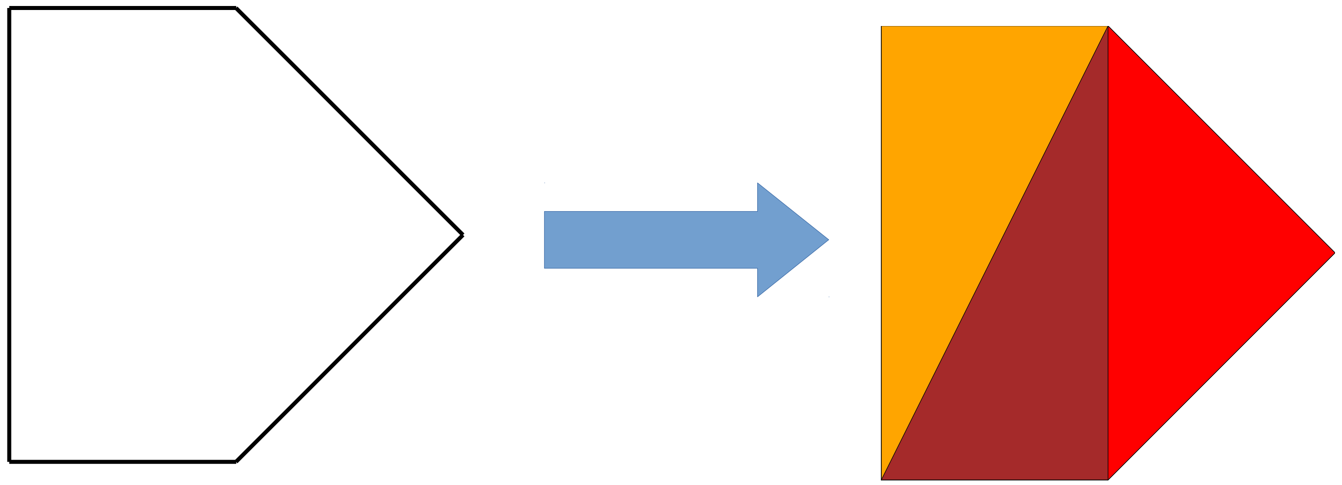
Parallel programming

HW2 assignment

FAKULTA
ELEKTROTECHNICKÁ
ČVUT V PRAZE

Triangulation of convex sets

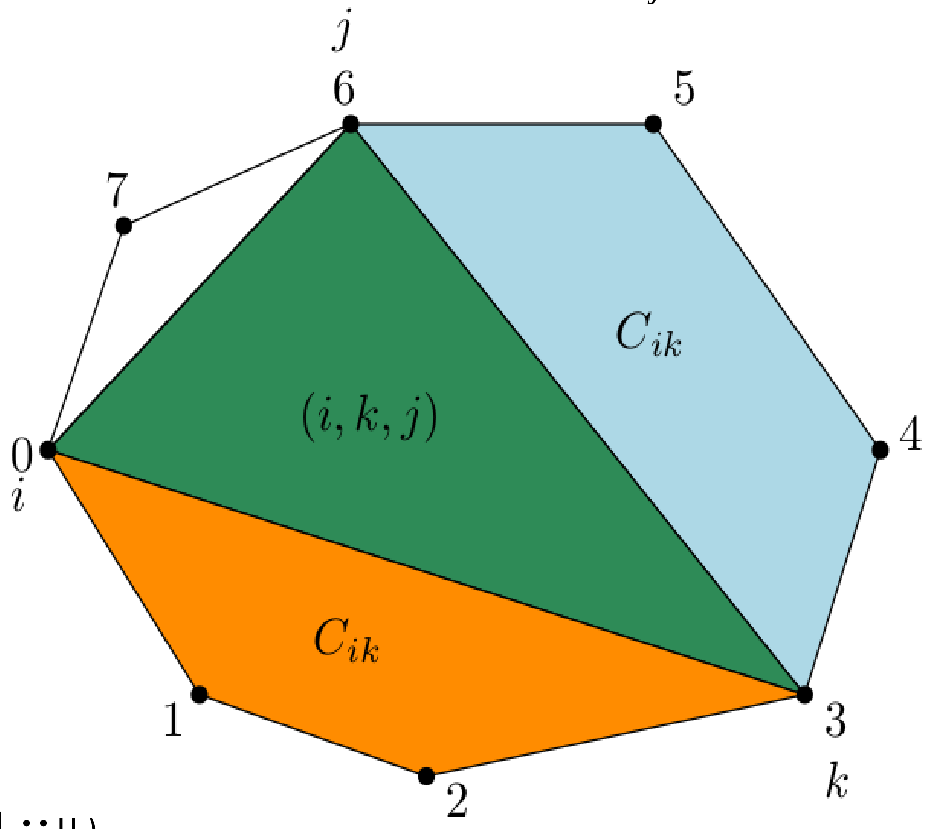
- Problem: Find the triangulation of a convex set such that a *cost* (sum of triangle perimeters) is minimal.



