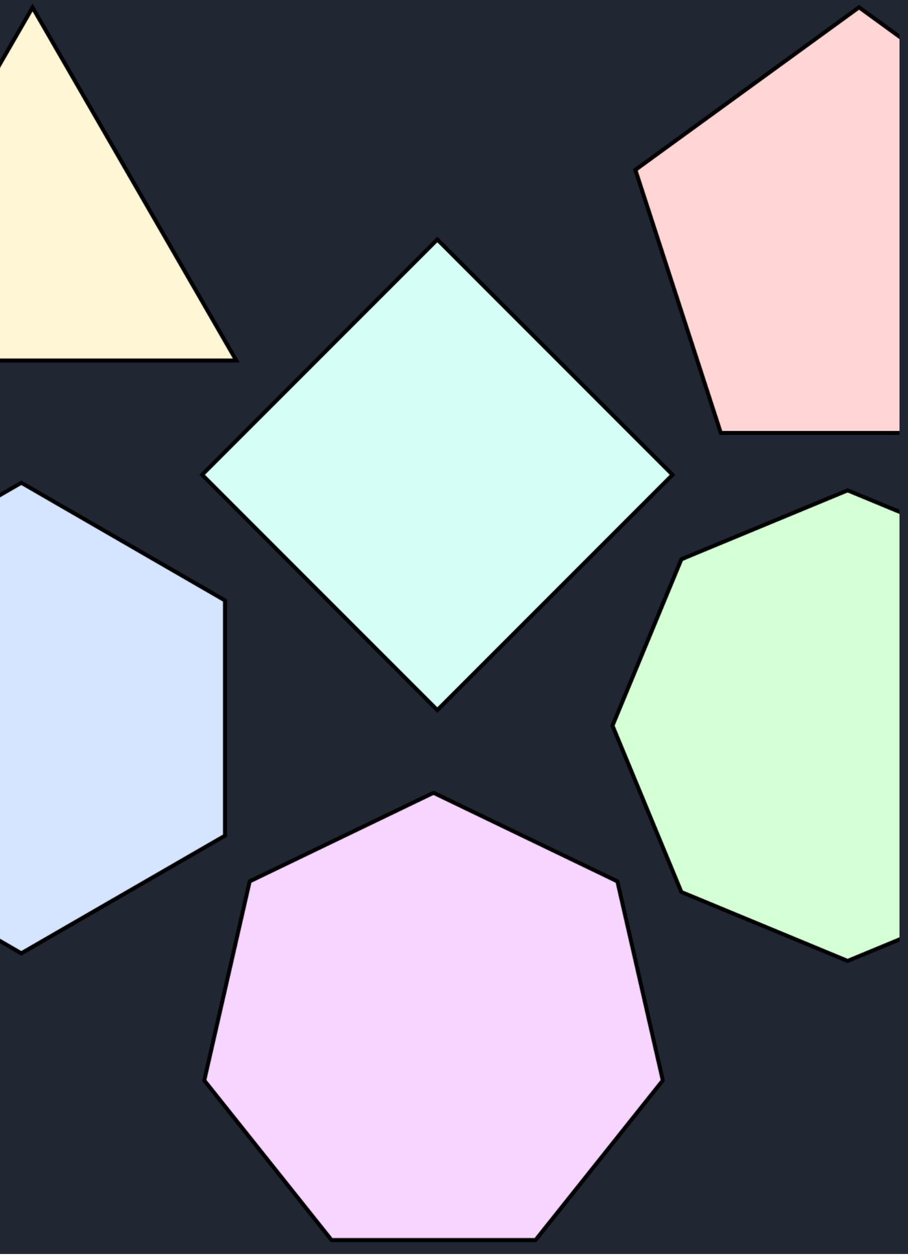
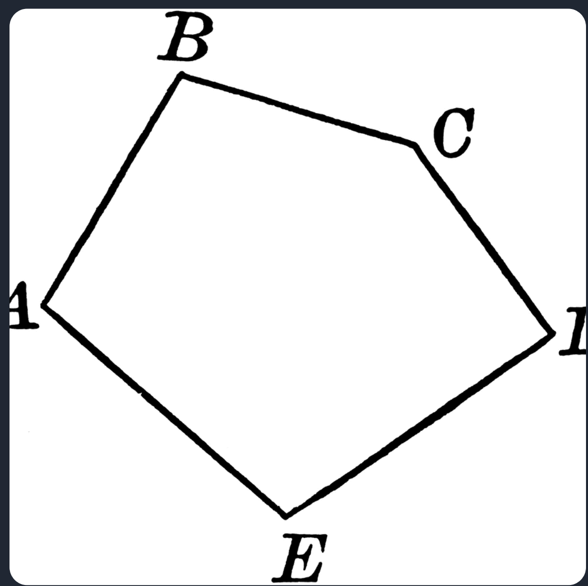


Computational Geometry Project: Minimum Weight Triangulation of a Simple Polygon

An in-depth look at the dynamic programming algorithm implementation to find the minimum weight triangulation of a simple polygon.

