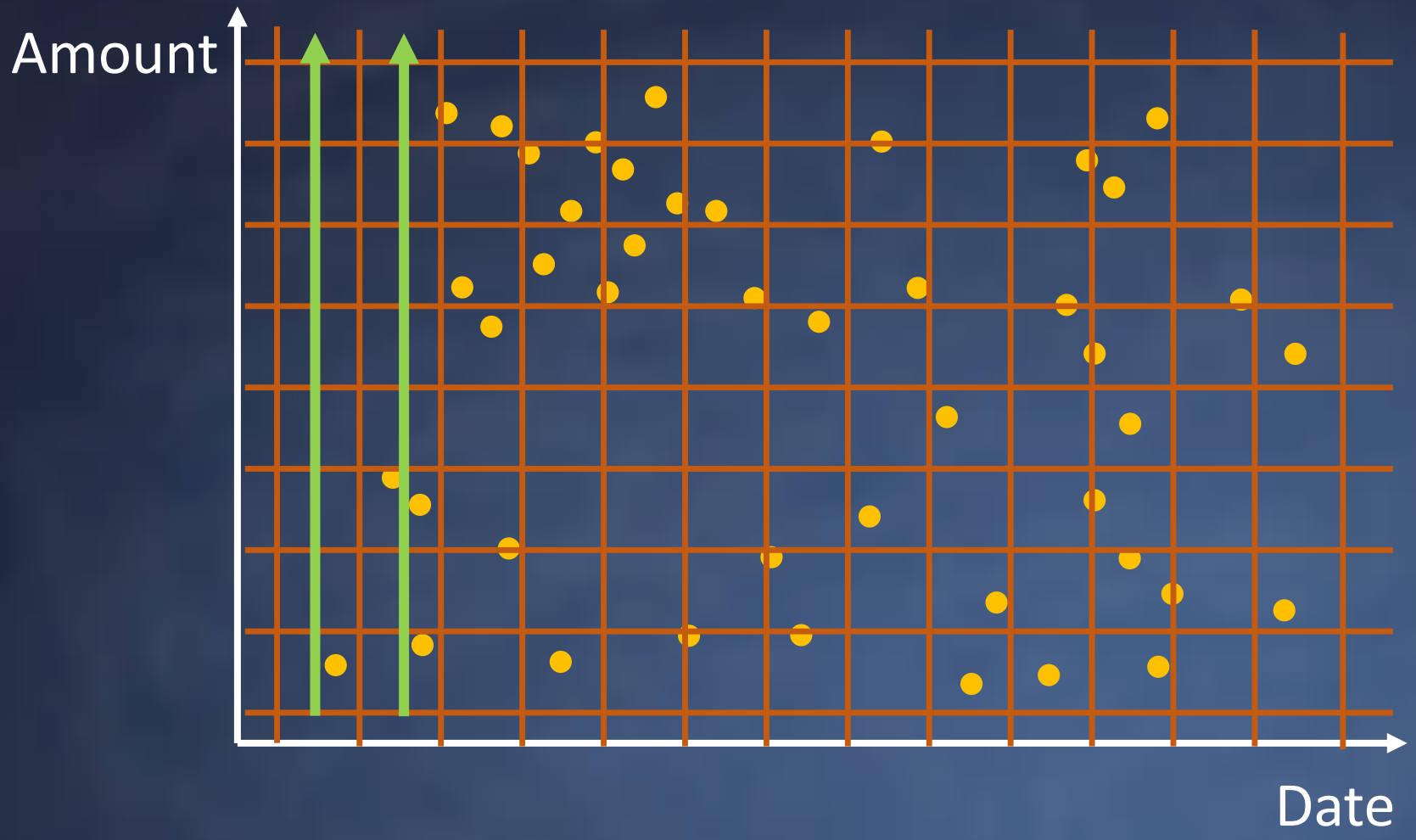
How To Build an Index For First and Last Name, Date, ID, and Email

id	date	first_name	last_name	email	address	zip	state	credit_card_nb	amount
1000	2017-01-01	Hobart	Spracklin	hspracklin0@dailymotion.com	20565 High Crossing Plaza	56372	Minnesota	4405-6975-7285-5160	$ 611.00
1001	2017-01-02	Billye	Binnion	bbinnion1@123-reg.co.uk	3698 Upham Point	20260	District of Columbia	3533-7150-7728-9850	$ 244.00
1002	2017-01-02	Johann	Brockley	jbrockley2@bizjournals.com	23844 Artisan Place	98516	Washington	67597-1193-7985-5100	$ 233.00
1003	2017-01-03	Artie	MacMenami	amacmenamin3@hao123.com	6276 Toban Trail	78759	Texas	3537-4829-6134-5000	$ 210.00
1004	2017-01-03	Delilah	O'Currigan	docurrigan4@chron.com	86016 New Castle Avenue	72199	Arkansas	3555-2017-2226-5780	$ 286.00
1005	2017-01-04	Gretta	Will	gwill5@yelp.com	0 Dottie Circle	68524	Nebraska	503844-1984-2085-5000	$ 870.00
1006	2017-01-04	Gordon	Kirsopp	gkirsopp6@utexas.edu	64060 Scott Park	20370	District of Columbia	633332-1895-2414-5000	$ 687.00
1007	2017-01-05	Bendick	Fagg	bfagg7@army.mil	94 Florence Hill	45440	Ohio	3528-9673-1815-8420	$ 733.00
1008	2017-01-05	Dimitry	Boyet	dboyet8@sakura.ne.jp	35886 Golf Plaza	30066	Georgia	3576-6991-4041-3170	$ 382.00
1009	2017-01-06	Ailsun	Beinke	abeinke9@si.edu	1 Badeau Place	46295	Indiana	56022-2011-8072-1400	$ 854.00
1010	2017-01-07	Lou	Hallows	lhallowsa@theguardian.com	1 Twin Pines Junction	91125	California	5602-2364-4079-0250	$ 150.00
1011	2017-01-09	Tiffani	Mathew	tmathewb@seattletimes.com	0456 Meadow Vale Lane	75260	Texas	6387-6943-8910-4580	$ 313.00
1012	2017-01-09	Perl	Bridie	pbridiec@hubpages.com	07 Bluestem Junction	33124	Florida	3539-8662-2397-5880	$ 558.00
1013	2017-01-09	Rosabelle	Blasik	rblasikd@delicious.com	7 Fairfield Pass	79699	Texas	5602-2297-6599-8560	$ 941.00
1014	2017-01-10	Meggi	Belamy	mbelamye@ask.com	0995 Manufacturers Street	10170	New York	3557-5094-7405-8340	$ 875.00
1015	2017-01-10	Tadio	Balderston	tbalderstonf@apache.org	80 Novick Road	75260	Texas	60485-3728-7119-9300	$ 954.00
1016	2017-01-11	Gianina	Oxteby	goxtebyg@google.pl	72674 Fuller Avenue	89505	Nevada	4-0415-9268-2397	$ 239.00
1017	2017-01-12	Brendan	Doody	bdoodyh@craigslist.org	87414 Golden Leaf Street	11480	New York	201-6348-4121-1314	$ 308.00
1018	2017-01-13	Conway	Coombs	ccoombsi@blogger.com	2810 Oakridge Park	32859	Florida	3529-1514-0357-9120	$ 60.00
1019	2017-01-14	Germaine	Bere	gberej@bravesites.com	82802 Oakridge Park	20041	District of Columbia	670961-0240-4054-9000	$ 95.00
1020	2017-01-15	Davide	Tolcharde	dtolchardek@redcross.org	89 Continental Avenue	79165	Texas	5018-7748-4325-9510	$ 137.00
1021	2017-01-16	Nigel	Artharg	narthargl@gizmodo.com	31 Mcbride Point	22301	Virginia	560225-6965-2870-0000	$ 496.00
1022	2017-01-17	Rickard	Trenholm	rtrenholmm@cbslocal.com	93 Hoepker Parkway	70593	Louisiana	3541-5241-5383-9970	$ 760.00
1023	2017-01-18	Juditha	Dwane	jdwanen@vk.com	7914 Eliot Lane	14276	New York	5456-4410-0914-3180	$ 474.00
1024	2017-01-19	Susan	Ilden	sildeno@aol.com	25204 Huxley Road	21684	Maryland	3574-8586-6367-9920	$ 83.00
1025	2017-01-20	Abbey	Triggle	atrigglep@google.com.au	47 Debra Pass	74184	Oklahoma	3538-6047-6315-7710	$ 513.00
1026	2017-01-21	Zsazsa	Dunster	zdunsterq@nature.com	7 Gerald Alley	40576	Kentucky	3562-0325-7709-3490	$ 952.00
1027	2017-01-22	Grantham	Friatt	gfriattr@seattletimes.com	774 Prairieview Circle	29225	South Carolina	3571-1171-9476-8780	$ 942.00
1028	2017-01-22	Ross	Gaudin	rgaudins@samsung.com	3102 Loeprich Trail	68197	Nebraska	5108-7578-4665-2710	$ 572.00
1029	2017-01-22	Aluino	Drover	adrovert@dagondesign.com	2717 Northridge Avenue	72199	Arkansas	670999-3171-8848-0000	$ 318.00
1030	2017-01-23	Shurlock	Braker	sbrakeru@huffingtonpost.com	30783 Jenna Alley	80945	Colorado	6331106-1894-9878-0000	$ 166.00

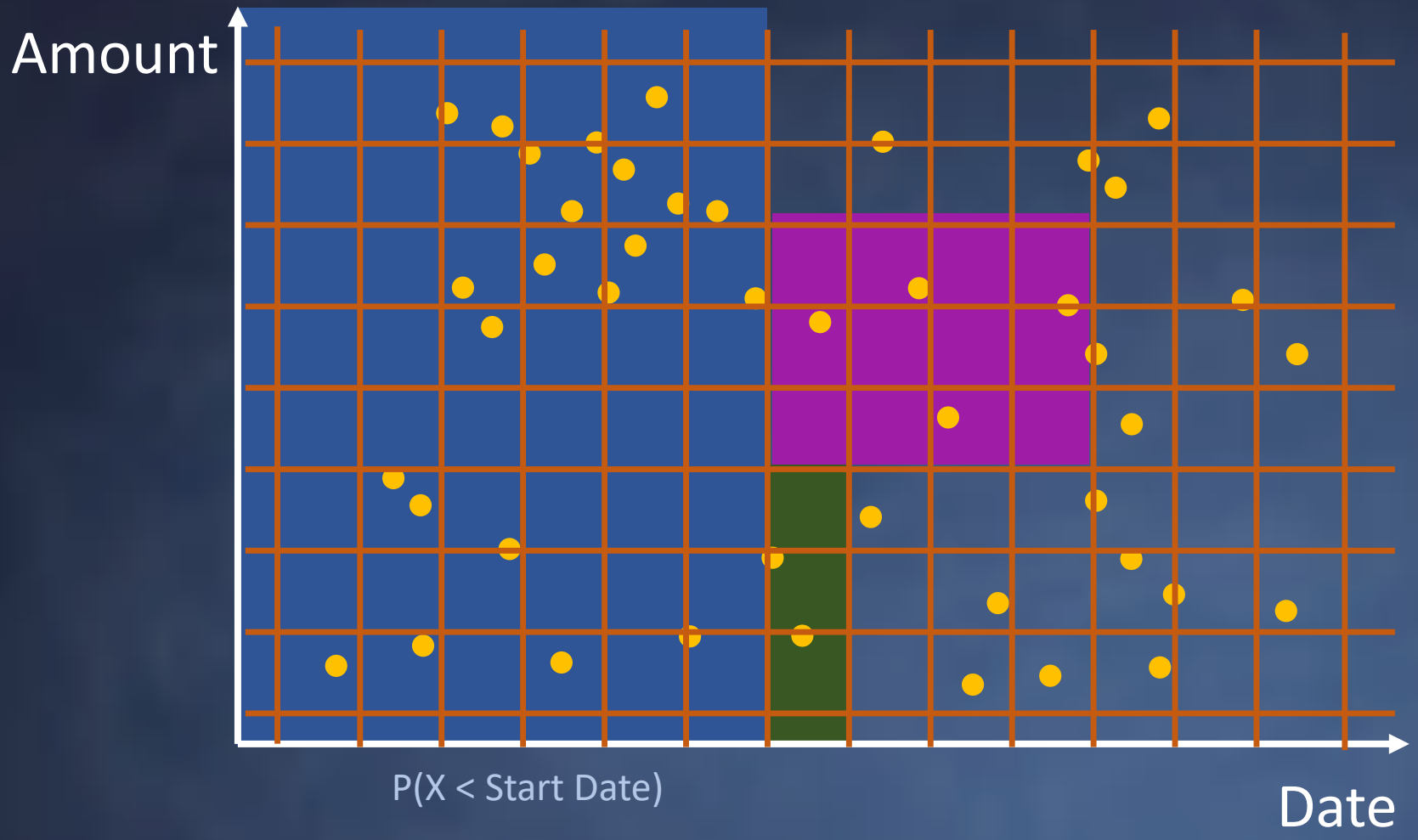
There is only one order on disk

Jump to results

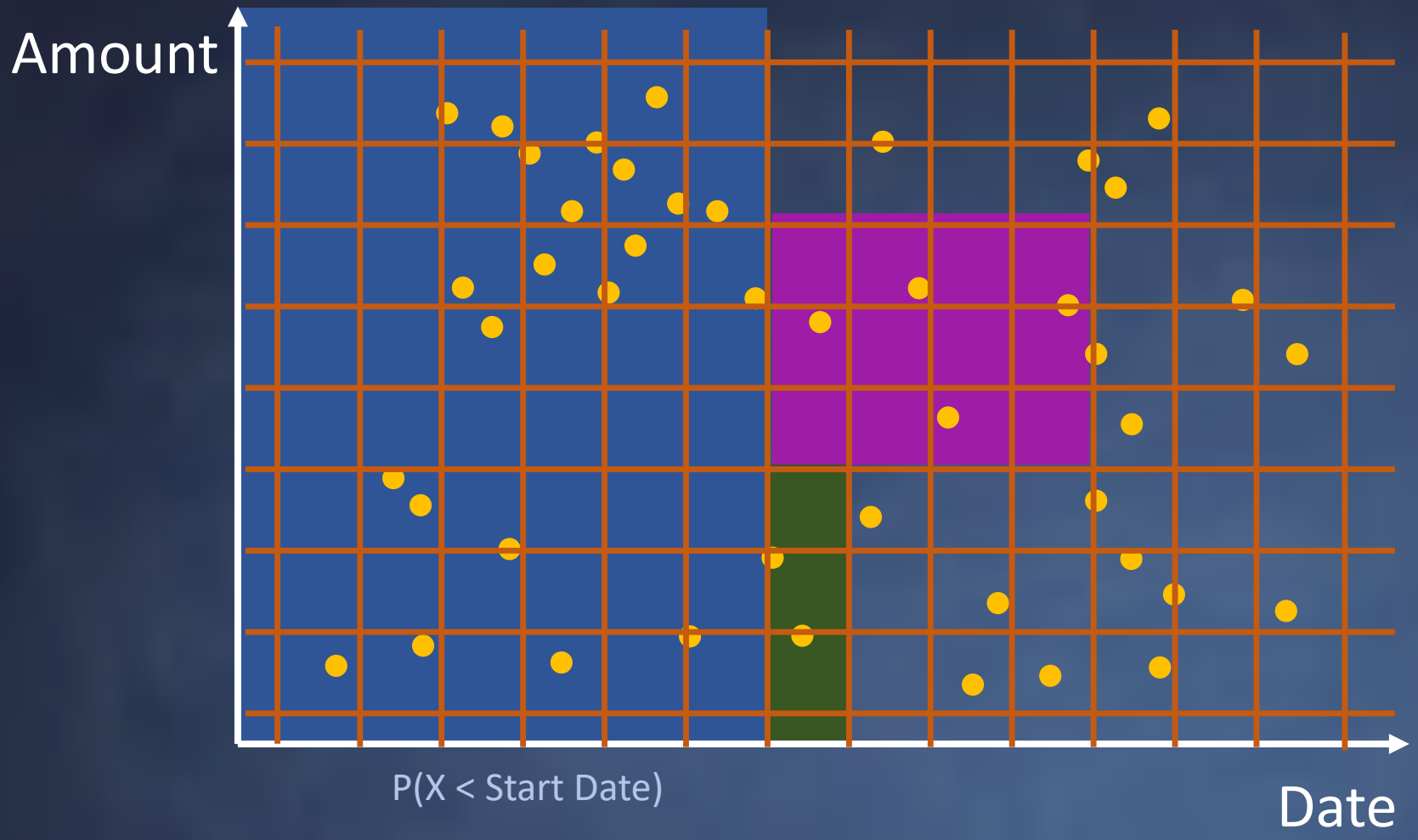

Stuff we do not need but still do not want to throw away
Plates
Cups
Coffee
Alcohol
BBQ Stuff
Spicies
Pans
Christmas Cookie Stuff
Garbage
Cleaning Stuff
Cutting boards
Cleaning Stuff
Cleaning Updates
Milk
Fruits



