# Algoritmos para união de círculos e polígonos

Luís Fernando Schultz Xavier da Silveira

Dissertação apresentada ao
Instituto de Matemática e Estatística
da Universidade de São Paulo
para obtenção do título de
Mestre em Ciências

Programa: Ciência da Computação
Orientador: Profª. Drª. Cristina Gomes Fernandes

Durante o desenvolvimento deste trabalho, o autor recebeu auxílio financeiro do CNPq.

6 de Março de 2016

# Algoritmos para união de círculos e polígonos

Esta versão da dissertação contém as correções e alterações sugeridas pela Comissão Julgadora durante a defesa da versão original do trabalho, realizada em 23/01/2015. Uma cópia da versão original está disponível no Instituto de Matemática e Estatística da Universidade de São Paulo.

Comissão Julgadora:

- Prof$^{\underline{a}}$. Dr$^{\underline{a}}$. Cristina Gomes Fernandes (orientadora) – IME-USP
- Prof. Dr. Jorge Stolfi – UNICAMP
- Prof. Dr. Arnaldo Mandel – IME-USP

# Agradecimentos

Agradeço enormemente meus pais e minha família, especialmente minha avó Neuza, por todo o apoio incondicional durante toda minha vida.

Agradeço minha namorada, Helena, pelo apoio que me deu e pela paciência que teve comigo.

Agradeço ao professor Daniel Santana de Freitas da Universidade Federal de Santa Catarina, também por toda a paciência que teve em me orientar na graduação e por nossas interessantes discussões.

Agradeço aos professores Jáuber Cavalcante de Oliveira e Ivan Pontual Costa e Silva por suas excelentes aulas de cálculo, que eu considero as minhas primeiras reais aulas de matemática.

Agradeço aos organizadores das diversas olimpíadas de matemática e computação, que não só contribuíram imensamente em meu aprendizado mas também que, para mim, representam a última esperança do ensino básico neste país.

Agradeço minha banca avaliadora pela leitura do trabalho e pelos comentários que foram tecidos. Em particular, agradeço o professor Arnaldo Mandel por suas simples ideias para as provas do lema 4 e da proposição 25.

# Resumo

Luís Fernando Schultz Xavier da Silveira.
**Algoritmos para união de círculos e polígonos**. 2014.
Dissertação (Mestrado) – Instituto de Matemática e Estatística, Universidade de São Paulo, 2014.

Este trabalho aborda dois problemas de geometria computacional: união de círculos e união de (vários) polígonos. Para o problema da união de círculos, os principais algoritmos da literatura são revisados e um algoritmo simples, porém ineficiente, é introduzido. Este algoritmo é então adaptado para resolver o problema da união de polígonos, produzindo um algoritmo que é competitivo com o estado da arte e, dependendo da aplicação, utiliza menos armazenamento.

**Palavras-chave:** algoritmos, geometria computacional, círculos, polígonos, linha de varredura.

# Abstract

Luís Fernando Schultz Xavier da Silveira.
**Algorithms for the union of circles and polygons**. 2014.
Dissertação (Mestrado) – Instituto de Matemática e Estatística, Universidade de São Paulo, 2014.

This work deals with two problems from the field of computational geometry: union of circles and union of (many) polygons. For the union of circles problem, the main algorithms in the literature are revised and a simple, albeit inefficient, algorithm is introduced. This algorithm is then adapted to solve the union of polygons problem, resulting in an algorithm that is competitive with the state of the art and, depending on the application, makes use of less storage.

**Keywords:** algorithms, computational geometry, circles, polygons, sweep-line.

# Conteúdo

# Introdução

A área de geometria computacional é uma área única em ciência da computação, pois ela liga as disciplinas aparentemente não relacionadas de geometria (usualmente euclidiana) e combinatória. Em geometria estudamos problemas matemáticos envolvendo entidades geométricas, como por exemplo o problema de obter a área de um triângulo a partir dos comprimentos de seus lados, enquanto que em combinatória estudamos problemas sobre estruturas discretas (ditas estruturas combinatórias), como o problema da árvore geradora mínima. Deste modo, não é surpreendente que geometria computacional seja uma área cujo enfoque são estruturas combinatórias definidas com base em propriedades geométricas. Por exemplo, o problema do casco convexo [4] consiste em encontrar uma estrutura combinatória (no caso, um subconjunto de um dado conjunto de pontos) que possui uma propriedade geométrica ("envolver" os pontos do conjunto).

Historicamente, ambas as áreas trataram de algoritmos. Em geometria, os gregos enfatizaram métodos de construção de entidades geométricas com régua e compasso e, em combinatória, muitas vezes buscamos algoritmos que constroem as estruturas combinatórias que desejamos. Em geometria computacional, no entanto, nossa noção de algoritmo é herdada da combinatória. Por exemplo, no problema do casco convexo, buscamos um algoritmo que tenha como entrada um conjunto de pontos e que devolva em sua saída nosso desejado subconjunto. Um excelente e elegante algoritmo para resolver a versão bidimensional deste problema deve-se a Andrew [4].

Os problemas que tratamos neste trabalho não têm nenhuma relação direta com o do casco convexo, no entanto. Estes problemas são derivados do que chamamos de geometria plana construtiva, uma técnica de composição de figuras (conjuntos de pontos) mais complexas a partir de figuras mais simples. Inicialmente começamos com figuras "primitivas", como círculos e polígonos, e usamos operações *booleanas* sobre estas figuras para obter novas figuras. Esta ideia é bastante presente no chamado *computer aided design*, onde projetistas aceleram seu trabalho com o uso do computador, usualmente munido de vários algoritmos de geometria computacional.

Claro, descrever um sistema completo de geometria plana construtiva é uma tarefa extremamente difícil. Em primeiro lugar, muitos dos problemas nesta área não têm soluções eficientes. Porém, mais importante do que isso, devido ao fato de que computadores são essencialmente máquinas de estados finitas ou, para os mais idealistas, máquinas de Turing, não é possível representar conjuntos quaisquer de pontos do plano em computadores. Por isso, todo sistema de geometria plana construtiva que possui a ambição de ser implementado precisa impor restrições para que um problema equivalente de geometria computacional que admite uma solução (idealmente eficiente) seja derivado do sistema.

No nosso caso, na verdade restringimos nosso modelo significativamente. Estamos interessados apenas em computar uniões. Ainda, nossas figuras primitivas devem ser todas círculos ou todas polígonos. Chegamos assim em dois problemas de geometria computacional: a união de círculos e a união de polígonos. Curiosamente, no entanto, não é exatamente simples definir estes problemas rigorosamente e um dos impedimentos é como representamos computacionalmente uma união de círculos ou uma união de polígonos. Tudo isso será tratado com o devido rigor adiante. Porém, para motivar um pouco mais o leitor, adiantamos que será possível calcular a área ou o centro de massa destas uniões facilmente.

Escolhemos estes dois problemas porque suas soluções elucidam várias técnicas importantes em algoritmos de geometria computacional relacionados a este tópico, entre eles a representação computacional de (alguns) conjuntos de pontos, diagramas de Voronoi e outros diagramas relacionados e também, é claro, linhas de varredura.

O problema da união de círculos é extremamente bem estudado na literatura. Muitos dos trabalhos estão preocupados, como nós, em produzir a "borda" da união, enquanto outros tratam apenas da obtenção de certas propriedades da união, como área e centro de massa. Adiante, iremos mostrar que um algoritmo que consegue computar a "borda" da união em um certo formato imediatamente produz algoritmos similarmente eficientes para computar a área e o centro de massa. Tendo isto em mente, apresentamos um resumo em ordem cronológica dos principais trabalhos da literatura de geometria computacional que lidam com este tópico:

- Em 1978, Shamos defende sua tese de doutorado [21], trabalho considerado por muitos o marco inicial da área de geometria computacional. Nela é enunciando o problema de computar a área da união de vários círculos;
- Em 1980, Brown defende também sua tese de doutorado [11]. Ele mostra como diversos problemas de geometria computacional podem ser reduzidos a outros problemas através de transformações geométricas. Em particular, ele mostra como transformar uma instância do problema da união de círculos em um programa linear em três dimensões onde há uma restrição para cada círculo. Ele também mostra como programas lineares gerais em três dimensões com $n$ restrições podem ser resolvidos em tempo $\mathcal{O}(n \log n)$ e espaço $\mathcal{O}(n)$. Ainda nesta tese, Brown demonstra que é impossível produzir um algoritmo para calcular a "borda" da união de $n$ círculos como um conjunto de curvas fechadas que seja mais rápido do que $\mathcal{O}(n \log n)$. Até hoje não há um algoritmo (determinístico) para computar a união (ou mesmo a área da união) mais rápido do que seu algoritmo e também foi provado que em modelos computacionais mais simples $\Omega(n \log n)$ é de fato um limite inferior para o problema. Desta forma, em apenas 2 anos, ele essencialmente resolve o problema proposto por Shamos, apresentando o primeiro algoritmo $\mathcal{O}(n \log n)$ para o problema, que será brevemente abordado na seção 3.5;
- Em 1983, Spirakis [25] publica um algoritmo simples que calcula a união de $n$ círculos em tempo $\mathcal{O}(n^2)$ e espaço $\mathcal{O}(n)$. Porém, mais importante do que isto, seu trabalho mostra algoritmos probabilísticos que, em particular, aproximam bem a área da união de $n$ círculos em tempo e espaço $\mathcal{O}(n)$, mas que não conseguem obter a "borda" da união;
- Em seu artigo publicado em 1985 [23], Sharir utiliza uma complicada estrutura combinatória para produzir um algoritmo que calcula a união de $n$ círculos em

tempo $\mathcal{O}(n \log^2 n)$ e espaço $\mathcal{O}(n)$;

- No mesmo ano, Imai, Iri e Murota publicam um artigo [19] que mostra como utilizar uma estrutura combinatória (um pouco) mais simples que a usada por Sharir e essencialmente a mesma ideia de algoritmo para produzir um algoritmo capaz de unir $n$ círculos em tempo $\mathcal{O}(n \log n)$ e espaço $\mathcal{O}(n)$. Este algoritmo é extremamente elegante e merece seu lugar na seção 3.6 deste trabalho; e
- Em 1988, Aurenhammer [5] publica um *survey* sobre algoritmos para computar a união de círculos e mostra seus próprios algoritmos para bolas em dimensões maiores. Observamos, no entanto, que este artigo diz em sua seção 5 que o algoritmo de Sharir [23] era o único algoritmo determinístico eficiente para computar a área da união de círculos, o que é minimamente curioso dado que ele cita ambos os algoritmos de Brown [11] e de Imai, Iri e Murota [19] em sua seção 2.

Observamos ao leitor que as complexidades destes algoritmos não necessariamente representam suas complexidades reais, mas suas complexidades algébricas, como definiremos na seção 1.5. Porém, estas complexidades são apropriadas para comparar estes algoritmos entre si.

Já o problema da união de polígonos foi, para nós, extremamente difícil de localizar na literatura. Claro, a literatura está repleta de bons algoritmos para unir dois polígonos, como os algoritmos apresentados em [20, seção 7.8] e [14, seção 2.4]. Porém, como a união de dois polígonos não necessariamente é um polígono, a técnica de divisão e conquista não se aplica imediatamente para produzir um algoritmo capaz de unir uma quantidade arbitrária de polígonos.

Na verdade, embora a técnica de divisão e conquista não se aplique imediatamente, se generalizarmos o problema um pouco, ela pode se aplicar. Existe o conceito de polígonos de Nef [10], que essencialmente são o resultado de operações *booleanas* sobre semiplanos (um outro tipo de sistema de geometria plana construtiva). Polígonos podem ser decompostos em triângulos, ou seja, triangulados. Cada triângulo é uma intersecção de três semiplanos e logo um polígono de Nef, e assim cada polígono, sendo a união de triângulos, também é um polígono de Nef. Triangulações de polígonos podem ser computadas eficientemente, um algoritmo ótimo devendo-se a Chazelle [12]. Ainda, a biblioteca CGAL [1] possui implementações de operações *booleanas* binárias eficientes sobre polígonos de Nef. A grande diferença, como o leitor pode já ter percebido, é que a união de dois polígonos de Nef é sempre um polígono de Nef, de forma que a ideia de divisão e conquista se aplica imediatamente. Embora não seja imediatamente claro quais complexidades este tipo de algoritmo pode atingir, na seção 4.2 fazemos uma estimativa para fins de comparação com o algoritmo apresentado neste trabalho.

Observamos também que ambas as bibliotecas CGAL [1] e LEDA [3] possuem implementações de algoritmos para unir um número arbitrário de polígonos. Ambas resolvem este problema generalizando o conceito de polígono para o que a biblioteca CGAL chama de polígonos com buracos (*polygon with holes* sendo o termo usado). A biblioteca CGAL entra um pouco nos detalhes de implementação em sua documentação, mas não estabelece as complexidades destes algoritmos. Novamente estimamos as complexidades dos algoritmos desta biblioteca e as comparamos com a de nosso algoritmo na seção 4.2.

Finalmente, o trabalho de Berberich *et al* [8] apresenta uma técnica extremamente geral baseada em computação algébrica e algoritmos de linha de varredura para realizar operações *booleanas* (regularizadas) sobre uma generalização de polígonos onde as arestas podem ser arcos de cônicas. Certamente, este trabalho trata de um problema muito mais geral do que os abordados aqui. Ele faz uso de várias técnicas de computação algébrica que estudamos durante o mestrado. Contudo, não ficou claro se o algoritmo nele apresentado, quando especializado para operar apenas em polígonos comuns, oferece uma performance superior ou igual a do nosso algoritmo.

O restante deste trabalho está organizado da seguinte forma:

- O capítulo 1 trata de todos os conceitos introdutórios que um leitor não familiar com análise real, geometria analítica, geometria computacional e alguns outros tópicos precisa conhecer para entender completamente este trabalho;
- O capítulo 2 trata de diversas propriedades de conjuntos e suas fronteiras, além de definir representações computacionais para certos tipos de conjunto. Ainda, ele mostra como algumas propriedades destes conjuntos (área e centro de massa) podem ser obtidas a partir desta representação;
- O capítulo 3 mostra o nosso primeiro algoritmo para o problema de encontrar a união de círculos e também os algoritmos ótimos de Brown e de Imai, Iri e Murota usando o *framework* desenvolvido nos capítulos anteriores; e
- O capítulo 4 mostra um algoritmo de linha de varredura (*sweep-line*) eficiente para resolver o problema da união de (possivelmente muitos) polígonos derivado da melhoria em que originalmente havíamos pensado para nosso algoritmo para o problema com círculos.

# Capítulo 1

# Preliminares e notação

Neste capítulo convencionamos parte da notação que usaremos neste trabalho. Também lembraremos o leitor de alguns resultados importantes que são pré-requisitos para ele.

## 1.1 Funções, conjuntos e números

Usamos duas notações semelhantes para categorizar e definir funções. Por exemplo, para categorizar uma função, escrevemos uma expressão da forma "seja $f : A \to B$ uma função" indicam que $f$ é uma função cujo domínio, denotado por $\mathrm{Dom}(f)$, é o conjunto $A$ e cujo contradomínio é o conjunto $B$.

Por outro lado, para definir uma função escrevemos algo como "seja

$$\begin{array}{rccc} s: & \mathbb{N} & \to & \mathbb{N} \\ & n & \mapsto & n+1 \end{array}$$

a função sucessor em $\mathbb{N}$". Esta notação é similar à anterior. A parte "$s : \mathbb{N} \to \mathbb{N}$" indica que $s$ é uma função de $\mathbb{N}$ em $\mathbb{N}$. O símbolo a esquerda de "$\mapsto$" é o nome de uma variável, que é a única variável livre na expressão a direita de "$\mapsto$". Para definir o comportamento da função $s$ em um valor (número natural), substitua a variável pelo valor e avalie a expressão. Porém, em alguns casos iremos usar várias variáveis a esquerda, principalmente quando o domínio é um produto cartesiano, e assim a expressão a direita deve ter exatamente estas variáveis como variáveis livres. Ilustramos isto definindo a rotação por 90 graus do plano como a função

$$\begin{array}{rccc} \rho_{\frac{\pi}{2}}: & \mathbb{R}^2 & \to & \mathbb{R}^2 \\ & (x,y) & \mapsto & (-y,x). \end{array}$$

Em vários momentos neste trabalho faremos uso de reticências, principalmente para definir conjuntos. Por exemplo, escrevemos $\{a, a+1, \ldots, b\}$, onde $a, b \in \mathbb{Z}$, para denotar o conjunto $\{x \in \mathbb{Z} : a \leq x \leq b\}$. Porém, é importante notar que nem sempre vale que $a \in \{a, a+1, \ldots, b\}$, pois podemos ter $b < a$. Isto é especialmente importante quando escrevemos o conjunto $\{0, 1, \ldots, n-1\}$ com $n \in \mathbb{N}$ quando $n = 0$.

Mostramos agora algumas notações que podem ser definidas mais formalmente.

**Definição 1** — Seja $f : A \to B$ uma função de um conjunto $A$ em um conjunto $B$ e seja $X \subseteq A$ um subconjunto de $A$. Denotamos então o conjunto $\{f(x) : x \in X\}$, dito a ***imagem*** de $X$ por $f$, por $f(X)$.

**Definição 2** — Seja $f : A \to B$ uma função de um conjunto $A$ em um conjunto $B$. Dizemos então que a ***imagem*** de $f$ é o conjunto $\operatorname{Im}(f) = f(A) \subseteq B$.

**Definição 3** — Seja $f : A \to B$ uma função de um conjunto $A$ em um conjunto $B$ e seja $Y \subseteq B$ um subconjunto de $B$. Denotamos então o conjunto $\{x \in A : f(x) \in Y\}$, dito a ***imagem inversa*** de $Y$ por $f$, por $f^{-1}(Y)$.

**Definição 4** — O ***sinal*** de um número real $x \in \mathbb{R}$ é

$$\operatorname{sign} x = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1, & x > 0. \end{cases}$$

## 1.2 Ordens

Neste trabalho, fazemos alguns usos do conceito de ordens parciais.

**Definição 5** — Uma ***ordem parcial*** sobre um conjunto $S$ é uma relação binária $< \subseteq S \times S$ que satisfaz duas propriedades:

- Irreflexividade: para todo elemento $a \in S$, jamais vale que $a < a$; e
- Transitividade: para todos os elementos $a, b, c \in S$, se $a < b$ e $b < c$, então $a < c$.

É comum o uso do símbolo "$>$" para denotar a relação transposta da relação $<$. Assim, "$a > b$" significa $b < a$ e "$>$" por si só significa $\big\{(b, a) : (a, b) \in <\big\}$.

