Learning to Rank in Theory and Practice

From Gradient Boosting to Neural Networks and Unbiased Learning

Tutorial @ ACM SIGIR 2019

http://ltr-tutorial-sigir19.isti.cnr.it/

Session I: Efficiency/Effectiveness Trade-offs

Claudio Lucchese

Ca' Foscari University of Venice

Venice, Italy

Franco Maria Nardini

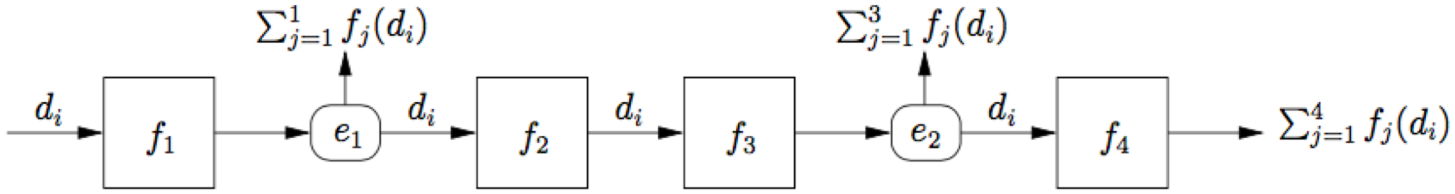
HPC Lab, ISTI-CNR

Pisa, Italy

Approximate Score Computation and Efficient Cascades

Early Exit Optimizations for Additive Machine Learned Ranking Systems [CZC+10]

- Why short-circuiting the scoring process in additive ensembles
  - for each query, few highly relevant documents and many irrelevant ones
  - most users view only the first few result pages
- Cambazoglu *et al.* introduce additive ensembles with early exits



- Four techniques
  - early exits using {Score, Capacity, Rank, Proximity} thresholds
- Evaluation on a state-of-the-art ML platform with GBRT
- With EPT, up to four times faster without loss in quality

[CZC+10] B. B. Cambazoglu, H. Zaragoza, O. Chapelle, J. Chen, C. Liao, Z. Zheng, and J. Degenhardt. *Early exit optimizations for additive machine learned ranking systems*. In Proc. ACM WSDM, 2010.

Efficient Cost-Aware Cascade Ranking in Multi-Stage Retrieval [CGBC17b]

- Cascade ranking model as a sequence of LtR models (*stages*)
  - ascending order of model complexity, only a fraction of documents in each stage advance to the next stage
- Chen *et al.* define the problem on how best to balance feature importance and feature costs in multi-stage cascade ranking models
  - three cost-aware heuristics to assign features to each stage
  - cost-aware L_1 regularization to learn each stage
    - Automatic feature selection while jointly optimize efficiency and effectiveness
- Experiments
  - Yahoo! Learning to Rank, gov
- Comparisons against [WLM11b]

[CGBC17b] R. Chen, L. Gallagher, R. Blanco, and J. S. Culpepper. *Efficient cost-aware cascade ranking in multi-stage retrieval*. In Proc. ACM SIGIR, 2017.

Efficient Cost-Aware Cascade Ranking in Multi-Stage Retrieval

System	ERR@k			NDCG@k			P@k			Cost
	@5	@10	@20	@5	@10	@20	@5	@10	@20	
Ground Truth Models										
GBDT-BL	**0.4605**	**0.4751**	**0.4789**	**0.7448**	**0.7872**	**0.8279**	0.8323	**0.7577**	**0.5967**	15988
GBRT-BL	0.4598	0.4744	0.4782	0.7420	0.7852	0.8264	0.8322	0.7562	0.5962	15876
LambdaMART-BL	0.4526	0.4674	0.4712	0.7314	0.7768	0.8203	**0.8330**	0.7564	0.5964	15856
Cascade Models (including Baseline) [a]										
WLM-BL	0.3679	0.3876	0.3933	0.5886	0.6506	0.7088	0.7832	0.7171	0.5673	99
LM-C3-C	0.3950	0.4127	0.4175	0.6461	0.7067	0.7638	0.8086**	0.7364**	0.5856**	1871
LM-C3-E	0.3871	0.4039	0.4089	0.6503	0.7033	0.7618	0.8192**	0.7413**	0.5885**	1580
LM-C3-F	0.3876	0.4047	0.4093	0.6541	0.7113	0.7666	**0.8226****	**0.7483****	**0.5915****	5278
GBDT-C3-C	0.4191	0.4357	0.4405	0.6535	0.7100	0.7631	0.7878*	0.7245**	0.5781**	1760
GBDT-C3-E	0.4264	0.4419	0.4466	0.6721	0.7180	0.7703	0.7942**	0.7241**	0.5778**	1535
GBDT-C3-F	0.4178	0.4350	0.4395	0.6554	0.7163	0.7672	0.7866	0.7310**	0.5819**	4953
GBRT-C3-C	0.4025	0.4203	0.4254	0.6304	0.6931	0.7488	0.7743	0.7168	0.5737**	1760
GBRT-C3-E	0.4100	0.4260	0.4313	0.6380	0.6867	0.7431	0.7697	0.7009	0.5637**	1535
GBRT-C3-F	0.4158	0.4332	0.4378	0.6479	0.7094	0.7612	0.7862	0.7294**	0.5802**	4949
LambdaMART-C3-C	0.4163	0.4332	0.4379	0.6577	0.7145	0.7673	0.7994**	0.7328**	0.5820**	1760
LambdaMART-C3-E	0.4183	0.4346	0.4394	0.6629	0.7133	0.7671	0.7968**	0.7268**	0.5786**	1535
LambdaMART-C3-F	**0.4353**	**0.4513**	**0.4557**	**0.6847**	**0.7354**	**0.7851**	0.8060**	0.7379**	0.5847**	4929

Joint Optimization of Cascade Ranking Models [GCBC19]

- Cost-aware approaches for LtR exist but no generalization to cascade
  - Stage-wise ranker trained independently while strict dependency exists
- Novel method for learning a globally optimized cascade architecture
  - Backpropagation on top of a cost efficient gradient boosting [PDHN17]
- Three types of cascades
  - Independent chaining, Full chaining, Weak chaining
- Experiments
  - Yahoo! Learning to Rank and MSLR-WEB10K
- Globally learning the cascade achieves much better trade-offs between efficiency and effectiveness than previous approaches

[CGBC19] L. Gallagher, R. Chen, R. Blanco, J. S. Culpepper, *Joint Optimization of Cascade Ranking Models*. In Proc. ACM WSDM 2019.