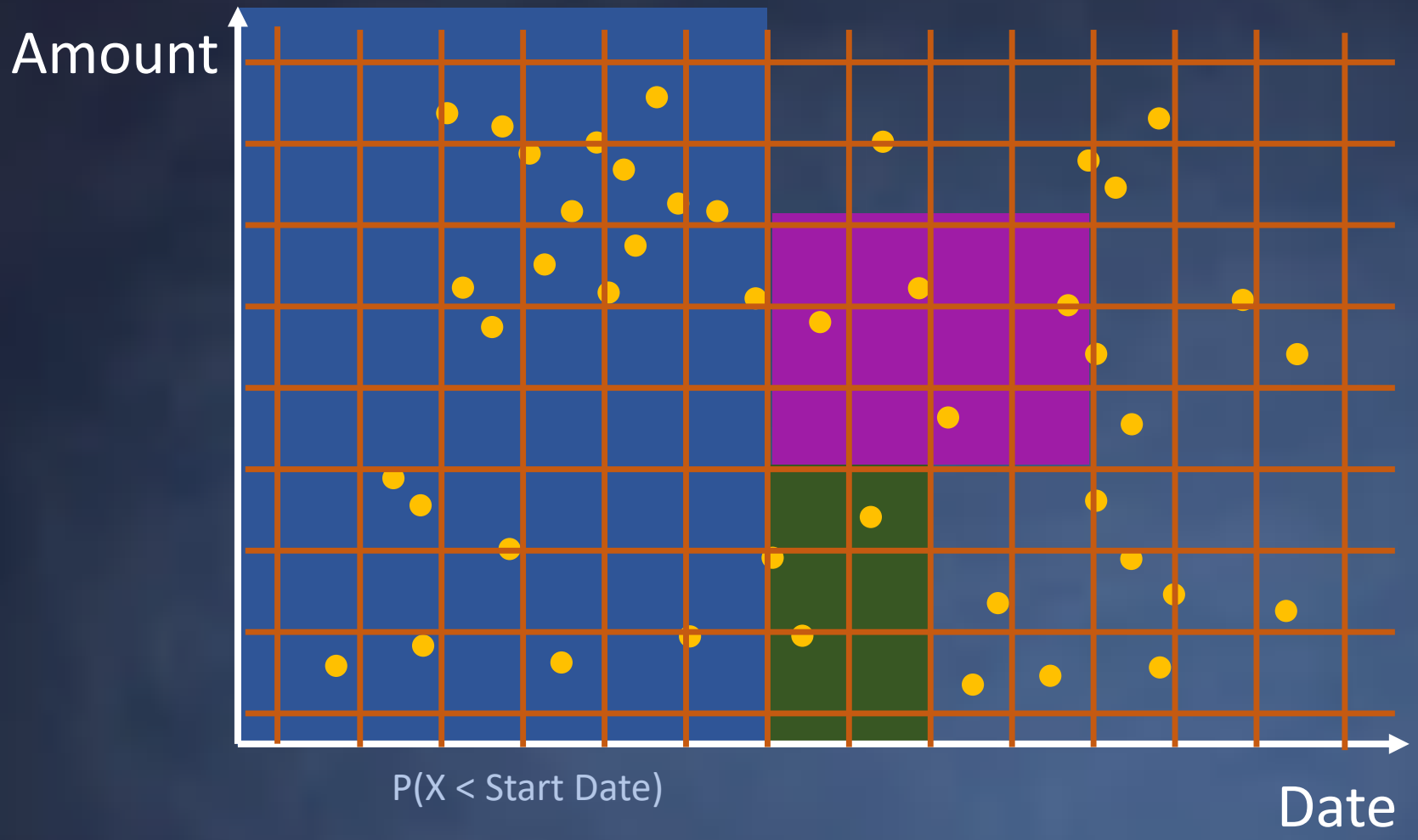
Storage:

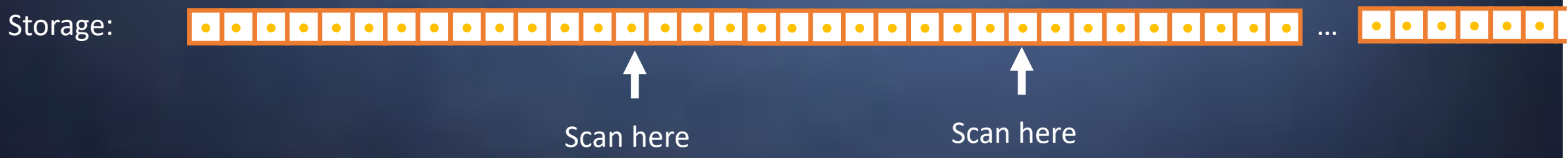


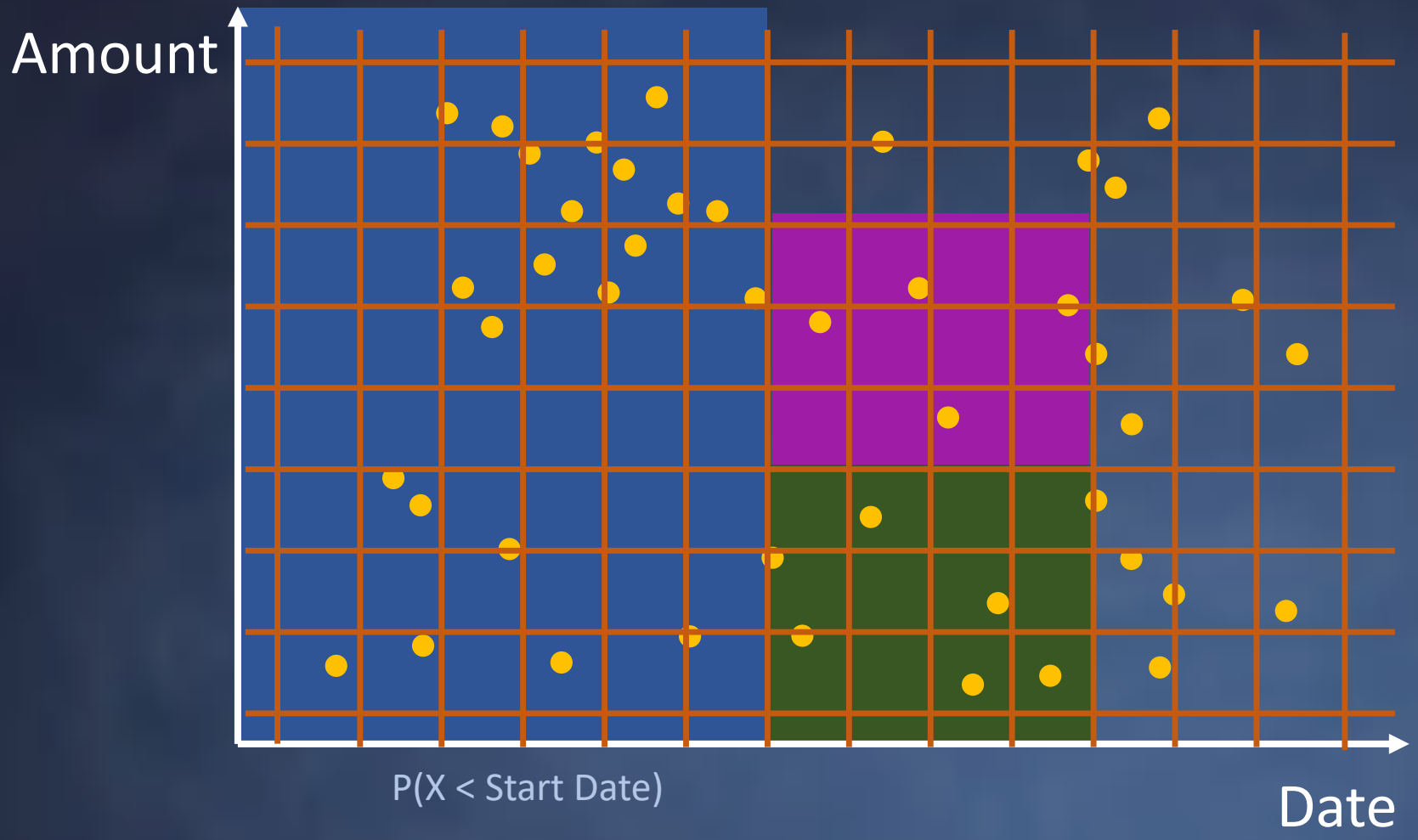
Storage:

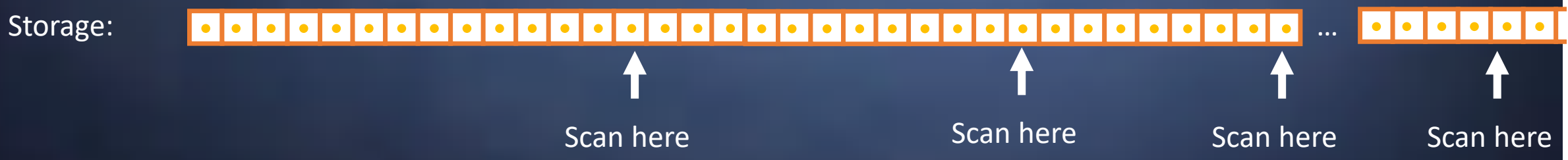

Amount
P(X < Start Date)
Date


Storage:
...
Scan here


Amount
P(X < Start Date)
Date


Storage:
...
Scan here
Scan here

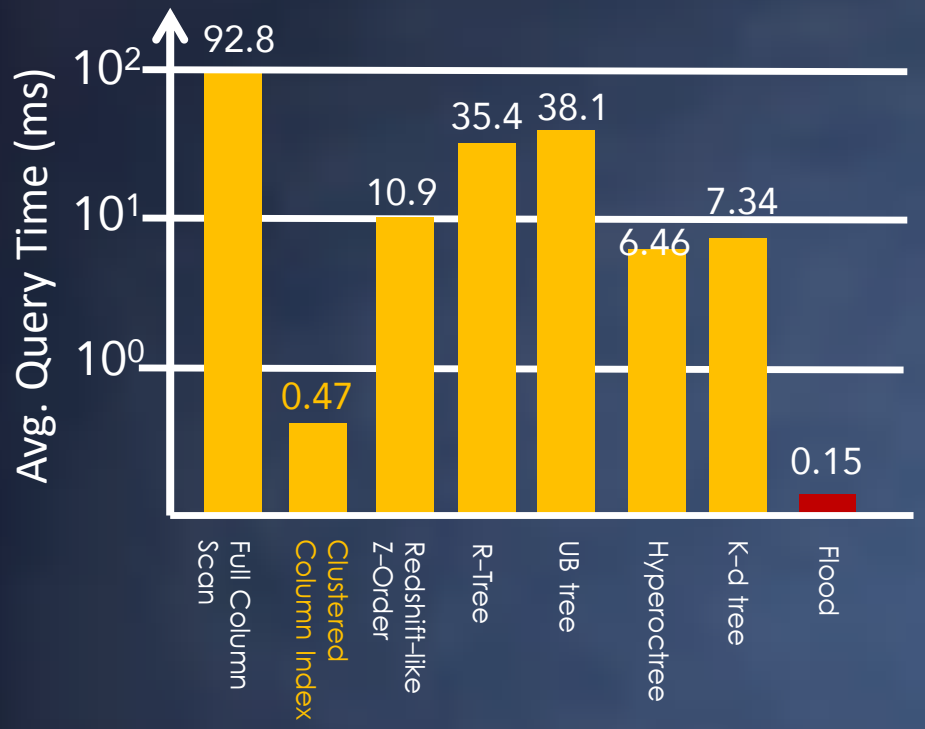

Amount
P(X < Start Date)
Date


Storage:
...
Scan here
Scan here
Scan here
Scan here

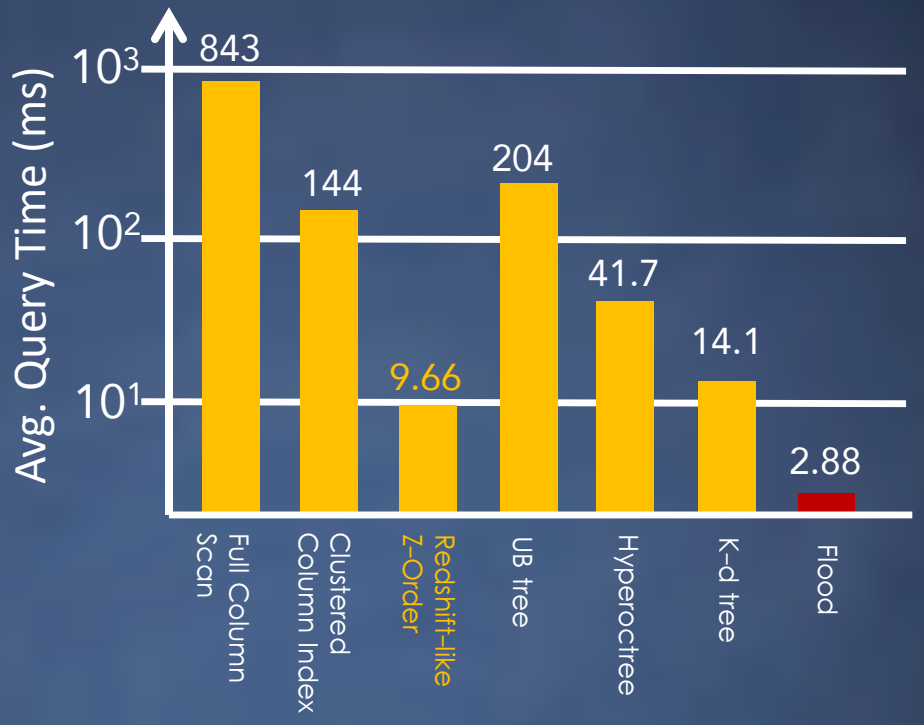
Models and Layout Can Be Data and Workload Dependent

Initial Results

Sales DW Workload



Perf-Monitoring Workload



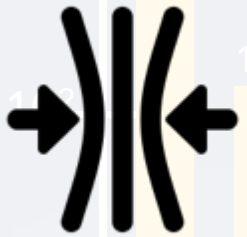
In-Memory Column Store, Single Threaded, Traditional Data Compression, No SIMD Opt.
Queries: Real-world workload: only scan-filter-aggregate (no joins, etc)

Initial Results

Note: We are still at the beginning!

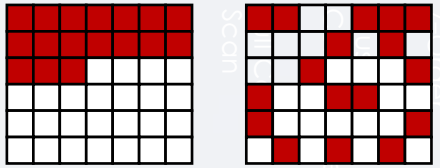
Unexplored opportunities

GPUs/FPGAs



ML Compression

Re-Generating IDs

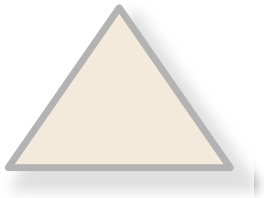


Sparse vs dense storage

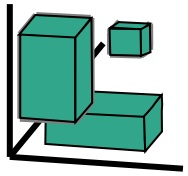
In-Memory Column Store, Single Threaded, Traditional Data Compression, No SIMD Opt.
Queries: Real world workload: only scan, filter, aggregate (no joins, etc)