- **Applications:**
  - Triangulation is useful in graphics to decompose a complex object into triangle mesh, which can be effectively processed by graphics cards.

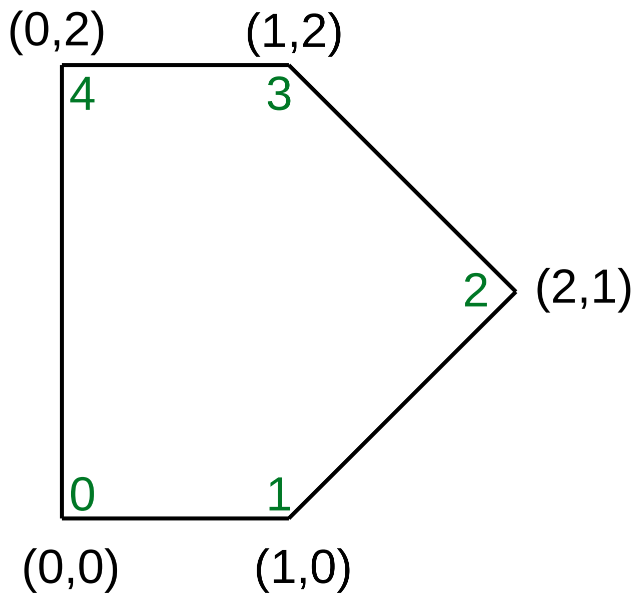
Solving triangulation by using dynamic programming

- Number the points of a polygon by numbers in counter-clockwise order
- Let C_{ij} be a minimal cost of a sub-triangulation from point i to j.
- The following recursive formula minimizes the triangulation cost for C_{ij}



```
if j < i+2
    $C_{ij} = 0$
else
    $C_{ij} = \min_{k=i+1}^{j-1} \left( C_{ik} + C_{kj} + \|ik\| + \|kj\| + \|ji\| \right)$
```

