Geometria Computacional

Introdução

Prof. Edson Alves

2018

Faculdade UnB Gama

Sumário

Geometria Computacional

Geometria Computacional e Programação Competitiva

- A Geometria Computacional é uma área recente (anos 70), porém é um tópico frequente em maratonas de programação
- Em geral, os competidores devem postergar a solução de problemas desta área para o meio ou o fim do *contest*, pois estes problemas
  1. possuem muitos *corner cases*, os quais devem ser tratados com atenção
  2. erros de precisão inerentes às variáveis em ponto flutuante podem levar ao WA
  3. envolvem operações triviais quando feitas com caneta e lápis, mas que possuem implementações sofisticadas
  4. apresentam soluções que podem ter codificações longas e tediosas
- Para atacar tais problemas, o competidor tem que se preparar previamente registrando, em suas anotações, as fórmulas básicas e implementações testadas dos algoritmos clássicos da geometria

Observações sobre problemas de Geometria Computacional

- trate todos os *corner cases* ou use implementações que os evitem;
- evite o uso de variáveis do tipo ponto flutuante sempre que possível;
- se for possível usar variáveis em ponto flutuante, defina um limiar ε (por exemplo, $\varepsilon = 10^{-6}$) para o teste de igualdades:
  i. $a = b$ se, e somente se, $|a - b| < \varepsilon$
  ii. $a \leq 0$ se, e somente se, $a < \varepsilon$
  iii. $a \geq 0$ se, e somente se, $a > -\varepsilon$

```
1 // Comparação de igualdade entre variáveis do tipo ponto flutuante
2 #define EPS 1e-9
3
4 bool equals(double a, double b)
5 {
6     return fabs(a - b) < EPS;
7 }
```

Observações sobre problemas de Geometria Computacional

- caso seja necessário usar variáveis do tipo ponto flutuante, utilize **double** ao invés de **float**
- se a precisão do tipo **float** (32 *bits*) e do **double** (64 *bits*) não forem suficientes, utilize o tipo **long double** (80 *bits*)
- tome cuidado com a impressão do zero em ponto flutuante: em determinados casos, a saída ser precedida com o sinal negativo

```
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << -1.0 * 0 << '\n';
6
7     return 0;
8 }
```

Abordagens para um problema de Geometria Computacional

- Há três abordagens possíveis na busca da solução de um problema de Geometria Computacional: Geometria Analítica, Geometria Plana e Álgebra Linear
- Na Geometria Analítica:
  - as figuras geométricas são localizadas no espaço através das coordenadas de seus pontos ou vértices
  - são usadas equações para representar figuras e relações
  - novas relações podem ser deduzidas através da combinação e manipulação destas expressões
- Na Geometria Plana:
  - as figuras são descritas por suas propriedades
  - a posição absoluta no espaço não é importante, apenas a distância relativa entre duas ou mais figuras
  - as relações são descobertas através de simetrias e semelhanças

Abordagens para um problema de Geometria Computacional

- Na Álgebra Linear:
  - segmentos de retas são interpretados como vetores
  - transformações como rotações e translações podem simplificar problemas ao deslocar os entes geométricos para uma nova origem
  - transformações lineares são representadas como matrizes, e sua composição equivale a uma multiplicação matricial
- A Geometria Analítica é a mais comum entre as três abordagens
- A Geometria Plana é a abordagem menos comum, mas pode simplificar os problemas quando bem utilizada
- A Álgebra Linear é útil para descobrir relações que seriam trabalhosas de se verificar utilizando apenas a Geometria Analítica

Referências

1. **HALIM**, Felix; **HALIM**, Steve. *Competitive Programming 3*, 2010.
2. **LAAKSONEN**, Antti. *Competitive Programmer's Handbook*, 2018.
3. **De BERG**, Mark; **CHEONG**, Otfried. *Computational Geometry: Algorithms and Applications*, 2008.
4. Jason Turner. *Negative Zero*. Youtube, 2017[1].

[1] https://www.youtube.com/watch?v=18v-fb_TepU