الجامعة الأهلية
AHLIA UNIVERSITY
BAHRAIN

CELEBRATING 20 YEARS OF EXCELLENCE

Machine Learning for Improving the Performance of Algorithms

Ebrahim Malalla

March 23, 2021

“We have some catching up to do in the area of machine learning ...”

Klaus Froehlich

Aim

Highlight recent advances in the new line of research of learning augmented algorithms and data structures.

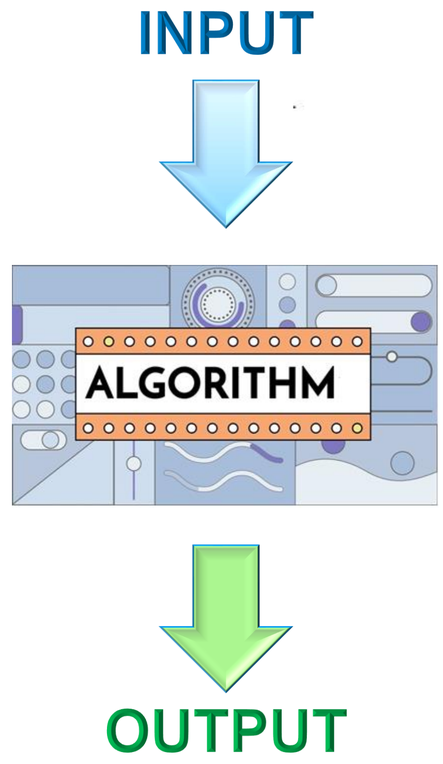
Outline

- Design and Analysis of Algorithms
- Research Framework
- Advantages and Challenges
- Toy Example: Binary Search

Design and Analysis of Algorithms

Classically, people design algorithms and analyze their asymptotic performances (time, space, quality) **based on the (instance) input:**

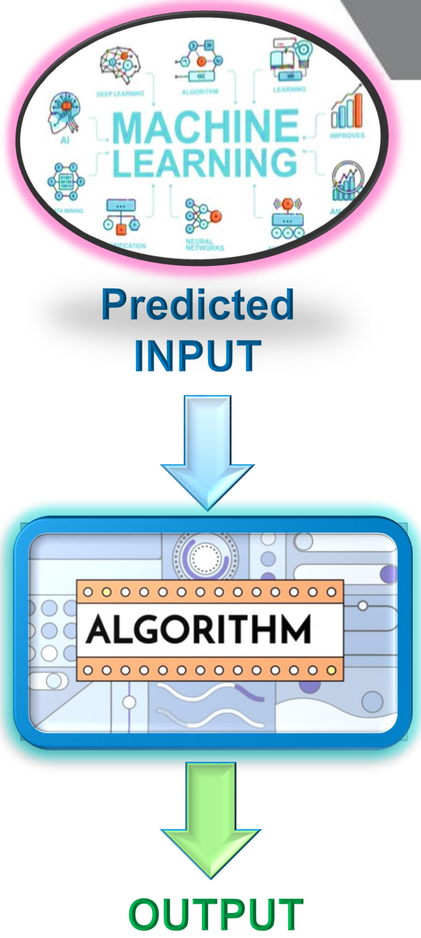
1. The best-case input: best-case scenario of all possible inputs.
2. The worst-case input: leading to the worst-case analysis of the performance.
3. The average-case input: average-case analysis of performance based on a randomly and uniformly chosen input.
4. The most likely input: probabilistic analysis of performance based on a randomly chosen input from a fixed probability distribution.



Design and Analysis of Algorithms

New area of research and paradigm shift in the design and analysis of algorithms,

5. The predicted input: assumes that some properties of the input is predicted by a ML method and the prediction is exploited to modify the behavior of algorithm to be more efficient. The performance of algorithm can be analyzed as a function of the accuracy of prediction.