Solving triangulation by using dynamic programming



$(0,1,2)=1+\sqrt{2}+\sqrt{5}\simeq 4.65$

$(1,2,3)=\sqrt{2}+\sqrt{2}+2\simeq 4.83$

$(2,3,4)=\sqrt{2}+1+\sqrt{5}\simeq 4.65$

$(0,1,3)=1+2+\sqrt{5}\simeq 5.24$

$(0,2,3)=\sqrt{5}+\sqrt{2}+\sqrt{5}\simeq 5.89$

$(1,2,4)=\sqrt{2}+\sqrt{5}+\sqrt{5}\simeq 5.89$

$(1,3,4)=2+1+\sqrt{5}\simeq 5.24$

$(0,1,4)=1+\sqrt{5}+2\simeq 5.24$

$(0,2,4)=\sqrt{5}+\sqrt{5}+2\simeq 6.47$

$(0,3,4)=\sqrt{5}+1+2\simeq 5.24$

C

j $\longrightarrow$

i $\downarrow$

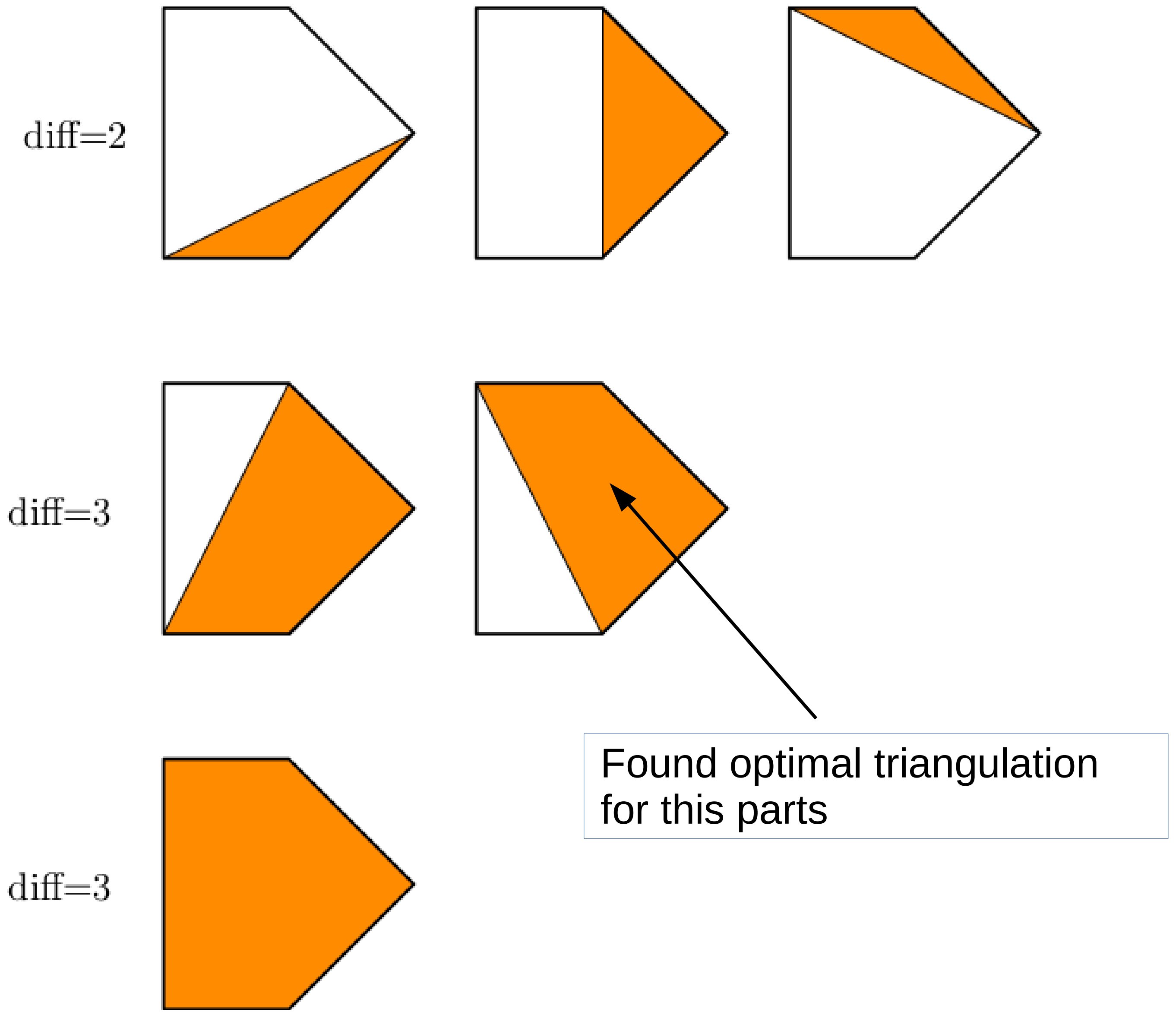
0	0	4.65	10.1	**15.3**
-	0	0	4.83	10.1
-	-	0	0	4.65
-	-	-	0	0
-	-	-	-	0

Pseudocode:

```
for (diff = 0; diff < n; ++diff)
    for (i = 0, j = diff; j < n; ++i, ++j)
        if (j < i+2)
            $C_{ij}$ = 0.0
        else
            $C_{ij}$ = inf;
            for (int k = i+1; k < j; ++k)
                cost = $C_{ik} + C_{kj} + \|ik\| + \|kj\| + \|ji\|$
                if ($C_{ij}$ > cost)
                    $C_{ij}$ = cost

return $C_{0\,n-1}$
```

Solving triangulation by using dynamic programming



Found optimal triangulation for this parts

HW1 assignment

- Use the **code skeleton**
  - reads test problems, measures runtime, generates SVG files to show the triangulation
  - try ‘--help’ to see the supported program arguments
- **Assignment:**
  - implement a parallel and vectorized min-cost triangulation
  - parallelize the code using OpenMP 4.0 or higher
  - upload your solution to UploadSystem
- **Flags for g++ (used by UploadSystem)**
  - `-fopt-info-vec -Ofast -std=c++14 -march=native -fopenmp`

Tricky issues

Tricky issues:

- Do not forget to optimize the sequential code (memory access, vectorization, ...).
- You should carefully think about which OpenMP schedule to use and what is parallelizable.
- Make you sure that your code is vectorized, e.g. use '-fopt-info-vec' option with GCC. See the compiler output after uploading to UploadSystem, it should inform you whether the vectorization is successful.
- Square root can be vectorized if you use:
  - GCC 4.9 and newer, glibc 2.22 and newer
  - Intel compiler 15 and newer
- If you use Clang it may be beneficial to tune your code without the square root. Afterwards, you can add the square root and verify the vectorization in the output of UploadSystem.
- Compile your code with '-march=native -Ofast -fopenmp' to maximize the probability of the vectorization.