Further Reading

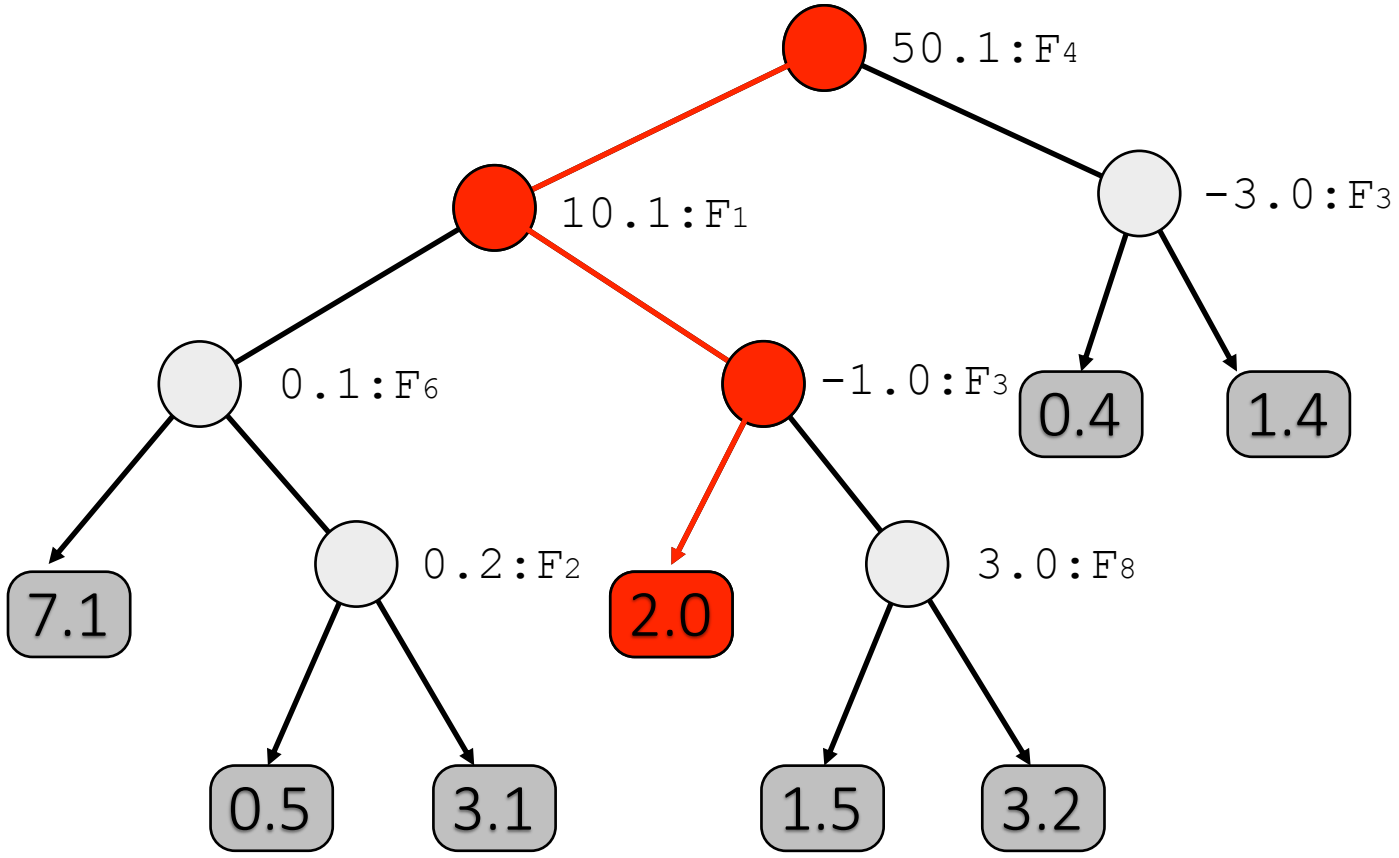
- Wang *et al.* propose a cascade ranking model for efficient ranked retrieval [WLM11b]
  - Retrieval as a multi-stage progressive refinement problem, where each stage considers successively richer and more complex ranking models, but over successively smaller candidate document sets
  - Boosting algorithm (modified AdaRank) to jointly learn the model structure and the set of documents to prune at each stage
  - Experiments show the model is able to simultaneously achieve high effectiveness and fast retrieval
- Xu *et al.* propose to post-process classifiers to reduce their test time complexity [XKW+14a]
  - Focus on execution time and feature extraction cost with skewed classes
  - Reduction of the average cost of a classifier during test time by an order of magnitude on real-world Web search ranking data sets

[WLM11b] L. Wang, J. Lin, and D. Metzler. *A cascade ranking model for efficient ranked retrieval*. In Proc. ACM SIGIR, 2011.
[XKW+14a] Z. Xu, M. J. Kusner, K. Q. Weinberger, M. Chen, and O. Chapelle. *Classifier cascades and trees for minimizing feature evaluation cost*. JMLR, 2014.