Fundamental Algorithms & Data Structures

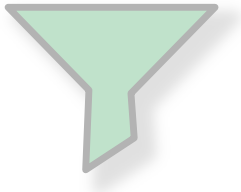
Tree



Multi-Dim Index

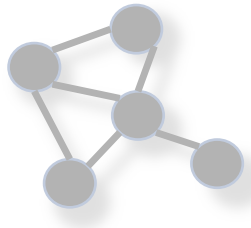


Bloom-Filter



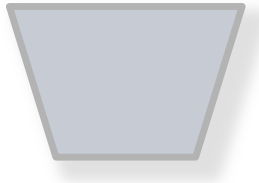
Sorting

Scheduling

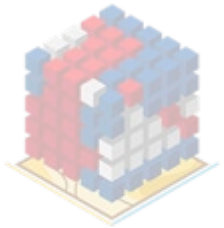


Range-Filter

Hash-Map



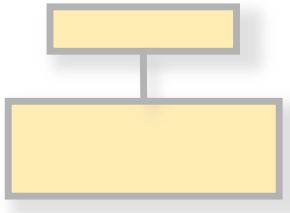
Data Cubes



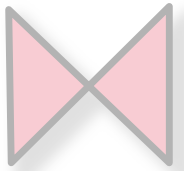
DNA-Search

SQL Query Optimizer

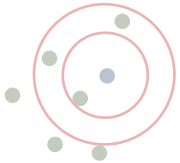
Cache Policy



Join

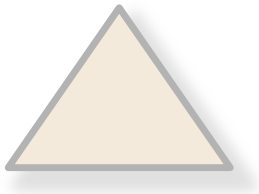


Nearest Neighbor

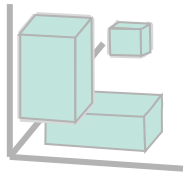


Fundamental Algorithms & Data Structures

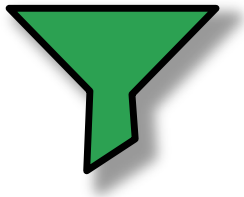
Tree



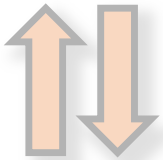
Multi-Dim Index



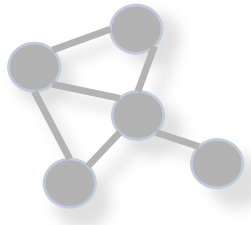
Bloom-Filter



Sorting

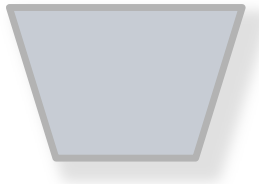


Scheduling

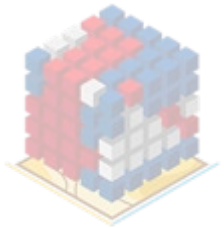


Range-Filter

Hash-Map



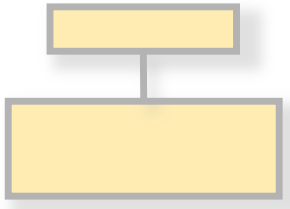
Data Cubes



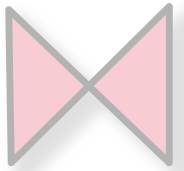
DNA-Search

SQL Query Optimizer

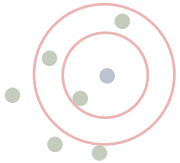
Cache Policy



Join

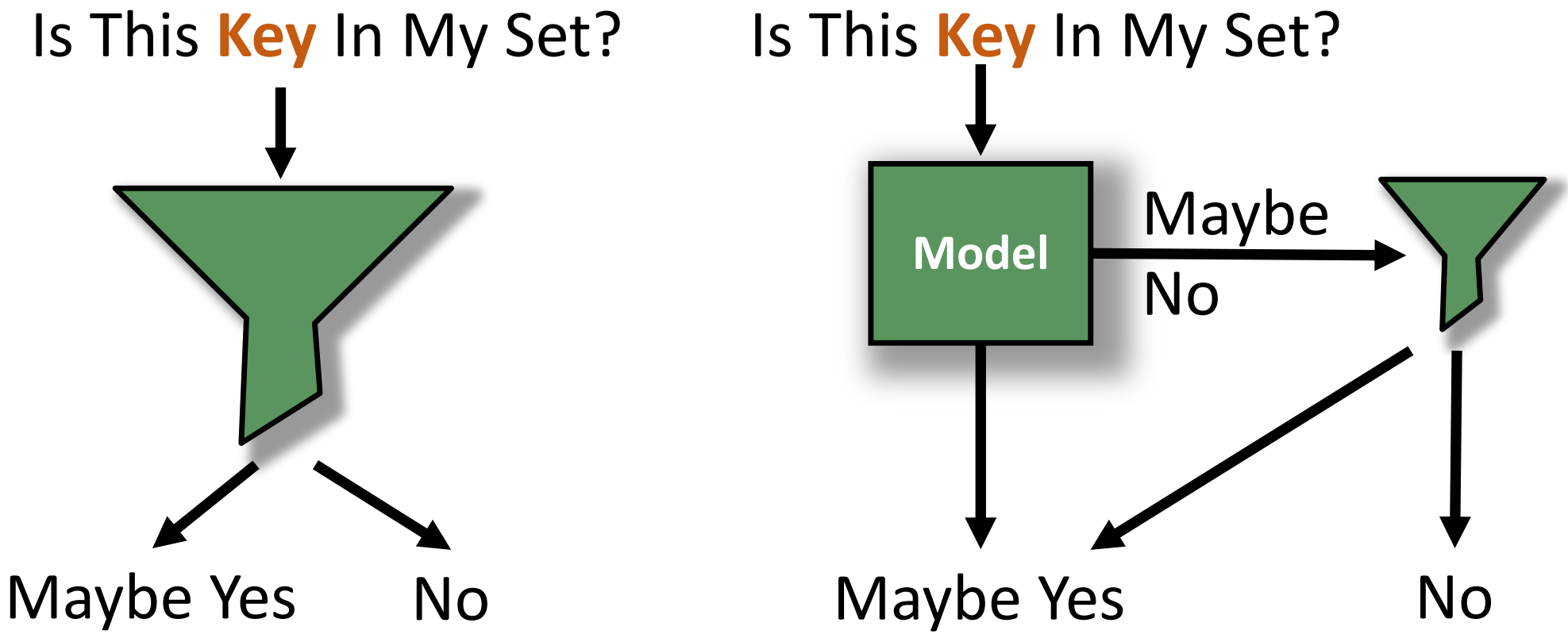


Nearest Neighbor



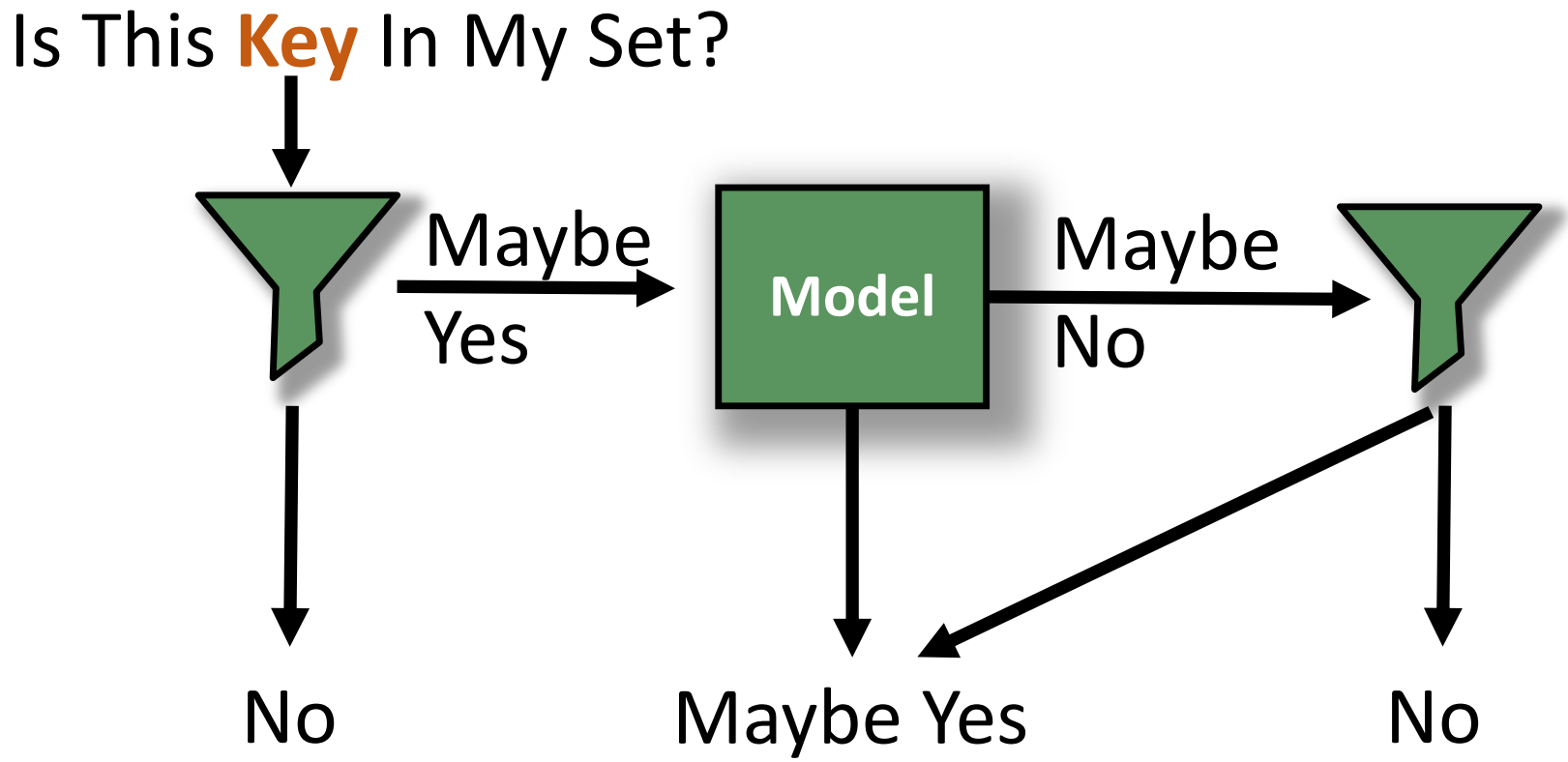
.....

Bloom Filter- Approach 1



36% Space Improvement over Bloom Filter at Same False Positive Rate

Sandwiched Bloom Filter



Michael Mitzenmacher: A Model for Learned Bloom Filters and Optimizing by Sandwiching. NeurIPS 2018: 462-471

Follow up work

Michael Mitzenmacher: **A Model for Learned Bloom Filters and Optimizing by Sandwiching**. NeurIPS 2018: 462-471

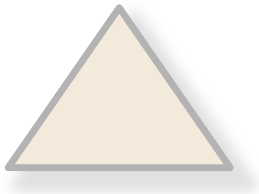
Jack W. Rae, Sergey Bartunov, Timothy P. Lillicrap: **Meta-Learning Neural Bloom Filters**. ICML 2019: 5271-5280

Stephen Macke, Alex Beutel, Tim Kraska, Maheswaran Sathiamoorthy, Derek Zhiyuan Cheng, Ed H. Chi: **Lifting the Curse of Multidimensional Data with Learned Existence Indexes**, ML for Systems workshop at NeurIPS, 2018

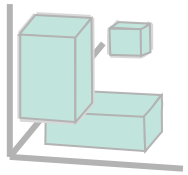
….

Fundamental Algorithms & Data Structures

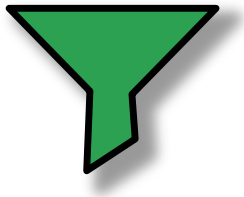
Tree



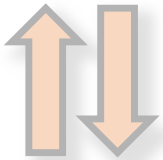
Multi-Dim Index



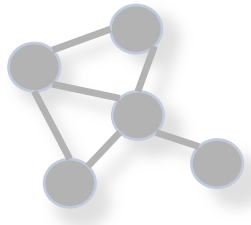
Bloom-Filter



Sorting

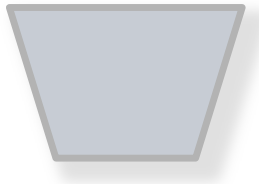


Scheduling

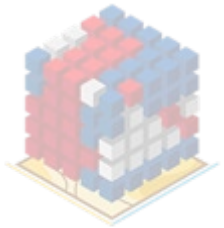


Range-Filter

Hash-Map



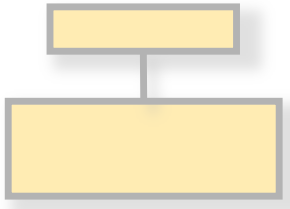
Data Cubes



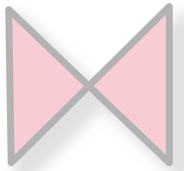
DNA-Search

SQL Query Optimizer

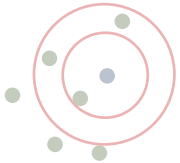
Cache Policy



Join



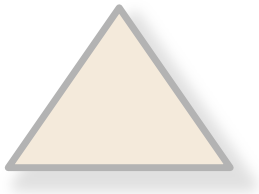
Nearest Neighbor



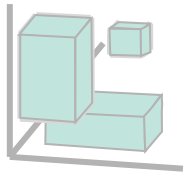
.....