Há uma propriedade tão comum de ordens parciais que usualmente ela é incluída na própria definição.

**Proposição 1** — Seja $<$ uma ordem parcial sobre um conjunto $S$. Então, para todos os elementos $a, b \in S$, jamais vale simultaneamente que $a < b$ e $b < a$.

**Prova** — Se $a < b$ e $b < a$, então, pela transitividade, $a < a$, o que é uma contradição devido à irreflexividade de $<$. $\square$

Uma ideia muito importante quando trabalhamos com ordens parciais é a noção de comparabilidade e incomparabilidade.

**Definição 6** — Seja $<$ uma ordem parcial sobre um conjunto $S$. Dizemos então que dois elementos $a, b \in S$ são ***comparáveis*** por $<$ se $a < b$ ou se $b < a$.

**Definição 7** — Seja $<$ uma ordem parcial sobre um conjunto $S$. Dizemos então que a ***relação de incomparabilidade*** de $<$ é a relação binária $\lesssim \subseteq S \times S$ definida por $a \lesssim b$ se, e somente se, $a$ e $b$ não são elementos comparáveis por $<$.

Imaginamos que o leitor já esteja familiarizado com a definição a seguir.

**Definição 8** — Uma ***ordem total*** sobre um conjunto $S$ é uma ordem parcial $<$ sobre $S$ tal que todos os elementos de $S$ são dois a dois comparáveis por $<$.

Apresentamos então o conceito de uma ordem estrita fraca, que vagamente corresponde ao conceito de que tipo de objeto matemático um procedimento de comparação passado a uma rotina de ordenação em uma linguagem de programação deve implementar.

**Definição 9** — Dizemos que uma ordem parcial $<$ sobre um conjunto $S$ é uma ***ordem estrita fraca*** se a relação $\lesssim$ de incomparabilidade de $<$ é transitiva, ou seja, se, para todos os elementos $a, b, c \in S$ tais que $a \lesssim b$ e $b \lesssim c$, sempre vale que $a \lesssim c$.

Uma consequência desta definição é a seguinte.

**Proposição 2** — A relação de incomparabilidade de toda ordem estrita fraca é uma relação de equivalência.

## 1.3 Análise real e geometria analítica

A seguir mostramos várias definições de análise real em várias e duas dimensões que serão usados no decorrer deste trabalho. Quando a menção de um resultado destas áreas valer a pena, o resultado correspondente será também enunciado.

**Definição 10** — Dados uma dimensão $d \in \mathbb{N}$, um ponto $x \in \mathbb{R}^d$ e um raio $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$, dizemos que a ***bola aberta*** de centro $x$ e raio $\varepsilon$ é o conjunto

$$B_\varepsilon(x) = \big\{y \in \mathbb{R}^d : ||y - x|| < \varepsilon\big\}.$$

**Definição 11** — O ***interior*** de um conjunto $X \subseteq \mathbb{R}^d$ (onde $d \in \mathbb{N}$ é uma dimensão) é o conjunto

$$X^\circ = \big\{x \in X : \exists \varepsilon \in \mathbb{R}, \varepsilon > 0 : B_\varepsilon(x) \subseteq X\big\}.$$

**Definição 12** — A ***fronteira*** de um conjunto $X \subseteq \mathbb{R}^d$ (onde $d \in \mathbb{N}$ é uma dimensão) é o conjunto

$$\partial X = \mathbb{R}^d \setminus \Big( X^\circ \cup \big(\mathbb{R}^d \setminus X\big)^\circ \Big).$$

**Proposição 3** — Seja $d \in \mathbb{N}$ uma dimensão, seja $X \subseteq \mathbb{R}^d$ um conjunto aberto e seja $f : X \to \mathbb{R}^d$ uma função contínua. Então, para todo conjunto aberto $Y \subseteq \mathbb{R}^d$, a imagem inversa $f^{-1}(Y) \subseteq X$ é também um conjunto aberto.

**Proposição 4** — Seja $d \in \mathbb{N}$ uma dimensão, seja $\Omega \subseteq \mathbb{R}^d$ um conjunto aberto, seja $f : \Omega \to \mathbb{R}^d$ uma função contínua e seja $X \subseteq \Omega$ um subconjunto de $\Omega$. Então $f(\partial X \cap \Omega) \subseteq \partial f(X)$.

**Corolário 1** — Seja $d \in \mathbb{N}$ uma dimensão, sejam $\Omega_0, \Omega_1 \subseteq \mathbb{R}^d$ conjuntos abertos, seja $f : \Omega_0 \to \Omega_1$ uma função bijetiva contínua cuja inversa $f^{-1}$ é contínua e seja $X \subseteq \Omega_0$ um subconjunto de $\Omega_0$. Então $f(\partial X \cap \Omega_0) = \partial f(X) \cap \Omega_1$.

**Definição 13** — Um conjunto $X \subseteq \mathbb{R}^d$ (onde $d \in \mathbb{N}$ é uma dimensão) é dito ***limitado*** se existe algum $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ tal que $X \subseteq B_\varepsilon(0)$ (0 sendo a origem).

**Definição 14** — Um conjunto $X \subseteq \mathbb{R}^d$ (onde $d \in \mathbb{N}$ é uma dimensão) é dito ***conexo*** se não existirem dois conjuntos $A, B \subseteq \mathbb{R}^d$ abertos, disjuntos, que intersectam $X$ e tais que $X \subseteq A \cup B$.

**Lema 1** — Se $\mathcal{C}$ é uma família de conjuntos conexos cuja intersecção é não-vazia, então a união dos conjuntos em $\mathcal{C}$ é conexa.

**Corolário 2** — Se $d \in \mathbb{N}$ é uma dimensão, $X \subseteq \mathbb{R}^d$ e $x \in X$, a união de todos os subconjuntos de $X$ que são conexos e contém $x$ é em si um subconjunto conexo de $X$ que contém $x$.

**Definição 15** — Se $d \in \mathbb{N}$ é uma dimensão, $X \subseteq \mathbb{R}^d$, $x \in X$, então dizemos que a ***componente conexa*** de $x$ em $X$ é a união de todos os subconjuntos conexos de $X$ que contém $x$.

**Definição 16** — O ***conjunto das componentes conexas*** de um conjunto $X \subseteq \mathbb{R}^d$ (onde $d \in \mathbb{N}$ é uma dimensão) é o conjunto de todas as componentes conexas de cada elemento $x \in X$ em $X$.

**Definição 17** — Seja $d \in \mathbb{N}$ uma dimensão e seja $X \subseteq \mathbb{R}^d$ um conjunto. Dizemos então que a ***função característica*** de $X$ é a função $\chi_X : \mathbb{R}^d \to \{0, 1\}$ tal que, para todo $x \in \mathbb{R}^d$, $\chi_X(x) \neq 0$ se, e somente se, $x \in X$.

**Definição 18** — Seja $X \subseteq \mathbb{R}^2$ um conjunto. Se, para todo $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$, existirem listas $x_0 x_1 \cdots x_{n-1}$ de pontos em $\mathbb{R}^2$ e $\varepsilon_0 \varepsilon_1 \cdots \varepsilon_{n-1}$ de números reais positivos tais que

$$X \subseteq \bigcup_{k=0}^{n-1} B_{\varepsilon_k}(x_k) \quad \text{e} \quad \sum_{k=0}^{n-1} \varepsilon_k^2 < \varepsilon,$$

dizemos que o conjunto $X$ possui ***medida de Jordan nula*** e que ele é ***J-desprezível***.

**Definição 19** — Um conjunto $X \subseteq \mathbb{R}^2$ é dito ***Jordan mensurável***, ou simplesmente ***J-mensurável***, se $X$ for limitado e $\partial X$ for J-desprezível.

**Definição 20** — Seja $X \subseteq \mathbb{R}^2$ um conjunto J-mensurável. Dizemos então que a ***área*** de $X$ é o valor

$$\operatorname{area} X = \int_X 1 \, \mathrm{d}p.$$

**Definição 21** — Seja $X \subseteq \mathbb{R}^2$ um conjunto J-mensurável tal que $\operatorname{area} X > 0$. Dizemos então que o ***centro de massa*** ou ***centroide*** de $X$ é o ponto

$$\operatorname{centroid} X = \frac{\displaystyle\int_X p \, \mathrm{d}p}{\operatorname{area} X}.$$

Agora apresentamos quais definições e resultados de geometria analítica serão usados neste trabalho.

**Definição 22** — Uma função $\gamma : [0, 1] \to \mathbb{R}^2$ contínua é dita uma ***curva***.

**Definição 23** — A ***curva reversa*** de uma curva $\gamma$ é a curva

$$\begin{aligned} \operatorname{rev} \gamma : \quad [0,1] &\to \mathbb{R}^2 \\ t &\mapsto \gamma(1-t). \end{aligned}$$

**Definição 24** — Sejam $\gamma_0, \gamma_1, \ldots, \gamma_{n-1}$ curvas tais que $n \neq 0$ e $\gamma_{i-1}(1) = \gamma_i(0)$ para todo $i \in \mathbb{N}$ tal que $0 < i < n$. Dizemos então que a ***concatenação*** de $\gamma_0, \gamma_1, \ldots, \gamma_{n-1}$ é a curva $\gamma : [0,1] \to \mathbb{R}^2$ dada por

$$\gamma\Big(\frac{i+t}{n}\Big) = \gamma_i(t)$$

para todo $t \in [0,1]$ e $i \in \mathbb{N}$ com $0 \leq i < n$. Ainda, denotamos a curva $\gamma$ por $\gamma_0\gamma_1 \cdots \gamma_{n-1}$ ou por

$$\prod_{i=0}^{n-1} \gamma_i.$$

**Teorema 1** (do valor intermediário para curvas) — Seja $A \subseteq \mathbb{R}^2$ um conjunto aberto e seja $\gamma : [0,1] \to \mathbb{R}^2$ uma curva tal que $\gamma(0) \in A$ e $\gamma(1) \notin A$. Então existe um valor real $a \in [0,1]$ tal que $\gamma(a) \in \partial A$.

**Definição 25** — Sejam $u, v \in \mathbb{R}^2$ dois pontos no plano tais que $u \neq v$. Então a curva

$$\begin{array}{rccl} \gamma : & [0,1] & \to & \mathbb{R}^2 \\ & t & \mapsto & (1-t)u + tv \end{array}$$

é dita a ***parametrização canônica*** do segmento orientado $\overrightarrow{uv}$ e será denotada por $\widetilde{uv}$.

**Definição 26** — Um ***círculo*** $C$ é um subconjunto de $\mathbb{R}^2$ da forma

$$\{p \in \mathbb{R}^2 : ||p - c|| \leq r\},$$

onde $c = (c_x, c_y) \in \mathbb{R}^2$ é um ponto, dito o ***centro*** de $C$, e $r$ é um número real positivo, dito o ***raio*** de $C$. Ainda, a curva

$$\begin{array}{rccl} \widetilde{C} : & [0,1] & \to & \mathbb{R}^2 \\ & t & \mapsto & (c_x + r\cos 2\pi t, c_y + r\sin 2\pi t) \end{array}$$

é dita a ***parametrização canônica*** de $C$.

**Definição 27** — Seja $C$ um círculo e sejam $p, q \in \partial C$ dois pontos em sua fronteira. Considere então a extensão periódica $\gamma : \mathbb{R} \to \mathbb{R}^2$ de $\widetilde{C}$ e os dois únicos argumentos $t_0, t_1 \in \mathbb{R}$ tais que $0 \leq t_0 < 1$, $t_0 \leq t_1 < t_0 + 1$, $\gamma(t_0) = p$ e $\gamma(t_1) = q$. Dizemos que o ***arco de circunferência*** $\alpha$ entre $p$ e $q$ sobre $C$ é a imagem $\operatorname{arc}(C, p, q)$ da curva

$$\begin{array}{rccl} \gamma' : & [0,1] & \to & \mathbb{R}^2 \\ & t & \mapsto & \gamma\big((1-t)t_0 + tt_1\big). \end{array}$$

Também dizemos que a ***parametrização canônica*** deste arco é a curva $\widetilde{\alpha} = \gamma'$.

**Definição 28** — Um ***arco de circunferência orientado*** é um par $\alpha = (\beta, \sigma)$ onde $\beta = \operatorname{arc}(C, p, q)$ é um arco de circunferência, sendo $C$ um círculo e $p, q \in \partial C$, e $\sigma \in \{-1, 1\}$ é um sinal. O arco orientado $\alpha$ está em ***sentido anti-horário*** se $\sigma > 0$ e em ***sentido horário*** se $\sigma < 0$. Denotaremos $\alpha$ por $\overrightarrow{\operatorname{arc}}(C, p, q)$ se ele estiver em sentido anti-horário e por $\overleftarrow{\operatorname{arc}}(C, p, q)$ se ele estiver em sentido horário. Diremos também que a ***parametrização canônica*** de $\alpha$ é

$$\widetilde{\alpha} = \begin{cases} \widetilde{\beta}, & \sigma > 0 \\ \operatorname{rev} \widetilde{\beta}, & \sigma < 0. \end{cases}$$

Finalmente, denotaremos o arco de circunferência orientado $(\beta, -\sigma)$ por $\operatorname{rev} \alpha$.

**Definição 29** — Uma curva $\gamma : [0,1] \to \mathbb{R}^2$ tal que $\gamma(0) = \gamma(1)$ e cuja restrição $\gamma|_{[0,1)}$ ao intervalo semiaberto $[0,1)$ é injetiva é dita uma ***curva de Jordan***.

**Teorema 2** (da curva de Jordan) — Se $\gamma$ é uma curva de Jordan, então o conjunto $\mathbb{R}^2 \setminus \operatorname{Im}(\gamma)$ possui exatamente duas componentes conexas, uma limitada e a outra não.

**Prova** — Esta demonstração é surpreendentemente não trivial e pode ser encontrada em [9]. $\square$

**Definição 30** — Seja $\gamma$ uma curva de Jordan. Iremos chamar a componente conexa limitada de $\mathbb{R}^2 \setminus \operatorname{Im}(\gamma)$ de o ***interior*** de $\gamma$ e, com um certo abuso de notação, denotaremos este interior por $\gamma^\circ$.

**Definição 31** — Sejam $u = (u_x, u_y)$ e $v = (v_x, v_y)$ dois vetores em $\mathbb{R}^2$. O ***produto cruzado*** de $u$ e $v$ é o número real

$$u \times v = u_x v_y - u_y v_x = \det \begin{bmatrix} u_x & v_x \\ u_y & v_y \end{bmatrix}.$$

**Definição 32** — Uma ***curva poligonal*** é uma concatenação de parametrizações canônicas de segmentos de reta.

**Definição 33** — Seja $\gamma : [0,1] \to \mathbb{R}^2$ uma curva de Jordan poligonal e seja $\delta : \mathbb{R} \to \mathbb{R}^2$ sua extensão periódica. Considere o ponto $p$ com maior abscissa dentre os pontos de menor ordenada de $\mathrm{Im}(\gamma)$ e seja $t$ o único valor em $[0,1)$ tal que $\gamma(t) = p$. Se existe $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ tal que para todos os valores $u, v \in \mathbb{R}$ com $t - \varepsilon < u < t < v < t + \varepsilon$ vale que $\big(\delta(v) - p\big) \times \big(\delta(u) - p\big) > 0$, então dizemos que $\gamma$ está em ***sentido anti-horário***.

**Definição 34** — Um ***polígono*** é um subconjunto de $\mathbb{R}^2$ que pode ser expresso na forma $\mathrm{Im}(\gamma) \cup \gamma^\circ$ para alguma curva de Jordan poligonal $\gamma$.

**Definição 35** — Seja $\gamma$ uma curva de Jordan poligonal e seja $P = \mathrm{Im}(\gamma) \cup \gamma^\circ$ um polígono. Então, sendo $\delta : \mathbb{R} \to \mathbb{R}^2$ a extensão periódica de $\gamma$, definimos o conjunto de ***vértices*** de $P$, denotado por $V(P)$, como a imagem por $\delta$ das abscissas em que $\delta$ não é diferenciável.

**Definição 36** — Seja $\gamma$ uma curva de Jordan poligonal em sentido anti-horário com extensão periódica $\delta : \mathbb{R} \to \mathbb{R}^2$ e seja $P = \mathrm{Im}(\gamma) \cup \gamma^\circ$ um polígono. Seja ainda $L = v_0 v_1 \ldots v_{n-1}$ uma lista de todos os vértices do polígono $P$ e $t$ o único valor em $[0,1)$ tal que $\gamma(t) = v_0$. Dizemos então que a lista $L$ está em ***sentido anti-horário*** se, para todos os números naturais $i, j \in \mathbb{N}$ com $i < j < n$, vale que $\delta|_{[t,t+1)}^{-1}(v_i) < \delta|_{[t,t+1)}^{-1}(v_j)$.

**Definição 37** — Seja $P$ um polígono e $L = v_0 v_1 \ldots v_{n-1}$ uma lista dos vértices de $P$ em sentido anti-horário. Definindo $v_n = v_0$, dizemos que o conjunto das arestas de $P$ é o conjunto de segmentos orientados

$$E(P) = \left\{ \overrightarrow{v_i v_{i+1}} : i \in \{0, 1, \ldots, n-1\} \right\}.$$

**Definição 38** — Seja $P$ um polígono e seja $L = v_0 v_1 \ldots v_{n-1}$ a única lista dos vértices de $P$ em sentido anti-horário em que o vértice $v_0$ é o vértice de maior abscissa dentre todos os vértices de $P$ com menor ordenada. Definindo $v_n = v_0$, dizemos que a ***parametrização canônica*** de $P$ é a curva

$$\widetilde{P} = \prod_{i=0}^{n-1} \widetilde{v_i v_{i+1}}.$$

## 1.4 Modelo de computação

Nesta seção tratamos de um assunto pouco relacionado ao restante deste trabalho, porém importante: o modelo de computação adotado.

Ao analisarmos qualquer algoritmo, devemos sempre estabelecer um modelo de computação. Por exemplo, podemos assumir que o computador que irá executar um algoritmo é uma máquina de Turing [24, seção 3.1]. Porém, isto não é muito realista. De fato, algoritmos comuns de ordenação podem requerer um número pelo menos quadrático de operações em uma máquina de Turing para concluir devido à necessidade de "ir e voltar" na fita.

Há outros modelos de computação, como o cálculo $\lambda$. Dentre estes, há vários modelos que representam como a computação é feita no mundo real com bastante precisão. Alguns destes modelos são extensões do cálculo $\lambda$, enquanto que outros são (tais como a máquina de Turing) extensões de autômatos finitos.