Efficient Traversal of Tree-based Models

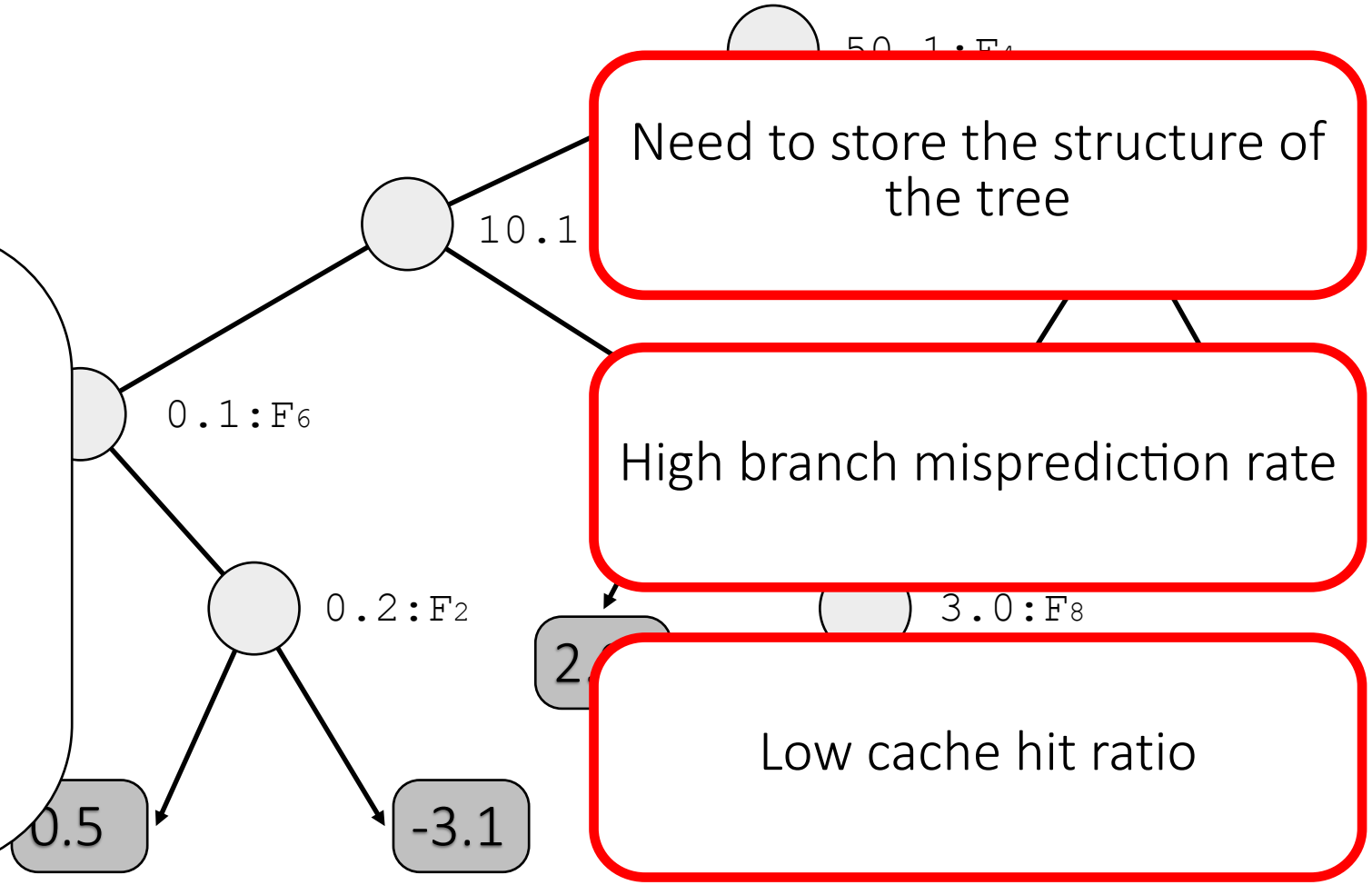
Efficient Traversal of Tree-based models

- From Yahoo! Learning to Rank Challenge Overview: *"The winner proposal used a linear combination of 12 ranking models, 8 of which were LambdaMART boosted tree models, having each up to 3,000 trees"* [YLTRC].

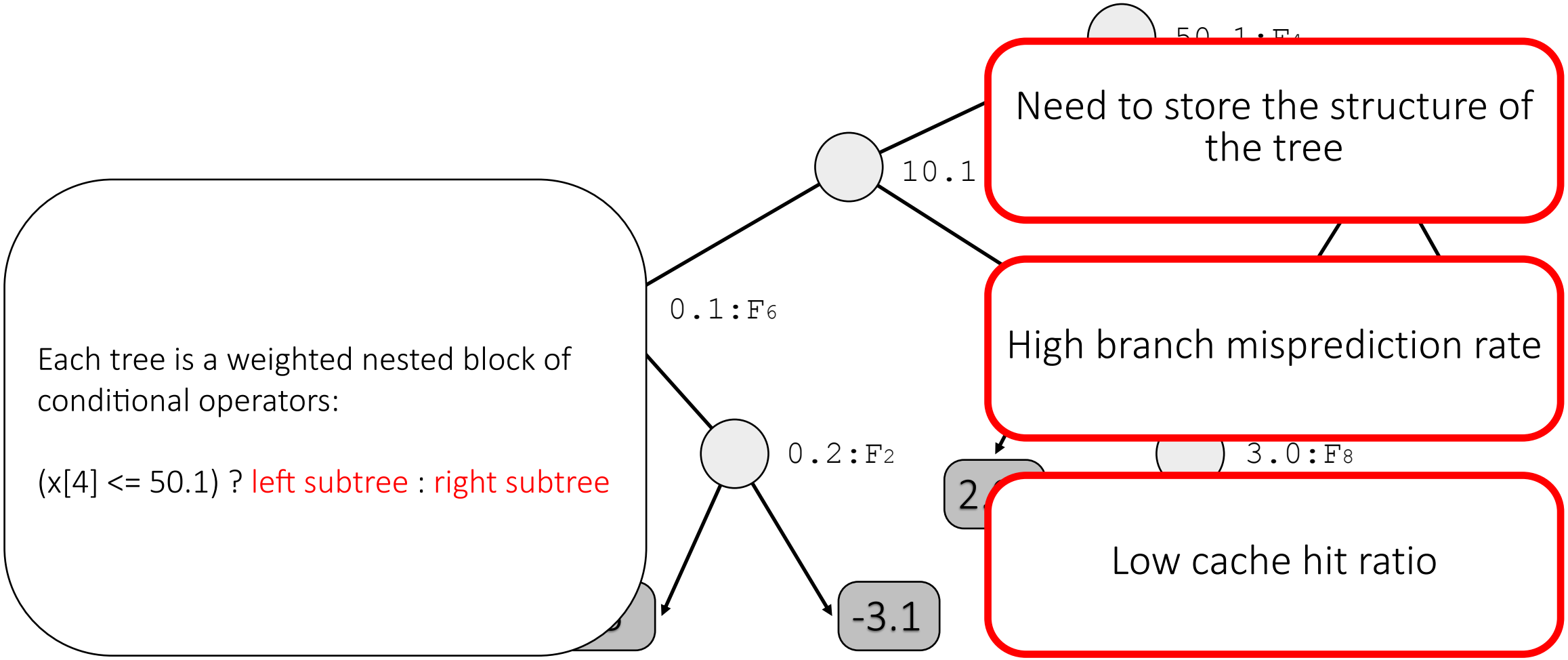


If-Then-Else

```
if (x[4] <= 50.1) {
// recurses on the left subtree
...
} else {
// recurses on the right subtree
   if(x[3] <= -3.0)
      result = 0.4;
   else
      result = -1.4;
}
```