Fundamental Algorithms & Data Structures

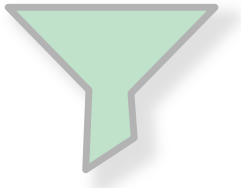
Tree



Multi-Dim Index

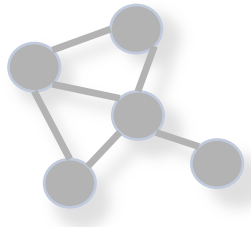


Bloom-Filter



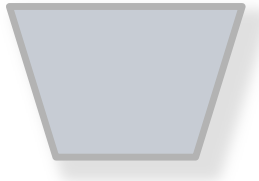
Sorting

Scheduling

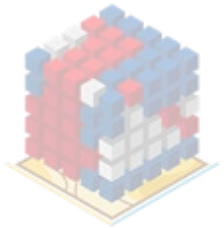


Range-Filter

Hash-Map



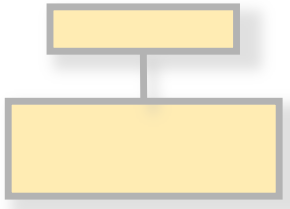
Data Cubes



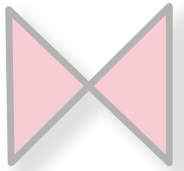
DNA-Search

SQL Query Optimizer

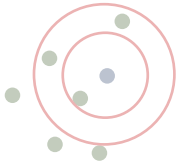
Cache Policy



Join

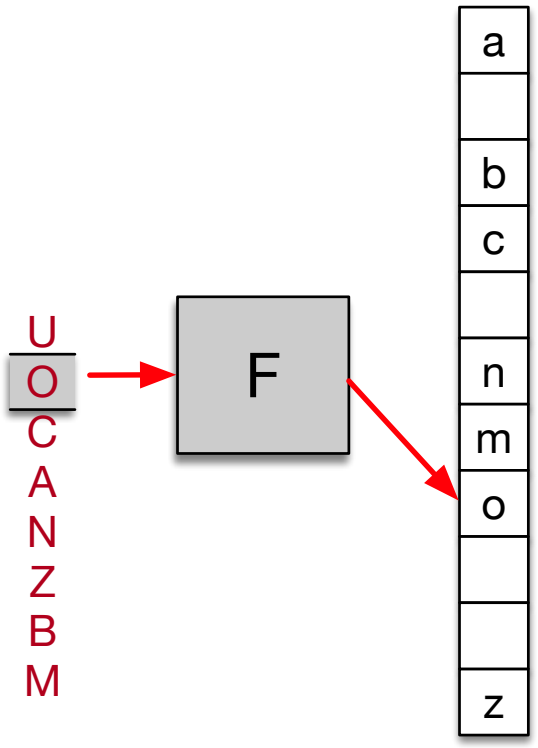


Nearest Neighbor



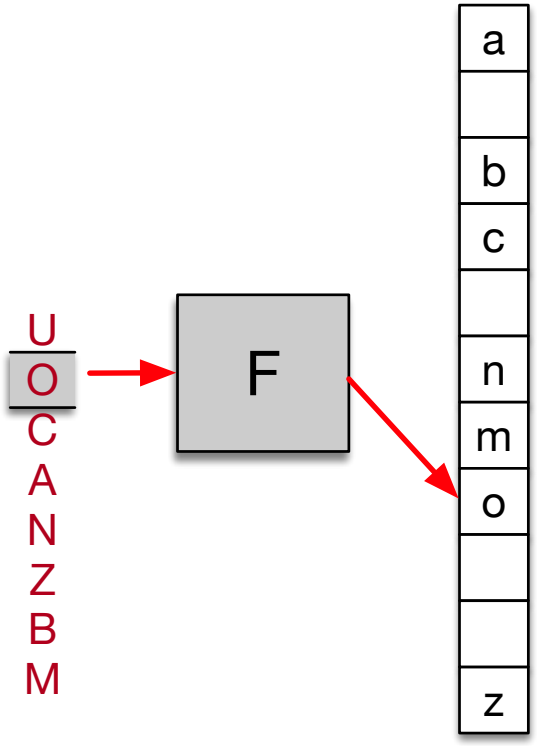
Sorting

(a) CDF Model Pre-Sorts

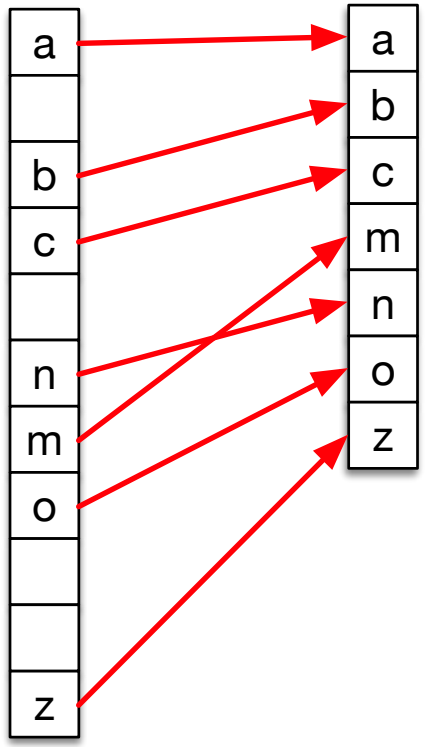


Sorting

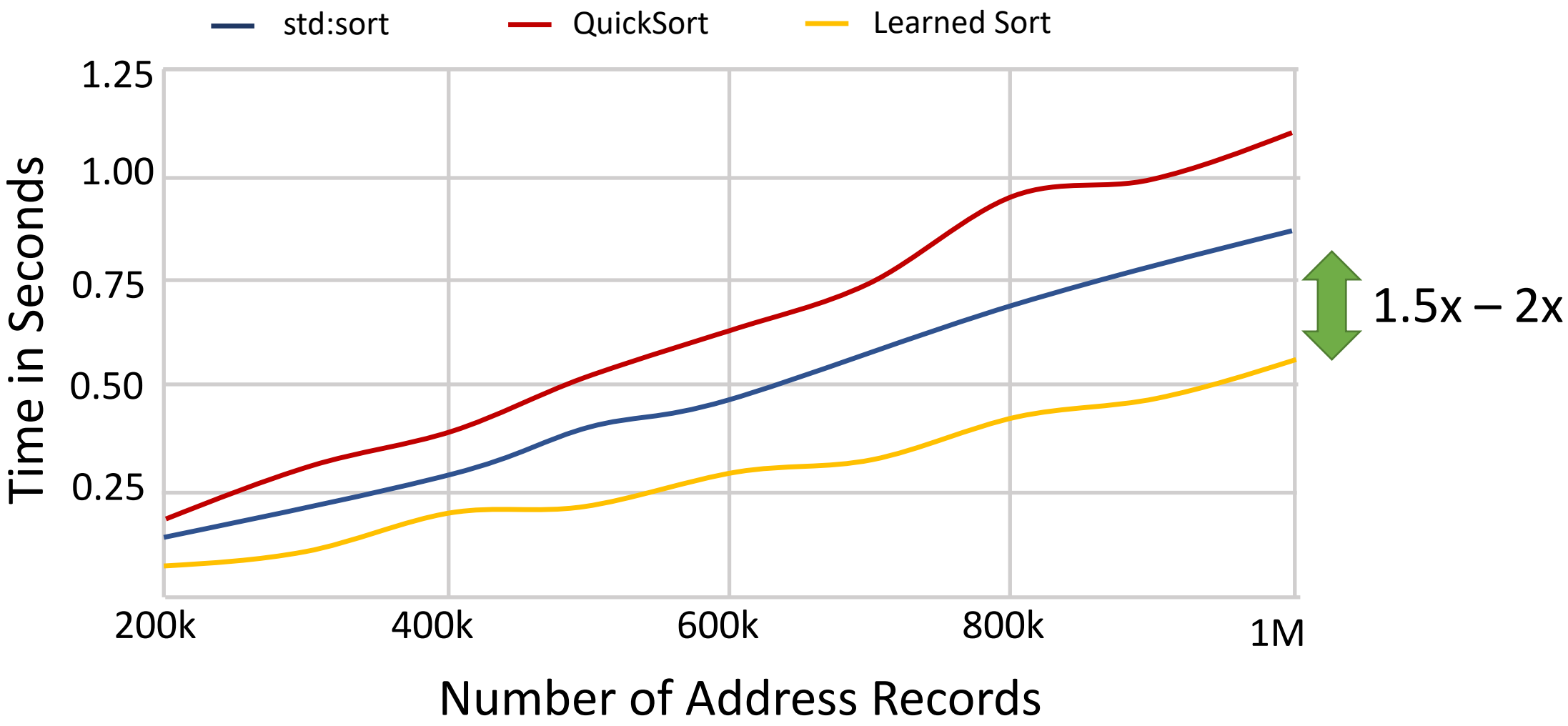
(a) CDF Model Pre-Sorts



(b) Compact & local sort

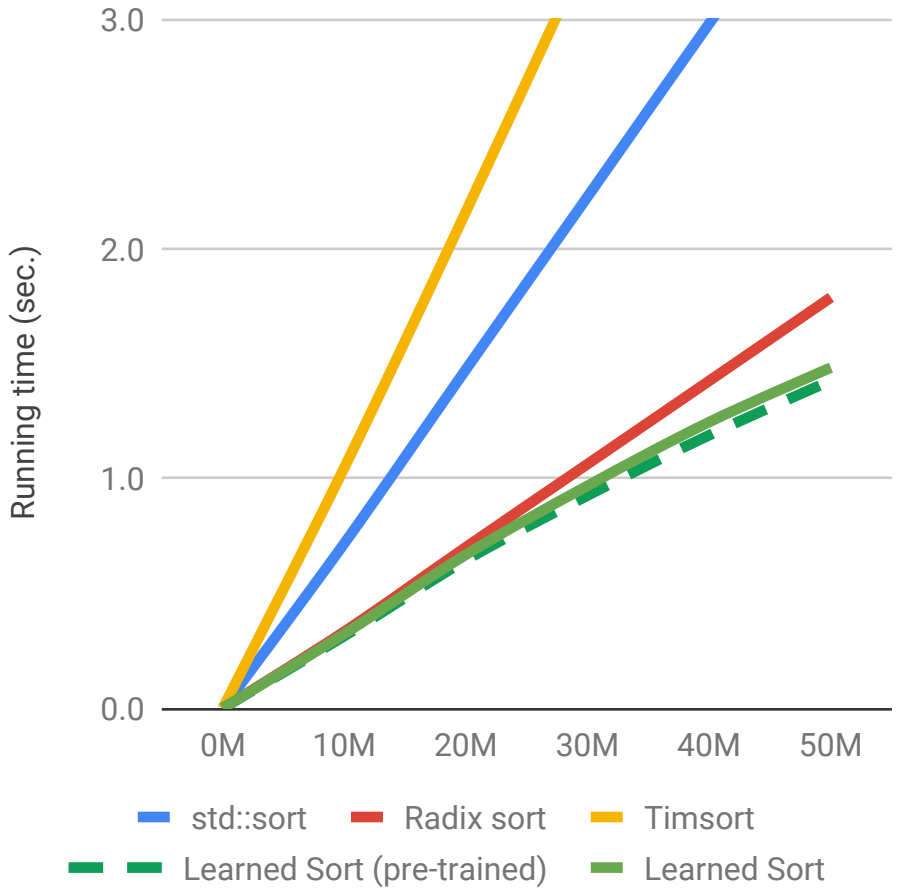


Initial Results

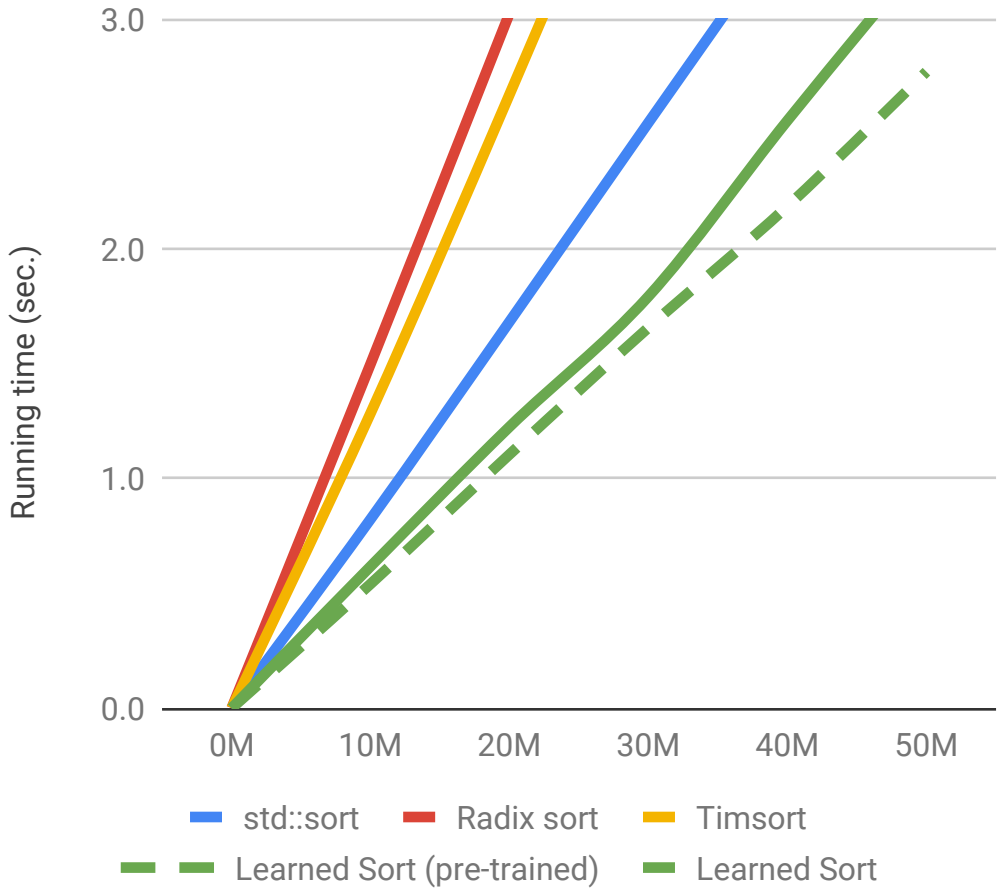


Initial Results

32-bit ints; normal distribution (μ=0, σ=1e6)

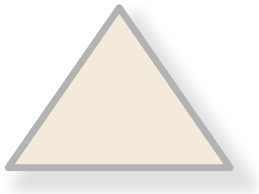


128-bit ints; normal distribution (μ=0, σ=1e6)

