# Project Idea:
# Algorithms for Shortest Paths, Distance Oracles, and Routing

## Background

Shortest paths is a classical problem in computer science which has found numerous practical applications. Real-life networks such as road maps are often of considerable size and efficient algorithms and data structures are needed for solving the shortest path problem on these networks.

Even an algorithm like Dijkstra which runs in near-linear time is often too slow in practice as it may explore a large part of the graph to find a shortest path between a single vertex pair. An algorithm like A* is often much faster when provided with a good heuristic for estimating shortest path distances. Another variant which often beats Dijkstra in practice is bidirectional search.

Even better time bounds can be obtained if we allow preprocessing the input graph and building a data structure on top of it. One extreme example is a giant look-up table which for each vertex pair $(u, v)$ stores a shortest path from $u$ to $v$ (or just the distance between them). This allows for very fast running time (a single look-up) but the drawback is obviously the large space requirement. If we do not require the paths output to be shortest but allow some approximation (for instance, the weight of the path output is required to be within a small factor of the shortest one) then data structures are known which offer good tradeoffs between time, space, and approximation. These data structures are often referred to as distance oracles.

The tradeoffs we can guarantee depend on the desired generality. We have some results for general undirected graphs [TZ05, PRT12, Wul13], but much better results are known if the graphs are planar [Tho04] or in geometric settings where we look for distances avoiding obtacles in the plane [Tho07].

There is also the routing variant were we want a small routing table at each node telling us which outgoing edge to use to get quickly to a desired destination [AGM+08].

## Generic project description

- Implement various (exact and/or approximate) shortest path algorithms and data structures. You are quite free in choosing which ones to implement. The cited papers are just some of the options.
- Experimentally test the performance of these on suitable graph instances.
- Performance measures may include running time, space requirement, quality of solution found (in case of approximate shortest paths).

- Compare the results of these experiments with the theoretical bounds to see how they match up.
- **Write the report**. Remember that at the end of the day, your grade will be determined by the report in which you describe the great work you have done and the theory behind it.

You can also chose a more theoretical focus.

## Supervision

- Mikkel Thorup.

## References


[AGM+08] Ittai Abraham, Cyril Gavoille, Dahlia Malkhi, Noam Nisan, and Mikkel Thorup. Compact name-independent routing with minimum stretch. *ACM Transactions on Algorithms*, 4(3), 2008.

[PRT12] Mihai Pătraşcu, Liam Roditty, and Mikkel Thorup. A new infinity of distance oracles for sparse graphs. In *Proceedings of the 53nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 738–747, 2012.

[Tho04] Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *Journal of the ACM*, 51(6):993–1024, 2004. Announced at FOCS'01.

[Tho07] Mikkel Thorup. Compact oracles for approximate distances around obstacles in the plane. In *Proc. 15th European Symposium on Algorithms (ESA)*, pages 383–394, 2007.

[TZ05] Mikkel Thorup and Uri Zwick. Approximate distance oracles. *Journal of the ACM*, 52(1):1–24, 2005. Announced at STOC'01.

[Wul13] Christian Wulff-Nilsen. Approximate distance oracles with improved query time. In *Proc. 24th ACM/SIAM Symposium on Discrete Algorithms (SODA)*, pages 539–549, 2013.