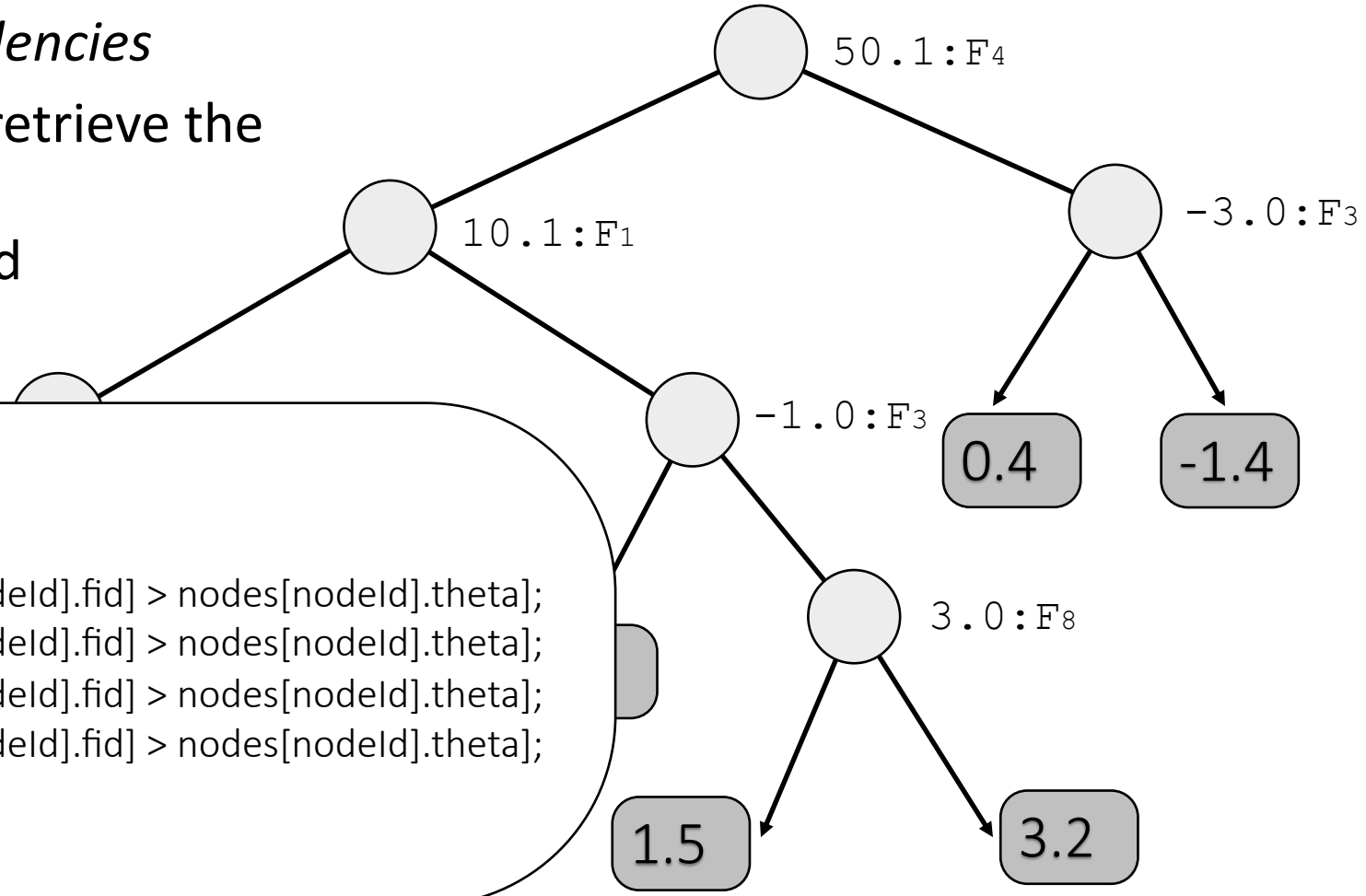


Conditional Operators



VPred [ALdV14]

- From *control* to *data dependencies*
- Output of a test as index to retrieve the next node to process
- The visit is statically un-rolled
- 16 docs at the same time

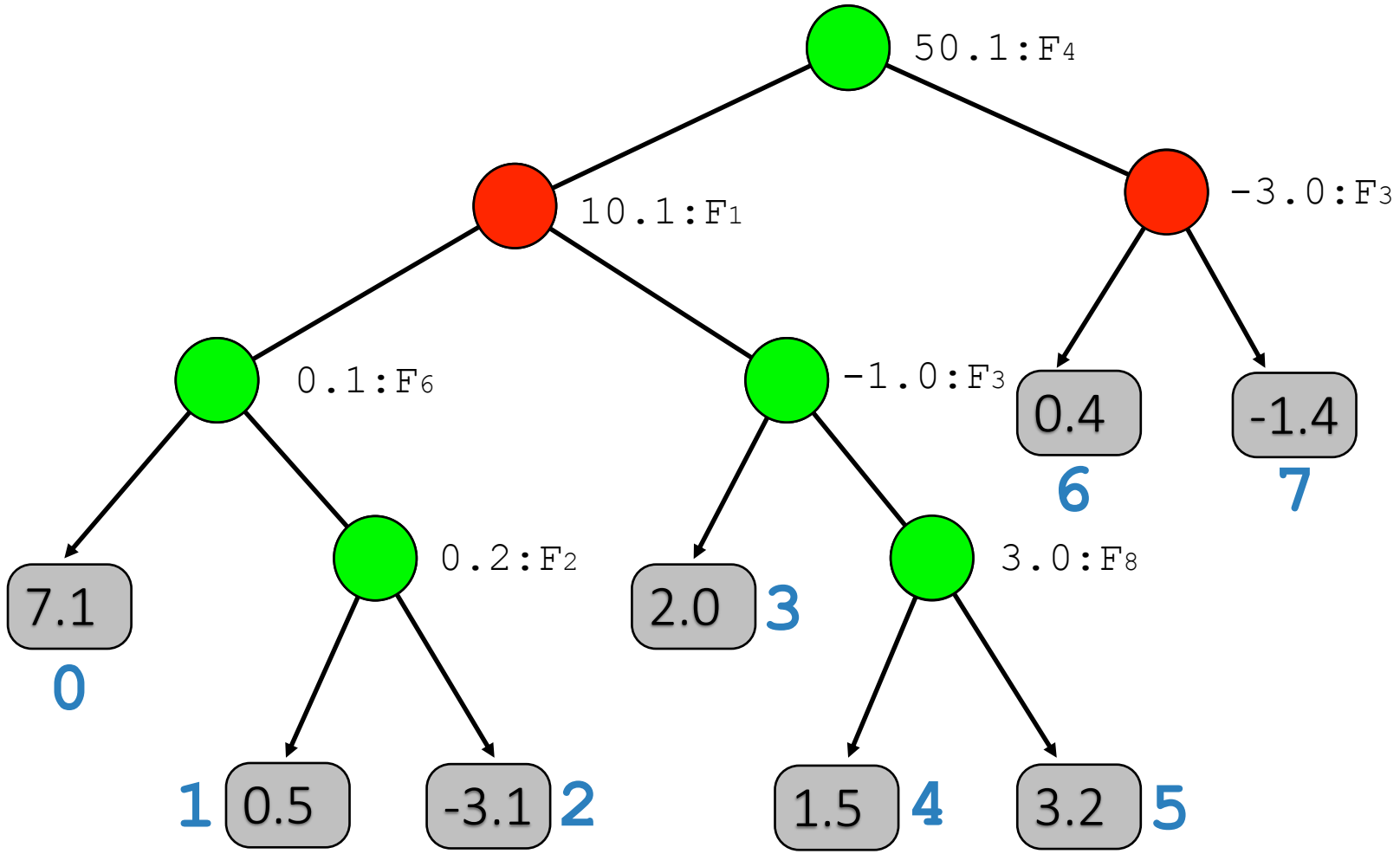



```
double depth4(float* x, Node* nodes) {
        int nodeId = 0;
        nodeId = nodes->children[x[nodes[nodeId].fid] > nodes[nodeId].theta];
        nodeId = nodes->children[x[nodes[nodeId].fid] > nodes[nodeId].theta];
        nodeId = nodes->children[x[nodes[nodeId].fid] > nodes[nodeId].theta];
        nodeId = nodes->children[x[nodes[nodeId].fid] > nodes[nodeId].theta];
        return scores[nodeId];
        }
```