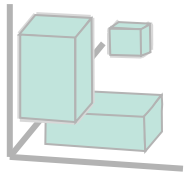


Fundamental Algorithms & Data Structures

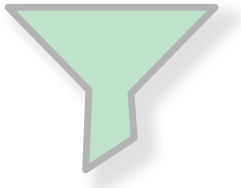
Tree



Multi-Dim Index

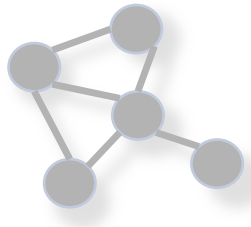


Bloom-Filter



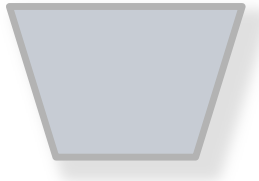
Sorting

Scheduling

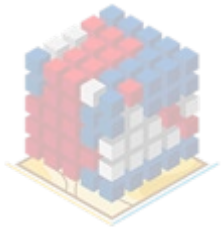


Range-Filter

Hash-Map



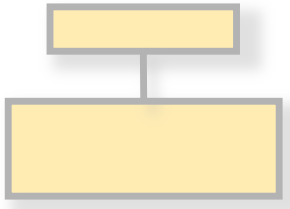
Data Cubes



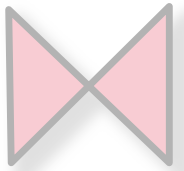
DNA-Search

SQL Query Optimizer

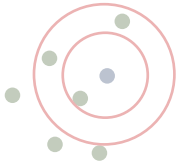
Cache Policy



Join

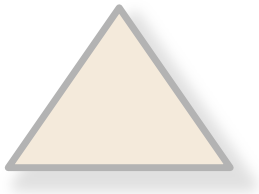


Nearest Neighbor

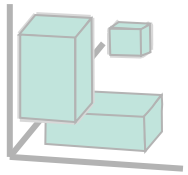


Fundamental Algorithms & Data Structures

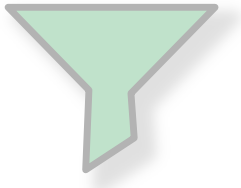
Tree



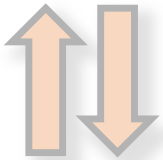
Multi-Dim Index



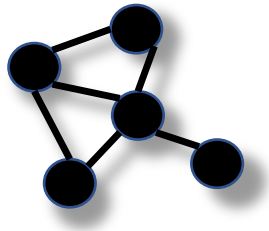
Bloom-Filter



Sorting

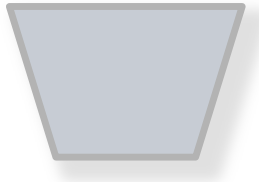


Scheduling

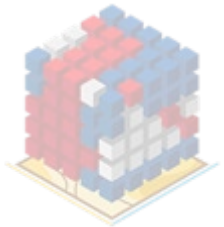


Range-Filter

Hash-Map



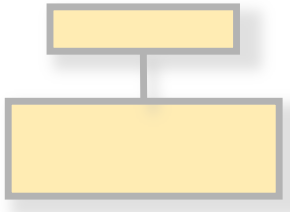
Data Cubes



DNA-Search

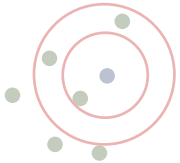
SQL Query Optimizer

Cache Policy



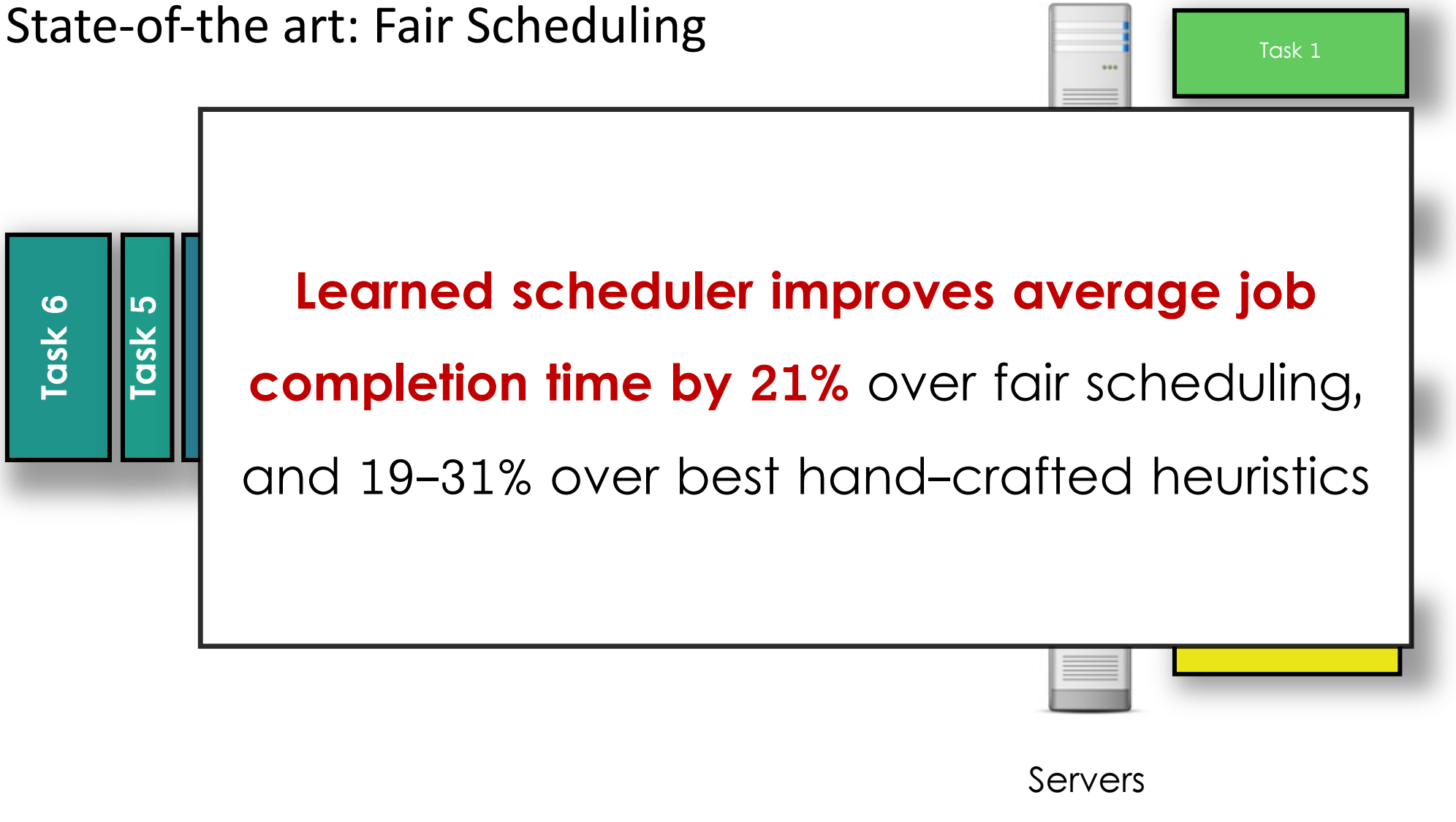
Join

Nearest Neighbor



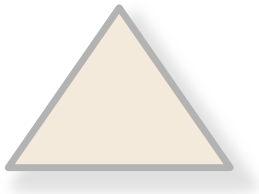
Scheduling

State-of-the art: Fair Scheduling

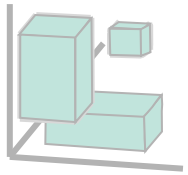


Fundamental Algorithms & Data Structures

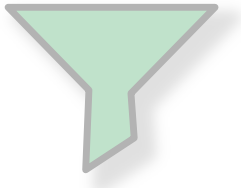
Tree



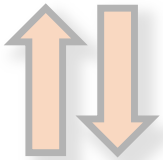
Multi-Dim Index



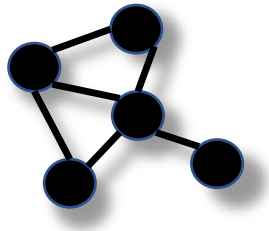
Bloom-Filter



Sorting

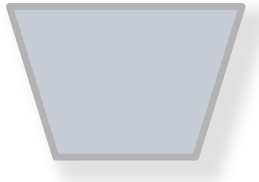


Scheduling

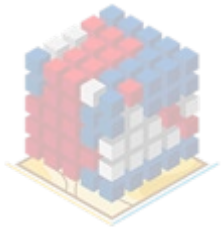


Range-Filter

Hash-Map



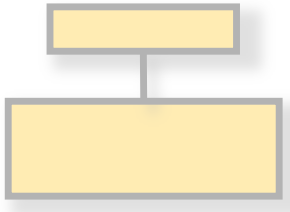
Data Cubes



DNA-Search

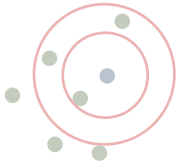
SQL Query Optimizer

Cache Policy



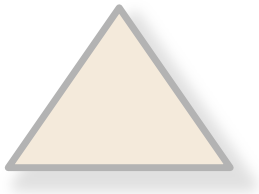
Join

Nearest Neighbor

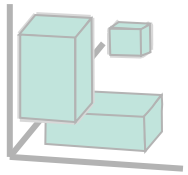


Fundamental Algorithms & Data Structures

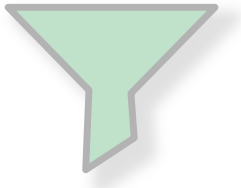
Tree



Multi-Dim Index

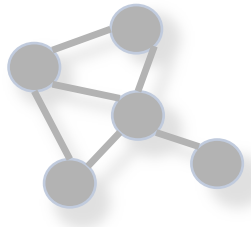


Bloom-Filter



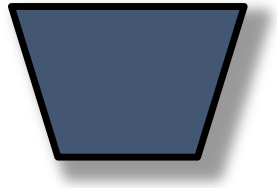
Sorting

Scheduling

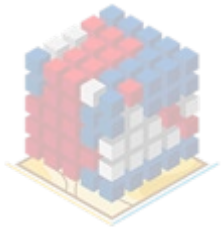


Range-Filter

Hash-Map



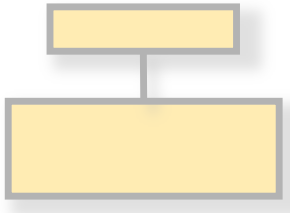
Data Cubes



DNA-Search

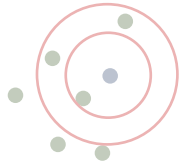
SQL Query Optimizer

Cache Policy

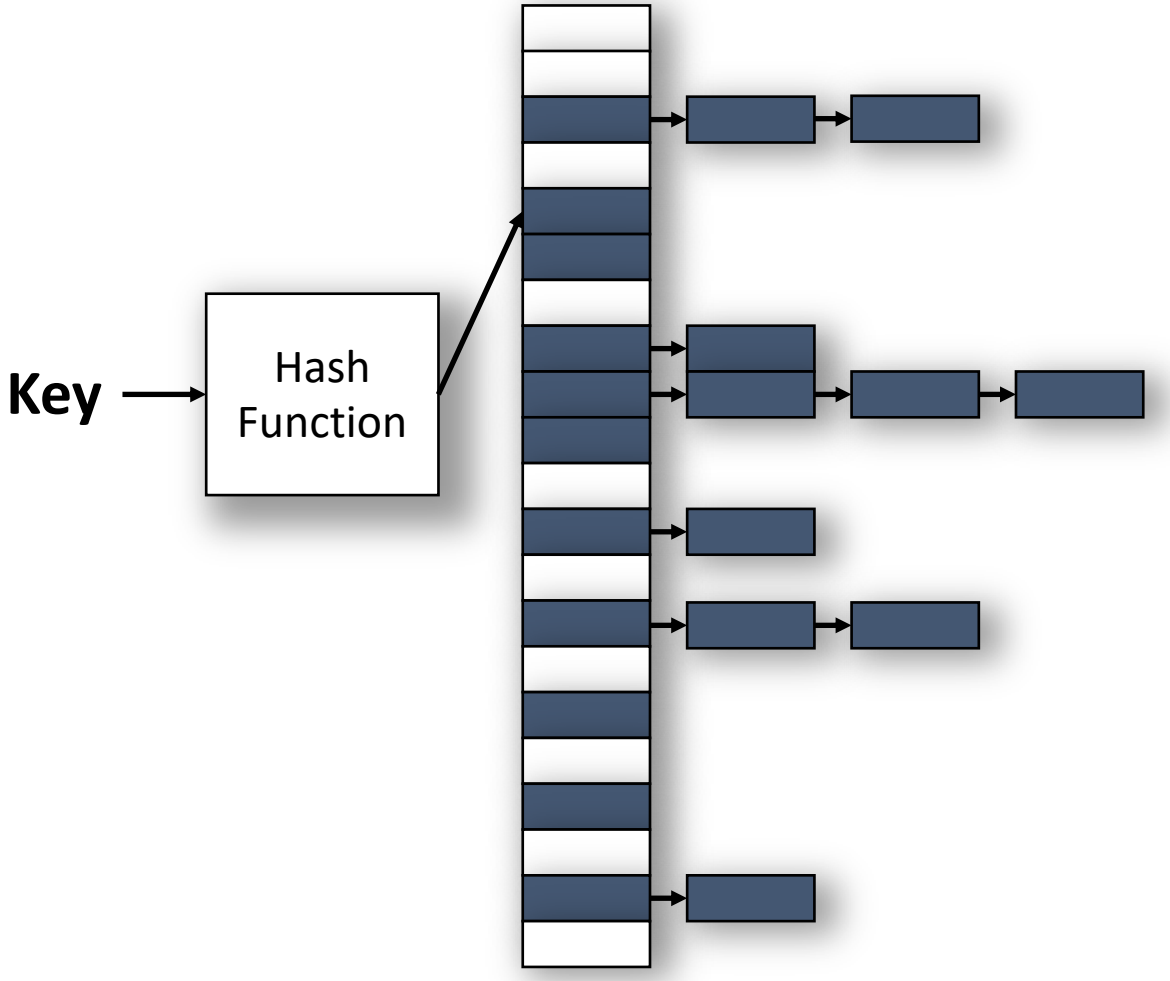


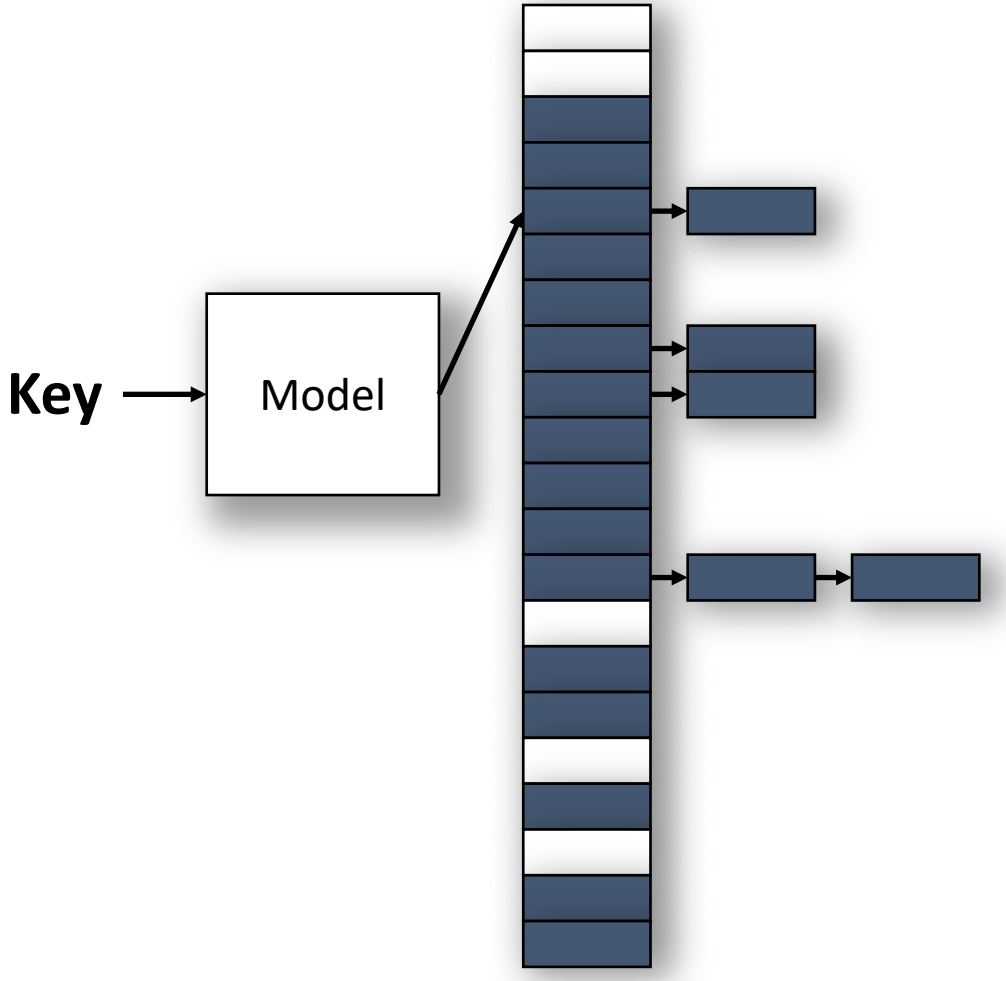
Join

Nearest Neighbor



Hash Map





Goal: Reduce Conflicts

Hash Map - Results

	% Conflicts Hash Map	% Conflicts Model	Reduction
Map Data	35.3%	07.9%	77.5%
Web Data	35.3%	24.7%	30.0%
Log Normal	35.4%	25.9%	26.7%

25% - 70% Reduction in Hash-Map Conflicts

Skip

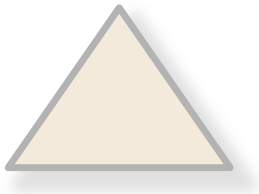

Independent
of Hash-Map
Architecture

Hash Map – Example Results

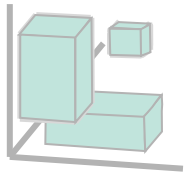
Type	Time (ns)	Utilization
Stanford AVX Cuckoo, 4 Byte value	31ns	99%
Stanford AVX Cuckoo, 20 Byte record - Standard Hash	43ns	99%
Commercial Cuckoo, 20Byte record - Standard Hash	90ns	95%
In-place chained Hash-map, 20Byte record, **learned hash functions**	35ns	100%

Fundamental Algorithms & Data Structures

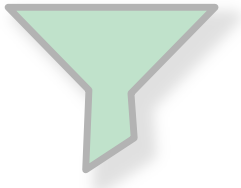
Tree



Multi-Dim Index

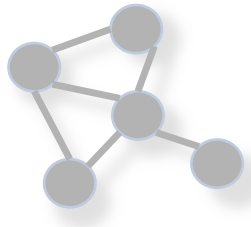


Bloom-Filter



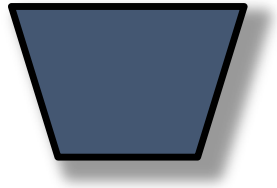
Sorting

Scheduling

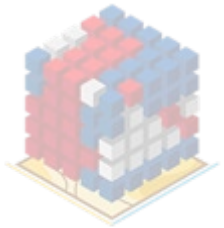


Range-Filter

Hash-Map



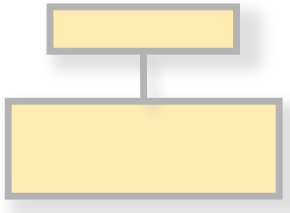
Data Cubes



DNA-Search

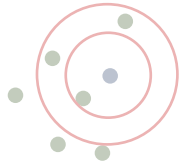
SQL Query Optimizer

Cache Policy



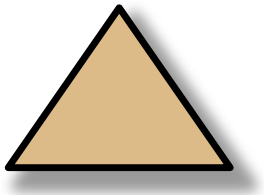
Join

Nearest Neighbor

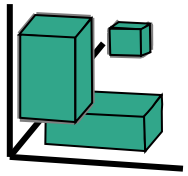


Fundamental Algorithms & Data Structures

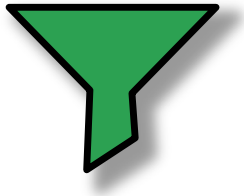
Tree



Multi-Dim Index

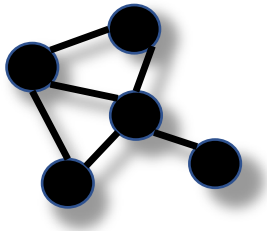


Bloom-Filter

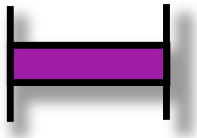


Sorting

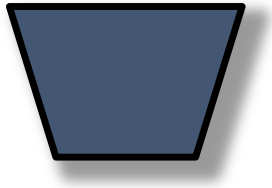
Scheduling



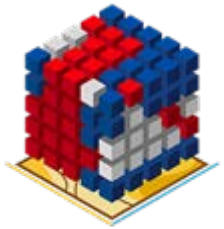
Range-Filter



Hash-Map



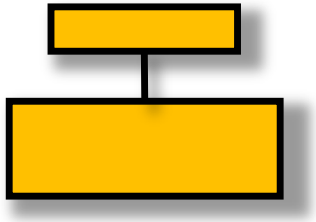
Data Cubes



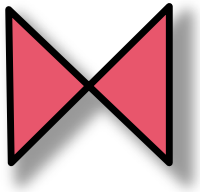
DNA-Search

SQL Query Optimizer

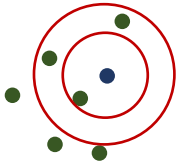
Cache Policy



Join



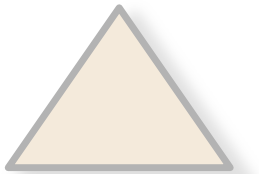
Nearest Neighbor



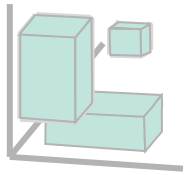
.....