A escolha de modelo computacional que fizemos foi o RAM (*random access machine*) com custos logarítmicos, que descrevemos a seguir. Esta escolha foi motivada pela abundância literária de tratados sobre este modelo, um dos quais está em [18, seção 5.1]. A descrição que oferecemos aqui não é a mesma que está nesta referência, mas é basicamente equivalente a ela.

Em primeiro lugar, um computador no modelo RAM com custos logarítmicos não pode ser programado: ele sempre executa o mesmo "programa" em todas as entradas, de forma que a solução de problemas diferentes em princípio requer a construção de computadores diferentes, bem como é o caso com máquinas de Turing. Existe, é claro, o conceito de máquinas de Turing universais [24, seção 4.2], mas, de certa forma, uma máquina de Turing universal também executa sempre o mesmo "programa" (o interpretador) em todas as suas entradas.

Cada um destes computadores é constituído de algumas partes. A primeira delas é o banco de registradores, uma coleção finita $R = \{r_0, r_1, \ldots, r_{n-1}\}$ de dispositivos, ditos registradores, capazes de armazenar um único número inteiro, inicialmente o número 0. A segunda parte é a chamada unidade lógica e aritmética ou ALU (*arithmetic logic unit*). A função desta unidade é repetidamente, quando invocada, acessar registradores no banco de registradores, realizar uma operação lógica ou aritmética nos valores destes registradores e depositar o resultado em outro registrador.

Seguem as operações que podem ser realizadas pela ALU. Cada uma delas é parametrizada sobre uma lista de índices de registradores, não necessariamente distintos. Ainda, cada operação acessa primeiro todos os registradores que tem de acessar antes de depositar valores nos registradores:

- `clear` $(i)$: deposita o valor 0 no registrador $r_i$;
- `inc` $(i)$: obtém o valor $a$ do registrador $r_i$ e deposita o valor $a+1$ no registrador $r_i$;
- `add` $(i, j, k)$: obtém o valor $a$ do registrador $r_j$ e o valor $b$ do registrador $r_k$ e deposita no registrador $r_i$ o valor $a + b$;
- `sub` $(i, j, k)$: obtém o valor $a$ do registrador $r_j$ e o valor $b$ do registrador $r_k$ e deposita no registrador $r_i$ o valor $a - b$;

- `and` $(i, j, k)$: obtém o valor absoluto

$$a = \sum_{d=0}^{\infty} a_d 2^d \in \mathbb{N}$$

do valor do registrador $r_j$ (com $a_d \in \{0, 1\}$ para todo $d \in \mathbb{N}$) e o valor absoluto

$$b = \sum_{d=0}^{\infty} b_d 2^d \in \mathbb{N}$$

do valor do registrador $r_k$ (com $b_d \in \{0, 1\}$ para todo $d \in \mathbb{N}$) e deposita no registrador $r_i$ o valor

$$\sum_{d=0}^{\infty} a_d b_d 2^d;$$

- `or` $(i, j, k)$: obtém o valor absoluto

$$a = \sum_{d=0}^{\infty} a_d 2^d \in \mathbb{N}$$

do valor do registrador $r_j$ (com $a_d \in \{0, 1\}$ para todo $d \in \mathbb{N}$) e o valor absoluto

$$b = \sum_{d=0}^{\infty} b_d 2^d \in \mathbb{N}$$

do valor do registrador $r_k$ (com $b_d \in \{0, 1\}$ para todo $d \in \mathbb{N}$) e deposita no registrador $r_i$ o valor

$$\sum_{d=0}^{\infty} \big(1 - (1 - a_d)(1 - b_d)\big) 2^d;$$

- `lshift` $(i, j, k)$: obtém o valor $a$ do registrador $r_j$, o valor absoluto $b$ do valor do registrador $r_k$ e deposita no registrador $r_i$ o valor $2^b a$;
- `popcount` $(i, j)$: obtém o valor absoluto

$$a = \sum_{d=0}^{\infty} a_d 2^d \in \mathbb{N}$$

do valor do registrador $r_j$ (com $a_d \in \{0, 1\}$ para todo $d \in \mathbb{N}$) e deposita no registrador $r_i$ o número de índices $d \in \mathbb{N}$ tais que $a_d \neq 0$, ou seja, o valor

$$\sum_{d=0}^{\infty} a_d; \text{ e}$$

- `bits` $(i, j, k, l)$: obtém o valor $a$ do registrador $r_j$, o valor $b$ do registrador $r_k$ e o valor $c$ do registrador $r_l$ e deposita no registrador $r_i$ o valor

$$\left\lfloor \frac{|c|}{2^{|a|}} \right\rfloor \bmod 2^{|b|}.$$

Esta última instrução `bits` é tal que se

$$|c| = \sum_{d=0}^{\infty} c_d 2^d \in \mathbb{N}$$

com $c_d \in \{0, 1\}$ para todo $d \in \mathbb{N}$, então ela coloca o valor

$$\sum_{d=0}^{|b|-1} c_{|a|+d} 2^d$$

no registrador $r_i$. Em outras palavras, ela deposita em $r_i$ os *bits* de índices entre $|a|$ (inclusive) e $|a| + |b|$ (exclusive) do valor $|c|$.

Cabe ressaltar que com estas operações é possível replicar as demais operações em um conjunto de instruções usual, como a operação de "ou exclusivo" *bit* a *bit*. As operações de multiplicação de inteiros e de divisão inteira (com quociente e resto) serão abordadas mais adiante, no entanto.

Uma outra parte do computador é a memória, que consiste em um vetor infinito de células $m_0, m_1, \ldots$. Cada uma destas células é capaz de armazenar um número inteiro, e todas elas começam armazenando o número 0. A memória tem acesso ao banco de registradores e é capaz de, quando invocada, realizar duas operações:

- `load` $(i, j)$: obtém o valor $a$ do registrador $r_j$ e deposita no registrador $r_i$ o valor armazenado na célula de memória $m_{|a|}$; e
- `store` $(i, j)$: obtém o valor $a$ do registrador $r_j$ e deposita na célula de memória $m_{|a|}$ o valor do registrador $r_i$.

Finalmente, a última e mais interessante parte do computador é sua unidade de processamento central ou CPU (*central processing unit*). Em nosso modelo, representamos a CPU como um autômato finito [24, seção 1.1] $\mathcal{C} = (Q, \Sigma, \delta, q_0, F)$, onde $Q$ é seu conjunto de estados, $\Sigma = \{-1, 0, 1\}$ é o alfabeto sobre o qual o autômato trabalha, $\delta : Q \times \Sigma \to Q$ é sua função de transição, $q_0 \in Q$ é seu estado inicial e $F \subseteq Q$ é seu conjunto de estados finais.

É dentro deste autômato que reside o "programa" que executa neste computador. Inclusive, um compilador para este modelo recebe como entrada o código fonte do programa em linguagem de alto nível e produz como saída um autômato finito. O que falta, no entanto, é "conectar" este autômato ao resto de nossa máquina.

Para isso, para cada operação

$$o \in \{\texttt{clear}, \texttt{inc}, \texttt{add}, \texttt{sub}, \texttt{and}, \texttt{or}, \texttt{lshift}, \texttt{popcount}, \texttt{bits}, \texttt{load}, \texttt{store}\}$$

de aridade (número de parâmetros) $d \in \mathbb{N}$ e para cada $d$-upla de índices de registradores $I = (i_0, i_1, \ldots, i_{d-1})$ com $i_j \in \{0, 1, \ldots, n-1\}$ para todo $j \in \{0, 1, \ldots, d-1\}$, associamos um subconjunto $E(o, I) \subseteq Q$ dos estados do autômato denominado o conjunto de emissão da operação $o$ com parâmetros $i_0, i_1, \ldots, i_{d-1}$.

Todos estes conjuntos de emissão $E(o, I)$ devem ser dois a dois disjuntos. A razão disso é que, em nosso modelo, toda vez que a CPU $\mathcal{C}$ entra em um estado em um conjunto de emissão (incluindo a "entrada" em $q_0$ antes de qualquer caractere do alfabeto ser processado), a operação correspondente é executada com os parâmetros correspondentes. Assim, desejamos que apenas uma operação seja executada por vez. Um bom motivo para isso é o nosso modelo de transferência do resultado de uma operação para a CPU: o próximo caractere que o autômato irá processar é o sinal ($-1$, 0 ou 1) do resultado da operação. No caso da operação `clear`, este sinal é considerado como sendo 0 e, no caso da operação `store`, este sinal é o sinal do valor armazenado na memória.

O funcionamento do autômato se resume então a emitir operações, processar o caractere (sinal) correspondente a seus resultados através da função de transição, o que resulta em mudanças de estado. Isto é feito até, claro, o autômato atingir um estado que não corresponde à emissão de nenhuma operação (podendo nenhuma operação ser emitida caso $q_0$ seja um destes estados). Se o estado em que o autômato termina for um elemento de $F$, a computação aceita e, caso contrário, rejeita.

Falta ainda mostrar como podemos alimentar um tal computador com uma entrada e como obter dele uma saída. Convencionamos que uma entrada neste modelo é uma lista de números inteiros. Para passar uma entrada $a_0a_1 \cdots a_{M-1}$ para o computador, devemos escrever nas células de memória $m_0, m_1, \ldots, m_M$ os valores $M, a_0, a_1, \ldots, a_{M-1}$ antes do programa começar. Se o problema que estamos tentando resolver for de decisão, a aceitação ou rejeição da CPU é suficiente. Porém, se quisermos colocar na saída uma lista de inteiros $b_0b_1 \cdots b_{N-1}$, então, de modo similar à entrada, devemos deixar nas células de memória $m_0, m_1, \ldots, m_N$ os valores $N, b_0, b_1, \ldots, b_{N-1}$ ao término da computação.

Com isto a descrição do modelo está quase concluída, restando estabelecer os custos das operações. Os custos que assinalamos às operações têm como propósito refletir o custo da operação em um modelo de computação realista. Por isso, os custos são diretamente proporcionais ao número de *bits* que a operação em questão manipula, ao menos assintoticamente. Mais precisamente, temos os seguintes custos em termos da notação usada na definição das operações, onde $D(a) = \lceil \log_2(|a| + 1) \rceil$ é o número de dígitos binários na representação do inteiro $|a|$:

| Operação | Custo |
|---:|---|
| `clear` | 1 |
| `add` | $D(a) + D(b)$ |
| `sub` | $D(a) + D(b)$ |
| `and` | $D(a) + D(b)$ |
| `or` | $D(a) + D(b)$ |
| `lshift` | $D(a) + \lvert b \rvert$ |
| `popcount` | $D(a)$ |
| `bits` | $D(a) + D(b) + \lvert b \rvert$ |
| `load` | $D(a) + D(m_{\lvert a \rvert})$ |
| `store` | $D(a) + D(r_i)$ |

Note que o custo da operação `bits` não envolve o valor $c$, mas apenas os índices dos *bits* que queremos extrair ($a$ e $b$) e quantos *bits* são extraídos ($b$). Ainda, há um (muito leve) abuso de notação quando escrevemos $D(m_{|a|})$ e $D(r_i)$.

Este modelo tem custos excepcionalmente realistas, e justamente por este motivo não incluímos as instruções de multiplicação e divisão inteira nele. Ao contrário das operações que incluímos, como soma e subtração, não se conhece um algoritmo linear no número de *bits* para multiplicar e dividir. Por isso deixamos estas operações a serem implementadas pelos usuários do modelo (ou pelo compilador anteriormente descrito). Na verdade, se você implementar os algoritmos quadráticos ingênuos, nenhum resultado neste trabalho é comprometido. Estes, no entanto, estão longe de ser os melhores algoritmos. Direcionamos o leitor interessado a explorar a implementação (e a documentação) da biblioteca GMP [2]. Lá, a multiplicação de inteiros é implementada com um algoritmo $\mathcal{O}(n \log n \log\log n)$ (onde $n$ é o maior número de *bits* nos operandos) baseado na *number theoretic transform*, uma aplicação extremamente interessante e profunda da transformada discreta de Fourier. Nós (os autores) não sabemos em detalhes como é feita a divisão inteira nesta biblioteca, mas sabemos que seu custo é o da multiplicação multiplicado por um fator polinomial em $\log_2 n$.

Isto define o custo em termos de tempo que uma execução no modelo possui. Resta definir o espaço utilizado por tal execução. Para isso seja $U \in \mathbb{N}$ o menor número natural maior do que $M$ (o tamanho da entrada) tal que todos os índices de memória utilizados na operação `store` sejam menores do que $U$. Seja, ainda, para cada índice $i \in \{0, 1, \ldots, U-1\}$, $U(m_i)$ o menor número natural maior do que todos os valores absolutos dos valores armazenados na célula $m_i$ durante a execução e, similarmente, seja, para cada índice $i \in \{0, 1, \ldots, n-1\}$, $U(r_i)$ o menor número natural maior do que todos os valores absolutos dos valores armazenados no registrador $r_i$ durante a execução. Definimos então o espaço ocupado pela computação como

$$1 + \sum_{i=0}^{U-1} D\big(U(m_i)\big) + \sum_{i=0}^{n-1} D\big(U(r_i)\big),$$

onde $D : \mathbb{N} \to \mathbb{N}$ é a função que computa o número de dígitos binários de um número usada anteriormente.

É importante notar um detalhe sobre os custos neste modelo, que ilustramos com um exemplo. Ao construirmos um computador em nosso modelo para ordenar uma permutação usando um conhecido algoritmo como o *mergesort*, se calcularmos os custos assintóticos, chegaremos a um custo da ordem de $\mathcal{O}(n \log^2 n)$ de tempo e $\mathcal{O}(n \log n)$ de espaço, sendo $n$ o tamanho da permutação. Isto está em contraste com os custos $\mathcal{O}(n \log n)$ e $\mathcal{O}(n)$ com os quais estamos acostumados. A razão disto, no entanto, é que, ao analisarmos o *mergesort* como de costume, atribuímos um custo $\mathcal{O}(1)$ às operações com inteiros e ponteiros neste algoritmo. Porém, como estes inteiros e ponteiros devem ter da ordem de $\mathcal{O}(\log n)$ *bits*, este fator adicional aparece nas complexidades. Convencionamos então que, ao usar a notação assintótica neste modelo, descontamos implicitamente um fator de $\mathcal{O}(\log B)$, onde $B$ é o total de *bits* na entrada de uma execução.

Observamos que este modelo de computação é Turing-equivalente [24, seção 3.3]. Porém, convenientemente não dissemos que este modelo, bem como uma máquina de Turing, é extremamente difícil de se programar. O compilador que descrevemos, embora exista, não é trivial de se construir. Porém, em um certo sentido, isto também é verdade para computadores modernos.

Nossa última observação é que este modelo não corresponde de fato a computadores modernos, mas apenas à realidade da computação atual. A razão disso é que qualquer computador fisicamente implementável é, aparentemente, uma máquina de estados finita (autômato finito). Se quiséssemos aproximar mais o modelo da realidade, temos que entender que ao resolver um problema em um computador real, a memória dele pode acabar. Neste caso devemos fazer um *backup* do estado atual da solução e transportá-lo para um computador mais potente para continuar a operação. Este computador mais potente também é uma máquina de estados finita e, se sua memória acabar, o processo deve ser repetido, sempre buscando computadores mais potentes.

## 1.5 Álgebra

Neste trabalho existem algumas poucas considerações algébricas, embora eu tenha dedicado uma boa parte de meu mestrado ao estudo do tratamento computacional de números algébricos.

Considere primeiramente a seguinte definição:

> **Definição 39** — Identificamos um conjunto $F \subseteq \mathbb{R}$ como um ***subcorpo real*** se forem válidas as seguintes propriedades:
>
> - Os números reais 0 e 1 são elementos de $F$;
> - Os números reais $x - y$ e $xy$ são elementos de $F$ sempre que $x$ e $y$ também o são; e
> - O número real $1/x$ é um elemento de $F$ sempre que $x$ é um elemento não-nulo de $F$.

Para fins de generalidade, neste trabalho devemos fixar um subcorpo real $\mathbb{F}$. Usaremos este subcorpo para representar as coordenadas de pontos (extremos de segmentos, centros de círculos e vértices de polígonos, dentre outros) e grandezas como o raio de círculos, tudo isso na entrada, durante a execução e na saída de nossos algoritmos. Esta é a nossa solução para o problema decorrente do fato de que computadores são incapazes de representar números reais arbitrários.

Após esta seção, iremos tratar o subcorpo $\mathbb{F}$ como fixado. Por isso, no restante dela, vamos especificar quais rotinas computacionais devem ser providenciadas para que um subcorpo real possa assumir o papel de $\mathbb{F}$. Mostraremos também como iremos avaliar a complexidade computacional de nossos algoritmos no tocante ao uso do subcorpo $\mathbb{F}$.

É perfeitamente possível assumir que $\mathbb{F} = \mathbb{Q}$ durante o resto deste trabalho sem nenhum inconveniente caso o leitor não precise de nenhuma generalidade maior. Neste caso é fácil observar que as rotinas computacionais requisitadas estão prontamente disponíveis na maioria dos ambientes computacionais contemporâneos, principalmente através da biblioteca GMP [2].

Para um subcorpo real $F$ poder assumir o papel de $\mathbb{F}$ durante o resto deste trabalho, os seguintes recursos computacionais devem estar disponíveis:

- Uma rotina F-Convert que converte números racionais em elementos de $F$ (observe que qualquer subcorpo real deve conter $\mathbb{Q}$);
- Uma rotina F-Difference para calcular a subtração de dois elementos de $F$;
- Uma rotina F-Product para calcular o produto de dois elementos de $F$;

- Uma rotina F-Inverse para calcular o inverso $1/x$ de cada elemento não-nulo $x$ de $F$;
- Uma rotina F-Less que determina se um elemento de $F$ é menor do que outro; e
- Uma rotina F-Approximation que recebe um elemento $x$ de $F$ e um número racional $\varepsilon$ maior do que 0 e devolve um número racional $a$ tal que $|a - x| < \varepsilon$.

Quanto à análise de complexidade de algoritmos que usam o subcorpo $\mathbb{F}$, o método que temos a oferecer é bem simples. Porém, neste nível de generalidade em que estamos trabalhando, ele pode ser considerado bastante prático.

Este método é baseado no conceito de ***níveis de complexidade algébrica***. Essencialmente, a cada valor representando um elemento de $\mathbb{F}$ computado por um programa, associamos um número natural chamado de o nível de complexidade algébrica daquele valor ou, por simplicidade, o nível daquele valor. A regra para deduzir o nível de um valor é a seguinte: o nível dos resultados de chamadas à rotina F-Convert é 0 e o nível dos resultados de chamadas às rotinas F-Difference, F-Product e F-Inverse é 1 mais o maior nível dentre seus argumentos. Dessa forma o nível de um valor é na verdade sua profundidade na árvore de computações do programa.