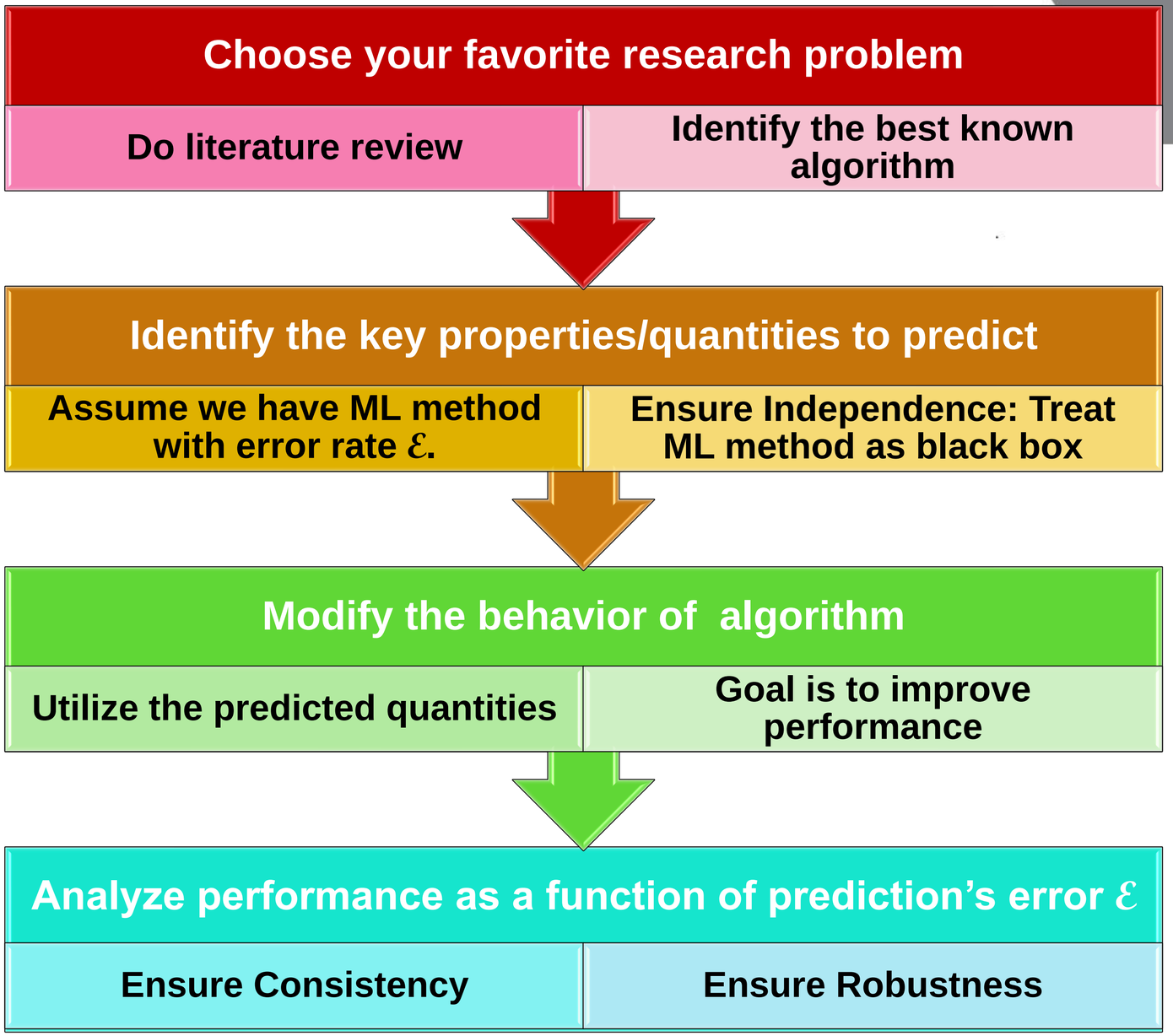


This led to an avalanche of research and emergence of a new line of algorithms and data structures: Learning-Augmented Algorithms, **aka,** Learned Algorithms, Data-driven Algorithms, Algorithms with Predictions, and Algorithms with ML Advice.

Research Framework

Main Idea: To reduce uncertainty of input by augmenting the online algorithm with some information about the future.

- Ideally, the better the prediction, the better the performance.
  - Consistency: Algorithms that have optimal or near-optimal performance when these predictions are good.
  - Robustness: when the predictions have large errors, the algorithms do not behave worse than the worst-case performance (the one with no prediction).



Assumptions

1. We assume nothing about the ML prediction methods, e.g.,
   - The performance/quality/accuracy of predictor
   - Types of prediction errors/distribution
2. We assume the following key properties of the algorithm:

Independence	• Algorithm should be independent of the predictor • No assumptions about error types / distribution
Consistence	• Better predictions lead to better performance. When **predictor is good** (with very small error), the online algorithm should have **optimal or near optimal** performance (close to the best offline algorithm).
Robustness	• Performance should **degrade gracefully** with bad predictions. When predictor is bad (large error), the performance of online algorithm with prediction should not be worse than the algorithm without predictions.