Fundamental Algorithms & Data Structures

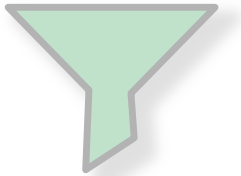
Tree



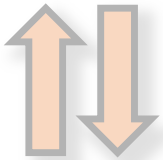
Multi-Dim Index



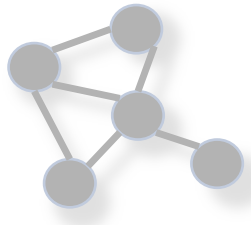
Bloom-Filter



Sorting

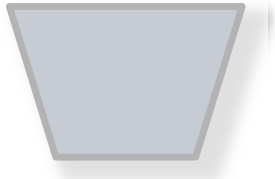


Scheduling

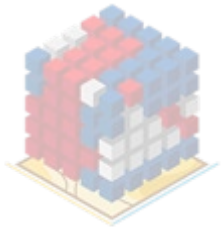


Range-Filter

Hash-Map



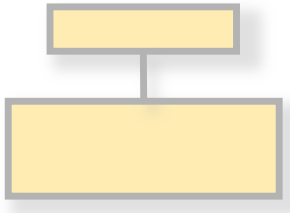
Data Cubes



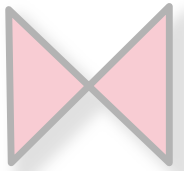
DNA-Search

SQL Query Optimizer

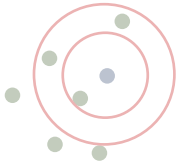
Cache Policy

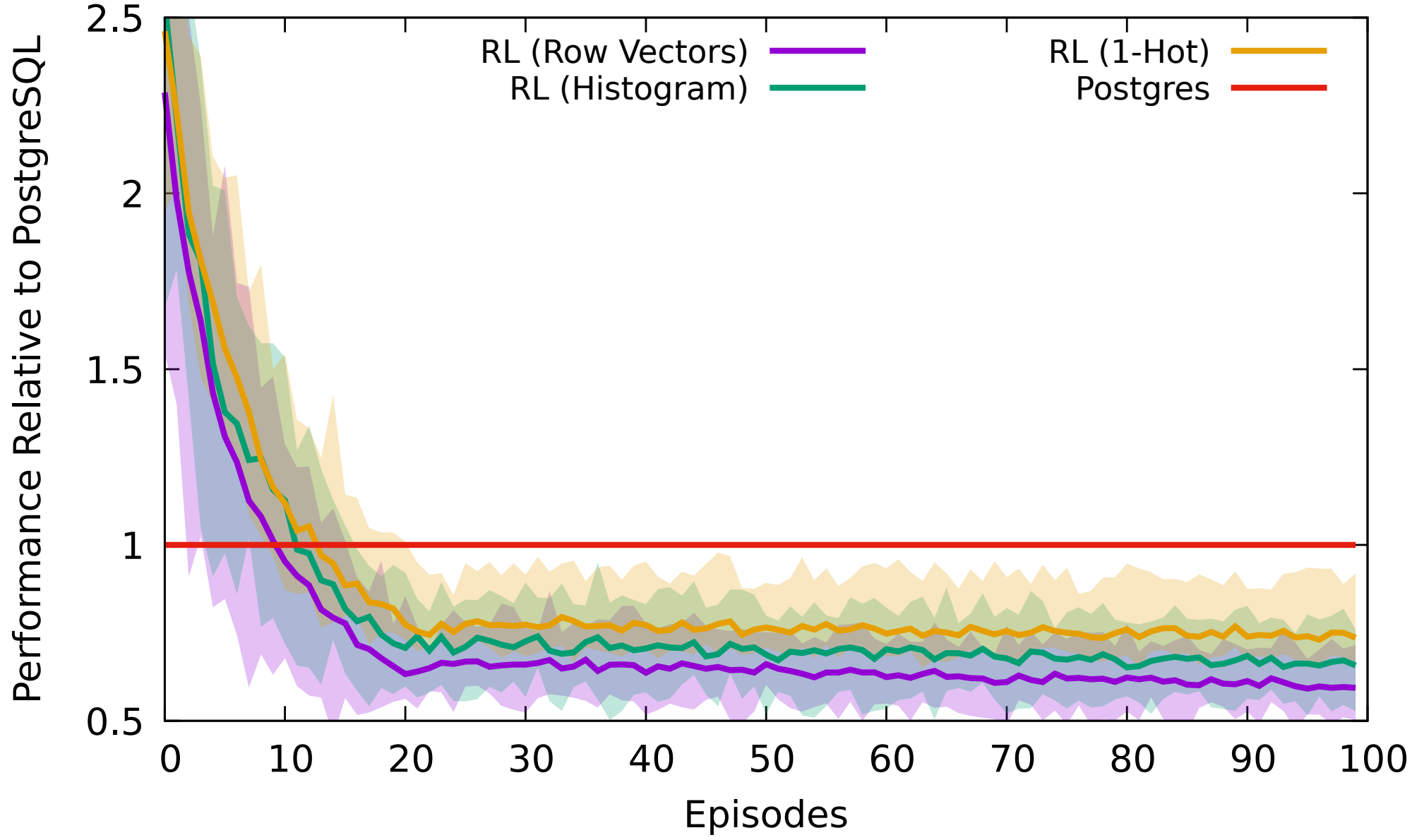


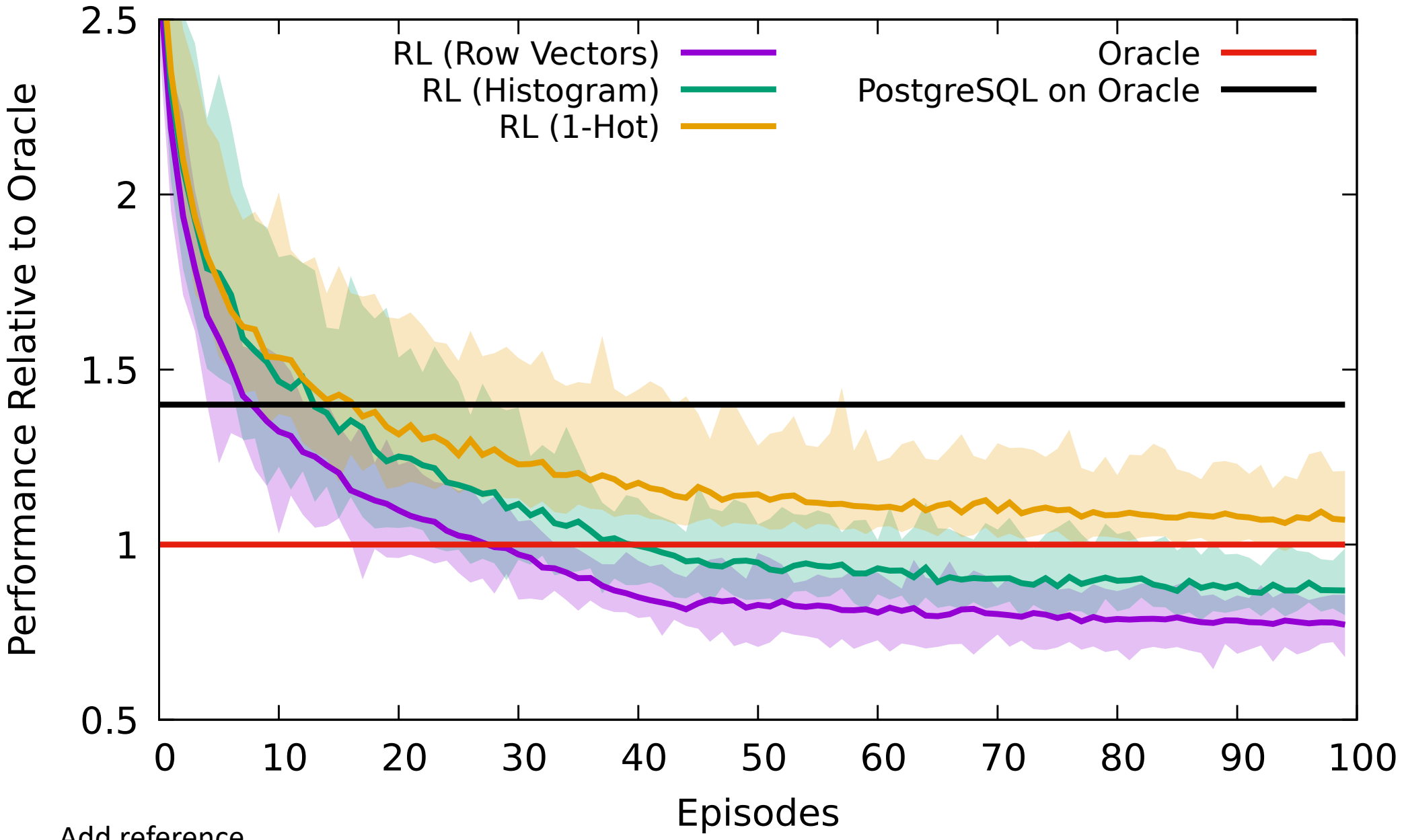
Join



Nearest Neighbor



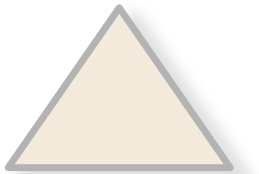

RL (Row Vectors)
RL (Histogram)
RL (1-Hot)
Postgres
Performance Relative to PostgreSQL
2.5
2
1.5
1
0.5
0
10
20
30
40
50
60
70
80
90
100
Episodes



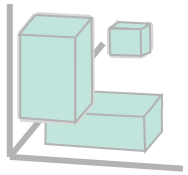
Add reference

Fundamental Algorithms & Data Structures

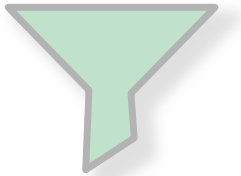
Tree



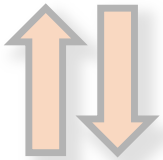
Multi-Dim Index



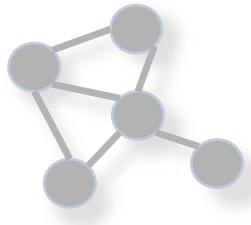
Bloom-Filter



Sorting

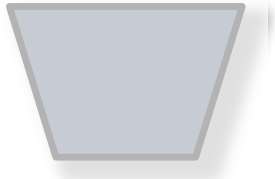


Scheduling

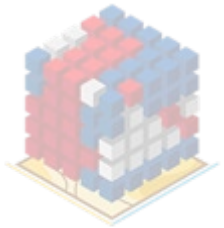


Range-Filter

Hash-Map



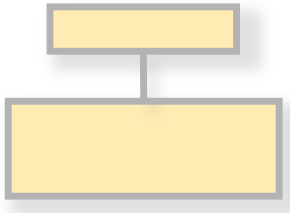
Data Cubes



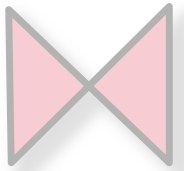
DNA-Search

SQL Query Optimizer

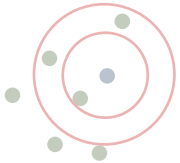
Cache Policy



Join

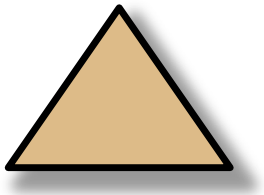


Nearest Neighbor

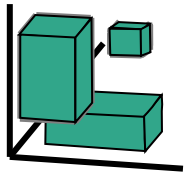


Fundamental Algorithms & Data Structures

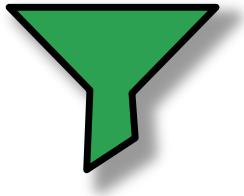
Tree



Multi-Dim Index

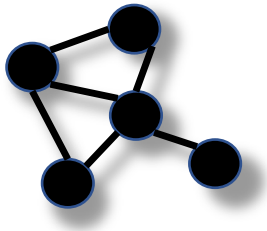


Bloom-Filter

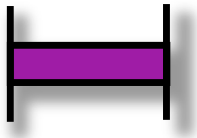


Sorting

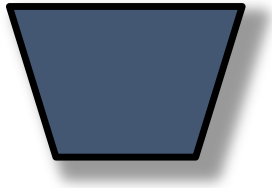
Scheduling



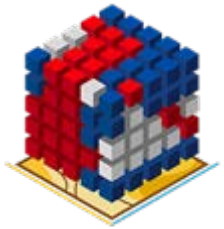
Range-Filter



Hash-Map



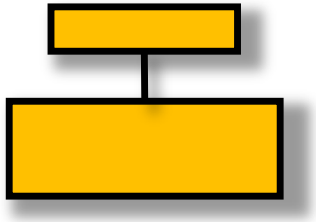
Data Cubes



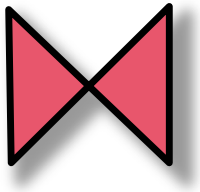
DNA-Search

SQL Query Optimizer

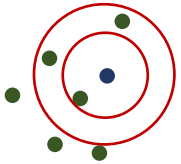
Cache Policy



Join



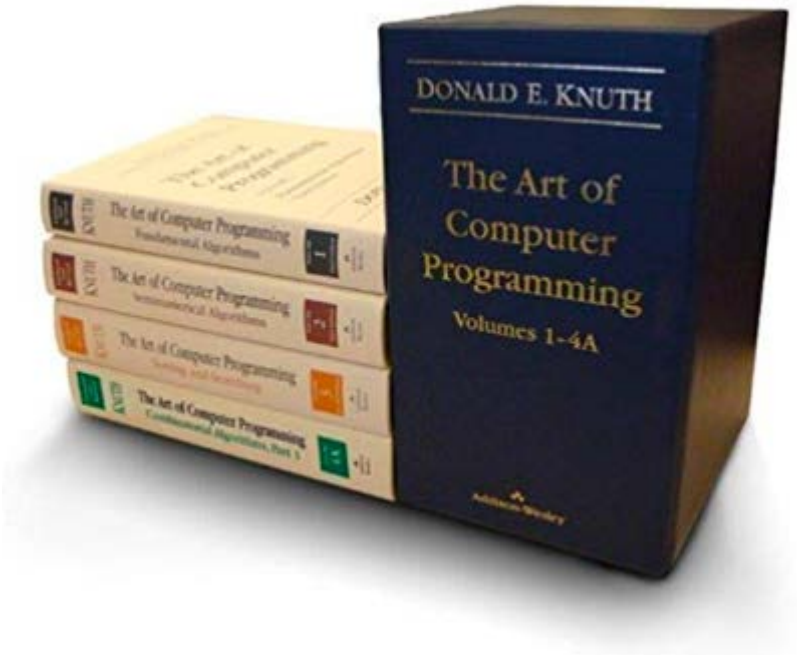
Nearest Neighbor



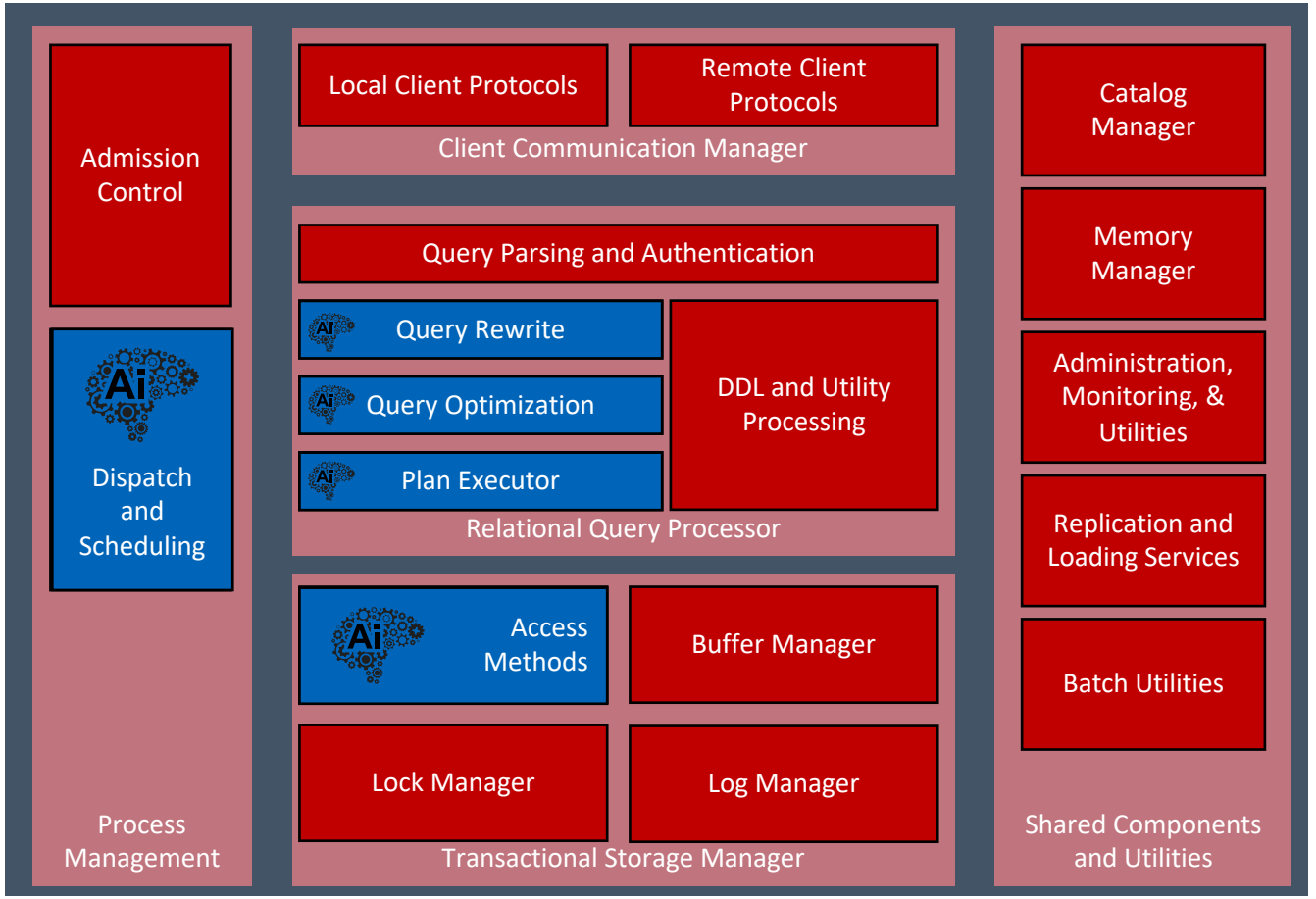
.....

We Have To ~~Rewrite The~~ Bible

Start A New

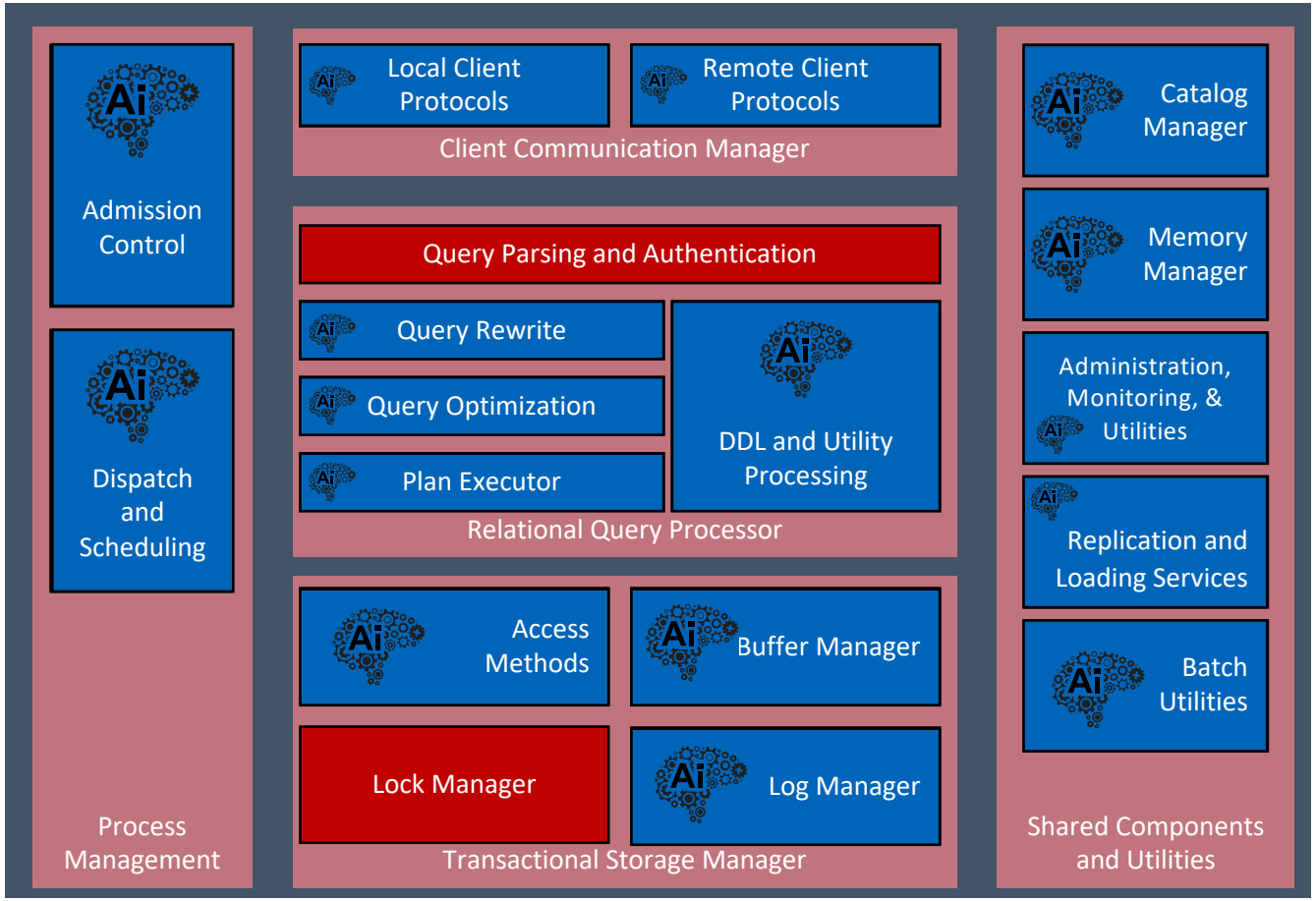


Database Architecture



Joseph M. Hellerstein, Michael Stonebraker, and James Hamilton: Architecture of a Database System. Foundations and Trends in Databases 1(2): 141-259 (2007)

Database Architecture



Joseph M. Hellerstein, Michael Stonebraker, and James Hamilton: Architecture of a Database System. Foundations and Trends in Databases 1(2): 141-259 (2007)