É importante notar que é impossível em geral associar um nível a cada elemento de $\mathbb{F}$. Níveis são associados apenas a valores computados por um programa e, assim, dois valores representando o mesmo elemento de $\mathbb{F}$ podem ter níveis diferentes. Por exemplo, se um programa faz a chamada

$$\text{F-Difference}\big(\text{F-Convert}(2), \text{F-Convert}(1)\big),$$

ele irá obter um valor de nível 1, embora ele represente o elemento $1 \in \mathbb{F}$, o mesmo elemento representado pelo resultado de uma chamada da forma F-Convert(1), que possui nível 0. Também é importante notar que estes níveis são apenas uma ferramenta de análise de complexidade, não afetando em nada a implementação dos programas (não existe, em particular, uma rotina que calcula o nível de um valor representando um elemento de $\mathbb{F}$).

Com isso em mente, dizemos que um programa é ***algebricamente tratável*** se existe um número natural maior do que todos os níveis de todos os valores que o programa pode produzir com todas as suas possíveis entradas (os níveis de complexidade algébrica de elementos de $\mathbb{F}$ presentes na entrada é considerado como sendo 0).

Ainda, dizemos que a ***complexidade algébrica*** de tempo ou espaço de um programa é a respectiva complexidade do programa obtido traduzindo o programa original da forma óbvia assumindo a presença de um oráculo que providencia as rotinas F-Difference, F-Product, F-Inverse e F-Less em tempo e espaço $\mathcal{O}(1)$.

Assim, se um programa é algebricamente tratável e se tivermos implementado todas as rotinas F-Difference, F-Product, F-Inverse e F-Less em tempo e espaço polinomiais, então as complexidades de tempo e espaço deste programa não são maiores do que suas respectivas complexidades algébricas multiplicadas por um fator polinomial no maior número de *bits* nas representações computacionais de elementos de $\mathbb{F}$ na entrada do programa (mais 1, para tratar o caso deste número ser 0).

Este método de análise pode parecer simples, mas ele é incrivelmente eficaz pelos seguintes motivos:

- É relativamente fácil mostrar que os programas desenvolvidos neste trabalho são algebricamente tratáveis;
- Analisar as complexidades algébricas destes programas é também relativamente fácil;
- Para todos os subcorpos $\mathbb{F}$ que chamaram nossa atenção, inclusive quando $\mathbb{F}$ é o subcorpo dos números reais algébricos, é possível implementar as rotinas mencionadas em tempo e espaço polinomiais; e
- É longe de ser trivial (pelo menos para nós), desenvolver um outro método de análise de complexidade que produza resultados mais justos do que este para os programas neste trabalho. Mesmo quando $\mathbb{F} = \mathbb{Q}$, uma análise de complexidade mais justa parece demandar uma incrivelmente complicada atenção a detalhes.

## 1.6 Co-rotinas

Existe um conceito em muitas linguagens de programação modernas que facilitará bastante a exposição de vários pontos deste trabalho e, sendo assim, decidimos dedicar esta seção à explicação deste conceito, que pode não ser familiar a muitos programadores.

Uma ***co-rotina*** é uma forma de representar sequências (finitas ou não) através de rotinas computacionais que podem retornar uma quantidade qualquer de resultados. Infelizmente, foge do escopo deste trabalho definir co-rotinas formalmente, e podemos apenas fornecer um exemplo.

O algoritmo 1 codifica uma co-rotina que recebe um número natural $n$ e devolve os números primos menores do que $n$ em ordem crescente.

Imediatamente percebemos onde uma co-rotina difere de uma rotina normal. Em primeiro lugar, ela retorna valores várias vezes. Para demarcar este fato, usamos a palavra reservada `yield` em vez da palavra reservada `return`.

Ainda, anotamos que o algoritmo "devolve os números primos menores do que $n$ em ordem crescente como uma co-rotina". Para entender o que isso significa, considere que um programa executa

$$s \leftarrow \textsc{Sieve}(10).$$

Neste caso qual valor está guardado na variável $s$? A resposta é uma instância da co-rotina. A variável $s$ representa então a sequência (no caso finita) dos números primos menores do que 10 e tais números podem ser acessados repetidamente invocando a variável $s$ como se ela fosse uma função. Assim, uma chamada subsequente da forma $s()$ produziria o número 2, uma próxima chamada produziria 3, seguido de 5 e, finalmente, uma nova chamada da forma $s()$ produziria o número 7. E, após isso, o que acontece se chamarmos $s()$. Neste caso o valor especial $\perp$ é devolvido, indicando que a sequência acabou. Na verdade futuras chamadas da forma $s()$ também produzirão $\perp$.

Para simplificar a notação, admitiremos em nossos programas expressões do tipo

`for` $x$ `in` $s$ `do` ... `done`

para iterar sobre os elementos da sequência.

**Algorithm 1** SIEVE ($n$)

`Pré-condição:` $n \in \mathbb{N}$

`Efeito:` devolve os números primos menores do que $n$ em ordem crescente como uma co-rotina

```
S ← um vetor de bits indexado por {2, 3, ..., n − 1} inicializado com zeros
i ← 2
while i² < n do
   if S_i = 0 then
      j ← i²
      while j < n do
         S_j ← 1
         j ← j + i
      done
   endif
   i ← i + 1
done
for i from 2 to n − 1 do
   if S_i = 0 then
      yield i
   endif
done
```

A implementação de co-rotinas em linguagens de programação varia bastante. A minha implementação favorita coloca no início do código (de máquina) da função uma tabela de pulos (estilo o `switch` da linguagem C) indexada por em quais lugares a execução daquele código pode começar, ou seja, o início e as linhas subsequentes às chamadas `yield`. Na variável que recebe a instância da co-rotina (no nosso caso $s$), guardamos um ponteiro para a função, o estado das variáveis locais e um índice na tabela de pulos. Ao chamar a instância da co-rotina, criamos um *stack frame*, o preenchemos com os valores das variáveis locais e executamos a função com o índice da tabela de pulos. Uma implementação de co-rotinas para a linguagem C de programação que utiliza este esquema é discutida em [26].

Há várias vantagens no uso de co-rotinas. No nosso caso, a vantagem mais aparente é não precisar guardar a sequência em uma estrutura de dados para iterar sobre ela. Isso, como mostraremos adiante, terá um impacto significativo na complexidade de espaço de um de nossos algoritmos. Vale ainda dizer que este efeito foi o que popularizou os *pipes* do sistema operacional UNIX, muito embora esta tecnologia não esteja no mesmo nível de abstração.

# Capítulo 2

# Representações de conjuntos

Este capítulo tem dois principais objetivos. O primeiro é introduzir um método de descrição de conjuntos baseado na relação que curvas orientadas têm com a fronteira do conjunto. Como veremos adiante, este método será extremamente conveniente ao definirmos a representação computacional de conjuntos que iremos adotar em nossos algoritmos.

Claro, representar computacionalmente conjuntos arbitrários de pontos é em geral impossível, e assim apenas alguns tipos de conjuntos poderão ser representados com este método.

Ainda, como o segundo principal objetivo deste capítulo, mostramos como usar uma tal representação de um conjunto para obter duas propriedades dele: sua área e seu centro de massa.

## 2.1 Representações de conjuntos

Considere as duas seguintes definições, que são a base de nosso método de representar conjuntos.

**Definição 40** — Seja $X \subseteq \mathbb{R}^2$ um conjunto e seja $\overrightarrow{uv}$ um segmento de reta orientado. Dizemos então que $\overrightarrow{uv}$ é ***compatível*** com $X$ se, para todo ponto $w \in uv \setminus \{u, v\}$, existir um valor $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ tal que, para todo $p \in B_\varepsilon(w)$, $p \in X$ se $(v - w) \times (p - w) > 0$ e $p \notin X$ se $(v - w) \times (p - w) < 0$.

**Definição 41** — Seja $X \subseteq \mathbb{R}^2$ um conjunto e seja $\alpha = \overrightarrow{\mathrm{arc}}(C, u, v)$ um arco de circunferência orientado em sentido anti-horário, onde $C$ é um círculo e $u, v \in \partial C$. Dizemos então que $\alpha$ é ***compatível*** com $X$ se, para todo $w \in \mathrm{arc}(C, u, v) \setminus \{u, v\}$, existir um $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ tal que, para todo $p \in B_\varepsilon(w)$, $p \in X$ se $p \in C^\circ$ e $p \notin X$ se $p \in \mathbb{R}^2 \setminus C$. Ainda, dizemos que o arco $\overleftarrow{\mathrm{arc}}(C, u, v)$ é ***compatível*** com $X$ se $\alpha$ é compatível com $\mathbb{R}^2 \setminus X$.

São imediatos os dois seguintes resultados sobre nossa noção de compatibilidade.

**Proposição 5** — Seja $X \subseteq \mathbb{R}^2$ um conjunto e seja $\overrightarrow{uv}$ um segmento de reta orientado compatível com $X$. Então $uv \subseteq \partial X$.

**Proposição 6** — Seja $X \subseteq \mathbb{R}^2$ um conjunto e seja $\alpha = \mathrm{arc}(C, u, v)$ um arco de circunferência, onde $C$ é um círculo e $u, v \in \partial C$. Então, se $\overrightarrow{\mathrm{arc}}(C, u, v)$ ou $\overleftarrow{\mathrm{arc}}(C, u, v)$ forem compatíveis com $X$, vale que $\alpha \subseteq \partial X$.

Agora mostramos nosso método de representação de conjuntos através da seguinte definição.

**Definição 42** — Seja $X \subseteq \mathbb{R}^2$ um conjunto de pontos e seja $\mathcal{R}$ um conjunto finito de segmentos orientados e arcos de circunferência orientados. Dizemos que $\mathcal{R}$ é uma ***boa representação*** de $X$ se as seguintes condições forem todas satisfeitas:

- Todo segmento orientado em $\mathcal{R}$ é compatível com $X$;
- Todo arco de circunferência orientado em $\mathcal{R}$ é compatível com $X$;
- Se $\overrightarrow{ab}$ e $\overrightarrow{uv}$ são dois segmentos orientados distintos em $\mathcal{R}$, então $ab \cap uv \subseteq \{a, b, u, v\}$;
- Se $\overrightarrow{ab}$ é um segmento de reta orientado em $\mathcal{R}$ e um elemento de $\mathcal{R}$ é uma orientação de um arco de circunferência $\alpha = \mathrm{arc}(C, u, v)$, onde $C$ é um círculo e $u, v \in \partial C$, então $ab \cap \alpha \subseteq \{a, b, u, v\}$;
- Se dois elementos distintos de $\mathcal{R}$ são respectivas orientações dos arcos de circunferência $\alpha = \mathrm{arc}(C, a, b)$ e $\beta = \mathrm{arc}(D, u, v)$, onde $C$ e $D$ são círculos, $a, b \in \partial C$ e $u, v \in \partial D$, então $\alpha \cap \beta \subseteq \{a, b, u, v\}$; e
- A fronteira de $X$ é

$$\partial X = \bigcup_{\overrightarrow{uv} \in \mathcal{R}} uv \cup \bigcup_{\overrightarrow{\mathrm{arc}}(C,u,v) \in \mathcal{R}} \mathrm{arc}(C, u, v) \cup \bigcup_{\overleftarrow{\mathrm{arc}}(C,u,v) \in \mathcal{R}} \mathrm{arc}(C, u, v).$$

Observe que, de acordo com esta definição, um conjunto pode admitir várias boas representações, mas não iremos entrar no mérito de se uma boa representação de um conjunto pode ser uma boa representação de outro conjunto.

Boas representações admitem uma certa analogia com polígonos, como mostramos agora.

**Definição 43** — Seja $\mathcal{R}$ uma boa representação de um conjunto $X \subseteq \mathbb{R}^2$. Dizemos então que os ***vértices*** de $\mathcal{R}$ são os elementos do conjunto (finito) $V(\mathcal{R}) \subseteq \mathbb{R}^2$ dos extremos de segmentos orientados de $\mathcal{R}$ e dos extremos de arcos de circunferência orientados de $\mathcal{R}$.

**Definição 44** — Seja $\mathcal{R}$ uma boa representação de um conjunto $X \subseteq \mathbb{R}^2$. Dizemos então que as ***arestas*** de $\mathcal{R}$ são os próprios elementos de $\mathcal{R}$.

Durante este trabalho, lidaremos principalmente com boas representações de conjuntos compostas apenas de segmentos orientados ou de arcos de circunferência orientados, raramente misturando ambos os tipos de curva em uma única representação. Porém, neste capítulo trabalhamos com esta estrutura mais geral, pois aqui esta mistura não traz muita complexidade adicional.

A seguir mostramos como obter a área e o centro de massa de conjuntos a partir de uma boa representação.

## 2.2 Relações *shoelace*

A chave para entender como calcular a área ou centro de massa de um conjunto a partir de uma boa representação para ele é generalizar os métodos de se calcular áreas e centros de massa de polígonos, também conhecidos como relações *shoelace* [20, seção 1.3].

Lembramos o leitor que se $P$ é um polígono, então

$$\operatorname{area} P = \frac{1}{2} \sum_{\overrightarrow{uv} \in E(P)} u \times v$$

e

$$\operatorname{centroid} P = \frac{1}{6(\operatorname{area} P)} \sum_{\overrightarrow{uv} \in P} (u \times v)(u + v).$$

Nosso primeiro passo então é demonstrar estas relações, e a técnica que utilizaremos para isso é conhecida como *ray casting*, uma caracterização extremamente útil de pertinência de um ponto em um polígono.

Intuitivamente, ao tentar decidir se um ponto $p \in \mathbb{R}^2$ pertence a um polígono $P$, consideramos uma semi-reta orientada que começa em $p$, também chamada de raio. Se "percorrermos" o raio, mantendo uma contagem $a$ do número de vezes que o raio "fura" o polígono de fora para dentro e uma contagem $b$ do número de vezes que o raio "fura" o polígono de dentro para fora, então, ao final do processo, "quase sempre" valerá que $p \in P$ se, e somente se, $b > a$. Mais precisamente, o conjunto dos pontos $p \in \mathbb{R}^2$ em que este processo pode falhar é J-desprezível.

A seguir definiremos formalmente todos os elementos da técnica de *ray casting* e mostraremos ambas as relações *shoelace*. Cabe, no entanto, observar que o que entendemos por *ray casting* neste trabalho não é o que usualmente encontramos na literatura. Lá, a técnica de *ray casting* normalmente visa o desenvolvimento de algoritmos para testar se um ponto pertence a um polígono. Aqui, ela é meramente um instrumento que usaremos em nossas demonstrações e, como isto é tecnicamente conveniente para nós, descrevemos uma versão simplificada (e incapaz de produzir um tal algoritmo correto) da técnica. Uma descrição mais completa do *ray casting* pode ser encontrada em [20].

**Definição 45** — Seja $p \in \mathbb{R}^2 \setminus \{0\}$ um ponto no plano diferente da origem. Definimos então o ***raio*** associado ao ponto $p$, também denotado por $\operatorname{ray} p$, como o conjunto $\{tp : t \in \mathbb{R} \wedge t \geq 1\}$.

**Definição 46** — Sejam $p, u, v \in \mathbb{R}^2$ pontos no plano. Definimos então o **ray casting** de $p$ no segmento orientado $\overrightarrow{uv}$, denotado por $\operatorname{rc}(p, \overrightarrow{uv})$, da seguinte forma. Se $p = 0$, se $p \neq 0$ mas $(\operatorname{ray} p) \cap uv = \{\}$ ou se $uv \subseteq \{tp : t \in \mathbb{R}\}$, então $\operatorname{rc}(p, \overrightarrow{uv}) = 0$. Caso contrário, o valor $c = p \times (v - u)$ deve ser diferente de 0. Definimos então

$$\operatorname{rc}(p, \overrightarrow{uv}) = \begin{cases} 1, & c > 0 \\ -1, & c < 0. \end{cases}$$

Intuitivamente, entendemos $\overrightarrow{uv}$ como uma aresta do polígono. Neste caso, $\operatorname{rc}(p, \overrightarrow{uv})$ é o número de vezes que o raio "fura" $\overrightarrow{uv}$ de dentro para fora menos o número de vezes que o raio "fura" $\overrightarrow{uv}$ de fora para dentro. O caso em que $\overrightarrow{uv}$ é paralelo ao raio não é de importância, e qualquer valor real poderia ter sido assinalado a $\operatorname{rc}(p, \overrightarrow{uv})$.

**Definição 47** — Seja $P$ um polígono e seja $p \in \mathbb{R}^2$ um ponto. Definimos então o **ray casting** de $p$ em $P$ como

$$\operatorname{rc}(p, P) = \sum_{\overrightarrow{uv} \in E(P)} \operatorname{rc}(p, \overrightarrow{uv}).$$

Em geral, não necessariamente é válido que $\operatorname{rc}(p, P) = \chi_P(p)$ para todo ponto $p \in \mathbb{R}^2$. Porém, isto é sempre verdade se $p \neq 0$, se $p$ não estiver na fronteira de $P$ e se $\operatorname{ray} p$ não intersectar nenhum vértice de $P$. Formalizamos isto a seguir.

**Definição 48** — O conjunto das ***falhas*** do *ray casting* em um polígono $P$ é o conjunto

$$\operatorname{fail} P = \partial P \cup \bigcup_{v \in V(P)} \{tv : t \in \mathbb{R}, 0 \leq t \leq 1\}.$$

**Proposição 7** — O conjunto das falhas do *ray casting* em um polígono é J-desprezível.

**Prova** — Imediato. □

**Proposição 8** — Seja $P$ um polígono e seja $p \in \mathbb{R}^2 \setminus (\operatorname{fail} P)$ um ponto no plano que não é uma falha do *ray casting* em $P$. Então $\operatorname{rc}(p, P) = \chi_P(p)$.

**Prova** — Como $p \notin \operatorname{fail} P$, não existem arestas em $P$ paralelas a $\operatorname{ray} p$ que intersectam $\operatorname{ray} p$. Assim o conjunto $T = \{t \in \mathbb{R} : t \geq 1 \wedge tp \in \partial P\}$ é finito e pode ser expresso como $\{t_0, t_1, \ldots, t_{n-1}\}$ para algum $n \in \mathbb{N}$ com $1 < t_0 < t_1 < \cdots < t_{n-1}$. Ainda, existe uma única aresta de $P$ que contém $t_i p$ para cada $i \in \{0, 1, \ldots, n-1\}$, que será denotada $e_i$.