Advantages of Research Framework

1. Improves online algorithms design and analysis that
    - avoid the worst-case scenario and have better performance guarantees both in theory and practice.
    - leverage the vast amount of modeling work in machine learning,
2. Isolates the method of prediction as a plug-and-play mechanism: The ML method applied is treated as black box, and algorithms are designed and analyzed independently of theses methods. This allow researchers to plug in richer ML models as the science evolve and new techniques are discovered.
3. Ensures that better predictions lead to better algorithm performance.
4. Ties the performance of algorithms to the accuracy of predictions: as ML techniques evolve and become more powerful in obtaining near-accurate predictions, we will obtain automatically better algorithms performance, hopefully near-optimal, for no additional cost.

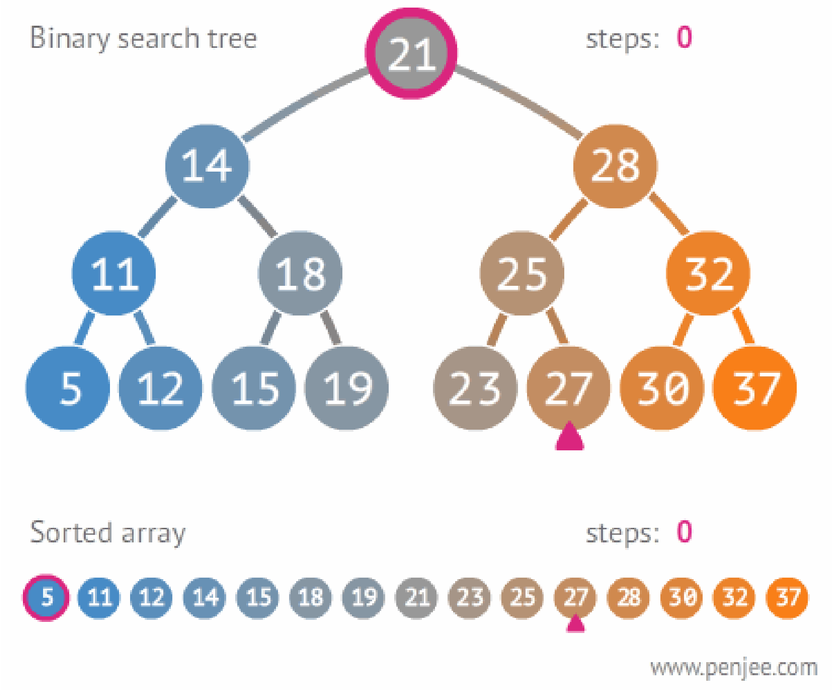
Challenges with Algorithms with Predictions

Overall question: How to incorporate (noisy, imperfect, non-uniform) ML predictions to improve performance (time, space, quality, competitive ratios) of classical algorithms.

1. Identify the quantities to predict. This depends on the problem/application.
2. Incorporate prediction into algorithm to achieve better performance and better bounds of analysis. .
3. Design **robust** algorithm that sufficiently cope with errors of predictions, i.e, improvement of performance is a decreasing function of prediction's error, but the worst-case performance (without any prediction) should be a lower bound.
4. Analyze and prove **consistency** of performance as a function of prediction's error, i.e., improvement should be an increasing function of the accuracy of prediction.

Motivating Toy Example: Binary Search

- **Problem:** Given a non-decreasingly sorted array A[n] and a query element q, search for q in A and returns either the position/index i s.t. A[i]=q, or state that q is not in A.
- **Solution:** Binary Search compares the value of q to the value of A[mid], and recursively search on the correct half of the array depending on whether q is smaller or larger than A[mid]. The worst-case search time is O(log n) probes.



Motivating Toy Example: Binary Search

- The behavior of such Binary Search does no mimic humans when **searching for a name in a contact list (telephone book), or a textbook in a library.**
- **Alternatively,** if we can somehow **predict the location of q**, we would certainly start looking at the location where we expect to find the query.
- **Assume we have a predictor h that predicts the location h(q) for every given query q.**
- **How can this improve the performance of binary search?**

Interpolation Search (Peterson, 1957)

- Instead of searching at mid, the predictor estimates the position of q, using the lowest and highest elements in the array as well as length of the array.
- The Linear Interpolation's predictor estimates q to be at the position

$$\mathbf{h(q) = low + \frac{q - A[low]}{A[high] - A[low]} (high - low)}$$, recursively.

- The average search time = O(log log n), when elements are uniformly distributed;
- The worst-case search time = O(n), which is worse than binary search. So, this is not a good predictor for our goal (not robust).