Instance Optimality

The best performance for every user/application

Building a system from scratch for every single user is not economical

Machine Learning makes it possible and we can leverage decades of ML research

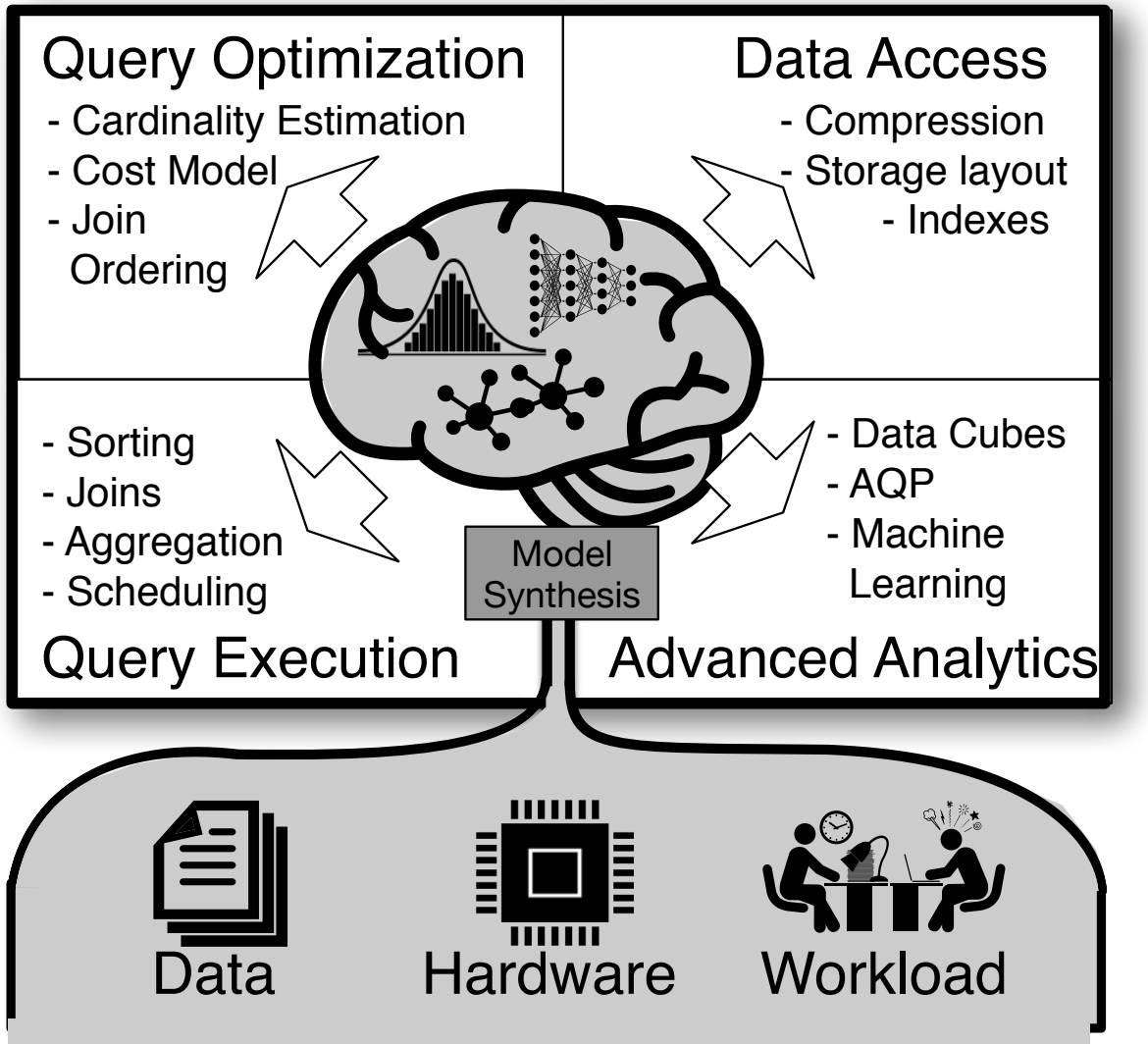
Instance Optimality

An algorithm *B* is instance optimal over a class of algorithm *A* and a database *d*, if for every choice of a ∈ A:

$$cost(B, d) \leq c \cdot cost(a, d) + c'$$

Ronald Fagin, Amnon Lotem, Moni Naor: Optimal Aggregation Algorithms for Middleware. PODS 2001

SageDB – Towards an Instance–Optimized Database System



Tim Kraska, Mohammad Alizadeh, Alex Beutel, Ed H. Chi, Ani Kristo, Guillaume Leclerc, Samuel Madden, Hongzi Mao, Vikram Nathan: **SageDB: A Learned Database System**. CIDR 2019

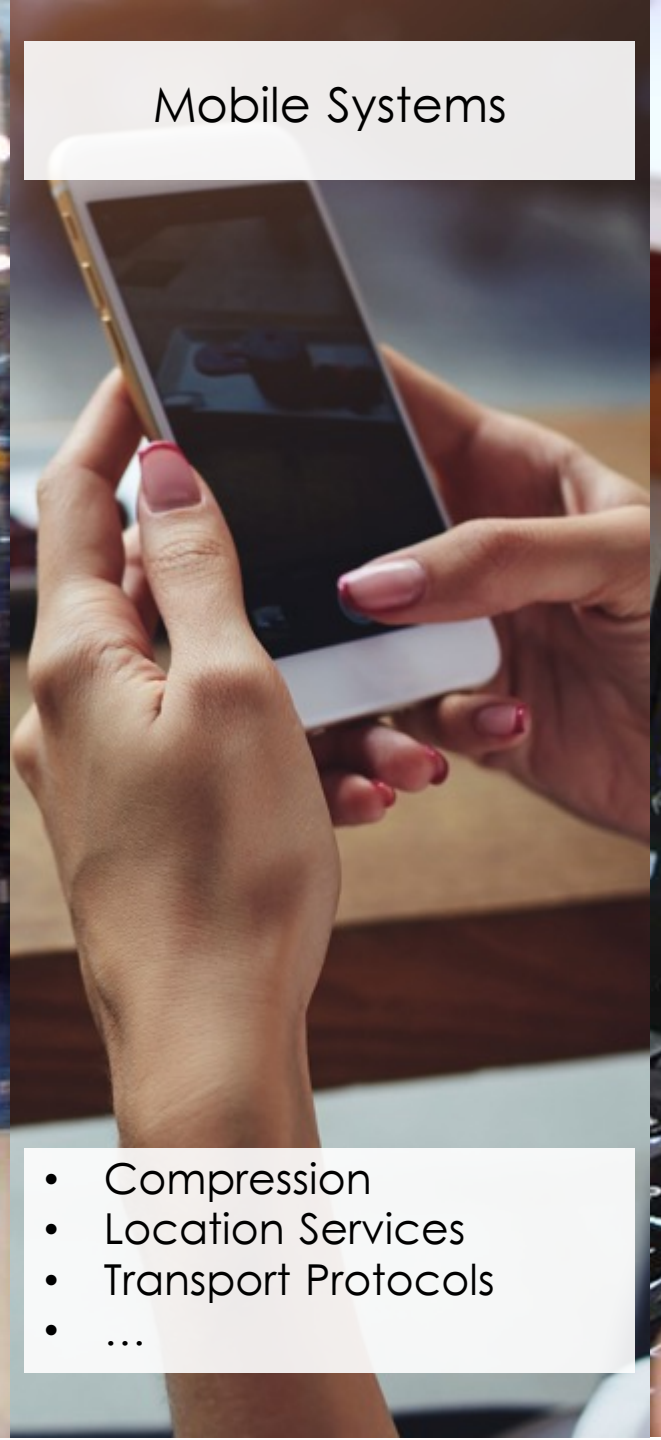
Data Science / Database Systems

- Index Structures
- Storage Manager
- Scheduling
- …

Network Systems

- Router Table
- Rule Engine
- Bandwidth Optimization
- …

Mobile Systems

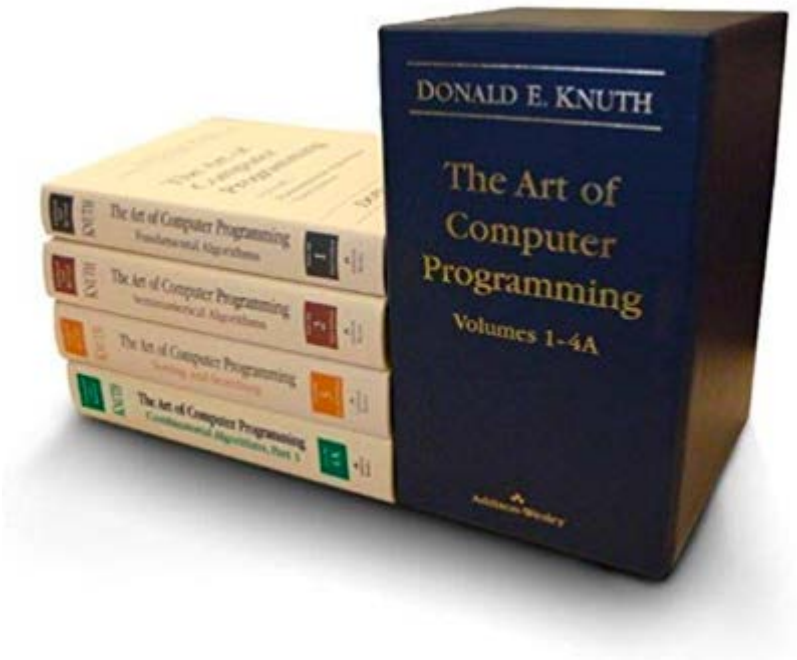


- Compression
- Location Services
- Transport Protocols
- …

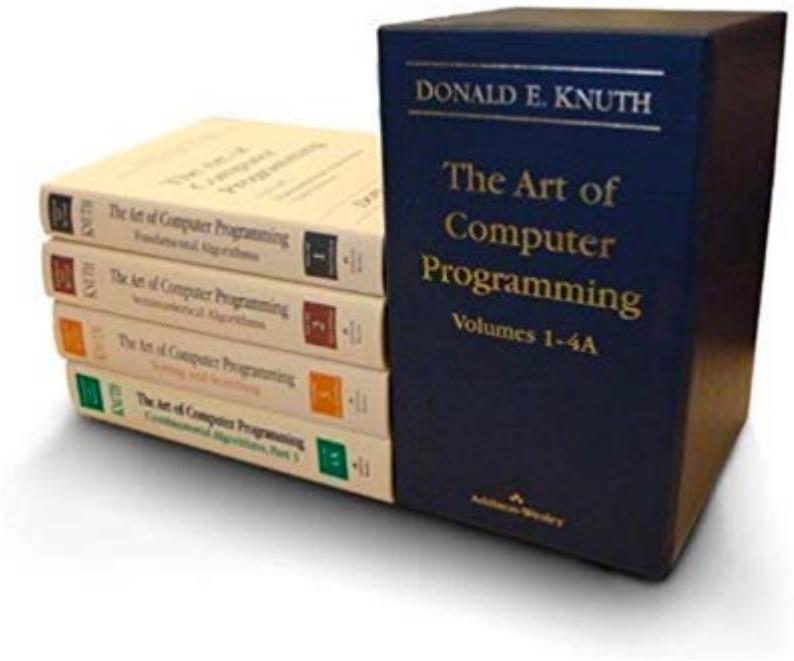
Applications

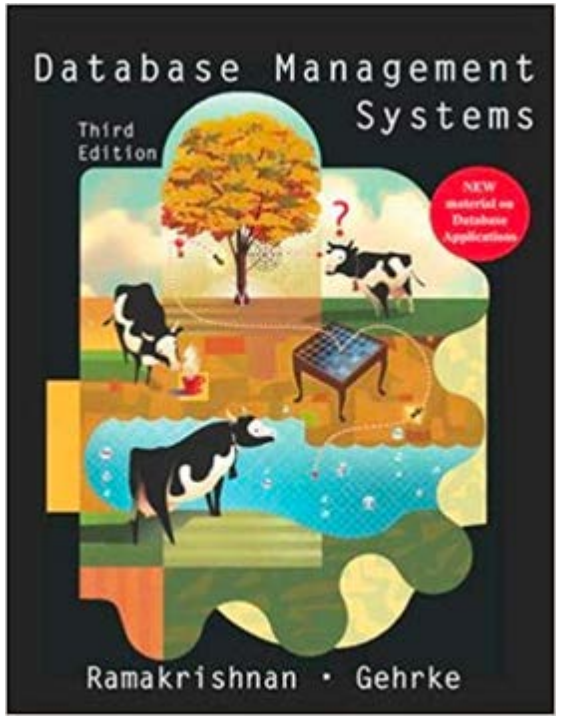
- Data-driven development
- Intelligent Assistants
- …

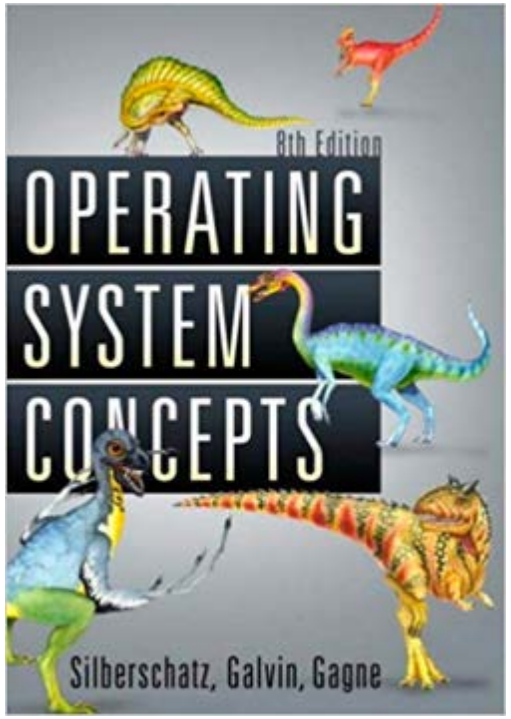
We Need To Start A New Bible



We Need To Start New Bibles







...

We Have To Start A New Bible

- Maria-Florina Balcan, Travis Dick, Tuomas Sandholm, and Ellen Vitercik: Learning to branch. ICML, 2018.
- Thodoris Lykouris and Sergei Vassilvitskii: Competitive caching with machine learned advice. ICML, 2018.
- Tim Kraska, Alex Beutel, Ed H Chi, Jeffrey Dean, and Neoklis Polyzotis. The case for learned index structures. SIGMOD, 2018.
- Michael Mitzenmacher. A model for learned bloom filters and optimizing by sandwiching. NeurIPS, 2018.
- Michael Mitzenmacher. Load balancing. unpublished
- Manish Purohit, Zoya Svitkina and Ravi Kumar. Improving Online Algorithms via ML Predictions. NeurIPS, 2018.
- Luca Baldassarre, Yen-Huan Li, Jonathan Scarlett, Baran G¨ozc¨u, Ilija Bogunovic, and Volkan Cevher. Learning-based compressive subsampling. J-STSP, 2016.
- Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed sensing using generative models. ICML, 2017.
- Ali Mousavi, Ankit B Patel, and Richard G Baraniuk. A deep learning approach to structured signal recovery. Allerton, 2015.
- Ali Mousavi, Gautam Dasarathy, Richard G. Baraniuk. A Data-Driven and Distributed Approach to Sparse Signal Representation and Recovery. ICLR 2019
- Hanjun Dai, Elias Khalil, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. NeurIPS, 2017.
- H Mao, R Netravali, M Alizadeh. Neural adaptive video streaming with pensieve. SIGCOMM, 2017
- H Mao, M Schwarzkopf, SB Venkatakrishnan, Z Meng, M Alizadeh. Learning Scheduling Algorithms for Data Processing Clusters. arXiv preprint arXiv:1810.01963
- Hyeji Kim, Yihan Jiang, Sreeram Kannan, Sewoong Oh, Pramod Viswanath. Deepcode: Feedback Codes via Deep Learning. NeurIPS, 2018
- Saeed Amizadeh, Sergiy Matusevych, Markus Weimer. Learning To Solve Circuit-SAT: An Unsupervised Differentiable Approach. ICLR, 2019
- Daniel Selsam, Matthew Lamm, Benedikt Bünz, Percy Liang, Leonardo de Moura, David L. Dill. Learning a SAT Solver from Single-Bit Supervision. ICLR, 2019
- Weiwei Kong, Christopher Liaw, Aranyak Mehta, D. Sivakumar. A new dog learns old tricks: RL finds classic optimization algorithms. ICLR, 2019
- Jun-Ting Hsieh, Shengjia Zhao, Stephan Eismann, Lucia Mirabella, Stefano Ermon. Learning Neural PDE Solvers with Convergence Guarantees. ICLR 2019
- Morteza Mardani, Qingyun Sun, David Donoho, Vardan Papyan, Hatef Monajemi, Shreyas Vasanawala, John Pauly. Neural Proximal Gradient Descent for Compressive Imaging. NeurIPS 2018
- Andrés Muñoz Medina and Sergei Vassilvitskii. Revenue optimization with approximate bid predictions. NIPS, 2017.
- MF Balcan, T Dick, C White. Data-Driven Clustering via Parameterized Lloyd's Families. NeurIPS, 2018
- Mislav Balunovic, Pavol Bielik, Martin Vechev. Learning to Solve SMT Formulas. NeurIPS, 2018
- Marcelo O. R. Prates, Pedro H. C. Avelar, Henrique Lemos, Luis Lamb, Moshe Vardi. Learning to Solve NP-Complete Problems - A Graph Neural Network for Decision TSP
- Jimenez, Daniel A. and Lin, Calvin. Neural Methods for Dynamic Branch Prediction, TOCS 2002
- Milad Hashemi, Kevin Swersky, Jamie A. Smith, Grant Ayers, Heiner Litz, Jichuan Chang, Christos Kozyrakis, Parthasarathy Ranganathan: Learning Memory Access Patterns. arXiv:1803.02329v1, 2018
- W. Xiang and H. Zhang and R. Cui and X. Chu and K. Li, W. Zhou. Pavo: A RNN-Based Learned Inverted Index, Supervised or Unsupervised?. IEEE Access, 2019
- Harrie Oosterhuis, J. Shane Culpepper, Maarten de Rijke. The Potential of Learned Index Structures for Index Compression. arXiv:1811.06678v1, 2018
- Stephen Macke, Alex Beutel, Tim Kraska, Maheswaran Sathiamoorthy, Derek Zhiyuan Cheng, and Ed H. Chi. Lifting the Curse of Multidimensional Data with Learned Existence Indexes. Workshop on ML for Systems at NeurIPS, 2018
- Zhe Wang, Dylan Cashman, Mingwei Li, Jixian Li, Matthew Berger, Joshua A. Levine, Remco Chang, Carlos Scheidegger: NeuralCubes: Deep Representations for Visual Data Exploration, CoRR abs/1808.08983, 2018