O resultado será agora provado por indução em $n$. Caso $n = 0$, devemos mostrar que $p \notin P$, pois claramente $\operatorname{rc}(p, P) = 0$. Para isso, assumimos, para fins de uma prova por contradição, que $p \in P$. Assim, como $p \notin \operatorname{fail} P$, devemos ter que $p \notin \partial P$ e logo que $p \in P^\circ$. Desse modo, o teorema do valor intermediário (teorema 1) garante a existência de um valor real $t \in \mathbb{R}$ com $t > 1$ tal que $tp \in \partial P$, o que contradiz $n = 0$, que deveria implicar $T = \{\}$. Assim, somos forçados a concluir que $p \notin P$, como desejado.

Já se $n \neq 0$, defina o ponto $p' \in \mathbb{R}^2$ por $p' = (t_0 + 1)p$ se $n = 1$ ou por

$$p' = \frac{t_0 + t_1}{2} p$$

se $n > 1$. Note então que $p' \notin \operatorname{fail} P$ e que $\operatorname{ray} p'$ intersecta precisamente as arestas $e_1, e_2 \ldots, e_{n-1}$. Assim, podemos usar a hipótese indutiva em $p'$, obtendo $\operatorname{rc}(p', P) = \chi_P(p')$. Porém, como $e_0$ é a única aresta de $P$ que intersecta $\operatorname{ray} p$ mas não intersecta $\operatorname{ray} p'$,

$$\begin{aligned} \operatorname{rc}(p, P) &= \operatorname{rc}(p, e_0) + \operatorname{rc}(p', P) \\ &= \operatorname{rc}(p, e_0) + \chi_P(p'). \end{aligned}$$

Note, no entanto, que $\chi_P(p) = 1 - \chi_P(p')$, pois $p \in P$ se, e somente se, $p' \notin P$, uma vez que $p, p' \notin \partial P$ e o segmento $pp'$ cruza exatamente uma aresta de $P$: a aresta $e_0$.

Segue que $\operatorname{rc}(p, P) = \operatorname{rc}(p, e_0) - \chi_P(p) + 1$. Se $p \notin P$, necessariamente temos que $\operatorname{rc}(p, e_0) = -1$ e, se $p \in P$, necessariamente temos que $\operatorname{rc}(p, e_0) = 1$. Em ambos os casos, é fácil verificar que $\operatorname{rc}(p, P) = \chi_P(p)$, provando o resultado. □

Em conjunto, as proposições 7 e 8 revelam um fato interessante. Se $P$ é um polígono, então ele, sendo um conjunto J-mensurável, está contido em algum retângulo aberto $R$ que contém também a origem. Logo

$$\begin{aligned} \operatorname{area} P &= \int_R \chi_P(p) \, \mathrm{d}p \\ &= \int_R \operatorname{rc}(p, P) \, \mathrm{d}p \\ &= \int_R \sum_{e \in E(P)} \operatorname{rc}(p, e) \, \mathrm{d}p \\ &= \sum_{e \in E(P)} \int_R \operatorname{rc}(p, e) \, \mathrm{d}p. \end{aligned}$$

Assim, se pudermos calcular estas últimas integrais de forma eficaz, conseguiremos calcular também a área do polígono.

**Proposição 9** — Seja $R$ um retângulo aberto que contém a origem e seja $\overrightarrow{uv}$ um segmento de reta orientado com $uv \subseteq R$. Então

$$\int_R \operatorname{rc}(p, \overrightarrow{uv}) \, \mathrm{d}p = \frac{1}{2}(u \times v).$$

**Prova** — Seja $T \subseteq R$ o triângulo cujos vértices são a origem, $u$ e $v$. Sabemos da geometria analítica que a área de $T$ é

$$\int_T 1 \, \mathrm{d}p = \operatorname{area} T = \frac{1}{2}|u \times v|.$$

Seja então $p \in \mathbb{R}^2 \setminus \{0\} \setminus \partial T$ um ponto fora da fronteira de $T$ e diferente da origem. Se adicionalmente $p \notin T$, então $(\operatorname{ray} p) \cap T = \{\}$, de modo que $(\operatorname{ray} p) \cap uv = \{\}$ e $\operatorname{rc}(p, \overrightarrow{uv}) = 0$. Se, por outro lado, $p \in T$, então $p \in T^\circ$, e assim o raio $\operatorname{ray} p$ não só intersecta o segmento $uv$, mas também não é paralelo a este segmento. Portanto, neste caso, $\operatorname{rc}(p, \overrightarrow{uv}) = \operatorname{sign}\big(p \times (v-u)\big)$. Porém, como $p \in T^\circ$, $\operatorname{sign}\big(p \times (v-u)\big) = \operatorname{sign}(u \times v)$.

Assim, temos

$$
\begin{aligned}
\int_R \operatorname{rc}(p, \overrightarrow{uv})\ \mathrm{d}p &= \int_T \operatorname{rc}(p, \overrightarrow{uv})\ \mathrm{d}p \\
&= \int_T \operatorname{sign}(u \times v)\ \mathrm{d}p \\
&= \operatorname{sign}(u \times v) \int_T 1\ \mathrm{d}p \\
&= \operatorname{sign}(u \times v)(\operatorname{area} T) \\
&= \frac{1}{2} \operatorname{sign}(u \times v)|u \times v| \\
&= \frac{1}{2}(u \times v),
\end{aligned}
$$

pois a fronteira de $T$ é J-desprezível. □

Com isso, a relação *shoelace* para áreas segue imediatamente.

**Corolário 3** (relação *shoelace* para áreas) — Se $P$ é um polígono, então

$$\operatorname{area} P = \frac{1}{2} \sum_{\overrightarrow{uv} \in E(P)} u \times v.$$

**Prova** — Seja $R$ um retângulo aberto contendo $P$ e a origem. Então, como discutimos acima e de acordo com a proposição 9,

$$\operatorname{area}(P) = \sum_{\overrightarrow{uv} \in E(P)} \int_R \operatorname{rc}(p, \overrightarrow{uv})\ \mathrm{d}p = \frac{1}{2} \sum_{\overrightarrow{uv} \in E(P)} u \times v.$$

□

Voltamos nossa atenção agora à relação *shoelace* restante, isto é, como calcular o centro de massa de polígonos. O método para isso é completamente análogo ao usado para se calcular áreas. Novamente, as proposições 7 e 8 têm um papel importante. Se $P$ é um polígono, então ele é um conjunto J-mensurável e está consequentemente contido em um retângulo aberto $R$ que contém também a origem. Assim,

$$\begin{aligned}
\operatorname{centroid} P &= \frac{1}{\operatorname{area} P}\int_R \chi_P(p)p\ \mathrm{d}p \\
&= \frac{1}{\operatorname{area} P}\int_R \operatorname{rc}(p,P)p\ \mathrm{d}p \\
&= \frac{1}{\operatorname{area} P}\int_R \Bigg(\sum_{e\in E(P)} \operatorname{rc}(p,e)\Bigg)p\ \mathrm{d}p \\
&= \frac{1}{\operatorname{area} P}\int_R \sum_{e\in E(P)} \operatorname{rc}(p,e)p\ \mathrm{d}p \\
&= \frac{1}{\operatorname{area} P}\sum_{e\in E(P)}\int_R \operatorname{rc}(p,e)p\ \mathrm{d}p.
\end{aligned}$$

Mostramos então como calcular estas últimas integrais.

**Proposição 10** — Seja $R$ um retângulo aberto contendo a origem e seja $\overrightarrow{uv}$ um segmento de reta orientado com $uv \subseteq R$. Então

$$\int_R \operatorname{rc}(p,\overrightarrow{uv})p\ \mathrm{d}p = \frac{1}{6}(u\times v)(u+v).$$

**Prova** — Esta demonstração é extremamente similar à demonstração da proposição 9. Da mesma forma que naquela demonstração, seja $T \subseteq R$ o triângulo cujos vértices são a origem, $u$ e $v$. É um fato conhecido que o centroide de $T$, também conhecido como seu baricentro, é dado por

$$\frac{\displaystyle\int_T p\ \mathrm{d}p}{\operatorname{area} T} = \operatorname{centroid} T = \frac{1}{3}(0+u+v) = \frac{1}{3}(u+v).$$

Assim, como $\operatorname{area} T = \frac{1}{2}|u\times v|$, temos que

$$\int_T p\ \mathrm{d}p = \frac{1}{6}|u\times v|(u+v).$$

Agora, do mesmo modo que na demonstração da proposição 9, se $p \in \mathbb{R}^2 \setminus \{0\} \setminus \partial T$ é um ponto fora da fronteira de $T$ e diferente de 0, então $\operatorname{rc}(p,\overrightarrow{uv}) = 0$ se $p \notin T$ e $\operatorname{rc}(p,\overrightarrow{uv}) = \operatorname{sign}(u\times v)$ se $p\in T$.

Deste modo, segue

$$\begin{aligned}
\int_R \mathrm{rc}(p, \overrightarrow{uv}) p \, \mathrm{d}p &= \int_T \mathrm{rc}(p, \overrightarrow{uv}) p \, \mathrm{d}p \\
&= \int_T \mathrm{sign}(u \times v) p \, \mathrm{d}p \\
&= \mathrm{sign}(u \times v) \int_T p \, \mathrm{d}p \\
&= \frac{1}{6} \mathrm{sign}(u \times v) |u \times v| (u + v) \\
&= \frac{1}{6} (u \times v)(u + v),
\end{aligned}$$

também devido ao fato de que $\partial T$ é Jordan desprezível. □

**Corolário 4** (relação *shoelace* para centros de massa) — Se $P$ é um polígono, então

$$\mathrm{centroid}\, P = \frac{1}{6(\mathrm{area}\, P)} \sum_{\overrightarrow{uv} \in E(P)} (u \times v)(u + v).$$

**Prova** — Seja $R$ um retângulo aberto contendo o polígono $P$ e a origem. Então, como foi discutido anteriormente e de acordo com a proposição 10,

$$\begin{aligned}
\mathrm{centroid}\, (P) &= \frac{1}{\mathrm{area}\, (P)} \sum_{\overrightarrow{uv} \in E(P)} \int_R \mathrm{rc}(p, \overrightarrow{uv}) p \, \mathrm{d}p \\
&= \frac{1}{6(\mathrm{area}\, P)} \sum_{\overrightarrow{uv} \in E(P)} (u \times v)(u + v).
\end{aligned}$$

□

## 2.3 Generalizações das relações *shoelace*

Nesta seção iremos generalizar as relações *shoelace* vistas na seção 2.2 para conjuntos de pontos com uma boa representação. A ideia é completamente análoga ao que foi visto naquela seção, mas existem bastante tecnicalidades, que cobrimos aqui.

O primeiro passo é generalizar a noção de *ray casting* para arcos de circunferência orientados.

**Definição 49** — Seja $\alpha = \big(\operatorname{arc}(C, u, v), \sigma\big)$ um arco de circunferência orientado com sentido $\sigma \in \{-1, 1\}$ e seja $p \in \mathbb{R}^2$ um ponto no plano. Definimos o **ray casting** de $p$ em $\alpha$, também denotado por $\operatorname{rc}(p, \alpha)$, da seguinte forma. Se $p = 0$, se $p \neq 0$ mas $(\operatorname{ray} p) \cap \operatorname{arc}(C, u, v) = \{\}$ ou se $p \neq 0$ e $|(\operatorname{ray} p) \cap \operatorname{arc}(C, u, v)| = 2$, então $\operatorname{rc}(p, \alpha) = 0$. Caso contrário, deve existir um ponto $q \in \mathbb{R}^2$ tal que $(\operatorname{ray} p) \cap \operatorname{arc}(C, u, v) = \{q\}$. Neste caso definimos

$$\operatorname{rc}(p, \alpha) = \begin{cases} 1, & c > 0 \\ 0, & c = 0 \\ -1, & c < 0, \end{cases}$$

onde $c = p \times \widetilde{\alpha}'\big(\widetilde{\alpha}^{-1}(q)\big)$, $\widetilde{\alpha}'$ é a derivada da função $\widetilde{\alpha}$ e $\widetilde{\alpha}^{-1}$ é a função inversa da função $\widetilde{\alpha}$.

Novamente, interpretamos o *ray casting* do ponto $p$ no arco de circunferência orientado $\alpha$ como o número de vezes que o raio de $p$ "sai" do arco $\alpha$ menos o número de vezes que ele "entra" no arco $\alpha$. Assim, o caso em que $p \neq 0$ e $|(\operatorname{ray} p) \cap \operatorname{arc}(C, u, v)| = 2$ deve ser considerado como 0, pois o raio necessariamente "sai" uma vez e "entra" uma vez.

Generalizamos agora a ideia de *ray casting* para conjuntos de segmentos de reta orientados e arcos de circunferência orientados.

**Definição 50** — Seja $\mathcal{R}$ um conjunto finito de segmentos de reta orientados e arcos de circunferência orientados e seja $p \in \mathbb{R}^2$ um ponto no plano. Definimos então o **ray casting** do ponto $p$ em $\mathcal{R}$ como

$$\operatorname{rc}(p, \mathcal{R}) = \sum_{\alpha \in \mathcal{R}} \operatorname{rc}(p, \alpha).$$

**Definição 51** — Seja $\mathcal{R}$ um conjunto finito de segmentos de reta orientados e arcos de circunferência orientados. Definimos então o conjunto das ***falhas*** do *ray casting* em $\mathcal{R}$ como o conjunto

$$\operatorname{fail}(\mathcal{R}) = \{0\} \cup B \cup \bigcup_{v \in V} \{tv : 0 \leq t \leq 1\},$$

onde $B \subseteq \mathbb{R}^2$ é o conjunto dos pontos no plano em algum elemento de $\mathcal{R}$ e $V \subseteq \mathbb{R}^2$ é o conjunto dos extremos dos elementos de $\mathcal{R}$.

**Proposição 11** — Seja $\mathcal{R}$ um conjunto finito de segmentos de reta orientados e arcos de circunferência orientados. Então o conjunto das falhas do *ray casting* em $\mathcal{R}$ é J-desprezível.

**Prova** — Imediato do fato de que há um número finito de extremos de elementos de $\mathcal{R}$. $\square$

De forma similar ao que fizemos anteriormente, relacionamos agora a função característica de um conjunto de pontos ao resultado do *ray casting*.

> **Proposição 12** — Seja $X \subseteq \mathbb{R}^2$ um conjunto J-mensurável de pontos e seja $\mathcal{R}$ uma boa representação de $X$. Então, para todo ponto $p \in \mathbb{R}^2 \setminus \operatorname{fail}(\mathcal{R})$, temos que $\operatorname{rc}(p, \mathcal{R}) = \chi_X(p)$.

**Prova** — Esta prova é extremamente similar à prova da proposição 8 e encorajamos o leitor a ler aquela prova antes de tentar ler esta caso ele já não tenha feito isto.

Consideramos (analogamente àquela demonstração) o conjunto $T \subseteq \mathbb{R}$ dos valores reais $t \in \mathbb{R}$ com $t > 1$ tal que $tp$ intersecta alguma aresta de $\mathcal{R}$. Como $p \notin \operatorname{fail}(\mathcal{R})$, não existe um segmento em $\mathcal{R}$ paralelo a $\operatorname{ray} p$ e que intersecta $\operatorname{ray} p$, de modo que $T$ é finito, podendo ser escrito na forma $\{t_0, t_1, \ldots, t_{n-1}\}$ com $1 < t_0 < t_1 < \cdots < t_{n-1}$. Ainda, como $\mathcal{R}$ é uma boa representação de $X$, as arestas de $\mathcal{R}$ não se intersectam, exceto possivelmente em seus extremos. Porém, como $p \notin \operatorname{fail}(\mathcal{R})$, $\operatorname{ray} p$ nunca intersecta um tal extremo. Assim, para cada $i \in \{0, 1, \ldots, n-1\}$, existe uma única aresta $e_i$ em $\mathcal{R}$ que contém o ponto $t_i p$. Claro, ao contrário do que ocorre com polígonos na demonstração da proposição 8, aqui podemos ter $e_i = e_j$ mesmo que $i \neq j$, mas neste caso esta aresta deve ser um arco de circunferência orientado.

Vamos agora mostrar que $\operatorname{rc}(p, \mathcal{R}) = \chi_X(p)$ por indução em $n$. No caso base, temos que $n = 0$ e devemos mostrar que $\chi_X(p) = 0$, ou seja, que $p \notin X$, pois $\operatorname{rc}(p, \mathcal{R})$ é claramente 0. De fato, se $p \in X$, então como $p \notin \operatorname{fail}(\mathcal{R})$, $p \notin \partial X$ e assim $p \in X^\circ$. Logo, como $X$ é J-mensurável e portanto limitado, o teorema do valor intermediário (teorema 1) garante a existência de um valor $t \in \mathbb{R}$ com $t > 1$ tal que $tp \in \partial X$. Como a fronteira de $X$ é a união das arestas de $\mathcal{R}$, devemos então concluir que existe uma aresta $e \in \mathcal{R}$ que contém o ponto $tp$, contradizendo $n = 0$. Assim, somos forçados a concluir que $p \notin X$, como desejado.

Se, no entanto, $n > 0$, defina o ponto $p' \in \mathbb{R}^2$ por $p' = (t_0 + 1)p$ se $n = 1$ ou por

$$p' = \frac{t_0 + t_1}{2} p$$

se $n > 1$. Claramente não há uma aresta em $\mathcal{R}$ contendo $p'$ e $\operatorname{ray} p' \subseteq \operatorname{ray} p$, de modo que $p' \notin \operatorname{fail}(\mathcal{R})$. Tratamos agora vários casos, que dependem do tipo da aresta $e_0$.

Primeiramente, se $e_0 = \overrightarrow{uv}$ é um segmento orientado, note que como $\overrightarrow{uv}$ é a única aresta em $\mathcal{R}$ que intersecta o segmento $pp'$, como $\mathcal{R}$ é uma boa representação de $X$ e como $p, p' \notin \partial X$, temos que $\chi_X(p) = 1 - \chi_X(p')$. Caso $p \in X$, então $p' \notin X$ e $\operatorname{rc}(p, e_0) = 1$. Pela hipótese indutiva, $\operatorname{rc}(p', \mathcal{R}) = \chi_X(p') = 0$, de forma que

$$\operatorname{rc}(p, \mathcal{R}) = \operatorname{rc}(p, e_0) + \operatorname{rc}(p', \mathcal{R}) = 1 + \chi_X(p') = 1 + 0 = 1,$$

pois $\operatorname{ray} p'$ intersecta exatamente as arestas $e_1, e_2, \ldots, e_{n-1}$ de $\mathcal{R}$. Similarmente, caso $p \notin X$, temos que $p' \in X$ e $\operatorname{rc}(p, e_0) = -1$. Novamente pela hipótese indutiva, $\operatorname{rc}(p', \mathcal{R}) = \chi_X(p') = 1$ e

$$\operatorname{rc}(p, \mathcal{R}) = \operatorname{rc}(p, e_0) + \operatorname{rc}(p', \mathcal{R}) = -1 + \chi_X(p') = -1 + 1 = 0.$$

Assim, temos que $\operatorname{rc}(p, \mathcal{R}) = \chi_X(p)$, como queríamos.