[ALdV14] N. Asadi, J. Lin, and A. P. de Vries. *Runtime optimizations for tree-based machine learning models*. IEEE TKDE, 2014.

QuickScorer [LNO+15]

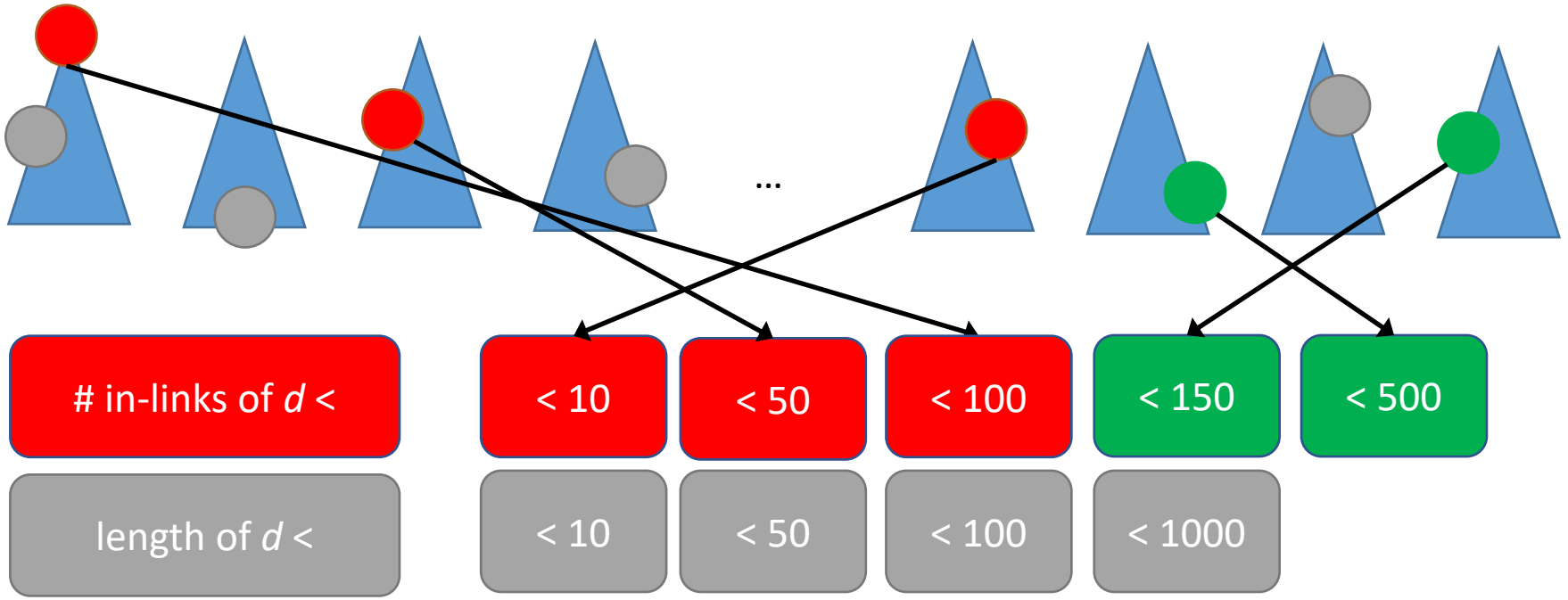
- Given a document, each node of a tree can be classified as True or False
- The exit leaf can be identified by knowing **all (and only) the false nodes of a tree**
- From per-tree scoring to per-feature scoring
  - Per-feature linear scan of thresholds in the forest



[LNO+15] C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, N. Tonellotto, and R. Venturini. *Quickscorer: A fast algorithm to rank documents with additive ensembles of regression trees*. In Proc. ACM SIGIR, 2015.

