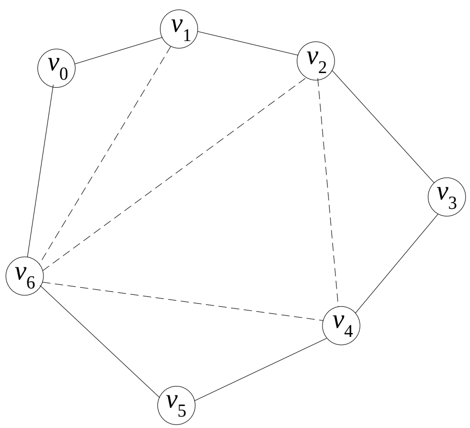
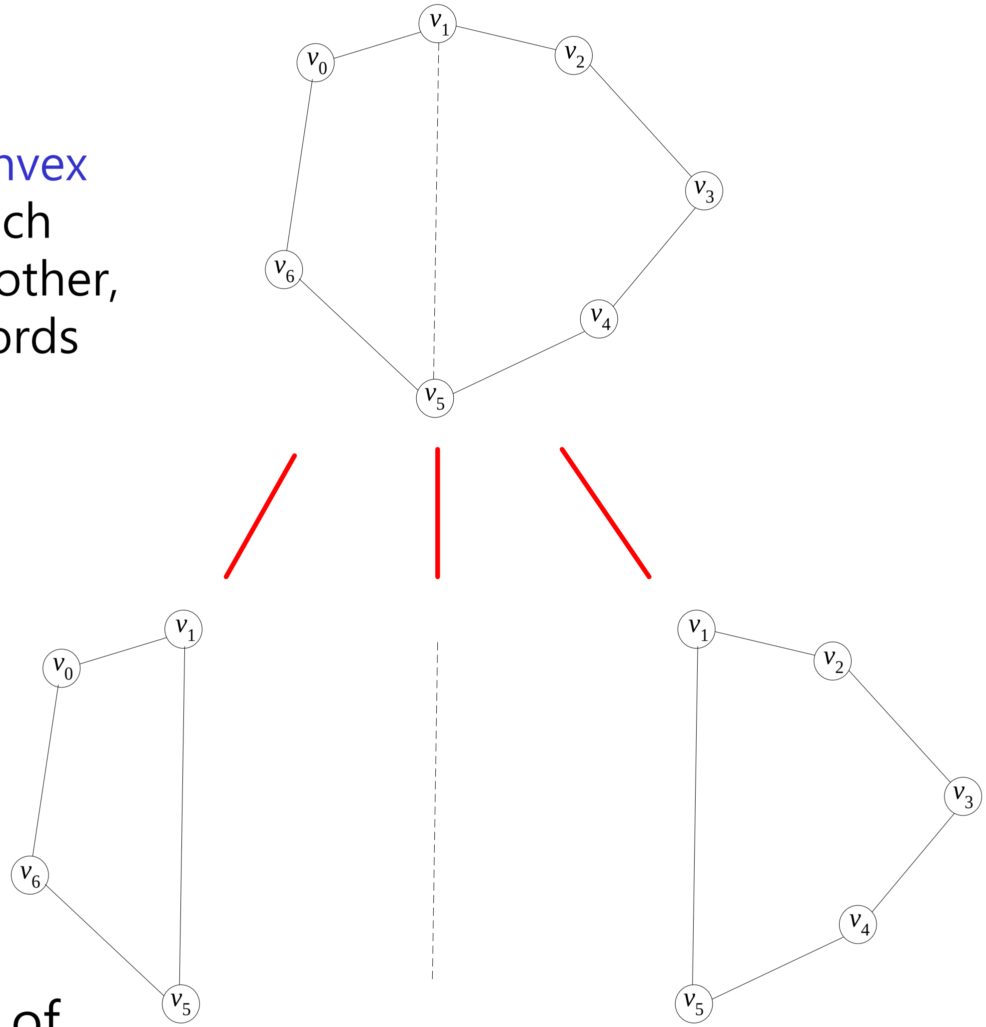
[A. Aho, J. Hopcroft, and J. Ullman, *Data Structures and Algorithms*, Addison-Wesley, 1983. 10.2]

Minimal Triangulation

- Problem
  - Given a set of *n* vertices for convex polygon, find a triangulation such that no two chords cross each other, and the total length of the chords selected is a minimum.