Tendo tratado do caso em que $e_0$ é um segmento orientado, suponha agora que $e_0$ é uma orientação de um arco de circunferência $\operatorname{arc}(C, u, v)$. Tratamos em separado o caso em que $C$ é tangente a $\operatorname{ray} p$. Como $C$ é apenas tangente a $\operatorname{ray} p$, então $\chi_X(p) = \chi_X(p')$ e $\operatorname{rc}(p, e_0) = 0$. Pela hipótese indutiva, $\operatorname{rc}(p', \mathcal{R}) = \chi_X(p')$ e assim

$$\operatorname{rc}(p, \mathcal{R}) = \operatorname{rc}(p, e_0) + \operatorname{rc}(p', \mathcal{R}) = 0 + \chi_X(p') = \chi_X(p),$$

também como desejado, pois novamente $\operatorname{ray} p'$ intersecta exatamente as arestas $e_1, e_2, \ldots, e_{n-1}$ de $\mathcal{R}$.

Voltamos agora nossa atenção ao caso em que $C$ não é tangente a $\operatorname{ray} p$. Dividimos este caso em dois casos. Primeiro consideramos que $\operatorname{rc}(p, e_0) \neq 0$. Este caso é extremamente similar ao caso em que $e_0$ é um segmento orientado. Brevemente, então, temos que se $p \in X$, então $p' \notin X$ e $\operatorname{rc}(p, e_0) = 1$. Assim, pela hipótese indutiva,

$$\operatorname{rc}(p, \mathcal{R}) = \operatorname{rc}(p, e_0) + \operatorname{rc}(p', \mathcal{R}) = 1 + \chi_X(p') = 1 + 0 = 1.$$

Ainda, se $p \notin X$, então $p' \in X$ e $\operatorname{rc}(p, e_0) = -1$, de forma que, pela hipótese indutiva,

$$\operatorname{rc}(p, \mathcal{R}) = \operatorname{rc}(p, e_0) + \operatorname{rc}(p', \mathcal{R}) = -1 + \chi_X(p') = -1 + 1 = 0.$$

Observe que usamos fortemente o fato de que $\operatorname{ray} p'$ intersecta exatamente as arestas $e_1, e_2, \ldots, e_{n-1}$ para deduzir $\operatorname{rc}(p, \mathcal{R})$ a partir de $\operatorname{rc}(p', \mathcal{R})$. Como $e_0$ é um arco de circunferência, temos de ter o cuidado de garantir que $e_0$ apareça uma vez apenas na lista para estas deduções serem válidas. No entanto, o fato de que $\operatorname{rc}(p, e_0) \neq 0$ garante que $\operatorname{ray} p$ tem apenas uma intersecção com $e_0$.

Finalmente chegamos ao caso mais interessante em que $e_0$ é a orientação de um arco de circunferência $\operatorname{arc}(C, u, v)$ no sentido $\sigma \in \{-1, 1\}$, $C$ não é tangente a $\operatorname{ray} p$ e $\operatorname{rc}(p, e_0) = 0$. Como o círculo $C$ não é tangente a $\operatorname{ray} p$, devemos concluir que $e_0$ aparece exatamente duas vezes na lista $e_0 e_1 \cdots e_{n-1}$, digamos nos índices 0 e $i$ com $i \in \{1, 2, \ldots, n-1\}$. Claro, como a aresta $e_0$ cruza o segmento $pp'$ (apenas uma vez), temos que $\chi_X(p) = 1 - \chi_X(p')$. Ainda, como a aresta $e_0$ intersecta $\operatorname{ray} p$ entre $p$ e $p'$ e depois de $p'$, segue que $p \notin C$ e $p' \in C^\circ$. Assim, temos que $\operatorname{rc}(p', e_0) = \sigma$. Note que $\operatorname{rc}(p, \mathcal{R}) = \operatorname{rc}(p', \mathcal{R}) - \sigma$, pois $\operatorname{ray} p$ e $\operatorname{ray} p'$ intersectam precisamente as mesmas arestas (como $i > 0$ temos que $\operatorname{ray} p'$ intersecta $e_0 = e_i$), mas $\operatorname{rc}(p, e_0) = 0$ enquanto que $\operatorname{rc}(p', e_0) = \sigma$. Observe também que $p \in X$ se, e somente se, $\sigma = -1$ e que, portanto, $\chi_X(p) = \dfrac{1 - \sigma}{2}$, ou seja, $\sigma = 1 - 2\chi_X(p)$. Logo, pela hipótese indutiva, temos que

$$\begin{aligned}
\operatorname{rc}(p, \mathcal{R}) &= \operatorname{rc}(p', \mathcal{R}) - \sigma \\
&= \chi_X(p') - \sigma \\
&= 1 - \chi_X(p) - \sigma \\
&= 1 - \chi_X(p) - \big(1 - 2\chi_X(p)\big) \\
&= 1 - 1 - \chi_X(p) + 2\chi_X(p) \\
&= \chi_X(p),
\end{aligned}$$

concluindo a demonstração. $\square$

Também como anteriormente, seja agora $X \subseteq \mathbb{R}^2$ um conjunto J-mensurável e seja $\mathcal{R}$ uma boa representação de $X$. Considere ainda um retângulo aberto $R$ contendo $X$ e a origem. Temos então que

$$\begin{aligned}
\operatorname{area} X &= \int_R \chi_X(p)\ \mathrm{d}p \\
&= \int_R \operatorname{rc}(p, \mathcal{R})\ \mathrm{d}p \\
&= \int_R \sum_{\alpha\in\mathcal{R}} \operatorname{rc}(p, \alpha)\ \mathrm{d}p \\
&= \sum_{\alpha\in\mathcal{R}} \int_R \operatorname{rc}(p, \alpha)\ \mathrm{d}p.
\end{aligned}$$

Novamente, estamos interessados em computar

$$\int_R \operatorname{rc}(p, \alpha)\ \mathrm{d}p$$

para cada elemento $\alpha$ de $\mathcal{R}$. Se $\alpha$ for um segmento de reta orientado, já sabemos como calcular esta integral. Resta então tratarmos do caso em que $\alpha$ é um arco de circunferência orientado.

**Lema 2** — Seja $\gamma : [0, 1] \to \mathbb{R}^2$ a parametrização canônica de um arco de circunferência orientado, seja $\gamma' : [0, 1] \to \mathbb{R}^2$ a função derivada de $\gamma$ e seja $F : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ uma função contínua que satisfaz:

- $F(u, v_0) + F(u, v_1) = F(u, v_0 + v_1)$ para todos os vetores $u, v_0, v_1 \in \mathbb{R}^2$;
- $F(u, \lambda v) = \lambda F(u, v)$ para todos os vetores $u, v \in \mathbb{R}^2$ e todo escalar real $\lambda \in \mathbb{R}$; e
- Existe um número natural $k \in \mathbb{N}$ tal que $\left|F(u, v)\right| \leq ||u||^k\, ||v||$ para todos os vetores $u, v \in \mathbb{R}^2$.

Então

$$\lim_{n\to\infty} \sum_{i=0}^{n-1} F\Big(\gamma\big({}^{i}/_{n}\big), \gamma\big({}^{i+1}/_{n}\big) - \gamma\big({}^{i}/_{n}\big)\Big) = \int_0^1 F\big(\gamma(t), \gamma'(t)\big)\ \mathrm{d}t.$$

**Prova** — Seja $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ um valor real positivo arbitrário. Vamos mostrar que

$$\left| \int_0^1 F\big(\gamma(t), \gamma'(t)\big)\ \mathrm{d}t - \lim_{n\to\infty} \sum_{i=0}^{n-1} F\Big(\gamma\big({}^{i}/_{n}\big), \gamma\big({}^{i+1}/_{n}\big) - \gamma\big({}^{i}/_{n}\big)\Big) \right| < \varepsilon.$$

Começamos observando que o método de integração por retângulos nos garante que

$$\int_0^1 F\big(\gamma(t), \gamma'(t)\big)\ \mathrm{d}t = \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} F\Big(\gamma\big({}^{i}/_{n}\big), \gamma'\big({}^{i}/_{n}\big)\Big),$$

pois $\gamma'$ é contínua.

Note agora que

$$\gamma'(t) = \lim_{h\to 0} \frac{\gamma(t+h) - \gamma(t)}{h}$$

para todo $t \in [0,1]$ e que, como $\gamma$ é a parametrização canônica de um arco de circunferência orientado, o comprimento

$$\left\|\gamma'(t) - \frac{\gamma(t+h)-\gamma(t)}{h}\right\|$$

independe da escolha de $t$ uma vez fixado $h$. Juntando estes dois fatos, temos que existe um número natural não-nulo $n_0 \in \mathbb{N} \setminus \{0\}$ tal que, para todo $n \in \mathbb{N}$ com $n \geq n_0$ e para todo $i \in \{0, 1, \ldots, n-1\}$,

$$\left\|\gamma'\big({}^{i}\!/\!{}_{n}\big) - n\Big(\gamma\big({}^{i+1}\!/\!{}_{n}\big) - \gamma\big({}^{i}\!/\!{}_{n}\big)\Big)\right\| < \frac{\varepsilon}{M^k},$$

onde $M$ é um número real positivo maior ou igual à norma de todo ponto em $\operatorname{Im}(\gamma)$ (tal número existe porque claramente $\operatorname{Im}(\gamma)$ é um conjunto limitado).

Portanto,

$$\begin{aligned}
\left|\int_0^1 F\big(\gamma(t),\gamma'(t)\big)\,\mathrm{d}t - \lim_{n\to\infty}\sum_{i=0}^{n-1} F\Big(\gamma\big({}^{i}\!/\!{}_{n}\big), \gamma\big({}^{i+1}\!/\!{}_{n}\big) - \gamma\big({}^{i}\!/\!{}_{n}\big)\Big)\right| &= \\
\left|\lim_{n\to\infty}\frac{1}{n}\sum_{i=0}^{n-1} F\Big(\gamma\big({}^{i}\!/\!{}_{n}\big), \gamma'\big({}^{i}\!/\!{}_{n}\big)\Big) - \lim_{n\to\infty}\frac{1}{n}\sum_{i=0}^{n-1} nF\Big(\gamma\big({}^{i}\!/\!{}_{n}\big), \gamma\big({}^{i+1}\!/\!{}_{n}\big) - \gamma\big({}^{i}\!/\!{}_{n}\big)\Big)\right| &= \\
\left|\lim_{n\to\infty}\frac{1}{n}\sum_{i=0}^{n-1}\bigg(F\Big(\gamma\big({}^{i}\!/\!{}_{n}\big), \gamma'\big({}^{i}\!/\!{}_{n}\big)\Big) - F\bigg(\gamma\big({}^{i}\!/\!{}_{n}\big), n\Big(\gamma\big({}^{i+1}\!/\!{}_{n}\big) - \gamma\big({}^{i}\!/\!{}_{n}\big)\Big)\bigg)\bigg)\right| &= \\
\lim_{n\to\infty}\frac{1}{n}\left|\sum_{i=0}^{n-1} F\bigg(\gamma\big({}^{i}\!/\!{}_{n}\big), \gamma'\big({}^{i}\!/\!{}_{n}\big) - n\Big(\gamma\big({}^{i+1}\!/\!{}_{n}\big) - \gamma\big({}^{i}\!/\!{}_{n}\big)\Big)\bigg)\right| &\leq \\
\lim_{n\to\infty}\frac{1}{n}\sum_{i=0}^{n-1}\left|F\bigg(\gamma\big({}^{i}\!/\!{}_{n}\big), \gamma'\big({}^{i}\!/\!{}_{n}\big) - n\Big(\gamma\big({}^{i+1}\!/\!{}_{n}\big) - \gamma\big({}^{i}\!/\!{}_{n}\big)\Big)\bigg)\right| &\leq \\
\lim_{n\to\infty}\frac{1}{n}\sum_{i=0}^{n-1}\left\|\gamma\big({}^{i}\!/\!{}_{n}\big)\right\|^k \left\|\gamma'\big({}^{i}\!/\!{}_{n}\big) - n\Big(\gamma\big({}^{i+1}\!/\!{}_{n}\big) - \gamma\big({}^{i}\!/\!{}_{n}\big)\Big)\right\| &< \\
\lim_{n\to\infty}\frac{1}{n}\sum_{i=0}^{n-1} M^k\frac{\varepsilon}{M^k} &= \\
\lim_{n\to\infty}\frac{1}{n}\sum_{i=0}^{n-1}\varepsilon &= \varepsilon,
\end{aligned}$$

como queríamos. □

> **Proposição 13** — Seja $\alpha$ um arco de circunferência orientado, seja $R \subseteq \mathbb{R}^2$ um retângulo aberto contendo todos os pontos em $\alpha$ e a origem, seja $\gamma : [0,1] \to \mathbb{R}^2$ a parametrização canônica $\widetilde{\alpha}$ de $\alpha$ e seja $\gamma' : [0,1] \to \mathbb{R}^2$ a função derivada de $\gamma$. Então
>
> $$\int_R \operatorname{rc}(p,\alpha)\,\mathrm{d}p = \frac{1}{2}\int_0^1 \gamma(t)\times\gamma'(t)\,\mathrm{d}t.$$

**Prova** — A estratégia que usaremos para realizar esta demonstração consiste em discretizar o arco $\alpha$. Mais precisamente, se $\alpha$ é uma orientação de um arco de circunferência $\operatorname{arc}(C, u, v)$, então defina, para cada $n \in \mathbb{N} \setminus \{0\}$:

- O ponto $a_{n,i}$ como $\gamma\left({}^{i}\!/\!_{n}\right)$ para cada $i \in \{0, 1, \ldots, n\}$;
- O conjunto $A_{n,i}$ como o fecho do interior da curva de Jordan

$$\widetilde{\operatorname{arc}}(C, a_{n,i}, a_{n,i+1}) \cdot \widetilde{a_{n,i+1} a_{n,i}}$$

  para cada $i \in \{0, 1, \ldots, n-1\}$;
- O conjunto de segmentos de reta orientados e contidos em $R$ (pois $R$ é um conjunto convexo)

$$S_n = \left\{\overrightarrow{a_{n,i} a_{n,i+1}} : i \in \{0, 1, \ldots, n-1\}\right\}; \text{ e}$$

- O conjunto $A_n$ como a união

$$\bigcup_{i=0}^{n-1} A_{n,i}.$$

Assumimos (sem demonstração) os três seguintes fatos intuitivos:

- $A_n$ é um conjunto J-mensurável para todo $n \in \mathbb{N} \setminus \{0\}$;
- $\lim\limits_{n \to \infty} \operatorname{area} A_n = 0$; e
- Para todo $n \in \mathbb{N} \setminus \{0\}$, se um ponto $p \in \mathbb{R}^2$ é tal que

$$\operatorname{rc}(p, \alpha) \neq \operatorname{rc}(p, S_n) = \sum_{\overrightarrow{ab} \in S_n} \operatorname{rc}(p, \overrightarrow{ab}),$$

  então $p \in A_n$ ou $p \in \{t a_{n,i} : 0 \leq t \leq 1\}$ para algum $i \in \{0, 1, \ldots, n\}$, de forma que o conjunto dos pontos onde estes *ray castings* diferem é sempre J-mensurável e possui uma área arbitrariamente pequena com $n$ arbitrariamente grande.

Desta forma, podemos concluir que

$$\begin{aligned}
\int_R \operatorname{rc}(p, \alpha)\ \mathrm{d}p &= \lim_{n \to \infty} \int_R \operatorname{rc}(p, S_n)\ \mathrm{d}p \\
&= \lim_{n \to \infty} \int_R \sum_{\overrightarrow{ab} \in S_n} \operatorname{rc}(p, \overrightarrow{ab})\ \mathrm{d}p \\
&= \lim_{n \to \infty} \sum_{\overrightarrow{ab} \in S_n} \int_R \operatorname{rc}(p, \overrightarrow{ab})\ \mathrm{d}p.
\end{aligned}$$

Logo, pela proposição 9, temos que

$$\int_R \operatorname{rc}(p, \alpha)\ \mathrm{d}p = \lim_{n \to \infty} \sum_{\overrightarrow{ab} \in S_n} \frac{1}{2} a \times b = \frac{1}{2} \lim_{n \to \infty} \sum_{i=0}^{n-1} \gamma\left({}^{i}\!/\!_{n}\right) \times \gamma\left({}^{i+1}\!/\!_{n}\right),$$

restando apenas mostrar que

$$\lim_{n\to\infty}\sum_{i=0}^{n-1}\gamma\big({}^{i}/_{n}\big)\times\gamma\big({}^{i+1}/_{n}\big)=\lim_{n\to\infty}\sum_{i=0}^{n-1}\gamma\big({}^{i}/_{n}\big)\times\Big(\gamma\big({}^{i+1}/_{n}\big)-\gamma\big({}^{i}/_{n}\big)\Big)=\int_0^1\gamma(t)\times\gamma'(t)\ \mathrm{d}t.$$

Porém, isto é consequência direta do lema 2 (tome $F(u,v) = u \times v$ e verifique que as hipóteses do lema valem para esta escolha da função $F$). □

Na verdade podemos mostrar uma primitiva para esta integral.

**Proposição 14** — Seja $\alpha = \overrightarrow{\operatorname{arc}}(C, u, v)$ um arco de circunferência orientado em sentido anti-horário tal que:

- O centro do círculo $C$ é $c = (x, y) \in \mathbb{R}^2$;
- O círculo $C$ tem raio $r$;
- O único valor $t \in [0, 2\pi)$ tal que $u = c + r(\cos t, \sin t)$ é $\theta_0$; e
- O único valor $t \in (\theta_0, \theta_0 + 2\pi)$ tal que $v = c + r(\cos t, \sin t)$ é $\theta_1$.