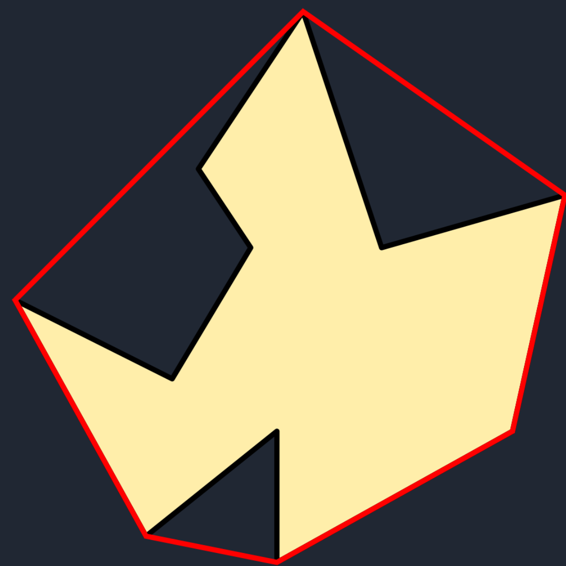
Input Polygon

A simple polygon can be defined as a closed path formed by the straight line segments in the plane, which encloses a region on one side. Our algorithm takes as input a simple polygon defined by its vertices, which can be taken as an ordered set of points that lie on the boundary of the polygon.



Convex Polygon

A polygon where no interior angle is greater than or equal to 180 degrees is a convex polygon.



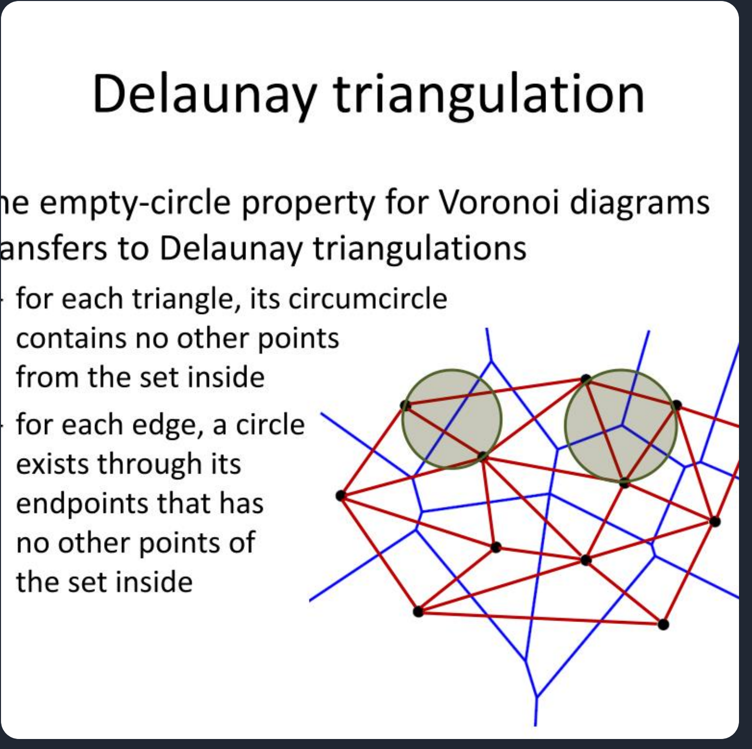
Concave Polygon

A polygon where at least one interior angle is greater than or equal to 180 degrees is a concave polygon.

Minimum Weight Triangulation

A minimum weight triangulation of a simple polygon P is a triangulation with a minimum sum of weights of its diagonals.

The weight of a diagonal is simply the Euclidean distance between the vertices determining it.



A Delaunay triangulation is the "nicest" triangulation, not necessarily the minimum weight one

Dynamic Programming

Dynamic programming is a technique used to solve problems in computer science, mathematics, economics and more. It's often applied to optimization problems, such as the minimum weight triangulation of a simple polygon.

1 Subproblems

Break the problem into smaller subproblems.

2 Optimal Solutions

Finding the optimal solutions of the subproblems.

3 Memory

Store the solutions to the subproblems in memory to avoid redundant calculations.

Dynamic Programming Approach

1. Let **MWT(i, j)** be the weight of the minimum-weight triangulation of the polygon for an edge ij, and assume we want to compute MWT(1, n), i.e. the minimum weight triangulation of the entire polygon.
2. Consider a chord between vertices i and j. This chord divides the polygon into two smaller polygons: Pi..j and Pj..k..i.
3. The weight of the triangulation that uses this chord is **W(i,j,k) = weight(i,j,k) + MWT(i,j) + MWT(j,k).** Here, weight(i,j,k) is the weight of the triangle formed by vertices i, j, and k.
4. We can then compute MWT(i, j) as the minimum of all possible chords:
    a. **MWT(i, j) = min { weight(i, k, j) + MWT(i, k) + MWT(k, j) }, if j > i + 1 , where i < j < k**
    b. **MWT(i, j) = 0, otherwise**

5.We start with the base case **MWT(i, i+1) = 0** for all i, since a line segment does not need to be triangulated.