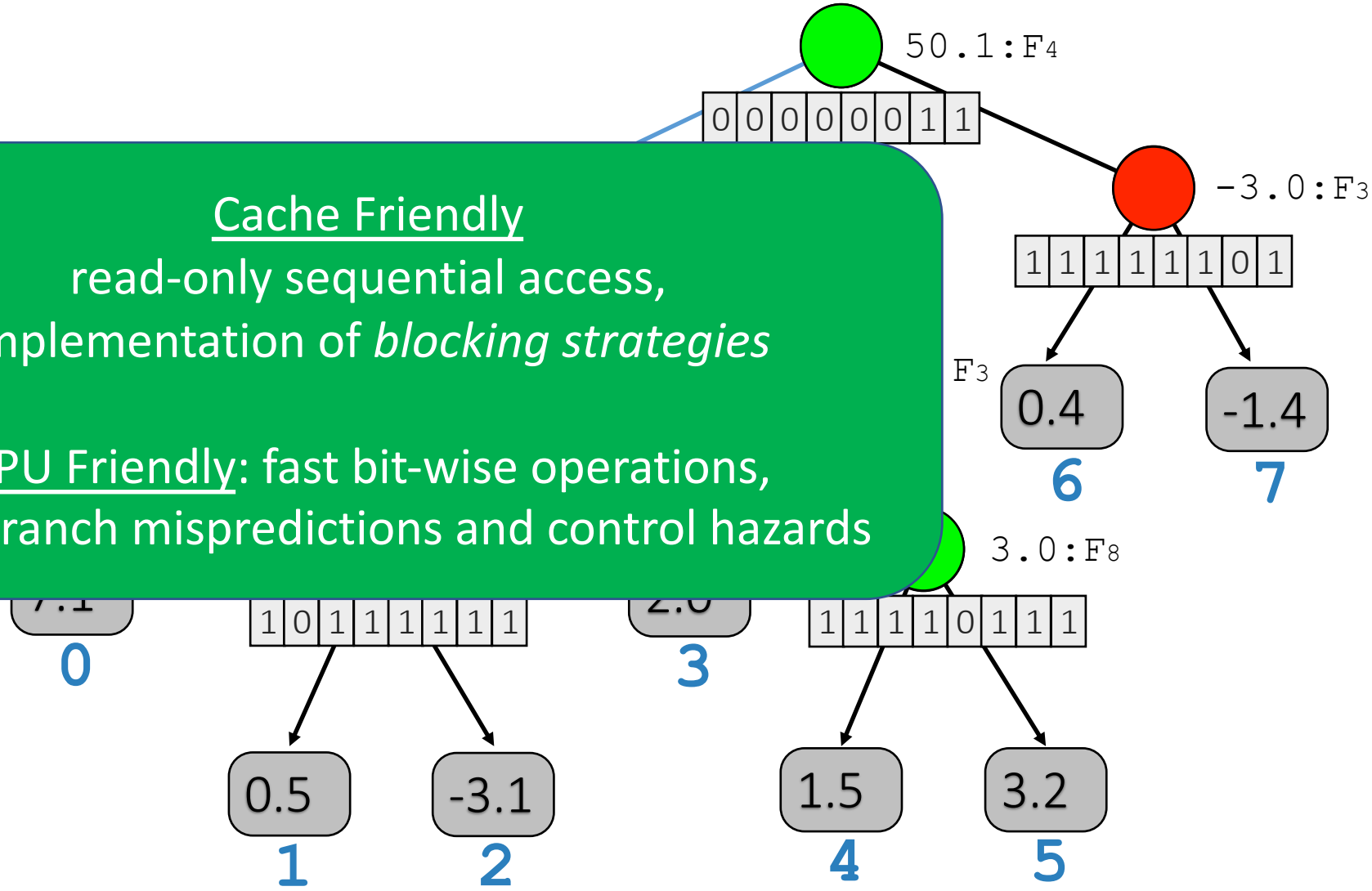
QuickScorer [LNO+15]

- Per-feature linear scan of thresholds in the ensemble
- Let's suppose our document has #inlinks = 120.



QuickScorer [LNO+15]

- Bitmasks storing leafs “disappearing” if th node is False
- ANDing masks o nodes lead to th identification of leaf
  - Leftmost bit set to resulting mask
- Few operations insensitive to node processing order



Cache Friendly
read-only sequential access,
implementation of *blocking strategies*

CPU Friendly: fast bit-wise operations,
little branch mispredictions and control hazards

QuickScorer [LNO+15]

- Public datasets: MSLR-Web10K and Yahoo LETOR
- Experiments on LambdaMART models: 1K, 5K, 10K, 20K trees and 8, 16, 32, 64 leaves.

Method	Λ	Number of trees/dataset							
		1,000		5,000		10,000		20,000	
		MSN-1	Y!S1	MSN-1	Y!S1	MSN-1	Y!S1	MSN-1	Y!S1
QS	8	**2.2** (–)	**4.3** (–)	**10.5** (–)	**14.3** (–)	**20.0** (–)	**25.4** (–)	**40.5** (–)	**48.1** (–)
VPred		7.9 (3.6x)	8.5 (2.0x)	40.2 (3.8x)	41.6 (2.9x)	80.5 (4.0x)	82.7 (3.3)	161.4 (4.0x)	164.8 (3.4x)
If-Then-Else		8.2 (3.7x)	10.3 (2.4x)	81.0 (7.7x)	85.8 (6.0x)	185.1 (9.3x)	185.8 (7.3x)	709.0 (17.5x)	772.2 (16.0x)
Struct+		21.2 (9.6x)	23.1 (5.4x)	107.7 (10.3x)	112.6 (7.9x)	373.7 (18.7x)	390.8 (15.4x)	1150.4 (28.4x)	1141.6 (23.7x)
QS	16	**2.9** (–)	**6.1** (–)	**16.2** (–)	**22.2** (–)	**32.4** (–)	**41.2** (–)	**67.8** (–)	**81.0** (–)
VPred		16.0 (5.5x)	16.5 (2.7x)	82.4 (5.0x)	82.8 (3.7x)	165.5 (5.1x)	165.2 (4.0x)	336.4 (4.9x)	336.1 (4.1x)
If-Then-Else		18.0 (6.2x)	21.8 (3.6x)	126.9 (7.8x)	130.0 (5.8x)	617.8 (19.0x)	406.6 (9.9x)	1767.3 (26.0x)	1711.4 (21.1x)
Struct+		42.6 (14.7x)	41.0 (6.7x)	424.3 (26.2x)	403.9 (18.2x)	1218.6 (37.6x)	1191.3 (28.9x)	2590.8 (38.2x)	2621.2 (32.4x)
QS	32	**5.2** (–)	**9.7** (–)	**27.1** (–)	**34.3** (–)	**59.6** (–)	**70.3** (–)	**155.8** (–)	**160.1** (–)
VPred		31.9 (6.1x)	31.6 (3.2x)	165.2 (6.0x)	162.2 (4.7x)	343.4 (5.7x)	336.6 (4.8x)	711.9 (4.5x)	694.8 (4.3x)
If-Then-Else		34.5 (6.6x)	36.2 (3.7x)	300.9 (11.1x)	277.7 (8.0x)	1396.8 (23.4x)	1389.8 (19.8x)	3179.4 (20.4x)	3105.2 (19.4x)
Struct+		69.1 (13.3x)	67.4 (6.9x)	928.6 (34.2x)	834.6 (24.3x)	1806.7 (30.3x)	1774.3 (25.2x)	4610.8 (29.6x)	4332.3 (27.0x)
QS	64	**9.5** (–)	**15.1** (–)	**56.3** (–)	**66.9** (–)	**157.5** (–)	**159.4** (–)	**425.1** (–)	**343.7** (–)
VPred		62.2 (6.5x)	57.6 (3.8x)	355.2 (6.3x)	334.9 (5.0x)	734.4 (4.7x)	706.8 (4.4x)	1309.7 (3.0x)	1420.7 (4.1x)
If-Then-Else		55.9 (5.9x)	55.1 (3.6x)	933.1 (16.6x)	935.3 (14.0x)	2496.5 (15.9x)	2428.6 (15.2x)	4662.0 (11.0x)	4809.6 (14.0x)
Struct+		109.8 (11.6x)	116.8 (7.7x)	1661.7 (29.5x)	1554.6 (23.2x)	3040.7 (19.3x)	2937.3 (18.4x)	5437.0 (12.8x)	5456.4 (15.9x)