Seja também $R \subseteq \mathbb{R}^2$ um retângulo aberto contendo todos os pontos em $\alpha$ e a origem. Então

$$\int_R \operatorname{rc}(p,\alpha)\ \mathrm{d}p = \frac{1}{2}\big(r^2(\theta_1-\theta_0) + rx(\sin\theta_1-\sin\theta_0) - ry(\cos\theta_1-\cos\theta_0)\big).$$

Ainda,

$$\int_R \operatorname{rc}(p,\operatorname{rev}\alpha)\ \mathrm{d}p = -\int_R \operatorname{rc}(p,\alpha)\ \mathrm{d}p.$$

**Prova** — Defina a função $\theta : [0,1] \to \mathbb{R}$ por $\theta(t) = (1-t)\theta_0 + t\theta_1$, de modo que $\theta(0) = \theta_0$ e $\theta(1) = \theta_1$. Assim, não há problemas em abreviarmos $\theta(t)$ por $\theta_t$ para cada $t \in [0,1]$. Ainda, se $\theta' : [0,1] \to \mathbb{R}$ é a função derivada de $\theta$, então $\theta'(t) = \theta_1 - \theta_0$ para todo $t \in [0,1]$.

Dessa forma, se $\gamma : [0,1] \to \mathbb{R}^2$ é a parametrização canônica $\widetilde{\alpha}$ de $\alpha$, então

$$\gamma(t) = c + r(\cos\theta_t, \sin\theta_t)$$

para todo $t \in [0,1]$. Logo, se $\gamma' : [0,1] \to \mathbb{R}^2$ é a função derivada de $\gamma$, então, pela regra da cadeia, $\gamma'(t) = r(\theta_1 - \theta_0)(-\sin\theta_t, \cos\theta_t)$ para cada $t \in [0,1]$.

Porém, a proposição 13 transforma o restante desta demonstração em um mero exercício de integração, uma vez que ela garante que

$$\begin{aligned}
\int_R \operatorname{rc}(p,\alpha)\ \mathrm{d}p &= \frac{1}{2}\int_0^1 \gamma(t)\times\gamma'(t)\ \mathrm{d}t \\
&= \frac{1}{2}\int_0^1 \big(c + r(\cos\theta_t,\sin\theta_t)\big)\times\big(r(\theta_1-\theta_0)(-\sin\theta_t,\cos\theta_t)\big)\ \mathrm{d}t \\
&= \frac{1}{2}r(\theta_1-\theta_0)\int_0^1 \big(c + r(\cos\theta_t,\sin\theta_t)\big)\times(-\sin\theta_t,\cos\theta_t)\ \mathrm{d}t \\
&= \frac{1}{2}r(\theta_1-\theta_0)\int_0^1 \big((x,y)\times(-\sin\theta_t,\cos\theta_t) + r(\cos\theta_t,\sin\theta_t)\times(-\sin\theta_t,\cos\theta_t)\big)\ \mathrm{d}t \\
&= \frac{1}{2}r(\theta_1-\theta_0)\int_0^1 \big(x\cos\theta_t + y\sin\theta_t + r(\cos^2\theta_t+\sin^2\theta_t)\big)\ \mathrm{d}t \\
&= \frac{1}{2}r(\theta_1-\theta_0)\int_0^1 (x\cos\theta_t + y\sin\theta_t + r)\ \mathrm{d}t \\
&= \frac{1}{2}r(\theta_1-\theta_0)\left(\frac{x\sin\theta_t}{\theta_1-\theta_0} - \frac{y\cos\theta_t}{\theta_1-\theta_0} + rt\right)\Bigg|_{t=0}^{1} \\
&= \frac{1}{2}\big(r^2(\theta_1-\theta_0)t + rx\sin\theta_t - ry\cos\theta_t\big)\Big|_{t=0}^{1} \\
&= \frac{1}{2}\big(r^2(\theta_1-\theta_0) + rx(\sin\theta_1-\sin\theta_0) - ry(\cos\theta_1-\cos\theta_0)\big).
\end{aligned}$$

Para completar a demonstração, observe que $\operatorname{rc}(p,\operatorname{rev}\alpha) = -\operatorname{rc}(p,\alpha)$ para todo ponto $p\in\mathbb{R}^2$. □

Um fato interessante (e importante em uma implementação) sobre a fórmula estabelecida por esta última proposição é que, se tivermos como entrada o círculo $C$ e os pontos $u$ e $v$, os senos e cossenos dos ângulos $\theta_0$ e $\theta_1$ podem ser computados de forma bastante simples. Por exemplo, temos que $\cos\theta_0$ é a diferença entre as abscissas do centro $c$ e de $u$ dividida pelo raio $r$. Porém, não vemos como computar a diferença entre os ângulos $\theta_0$ e $\theta_1$ de forma mais simples do que com uma rotina para computar arcos de cosseno.

A proposição 13 pode ser adaptada para segmentos de reta, como mostramos a seguir.

**Proposição 15** — Seja $\overrightarrow{uv}$ um segmento de reta orientado, seja $R\subseteq\mathbb{R}^2$ um retângulo aberto contendo todos os pontos em $uv$ e a origem, seja $\gamma:[0,1]\to\mathbb{R}^2$ a parametrização canônica $\widetilde{uv}$ de $\overrightarrow{uv}$ e seja $\gamma':[0,1]\to\mathbb{R}^2$ a função derivada de $\gamma$. Então

$$\int_R \operatorname{rc}(p,\overrightarrow{uv})\ \mathrm{d}p = \frac{1}{2}u\times v = \frac{1}{2}\int_0^1 \gamma(t)\times\gamma'(t)\ \mathrm{d}t.$$

**Prova** — Trata-se de um simples exercício de integração. □

Juntando tudo isso podemos estabelecer como obter a área de um conjunto J-mensurável a partir de uma boa representação deste conjunto.

**Teorema 3** — Seja $X \subseteq \mathbb{R}^2$ um conjunto J-mensurável e seja $\mathcal{R}$ uma boa representação de $X$. Para cada elemento $\alpha \in \mathcal{R}$, denote por $\widetilde{\alpha}'$ a derivada da parametrização canônica $\widetilde{\alpha}$ de $\alpha$. Então

$$\operatorname{area} X = \frac{1}{2} \sum_{\alpha \in \mathcal{R}} \int_0^1 \widetilde{\alpha}(t) \times \widetilde{\alpha}'(t)\, \mathrm{d}t.$$

**Prova** — Imediata tendo em vista o que já discutimos. □

Este raciocínio pode ser repetido a fim de calcularmos centros de massa. Devido à proposição 12, se $X \subseteq \mathbb{R}^2$ é um conjunto J-mensurável que admite uma boa representação $\mathcal{R}$ e $R$ é um retângulo aberto contendo $X$ e a origem, então

$$\begin{aligned}
\operatorname{centroid} X &= \frac{1}{\operatorname{area} X} \int_R \chi_X(p) p \,\mathrm{d}p \\
&= \frac{1}{\operatorname{area} X} \int_R \operatorname{rc}(p, \mathcal{R}) p \,\mathrm{d}p \\
&= \frac{1}{\operatorname{area} X} \int_R \Bigg( \sum_{\alpha \in \mathcal{R}} \operatorname{rc}(p, \alpha) \Bigg) p \,\mathrm{d}p \\
&= \frac{1}{\operatorname{area} X} \int_R \sum_{\alpha \in \mathcal{R}} \operatorname{rc}(p, \alpha) p \,\mathrm{d}p \\
&= \frac{1}{\operatorname{area} X} \sum_{\alpha \in \mathcal{R}} \int_R \operatorname{rc}(p, \alpha) p \,\mathrm{d}p.
\end{aligned}$$

Já sabemos, devido à proposição 10, como calcular estas últimas integrais no caso de $\alpha$ ser um segmento orientado. Vejamos então como tratar o caso em que $\alpha$ é um arco de circunferência orientado. Antes, porém, generalizamos o lema 2.

**Corolário 5** — Seja $\gamma : [0, 1] \to \mathbb{R}^2$ a parametrização canônica de um arco de circunferência orientado, seja $\gamma' : [0, 1] \to \mathbb{R}^2$ a função derivada de $\gamma$ e seja $F : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^2$ uma função contínua que satisfaz:

- $F(u, v_0) + F(u, v_1) = F(u, v_0 + v_1)$ para todos os vetores $u, v_0, v_1 \in \mathbb{R}^2$;
- $F(u, \lambda v) = \lambda F(u, v)$ para todos os vetores $u, v \in \mathbb{R}^2$ e todo escalar real $\lambda \in \mathbb{R}$; e
- Existe um número natural $k \in \mathbb{N}$ tal que $\big|\big|F(u, v)\big|\big| \leq ||u||^k \, ||v||$ para todos os vetores $u, v \in \mathbb{R}^2$.

Então

$$\lim_{n \to \infty} \sum_{i=0}^{n-1} F\Big(\gamma\big({}^{i}\!/\!{}_{n}\big), \gamma\big({}^{i+1}\!/\!{}_{n}\big) - \gamma\big({}^{i}\!/\!{}_{n}\big)\Big) = \int_0^1 F\big(\gamma(t), \gamma'(t)\big)\, \mathrm{d}t.$$

**Prova** — Tendo o lema 2, esta prova é bastante simples (e portanto o título “corolário”).

Primeiramente consideramos as funções componentes $F_0, F_1 : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ de $F$, de forma que $F(u, v) = \big(F_0(u, v), F_1(u, v)\big)$ para todos os vetores $u, v \in \mathbb{R}^2$. Basta então percebermos que

$$\int_0^1 F\big(\gamma(t), \gamma'(t)\big)\ \mathrm{d}t = \left(\int_0^1 F_0\big(\gamma(t), \gamma'(t)\big)\ \mathrm{d}t, \int_0^1 F_1\big(\gamma(t), \gamma'(t)\big)\ \mathrm{d}t\right) \in \mathbb{R}^2$$

e, similarmente, que

$$\lim_{n\to\infty} \sum_{i=0}^{n-1} F\Big(\gamma\big(i/n\big), \gamma\big(i+1/n\big) - \gamma\big(i/n\big)\Big) =$$

$$\left(\lim_{n\to\infty} \sum_{i=0}^{n-1} F_0\Big(\gamma\big(i/n\big), \gamma\big(i+1/n\big) - \gamma\big(i/n\big)\Big), \lim_{n\to\infty} \sum_{i=0}^{n-1} F_1\Big(\gamma\big(i/n\big), \gamma\big(i+1/n\big) - \gamma\big(i/n\big)\Big)\right) \in \mathbb{R}^2.$$

Desta forma, temos que

$$\lim_{n\to\infty} \sum_{i=0}^{n-1} F\Big(\gamma\big(i/n\big), \gamma\big(i+1/n\big) - \gamma\big(i/n\big)\Big) = \int_0^1 F\big(\gamma(t), \gamma'(t)\big)\ \mathrm{d}t$$

se, e somente se,

$$\lim_{n\to\infty} \sum_{i=0}^{n-1} F_0\Big(\gamma\big(i/n\big), \gamma\big(i+1/n\big) - \gamma\big(i/n\big)\Big) = \int_0^1 F_0\big(\gamma(t), \gamma'(t)\big)\ \mathrm{d}t$$

e

$$\lim_{n\to\infty} \sum_{i=0}^{n-1} F_1\Big(\gamma\big(i/n\big), \gamma\big(i+1/n\big) - \gamma\big(i/n\big)\Big) = \int_0^1 F_1\big(\gamma(t), \gamma'(t)\big)\ \mathrm{d}t.$$

Assim, resta mostrar que as funções $F_0$ e $F_1$ satisfazem as hipóteses do lema 2. Porém, isto é simples ao percebermos que, para todos os vetores $u, v \in \mathbb{R}^2$, ambos os números $\big|F_0(u, v)\big|$ e $\big|F_1(u, v)\big|$ são menores ou iguais que $\big|\big|F(u, v)\big|\big|$. □

**Proposição 16** — Seja $\alpha$ um arco de circunferência orientado, seja $R \subseteq \mathbb{R}^2$ um retângulo aberto contendo todos os pontos em $\alpha$ e a origem, seja $\gamma : [0, 1] \to \mathbb{R}^2$ a parametrização canônica $\widetilde{\alpha}$ de $\alpha$ e seja $\gamma' : [0, 1] \to \mathbb{R}^2$ a função derivada de $\gamma$. Então

$$\int_R \mathrm{rc}(p, \alpha) p\ \mathrm{d}p = \frac{1}{3} \int_0^1 \big(\gamma(t) \times \gamma'(t)\big)\gamma(t)\ \mathrm{d}t.$$

**Prova** — Esta demonstração é bastante similar à demonstração da proposição 13. Lembramos então que, naquela demonstração, discretizamos o arco $\alpha$. Isto é, supondo que $\alpha$ é uma orientação de um arco de circunferência $\mathrm{arc}(C, u, v)$, definimos, para cada $n \in \mathbb{N}$:

- O ponto $a_{n,i}$ como $\gamma\big(i/n\big)$ para cada $i \in \{0, 1, \ldots, n\}$;
- O conjunto $A_{n,i}$ como o fecho do interior da curva de Jordan

  $$\widetilde{\mathrm{arc}}(C, a_{n,i}, a_{n,i+1}) \cdot \widetilde{a_{n,i+1} a_{n,i}}$$

  para cada $i \in \{0, 1, \ldots, n-1\}$;

- O conjunto de segmentos de reta orientados e contidos em $R$ (pois $R$ é convexo)

$$S_n = \left\{\overrightarrow{a_{n,i}a_{n,i+1}} : i \in \{0, 1, \ldots, n-1\}\right\}; \text{ e}$$

- O conjunto $A_n$ como a união

$$\bigcup_{i=0}^{n-1} A_{n,i}.$$

Também havíamos estabelecido três fatos:

- $A_n$ é um conjunto J-mensurável para todo $n \in \mathbb{N} \setminus \{0\}$;
- $\lim_{n\to\infty} \operatorname{area} A_n = 0$; e
- Para todo $n \in \mathbb{N} \setminus \{0\}$, se um ponto $p \in \mathbb{R}^2$ é tal que

$$\operatorname{rc}(p, \alpha) \neq \operatorname{rc}(p, S_n) = \sum_{\overrightarrow{ab} \in S_n} \operatorname{rc}(p, \overrightarrow{ab}),$$

então $p \in A_n$ ou $p \in \{ta_{n,i} : 0 \leq t \leq 1\}$ para algum $i \in \{0, 1, \ldots, n\}$, de forma que o conjunto dos pontos onde estes *ray castings* diferem é sempre J-mensurável e possui uma área arbitrariamente pequena com $n$ arbitrariamente grande.

Assim, podemos concluir que

$$\begin{aligned}
\int_R \operatorname{rc}(p, \alpha) p \, \mathrm{d}p &= \lim_{n\to\infty} \int_R \operatorname{rc}(p, S_n) p \, \mathrm{d}p \\
&= \lim_{n\to\infty} \int_R \left( \sum_{\overrightarrow{ab} \in S_n} \operatorname{rc}(p, \overrightarrow{ab}) \right) p \, \mathrm{d}p \\
&= \lim_{n\to\infty} \int_R \sum_{\overrightarrow{ab} \in S_n} \operatorname{rc}(p, \overrightarrow{ab}) p \, \mathrm{d}p \\
&= \lim_{n\to\infty} \sum_{\overrightarrow{ab} \in S_n} \int_R \operatorname{rc}(p, \overrightarrow{ab}) p \, \mathrm{d}p.
\end{aligned}$$

Porém, a proposição 10 prontamente nos mostra que

$$\int_R \operatorname{rc}(p, \overrightarrow{uv}) p \, \mathrm{d}p = \frac{1}{6}(u \times v)(u + v)$$

para todo segmento de reta orientado $\overrightarrow{uv}$ com $uv \subseteq R$, de forma que

$$\begin{aligned}
\int_R \operatorname{rc}(p, \alpha) p \, \mathrm{d}p &= \lim_{n\to\infty} \sum_{\overrightarrow{ab} \in S_n} \frac{1}{6}(a \times b)(a + b) \\
&= \frac{1}{6} \lim_{n\to\infty} \sum_{i=0}^{n-1} \Big( \gamma\big(i/n\big) \times \gamma\big(i+1/n\big) \Big) \Big( \gamma\big(i/n\big) + \gamma\big(i+1/n\big) \Big).
\end{aligned}$$

Claramente podemos reescrever este limite como

$$\lim_{n\to\infty}\left(\frac{1}{6}\sum_{i=0}^{n-1}\Big(\gamma\big(i/n\big)\times\gamma\big(i+1/n\big)\Big)\gamma\big(i/n\big)\ +\ \frac{1}{6}\sum_{i=0}^{n-1}\Big(\gamma\big(i/n\big)\times\gamma\big(i+1/n\big)\Big)\gamma\big(i+1/n\big)\right)\ =$$

$$\lim_{n\to\infty}\left(2\frac{1}{6}\sum_{i=0}^{n-1}\Big(\gamma\big(i/n\big)\times\gamma\big(i+1/n\big)\Big)\gamma\big(i/n\big)\ +\ \frac{1}{6}\sum_{i=0}^{n-1}\Big(\gamma\big(i/n\big)\times\gamma\big(i+1/n\big)\Big)\Big(\gamma\big(i+1/n\big)-\gamma\big(i/n\big)\Big)\right).$$

Mostraremos em breve que

$$\lim_{n\to\infty}\sum_{i=0}^{n-1}\Big(\gamma\big(i/n\big)\times\gamma\big(i+1/n\big)\Big)\Big(\gamma\big(i+1/n\big)-\gamma\big(i/n\big)\Big)=0,$$

de forma que

$$\int_R \mathrm{rc}(p,\alpha)p\ \mathrm{d}p=\frac{1}{3}\lim_{n\to\infty}\sum_{i=0}^{n-1}\Big(\gamma\big(i/n\big)\times\gamma\big(i+1/n\big)\Big)\gamma\big(i/n\big).$$

Assim, uma vez que percebemos que a função

$$\begin{array}{rccc} F: & \mathbb{R}^2\times\mathbb{R}^2 & \to & \mathbb{R}^2 \\ & (u,v) & \mapsto & (u\times v)u \end{array}$$

satisfaz todas as hipóteses do corolário 5, concluímos que

$$\begin{aligned}
\int_R \mathrm{rc}(p,\alpha)p\ \mathrm{d}p &= \frac{1}{3}\lim_{n\to\infty}\sum_{i=0}^{n-1}\bigg(\gamma\big(i/n\big)\times\Big(\gamma\big(i+1/n\big)-\gamma\big(i/n\big)\Big)\bigg)\gamma\big(i/n\big)\\
&= \frac{1}{3}\lim_{n\to\infty}\sum_{i=0}^{n-1}F\Big(\gamma\big(i/n\big),\gamma\big(i+1/n\big)-\gamma\big(i/n\big)\Big)\\
&= \frac{1}{3}\int_0^1 F\big(\gamma(t),\gamma'(t)\big)\ \mathrm{d}t\\
&= \frac{1}{3}\int_0^1 \big(\gamma(t)\times\gamma'(t)\big)\gamma(t)\ \mathrm{d}t,
\end{aligned}$$

como desejado.

