1 計算幾何学とは

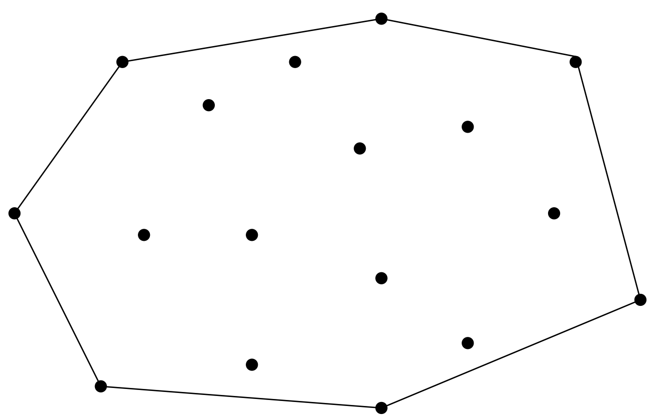
計算幾何学 (computational geometry) は，幾何学に**計算の複雑さ** (computational complexity) の理論を導入して，初等ユークリッド幾何学的な問題を計算機で効率よく処理したり，その問題の難しさを解析する**計算機科学** (computer science) の一分野です．1975 年ころ，Carnegie Mellon 大学の大学院生であった Shamos によって計算幾何学特有の方法論がまとめられて以来，急速に発展し，地理情報処理，VLSI の CAD，パターン認識，コンピュータグラフィクス，ロボティクスなどへの幅広い分野における応用が進んでいます．

1.1 計算幾何問題

例 1.1. **凸包問題**

n 個の点が与えられたとき，これらの点すべてを含む最小の凸多角形 (**凸包**，convex hull) を求めよ．

平面上の点集合に対しては $n \log n$ に比例する回数の基本演算で凸包を求めるアルゴリズムが知られています．

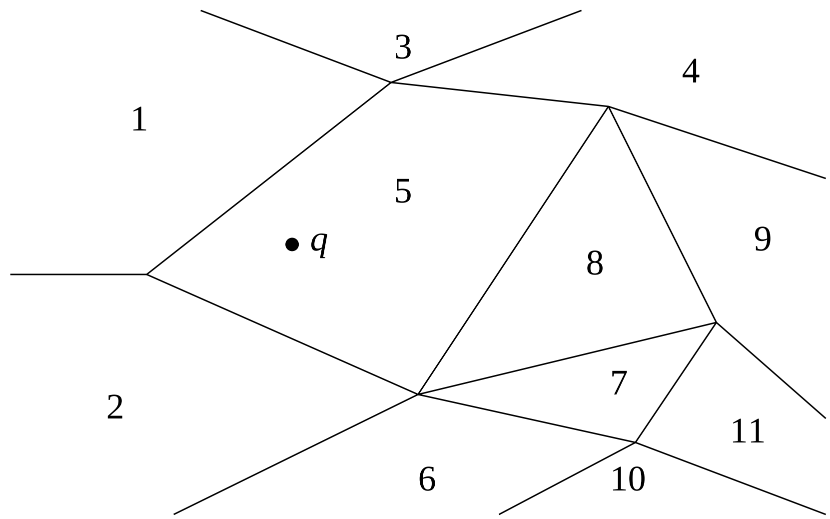


例 1.2. **点位置決定問題**

平面上に直線で描かれた n 点からなる平面地図に対して質問点 q を含む領域を求めよ．

この問題は計算幾何学における最も基本的な問題の 1 つで，$n \log n$ に比例する基本演算からな

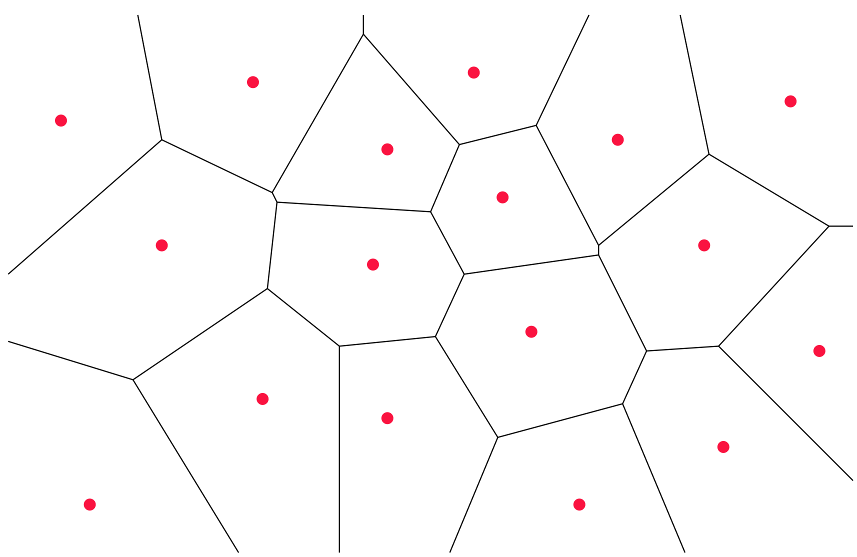
る前処理を施せば，1 回の質問に $\log n$ に比例する基本演算で答を与えるアルゴリズムが存在します．