Vectorized QuickScorer [LNO+16b]

- Instruction-level parallelism: SIMD capabilities of modern CPUs (SSE 4.2 and AVX 2)
- V-QuickScorer (vQS) exploits 128 bit registers (SSE 4.2) and 256 bit registers (AVX 2)
  - mask and score computation: with 32 leaves models, 4 docs in parallel (SSE 4.2), 8 docs in parallel (AVX 2)
- Public datasets: MSN10K, Yahoo LETOR and Istella
- Experiments on LambdaMART models: 1K, 10K trees and 32, 64 leaves

Λ	Method	Number of trees/dataset					
		1,000			10,000		
		MSN-1	Y!S1	istella	MSN-1	Y!S1	istella
32	QS	6.3 (–)	12.5 (–)	8.9 (–)	73.7 (–)	88.7 (–)	69.9 (–)
	vQS (SSE 4.2)	3.2 (2.0x)	5.2 (2.4x)	4.2 (2.1x)	46.2 (1.6x)	53.7 (1.7x)	38.6 (1.8x)
	vQS (AVX-2)	**2.6 (2.4x)**	**3.9 (3.2x)**	**3.1 (2.9x)**	**39.6 (1.9x)**	**43.7 (2.0x)**	**30.7 (2.3x)**
64	QS	11.9 (-)	18.8 (-)	14.3 (-)	183.7 (-)	182.7 (-)	162.2 (-)
	vQS (SSE 4.2)	10.2 (1.2x)	13.9 (1.4x)	11.0 (1.3x)	173.1 (1.1x)	164.3 (1.1x)	132.2 (1.2x)
	vQS (AVX-2)	**7.9 (1.5x)**	**10.5 (1.8x)**	**8.0 (1.8x)**	**138.2 (1.3x)**	**140.0 (1.3x)**	**104.2 (1.6x)**

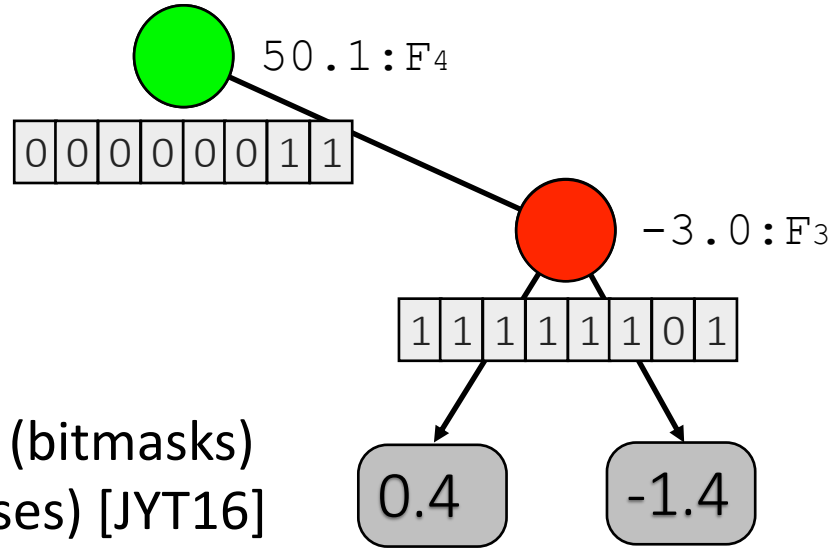
[LNO+16b] C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, N. Tonellotto, and R. Venturini. *Exploiting CPU SIMD extensions to speed-up document scoring with tree ensembles*. In Proc. ACM SIGIR, 2016.

Multi-thread and GPU-QuickScorer [LLN+18]

- Thread-level parallelism
  - OpenMP to distribute vQS among several processing cores
- Speedup
  - 16 cores machine
  - 6.3x – 14x over vQS, 20.7x – 35x over QS
- QuickScorer on GPU
- The GPU performs both
  - mask and score computation
- Speedup
  - NVIDIA GTX 1080
  - Up to 102.6x over QS

[LLN+18] F. Lettich, C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, N. Tonellotto, and R. Venturini. *Parallel Traversal of Large Ensembles of Decision Trees* . IEEE TPDS, 2018.

RapidScorer [YZZ+18]



- The data structure of QS and V-QS
  - linearly depends from the number of leaves of the trees (bitmasks)
  - for large trees, it is redundant and inefficient (cache misses) [JYT16]
- RapidScorer (RS)
  - industrial scenarios can exploit trees with a large number of leaves: up to 500
- Two solutions:
  - *Epitome*: 0's count not 1's while ANDing. Only store 0's
  - Collapse nodes of trees with same feature, threshold
- Experiments on two datasets: MSN10K and AdsCTR (not public)
  - Against QS and V-QS
  - Speedup ranging from 5.8x to 25x when using trees with more than 64 leaves

[YZZ+18] T. Ye, H. Zhou, W. Y. Zou, B. Gao, R. Zhang. *RapidScorer: Fast Tree Ensemble Evaluation by Maximizing Compactness in Data Level Parallelization*. In. Proc. ACM SIGKDD, 2018.

Further Reading

- Tang *et al.* [TJY14] investigates data traversal methods for fast score calculation with large ensembles of regression trees. Authors propose a 2D blocking scheme for better cache utilization.
  - Introduction of document and tree blocking for a better exploitation of cache layers of modern CPUs. The technique is used by Lucchese *et al.* [LNO+15].
- Jin *et al.* [JYT16] provide an analytical comparison of cache blocking methods. Moreover, they propose a technique to select a traversal method and its optimal blocking parameters for effective use of memory hierarchy.