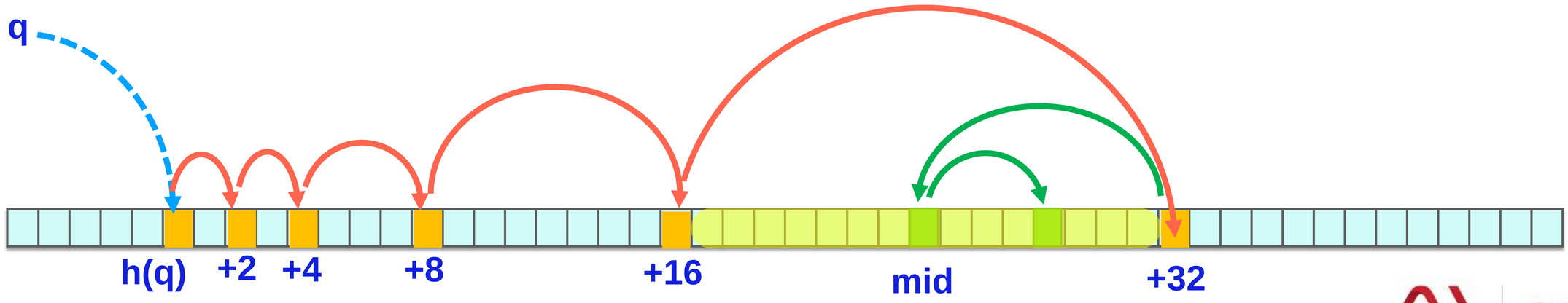
Binary Search with Predictions

Kraska et al. (2018) and Lykouris & Vassilvitskii (2018)

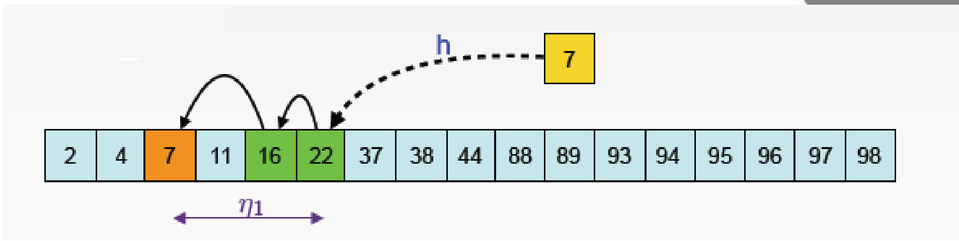
1. Perform exponential search to find upper & lower bounds of the interval containing q:

Probe first at h(q); if q is not found there, check if it is smaller or larger. If q > A[h(q)], probe elements at h(q) + 2, h(q) + 4, h(q) + 8, and so on, until we find an element larger than q (or we hit the end of the array).

2. Then we apply binary search on the last sub-interval which will contain q, if exists.



Binary Search with Predictions



Analysis:

- Let t(q) be the true position of q in A (or largest element < q, if $q \notin A$).
- The error of prediction $\mathcal{E}(q) = |h(q) - t(q)|$.
- The cost of running the algorithm is at most $2 \log |h(q) - t(q)| = 2 \log \mathcal{E}(q)$.
- The expected search time = $2\, \mathbf{E}_q [\log \mathcal{E}(q)] \leq 2 \log \mathbf{E}_q[\mathcal{E}(q)]$ (by Jensen's inequality).

- Thus, any predictor with an expected error of O(poly(log n)) leads to O(log log n) expected search time. Optimal solution is reached when expected error is constant.
- Moreover, since $\mathcal{E}(q) < n$, bad predictors cannot do much harm because worst-case search time is O(log n).

Other Algorithms with Predictions

Online Algorithms

- Scheduling and queueing
- Caching/Paging
- Ski rental (rent or buy)
- Matching

Streaming Algorithms

- Frequency estimation
- Counting sketches

Indexed Data Structures

- Hashing
- B-Trees

الجامعة الأهلية
AHLIA UNIVERSITY
BAHRAIN

CELEBRATING 20 YEARS OF EXCELLENCE

Thank you!

Consistency and Robustness

An algorithm is

- α-consistent, if its competitive ratio $\rightarrow \alpha$ as the prediction error $\rightarrow 0$,
- β-robust, if the competitive ratio $\leq \beta$ for any predictions.

- 1- consistent means optimal
- β should not be more than the worst-case performance of algorithm without prediction.