例 1.3. **全最近点問題**

n 個の点が与えられたとき，それぞれの点に最も近い点を求めよ．

素朴に，すべての点対間の距離を計算すれば n^2 に比例する基本演算で解けます．しかし，それぞれの点が最近点となる領域に平面を分割する**ボロノイ図** (Voronoi diagram) を構成すれば，点位置決定問題として $n \log n$ に比例する基本演算で答が得られます．



例 1.4.　**最近点対問題**

n 個の点が与えられたとき，最も距離の短い点対を求めよ．

この問題もボロノイ図を作っておけば，すべての点対について調べなくとも $n \log n$ に比例する基本演算で解けます．

例 1.5.　**線分の交差判定問題**

平面上に n 本の線分が与えられたとき，これらの線分の間に交差があるかどうかを判定せよ．

すべての線分対に対して交差の有無を調べても答は得られますが，$n \log n$ に比例する基本演算だけで判定ができます．

例 1.6.　**線分の交差列挙問題**

平面上に与えられた n 本の線分の交差判定だけでなく，実際に交点をすべて列挙せよ．

交点数は最大で $n(n-1)/2$ となるので，最悪の場合には n^2 に比例する基本演算が必要です．したがって，素朴に線分対すべての交差を調べる方法が最適ともいえますが，実際の交点数 k が $n(n-1)/2$ に満たない場合には，$(n+k)\log n$ に比例する基本演算で k 個の交点すべてを列挙することもできます．

例 1.7.　**可視性問題**

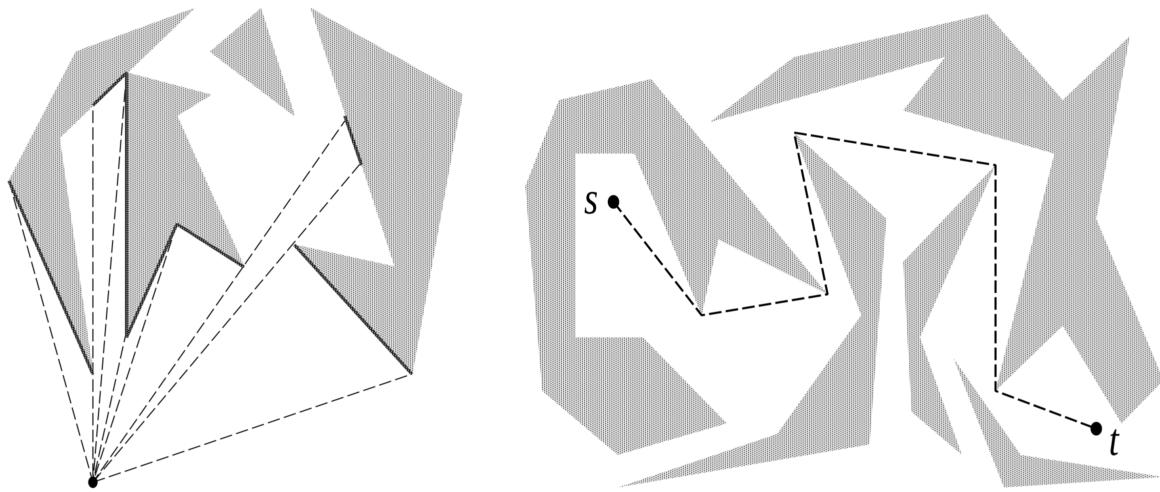
平面上にいくつかの多角形 (障害物) が与えられているとき，任意に指定された点 q から見える部分を求めよ．

多角形が 1 つだけの場合には，多角形の頂点数に比例する基本演算で可視部分を求めることができます．また，m 個の多角形が存在する場合には，n を頂点総数として $n \log m$ に比例する基本演算のアルゴリズムが最も高速とされています．

例 1.8.　**最短経路問題**

平面上にいくつかの多角形 (障害物) が与えられているとき，指定された 2 点を結ぶ最短経路を求めよ．

例 1.7 の可視性問題を解くアルゴリズムを使って互いに見える 2 点を枝で結べば，**可視グラフ** (visibility graph) を作ることができます．この可視グラフ上で数理計画法の最短路問題を解けば答が得られます．可視グラフをつくるのに，頂点総数を n として n^2 に比例する基本演算が必要ですが，グラフの特殊性から最短路の算出には n に比例する基本演算しか掛かりません．



例 1.9. 多角形の基本図形分解

多角形の内部を指定された基本図形に分解せよ．

基本図形として三角形，台形，凸多角形などが考えられますが，基本図形の数を最小化することが目的でなければ，いずれの場合も多角形の頂点の数 n に対して $n \log n$ に比例する基本演算で分解できます．しかし，最小個数の基本図形への分割は，後で説明する NP 完全とよばれる問題クラスに属することがわかっており，効率のよいアルゴリズムの存在は期待できません．