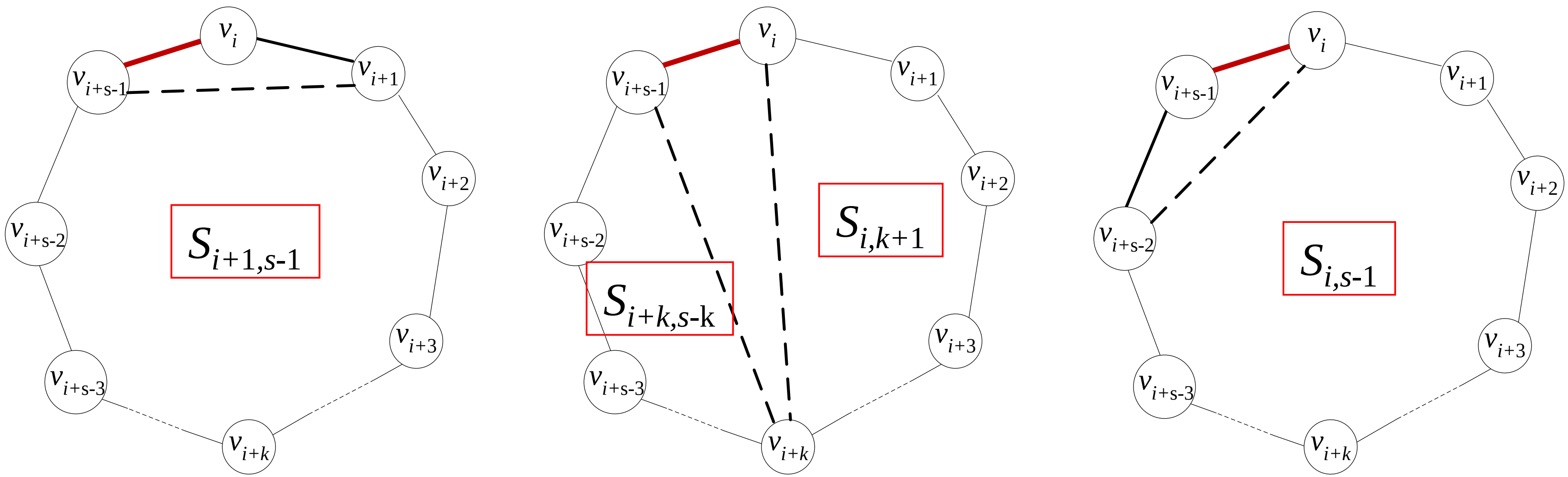




- Counting all possible selections of chords in an inefficient way results in an exponential algorithm.

A Dynamic Programming Approach

- The sub-problem S_{is} of size s beginning at vertex v_i
  - The minimal triangulation problem for the polygon formed by the s vertices beginning at v_i an proceeding clockwise, that is, v_i, v_{i+1}, … v_{i+s-1}.
- Three cases:



✓ Pick some k between 1 and s-2 and solve sub-problems $S_{i,k+1}$ and $S_{i+k,s-k}$.

- "Solving" any sub-problem of size three or less requires no action.

- Let C_{is} be the minimum cost of triangulating S_{is} for all i and s.

- The recursive formula for computing C_{is}

$$\boxed{C_{is} = \begin{cases} 0, & \text{if } s = 2 \text{ or } 3 \\ \min_{1 \le k \le s-2}[C_{i,k+1} + C_{i+k,s-k} + D(v_i, v_{i+k}) + D(v_{i+k}, v_{i+s-1})], & \text{if } s \ge 4 \end{cases}}$$

$$\text{where } D(v_p, v_q) = \begin{cases} \text{the length of the chord } (v_p, v_q), & \text{if } v_p \text{ and } v_q \text{ are not adjacent} \\ 0, & \text{otherwise} \end{cases}$$

- Table filling pattern: C_{65}

$$C_{65} = \begin{cases} k=1: & C_{62} + C_{74} + D(v_6, v_7) + D(v_7, v_{10}) \\ k=2: & C_{63} + C_{83} + D(v_6, v_8) + D(v_8, v_{10}) \\ k=3: & C_{64} + C_{92} + D(v_6, v_9) + D(v_9, v_{10}) \end{cases}$$

10										
9										
8										
7										
6										
5										
4										
3	0	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	0
s / i	0	1	2	3	4	5	6	7	8	9

- Table filling pattern: C_{19}

10									
9									
8									
7									
6									
5									
4									
3	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
s \ i	0	1	2	3	4	5	6	7	8

- To compute $C_{0,10}$.

s \ i									
10									
9									
8									
7									
6									
5									
4									
3	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0
s \ i	0	1	2	3	4	5	6	7	8

Cost for table filling after initialization

$$\sum_{i=4}^{n}(n-i+1)(i-2)$$

$$= n\sum_{i=4}^{n} i - \sum_{i=4}^{n} i^2 + O(n^2)$$

$$= n\sum_{i=1}^{n} i - \sum_{i=1}^{n} i^2 + O(n^2)$$

$$= \frac{n^3}{2} - \frac{n^3}{3} + O(n^2) = \frac{n^3}{6} + O(n^2)$$

- Time and Space complexity for computing C_{0n}
  - O(?) & O(?)

- **Finding solutions from the table**
  - The table does not immediately give us the triangulation itself.
  - During the computation of C_{is}, store together the value of k that gives the best solution.

$$C_{is} = \begin{cases} 0, & \text{if } s = 2 \text{ or } 3 \\ \min_{1 \le k \le s-2}[C_{i,k+1} + C_{i+k,s-k} + D(v_i, v_{i+k}) + D(v_{i+k}, v_{i+s-1})], & \text{if } s \ge 4 \end{cases}$$

$$\text{where } D(v_p, v_q) = \begin{cases} \text{the length of the chord } (v_p, v_q), & \text{if } v_p \text{ and } v_q \text{ are not adjacent} \\ 0, & \text{otherwise} \end{cases}$$