…

4.5 Ways to Instance Optimized Algorithms and Data Structures

1. **Configure/synthesize** traditional algorithm using a model

4.5 Ways to Instance Optimized Algorithms and Data Structures

1. **Configure/synthesize** traditional algorithm using a model
2. **CDF**: empirical CDF model of the data

4.5 Ways to Instance Optimized Algorithms and Data Structures

1. **Configure/synthesize** traditional algorithm using a model
2. **CDF**: empirical CDF model of the data
3. **Oracle**: prediction model

4.5 Ways to Instance Optimized Algorithms and Data Structures

1. **Configure/synthesize** traditional algorithm using a model
2. **CDF**: empirical CDF model of the data
3. **Oracle**: prediction model
4. **Full-Model:** learning the entire algorithm/data structure

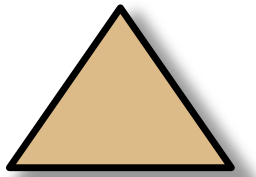
4.5 Ways to Instance Optimized Algorithms and Data Structures

1. **Configure/synthesize** traditional algorithm using a model
2. **CDF**: empirical CDF model of the data
3. **Oracle**: prediction model
4. **Full-Model:** learning the entire algorithm/data structure
5. **Hybrids:** Any combination of the above 4

Fundamental Algorithms & Data Structures

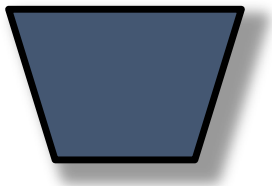
Our initial paper

Tree

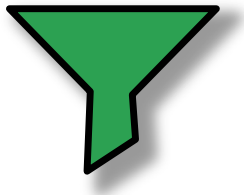


CDF-based

Hash-Map



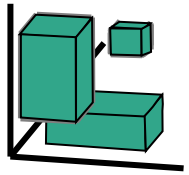
Bloom-Filter



Oracle -based

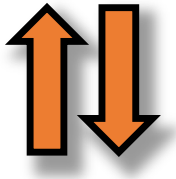
Work in Progress

Multi-Dim Index



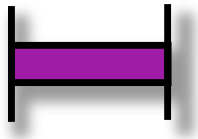
CDF-based

Sorting



CDF-based

Range-Filter



Oracle-based

DNA-Search

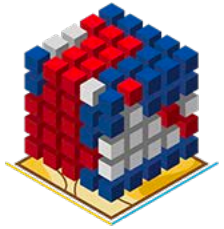
CDF-based

CDF-Synth.

F(x)

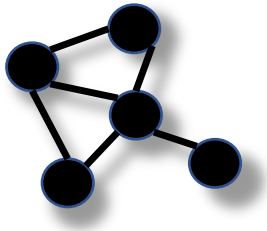
Work in Progress

Data Cubes



CDF-based/Full

Scheduling

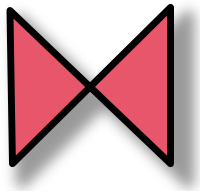


Full

SQL Query Optimizer

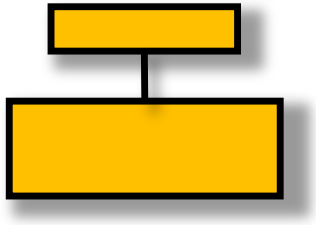
Full

Join



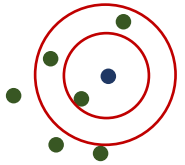
CDF-based

Cache Policy



Full

Nearest Neighbor



Full

.....

Data System for AI Lab DSAIL@CSAIL

Research Area

Data Systems for AI **AI for Data Systems**

System Faculty



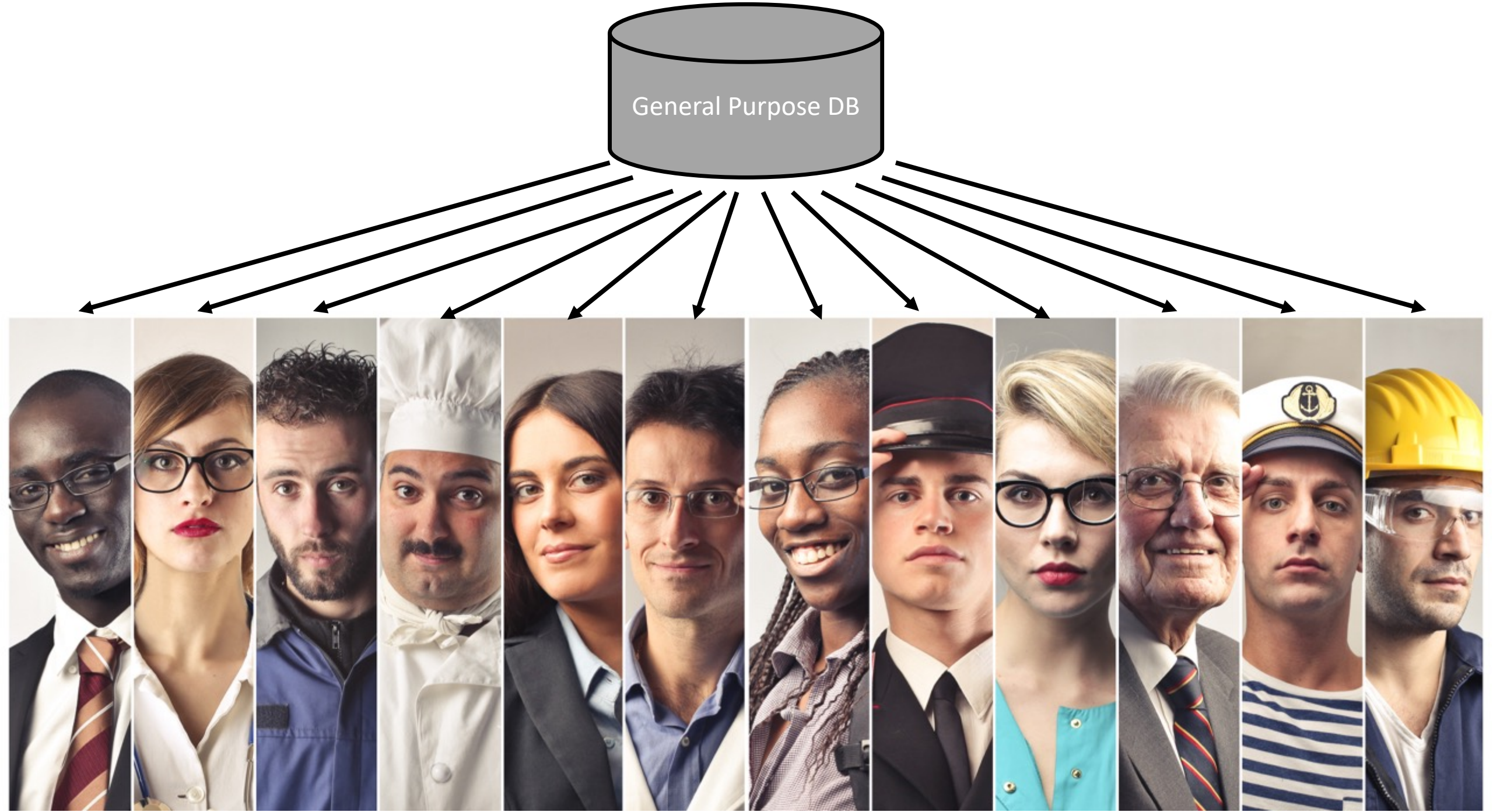

ML Faculty




Founding Sponsors

Google Microsoft intel

Conclusion


General Purpose DB

Instance Optimized

Customized Systems

Adaptation to data and workload

4.5 Ways to Instance Optimized Algorithms and Data Structures

1. **Configure/synthesize** traditional algorithm using a model
2. **CDF**: empirical CDF model of the data
3. **Oracle**: prediction model
4. **Full-Model:** learning the entire algorithm/data structure
5. **Hybrids:** Any combination of the above 4

Can Change the Complexity Class

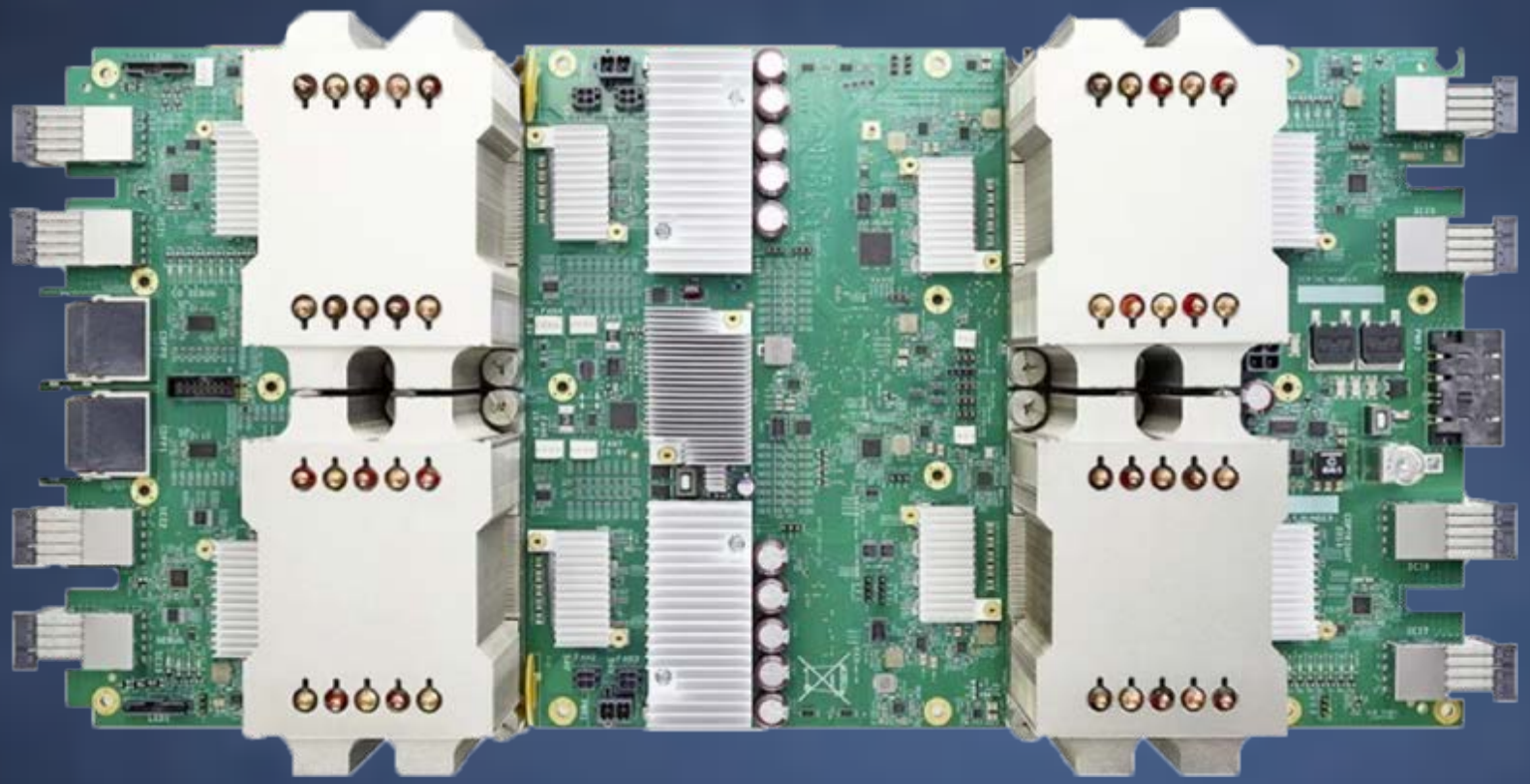
id	1000	1001	1002	1003	1004	1005	1006	1007	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	…

```
data_array[id - 1000]
```

date	2017-01-01	2017-01-02	2017-01-02	2017-01-03	2017-01-03	2017-01-04	2017-01-04	2017-01-05	2017-01-05	2017-01-06	2017-01-07	2017-01-09	2017-01-09	2017-01-09	2017-01-10	2017-01-10	2017-01-11	2017-01-12	2017-01-13	2017-01-14	2017-01-15	2017-01-16	2017-01-17	2017-01-18	2017-01-19	2017-01-20	2017-01-21	2017-01-22	2017-01-22	2017-01-22	2017-01-23	2017-01-24	2017-01-24	2017-01-26	2017-01-26	2017-01-26	2017-01-28	2017-01-29	2017-01-30	2017-01-30	2017-01-30	2017-01-31	2017-01-31	2017-02-01	2017-02-01	2017-02-02	2017-02-04	2017-02-05	2017-02-05	2017-02-06	2017-02-06	2017-02-06	2017-02-07	2017-02-07	2017-02-08	2017-02-08	2017-02-08	2017-02-09	2017-02-10	2017-02-10	2017-02-11	2017-02-12	2017-02-13	2017-02-13	2017-02-14	2017-02-14	2017-02-15	…

```
data_array[model(date)]
```

Big potential for GPUs/FPGAs/TPUs

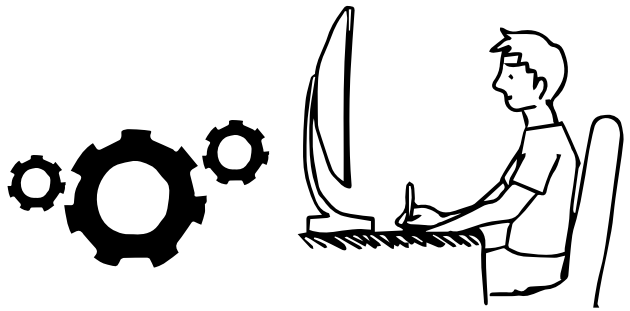


Open Challenges

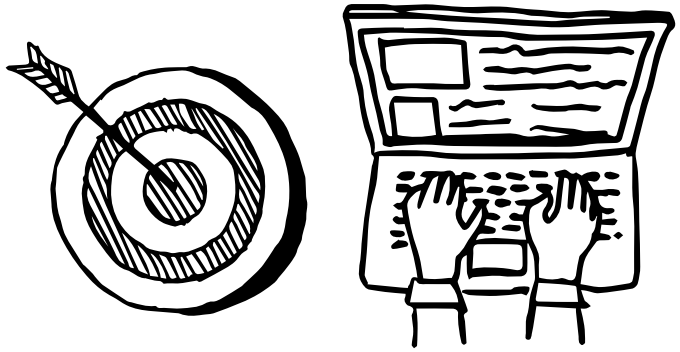
- Development
- Benchmarks
- Type of Models
- Other algorithms
- Adaptation to changes
- Inserts/Updates
- Complexity Analysis and Robustness Guarantees
- Predictive Modeling
- ….

How We Do Software Development Has To Change Fundamentally

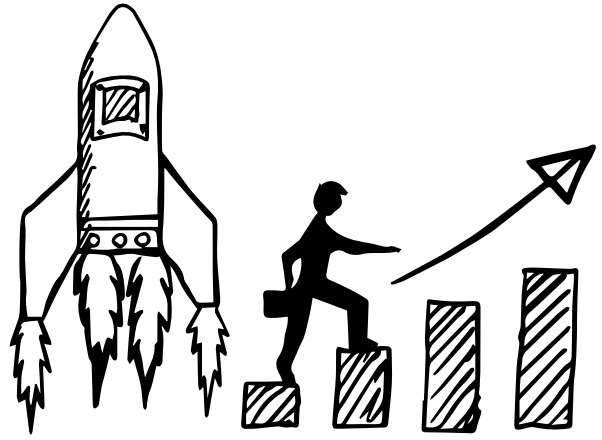
How We Do Software Development Has To Change Fundamentally



Implementation



Testing



Evolution

Open Challenges

- Development
- Benchmarks
- Type of Models
- Other algorithms
- Adaptation to changes
- Inserts/Updates
- Complexity Analysis and Robustness Guarantees
- Predictive Modeling
- ….

Learned Algorithms and Data Structures

- ML-Enhanced Algorithms, Data Structures, Systems Components Are On The Rise
- They Provide A New Way To Build Highly Customized Systems (i.e., Instance-Optimized Systems)
- Many Data-Related Problems Can Be Reduced to Learning A CDF model
- We Found That RMIs (Mixture Of Experts With Very Simple Models) Are A Very powerful Building Block
- What Makes The Approach So Powerful Is The Use Of Continuous Functions And Easy To Vectorized Math Operations (the HW has changed!!!)

DSAIL
Data Systems and AI Lab

Microsoft Google intel

LEARNED AND **SELF-DESIGNING** DATA STRUCTURES

DASlab
@ Harvard SEAS

MIT DSAIL
Data Systems and AI Lab

Stratos Idreos & Tim Kraska