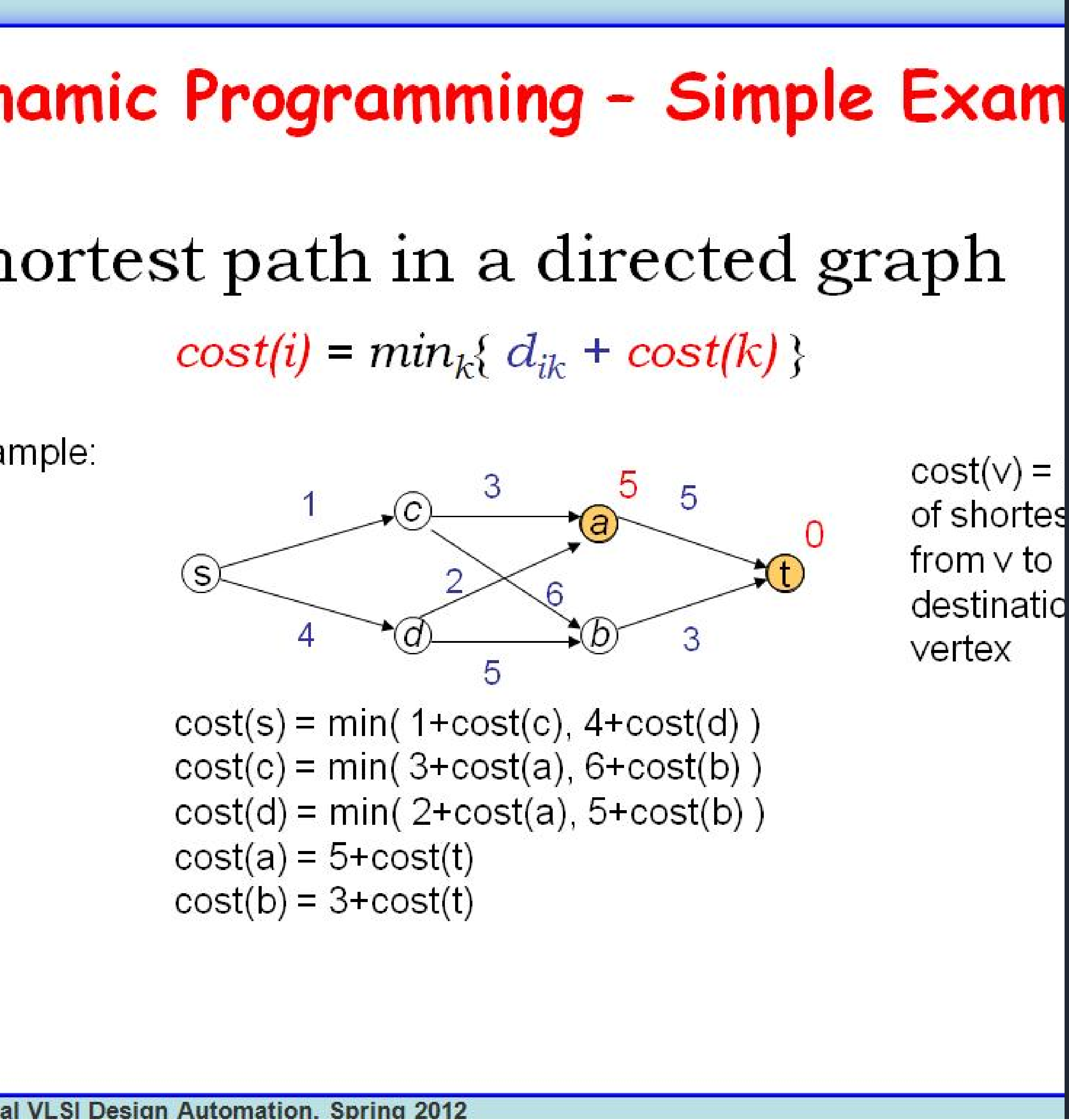
The final solution is then **MWT(1, n)**.

Time Complexity and Optimization

The time complexity of the above solution is exponential. On drawing a recursion tree, we can easily see that each subproblem gets repeated several times. The problem satisfies both **optimal substructure** and **overlapping subproblems** properties of dynamic programming, so the subproblem solutions can be derived using memoization or in a bottom-up manner.

The time complexity of the dynamic programming algorithm is $O(n^3)$, which could make it unsuitable for large polygons. There are other techniques as well, such as triangulation based on ear clipping or Delaunay triangulations, which reduces the time complexity to $O(n^2)$, which doesn't give such an optimized answer, however.

Note - My solution uses the iterative dynamic programming version of the recursive algorithm mentioned, just to make the solution more obvious.



Conclusion

Minimum weight triangulation is a classical problem in the field of computational geometry with a variety of applications. The dynamic programming approach used here is quite analogous to the Bellman Ford approach in determining all pairs shortest paths in a graph.

Demonstration

drive.google.com

Video demo