[TJY14] X. Tang, X. Jin, and T. Yang. *Cache-conscious runtime optimization for ranking ensembles*. In Proc. ACM SIGIR, 2014.
[JYT16] X. Jin, T. Yang, and X. Tang. *A comparison of cache blocking methods for fast execution of ensemble-based score computation*. In Proc. ACM SIGIR, 2016.

Thanks a lot for your attention!

References

Feature Selection

[DC10] V. Dang and B. Croft. *Feature selection for document ranking using best first search and coordinate ascent*. In ACM SIGIR workshop on feature generation and selection for information retrieval, 2010.

[GE03] I. Guyon and A. Elisseff. *An introduction to variable and feature selection*. JMLR, 2003.

[GLNP16] A. Gigli, C. Lucchese, F. M. Nardini, and R. Perego. *Fast feature selection for learning to rank*. In Proc. ACM ICTIR, 2016.

[HZL+10] G. Hua, M. Zhang, Y. Liu, S. Ma, and L. Ru. *Hierarchical feature selection for ranking*. In Proc. WWW 2010.

[LFC+12] L. Laporte, R. Flamary, S. Canu, S. Dˊejean, and J. Mothe. *Non-convex regularizations for feature selection in ranking with sparse SVM*. Transactions on Neural Networks and Learning Systems, 10(10), 2012.

[LPTY13] H.J. Lai, Y. Pan, Y. Tang, and R. Yu. *FSMRank: Feature selection algorithm for learning to rank*. Transactions on Neural Networks and Learning Systems, 24(6), 2013.

[NA14] K. D. Naini and I. S. Altingovde. *Exploiting result diversification methods for feature selection in learning to rank*. In Proc. ECIR, 2014.

[PCA+09] F. Pan, T. Converse, D. Ahn, F. Salvetti, and G. Donato. *Feature selection for ranking using boosted trees*. In Proc. ACM CIKM, 2009.

[XHW+14] Z. Xu, G. Huang, K.Q. Weinberger, A.X. Zheng. Gradient boosted feature selection. In Proc. ACM SIGKDD 2014

References

Optimizing Efficiency within the Learning Process

[AL13] N. Asadi and J. Lin. *Training efficient tree-based models for document ranking*. In Proc. ECIR, 2013.

[LNO+16a] C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, F. Silvestri, and S. Trani. *Post-learning optimization of tree ensembles for efficient ranking*. In Proc. ACM SIGIR, 2016.

[LNO+17] C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, and S. Trani. *X-DART: Blending dropout and pruning for efficient learning to rank*. In Proc. ACM SIGIR, 2017.

[LNO+18] C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, F. Silvestri, and S. Trani. *X-CLEAVER: Learning Ranking Ensembles by Growing and Pruning Trees*. ACM TIST, 2018.

[VGB15] R. Korlakai Vinayak and R. Gilad-Bachrach. *DART: Dropouts meet Multiple Additive Regression Trees*. In PMLR, 2015.

[WLM10] L. Wang, J. Lin, and D. Metzler. *Learning to efficiently rank*. In Proc. ACM SIGIR 2010.

[XKWC13] Z. Xu, M. J. Kusner, K. Q. Weinberger, and M. Chen. *Cost-sensitive tree of classifiers*. In Proc. ICML, 2013.

References

Approximate Score Computation and Efficient Cascades

[CGBC17b] R. Chen, L. Gallagher, R. Blanco, and J. S. Culpepper. *Efficient cost-aware cascade ranking in multi-stage retrieval*. In Proc. ACM SIGIR, 2017.

[CZC+10] B. B. Cambazoglu, H. Zaragoza, O. Chapelle, J. Chen, C. Liao, Z. Zheng, and J. Degenhardt. *Early exit optimizations for additive machine learned ranking systems*. In Proc. ACM WSDM, 2010.

[WLM11b] L. Wang, J. Lin, and D. Metzler. *A cascade ranking model for efficient ranked retrieval*. In Proc. ACM SIGIR, 2011.

[XKW+14a] Z. Xu, M. J. Kusner, K. Q. Weinberger, M. Chen, and O. Chapelle. *Classifier cascades and trees for minimizing feature evaluation cost*. JMLR, 2014.

References

Efficient Traversal of Tree-based Models

[ALdV14] N. Asadi, J. Lin, and A. P. de Vries. *Runtime optimizations for tree-based machine learning models*. IEEE TKDE, 2014.

[TJY14] X. Tang, X. Jin, and T. Yang. *Cache-conscious runtime optimization for ranking ensembles*. In Proc. ACM SIGIR, 2014.

[DLN+16] D. Dato, C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, N. Tonellotto, and R. Venturini. *Fast ranking with additive ensembles of oblivious and non-oblivious regression trees*. ACM TOIS, 2016.

[JYT16] X. Jin, T. Yang, and X. Tang. *A comparison of cache blocking methods for fast execution of ensemble-based score computation*. In Proc. ACM SIGIR, 2016.

[LNO+15] C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, N. Tonellotto, and R. Venturini. *Quickscorer: A fast algorithm to rank documents with additive ensembles of regression trees*. In Proc. ACM SIGIR, 2015.

[LNO+16b] C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, N. Tonellotto, and R. Venturini. *Exploiting CPU SIMD extensions to speed-up document scoring with tree ensembles*. . In Proc. ACM SIGIR, 2016.

[LLN+18] F. Lettich, C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, N. Tonellotto, and R. Venturini. *Parallel Traversal of Large Ensembles of Decision Trees*. IEEE TPDS, 2018.

[JYT16] X. Jin, T. Yang, and X. Tang. *A comparison of cache blocking methods for fast execution of ensemble-based score computation*. In Proc. ACM SIGIR, 2016.

[YZZ+18] T. Ye, H. Zhou, W. Y. Zou, B. Gao, R. Zhang. *RapidScorer: Fast Tree Ensemble Evaluation by Maximizing Compactness in Data Level Parallelization*. In. Proc. ACM SIGKDD, 2018.

References

Other

[QR] *QuickRank, A C++ suite of Learning to Rank algorithms*. http://quickrank.isti.cnr.it

[YLTRC] O. Chapelle and Y. Chang. *Yahoo! learning to rank challenge overview*. JMLR, 2011.

[CBY15] B. B. Cambazoglu and R. Baeza-Yates. *Scalability Challenges in Web Search Engines*. Morgan & Claypool Publishers, 2015.