Seja então, para cada $n\in\mathbb{N}$ e cada $i\in\{0,1,\ldots,n-1\}$, $T_{n,i}\subseteq\mathbb{R}^2$ o triângulo que possui como vértices a origem e os pontos $a_{n,i}$ e $a_{n,i+1}$, de forma que

$$\Big|\gamma\big(i/n\big)\times\gamma\big(i+1/n\big)\Big|=2(\operatorname{area}T_i).$$

Note agora que, dado $n\in\mathbb{N}$, cada ponto $p\in R$ no retângulo $R$ pode estar no máximo no interior de dois triângulos $T_{n,i}$ com $i\in\{0,1,\ldots,n-1\}$. De fato, se $p\in T_{n,i}^\circ$ para algum $i\in\{0,1,\ldots,n-1\}$, então a reta $\ell_p=\{tp : t\in\mathbb{R}\}$ separa os pontos $a_{n,i}$ e $a_{n,i+1}$, de modo que o teorema do valor intermediário (teorema 1) nos garante que $\ell_p$ intersecta $\operatorname{arc}(C,a_{n,i},a_{n,i+1})\subseteq\partial C$ em um ponto diferente de $a_{n,i}$ e $a_{n,i+1}$. Assim, se um ponto $p$ é tal que $p\in T_{n,i}^\circ\cap T_{n,j}^\circ\cap T_{n,k}^\circ$ com $i,j,k\in\{0,1,\ldots,n-1\}$

e $\big|\{i,j,k\}\big| = 3$, temos que a reta $\ell_p$ intersecta $\partial C$ em três pontos distintos (pois os arcos $\operatorname{arc}(C, a_{n,i}, a_{n,i+1})$, $\operatorname{arc}(C, a_{n,j}, a_{n,j+1})$ e $\operatorname{arc}(C, a_{n,k}, a_{n,k+1})$ nunca se intersectam exceto possivelmente em seus extremos), o que é uma clara contradição. Logo, segue que, para todo $n \in \mathbb{N}$,

$$\sum_{i=0}^{n-1} \operatorname{area} T_{n,i} \leq 2(\operatorname{area} R).$$

Ainda, note que, para cada $n \in \mathbb{N}$ e cada $i \in \{0, 1, \ldots, n-1\}$, a distância entre $\gamma\big({}^{i}\!/_{n}\big)$ e $\gamma\big({}^{i+1}\!/_{n}\big)$, ou seja, a distância $||a_{n,i+1} - a_{n,i}||$, é menor do que o comprimento do arco $\operatorname{arc}(C, a_{n,i}, a_{n,i+1})$, que por sua vez é menor do que uma $n$-ésima parte do perímetro $2\pi r$ do círculo $C$ (sendo $r$ o raio de $C$).

Assim, podemos concluir que

$$\begin{aligned}
\left|\left|\sum_{i=0}^{n-1}\Big(\gamma\big({}^{i}\!/_{n}\big) \times \gamma\big({}^{i+1}\!/_{n}\big)\Big)\Big(\gamma\big({}^{i+1}\!/_{n}\big) - \gamma\big({}^{i}\!/_{n}\big)\Big)\right|\right| &\leq \\
\sum_{i=0}^{n-1}\left|\left|\Big(\gamma\big({}^{i}\!/_{n}\big) \times \gamma\big({}^{i+1}\!/_{n}\big)\Big)\Big(\gamma\big({}^{i+1}\!/_{n}\big) - \gamma\big({}^{i}\!/_{n}\big)\Big)\right|\right| &= \\
\sum_{i=0}^{n-1}\left|\Big(\gamma\big({}^{i}\!/_{n}\big) \times \gamma\big({}^{i+1}\!/_{n}\big)\Big)\right| \left|\left|\Big(\gamma\big({}^{i+1}\!/_{n}\big) - \gamma\big({}^{i}\!/_{n}\big)\Big)\right|\right| &< \\
\sum_{i=0}^{n-1} 2(\operatorname{area} T_{n,i})\frac{2\pi r}{n} &= \\
\frac{4\pi r}{n}\sum_{i=0}^{n-1} \operatorname{area} T_{n,i} &\leq \\
\frac{4\pi r}{n}\, 2(\operatorname{area} R) &= \\
\frac{8\pi r(\operatorname{area} R)}{n}.
\end{aligned}$$

Assim, segue que

$$\lim_{n\to\infty} \sum_{i=0}^{n-1}\Big(\gamma\big({}^{i}\!/_{n}\big) \times \gamma\big({}^{i+1}\!/_{n}\big)\Big)\Big(\gamma\big({}^{i+1}\!/_{n}\big) - \gamma\big({}^{i}\!/_{n}\big)\Big) = 0,$$

concluindo a prova. □

Novamente mostramos uma primitiva para a integral acima.

**Proposição 17** — Seja $\alpha = \overrightarrow{\mathrm{arc}}(C, u, v)$ um arco de circunferência orientado em sentido anti-horário tal que:

- O centro do círculo $C$ é $c = (x, y) \in \mathbb{R}^2$;
- O círculo $C$ tem raio $r$;
- O único valor $t \in [0, 2\pi)$ tal que $u = c + r(\cos t, \sin t)$ é $\theta_0$; e
- O único valor $t \in (\theta_0, \theta_0 + 2\pi)$ tal que $v = c + r(\cos t, \sin t)$ é $\theta_1$.

Então

$$\int_R \mathrm{rc}(p, \alpha)p \; \mathrm{d}p \in \mathbb{R}^2$$

é um vetor cuja abscissa é

$$\frac{1}{2}xr^2(\theta_1 - \theta_0) + \frac{1}{3}(x^2r + r^3)(\sin\theta_1 - \sin\theta_0) - \frac{1}{3}xyr(\cos\theta_1 - \cos\theta_0)$$
$$+\frac{1}{6}yr^2(\sin^2\theta_1 - \sin^2\theta_0) + \frac{1}{12}xr^2(\sin 2\theta_1 - \sin 2\theta_0)$$

e cuja ordenada é

$$\frac{1}{2}yr^2(\theta_1 - \theta_0) - \frac{1}{3}(y^2r + r^3)(\cos\theta_1 - \cos\theta_0) + \frac{1}{3}xyr(\sin\theta_1 - \sin\theta_0)$$
$$+\frac{1}{6}xr^2(\sin^2\theta_1 - \sin^2\theta_0) - \frac{1}{12}yr^2(\sin 2\theta_1 - \sin 2\theta_0).$$

Ainda,

$$\int_R \mathrm{rc}(p, \mathrm{rev}\,\alpha)p \; \mathrm{d}p = -\int_R \mathrm{rc}(p, \alpha)p \; \mathrm{d}p.$$

**Prova** — Esta prova partilha vários elementos presentes na prova da proposição 14. Primeiramente definimos a função $\theta : [0, 1] \to \mathbb{R}$ por $\theta(t) = (1 - t)\theta_0 + t\theta_1$, de modo que $\theta(0) = \theta_0$ e $\theta(1) = \theta_1$. Observamos então que não há ambiguidade ao denotarmos $\theta(t)$ por $\theta_t$ para cada $t \in [0, 1]$. Também temos que se $\theta' : [0, 1] \to \mathbb{R}$ é a função derivada de $\theta$, então $\theta'(t) = \theta_1 - \theta_0$ para todo $t \in [0, 1]$.

Também como naquela demonstração, definimos $\gamma : [0, 1] \to \mathbb{R}^2$ como a parametrização canônica $\widetilde{\alpha}$ de $\alpha$, de modo que

$$\gamma(t) = c + r(\cos\theta_t, \sin\theta_t)$$

para todo $t \in [0, 1]$. Assim, se $\gamma' : [0, 1] \to \mathbb{R}^2$ é a função derivada de $\gamma$, então, novamente pela regra da cadeia, $\gamma'(t) = r(\theta_1 - \theta_0)(-\sin\theta_t, \cos\theta_t)$ para cada $t \in [0, 1]$.

A proposição 16 tem nesta prova o papel que a proposição 13 tinha na prova da proposição 14: ela transforma o restante desta demonstração em um exercício de integração, uma vez que ela estabelece que

$$\int_R \mathrm{rc}(p, \alpha)p \; \mathrm{d}p = \frac{1}{3}\int_0^1 \big(\gamma(t) \times \gamma'(t)\big)\gamma(t) \; \mathrm{d}t =$$
$$\frac{1}{3}\int_0^1 \Big(\big(c + r(\cos\theta_t, \sin\theta_t)\big) \times \big(r(\theta_1 - \theta_0)(-\sin\theta_t, \cos\theta_t)\big)\Big)\big(c + r(\cos\theta_t, \sin\theta_t)\big) \; \mathrm{d}t.$$

Na prova da proposição 13, havíamos calculado que, para todo $t \in [0,1]$,

$$\begin{aligned}
\big(c + r(\cos\theta_t, \sin\theta_t)\big) \times \big(r(\theta_1-\theta_0)(-\sin\theta_t, \cos\theta_t)\big) &= \\
(x + r\cos\theta_t, y + r\sin\theta_t) \times \big(-r(\theta_1-\theta_0)\sin\theta_t, r(\theta_1-\theta_0)\cos\theta_t\big) &= \\
(\theta_1-\theta_0)(xr\cos\theta_t + r^2\cos^2\theta_t + yr\sin\theta_t + r^2\sin^2\theta_t) &= \\
(\theta_1-\theta_0)\big(xr\cos\theta_t + yr\sin\theta_t + r^2(\cos^2\theta_t + \sin^2\theta_t)\big) &= \\
(\theta_1-\theta_0)(xr\cos\theta_t + yr\sin\theta_t + r^2).
\end{aligned}$$

Voltando então ao que fazíamos, temos que

$$\begin{aligned}
\int_R \mathrm{rc}(p,\alpha)p \; \mathrm{d}p &= \frac{1}{3}\int_0^1 (\theta_1-\theta_0)(xr\cos\theta_t + yr\sin\theta_t + r^2)\big(c + r(\cos\theta_t, \sin\theta_t)\big) \; \mathrm{d}t \\
&= \frac{1}{3}(\theta_1-\theta_0)\int_0^1 (xr\cos\theta_t + yr\sin\theta_t + r^2)(x + r\cos\theta_t, y + r\sin\theta_t) \; \mathrm{d}t.
\end{aligned}$$

Temos então que a abscissa e a ordenada de

$$\int_R \mathrm{rc}(p,\alpha)p \; \mathrm{d}p$$

são, respectivamente,

$$\frac{1}{3}(\theta_1-\theta_0)\int_0^1 (xr\cos\theta_t + yr\sin\theta_t + r^2)(x + r\cos\theta_t) \; \mathrm{d}t$$

e

$$\frac{1}{3}(\theta_1-\theta_0)\int_0^1 (xr\cos\theta_t + yr\sin\theta_t + r^2)(y + r\sin\theta_t) \; \mathrm{d}t.$$

Antes de propriamente calcularmos estas integrais, lembramos o leitor das seguintes identidades em integração trigonométrica:

$$\int \cos^2 t \; \mathrm{d}t = \frac{1}{2}t + \frac{1}{4}\sin 2t;$$

$$\int \sin^2 t \; \mathrm{d}t = \frac{1}{2}t - \frac{1}{4}\sin 2t; \text{ e}$$

$$\int \cos t \sin t \; \mathrm{d}t = \frac{1}{2}\sin^2 t.$$

Assim, ao aplicarmos o método de integração por substituição para a função $\theta$, temos

$$\int \cos^2\theta_t \; \mathrm{d}t = \frac{\frac{1}{2}\theta_t + \frac{1}{4}\sin 2\theta_t}{\theta_1-\theta_0};$$

$$\int \sin^2\theta_t \; \mathrm{d}t = \frac{\frac{1}{2}\theta_t - \frac{1}{4}\sin 2\theta_t}{\theta_1-\theta_0}; \text{ e}$$

$$\int \cos\theta_t \sin\theta_t \; \mathrm{d}t = \frac{\sin^2\theta_t}{2(\theta_1-\theta_0)}.$$

Agora sim, temos que a abscissa de

$$\int_R \mathrm{rc}(p, \alpha) p \ \mathrm{d}p$$

é

$$\begin{aligned}
\frac{1}{3}(\theta_1 - \theta_0) \int_0^1 (xr \cos\theta_t + yr \sin\theta_t + r^2)(x + r\cos\theta_t) \ \mathrm{d}t &= \\
\frac{1}{3}(\theta_1 - \theta_0) \int_0^1 (x^2 r \cos\theta_t + xyr \sin\theta_t + xr^2 + xr^2 \cos^2\theta_t + yr^2 \cos\theta_t \sin\theta_t + r^3 \cos\theta_t) \ \mathrm{d}t &= \\
\frac{1}{3}(\theta_1 - \theta_0) \int_0^1 \big(xr^2 + (x^2 r + r^3)\cos\theta_t + xyr \sin\theta_t + yr^2 \cos\theta_t \sin\theta_t + xr^2 \cos^2\theta_t\big) \ \mathrm{d}t &= \\
\frac{1}{3}(\theta_1 - \theta_0) \left( xr^2 t + \frac{(x^2 r + r^3)\sin\theta_t}{(\theta_1 - \theta_0)} - \frac{xyr\cos\theta_t}{(\theta_1 - \theta_0)} + \frac{yr^2 \sin^2\theta_t}{2(\theta_1 - \theta_0)} + \frac{xr^2(\frac{1}{2}\theta_t + \frac{1}{4}\sin 2\theta_t)}{(\theta_1 - \theta_0)} \right) \Bigg|_{t=0}^{1} &= \\
\frac{1}{3}\left( xr^2(\theta_1 - \theta_0)t + (x^2 r + r^3)\sin\theta_t - xyr\cos\theta_t + \frac{yr^2}{2}\sin^2\theta_t + \frac{xr^2}{2}\theta_t + \frac{xr^2}{4}\sin 2\theta_t \right) \Bigg|_{t=0}^{1} &= \\
\frac{1}{3}xr^2(\theta_1 - \theta_0) + \frac{1}{3}(x^2 r + r^3)(\sin\theta_1 - \sin\theta_0) - \frac{1}{3}xyr(\cos\theta_1 - \cos\theta_0) + & \\
\frac{1}{6}yr^2(\sin^2\theta_1 - \sin^2\theta_0) + \frac{1}{6}xr^2(\theta_1 - \theta_0) + \frac{1}{12}xr^2(\sin 2\theta_1 - \sin 2\theta_0) &= \\
\frac{1}{2}xr^2(\theta_1 - \theta_0) + \frac{1}{3}(x^2 r + r^3)(\sin\theta_1 - \sin\theta_0) - \frac{1}{3}xyr(\cos\theta_1 - \cos\theta_0) + & \\
\frac{1}{6}yr^2(\sin^2\theta_1 - \sin^2\theta_0) + \frac{1}{12}xr^2(\sin 2\theta_1 - \sin 2\theta_0), &
\end{aligned}$$

como desejado.

Os cálculos para se obter a ordenada de

$$\int_R \mathrm{rc}(p, \alpha) p \ \mathrm{d}p$$

são análogos e deixados como exercício. Ainda, para completar a demonstração, observe que $\mathrm{rc}(p, \mathrm{rev}\,\alpha) = -\,\mathrm{rc}(p, \alpha)$ para todo ponto $p \in \mathbb{R}^2$. □

Observamos que a relação estabelecida pela proposição 16 também é válida para segmentos orientados.

**Proposição 18** — Seja $\overrightarrow{uv}$ um segmento de reta orientado, seja $R \subseteq \mathbb{R}^2$ um retângulo aberto contendo todos os pontos em $uv$ e a origem, seja $\gamma : [0, 1] \to \mathbb{R}^2$ a parametrização canônica $\widetilde{uv}$ de $\overrightarrow{uv}$ e seja $\gamma' : [0, 1] \to \mathbb{R}^2$ a função derivada de $\gamma$. Então

$$\int_R \mathrm{rc}(p, \overrightarrow{uv}) p \ \mathrm{d}p = \frac{1}{6}(u \times v)(u + v) = \frac{1}{3} \int_0^1 \big(\gamma(t) \times \gamma'(t)\big)\gamma(t) \ \mathrm{d}t.$$

**Prova** — Trata-se também de uma simples integração. □

Finalmente, apresentamos o último resultado restante.

**Teorema 4** — Seja $X \subseteq \mathbb{R}^2$ um conjunto J-mensurável e seja $\mathcal{R}$ uma boa representação de $X$. Para cada elemento $\alpha \in \mathcal{R}$, denote por $\widetilde{\alpha}'$ a derivada da parametrização canônica $\widetilde{\alpha}$ de $\alpha$. Então

$$\operatorname{centroid} X = \frac{1}{3(\operatorname{area} X)} \sum_{\alpha \in \mathcal{R}} \int_0^1 \big(\widetilde{\alpha}(t) \times \widetilde{\alpha}'(t)\big)\widetilde{\alpha}(t)\ \mathrm{d}t.$$

**Prova** — Também segue imediatamente de tudo o que já discutimos. $\square$

# Capítulo 3

# União de círculos

## 3.1 I-coordenadas

Na seção 1.5, convencionamos escolher um subcorpo real $\mathbb{F}$ a fim de escrevermos nossos algoritmos da forma mais geral possível. Porém, nossos algoritmos para resolver o problema da união de círculos irão precisar manipular outros números reais além dos números em $\mathbb{F}$, mesmo quando $\mathbb{F} = \mathbb{Q}$. Além disso, não podemos nem sequer enunciar o problema da união de círculos sem antes tratar deles. Por causa disso, esta seção é dedicada a completar a parte numérica deste trabalho através da introdução e do estudo destes números.

A origem deste problema vem do fato de que em nossos algoritmos será possivelmente necessário representar pontos na intersecção das fronteiras de dois círculos distintos na entrada. Por exemplo, se $\mathbb{F} = \mathbb{Q}$ e a entrada possuir dois círculos $C$ e $D$ de raio 1 cujos centros são, respectivamente, $(0, 0)$ e $(1, 0)$, então o ponto $(1/2, \sqrt{3}/2)$ estará na intersecção $\partial C \cap \partial D$ e poderá ter que ser representado computacionalmente. No entanto, o número $\sqrt{3}/2$ não é um número racional.

A técnica que usaremos para resolver este problema esteve presente em nosso trabalho desde a primeira implementação que fizemos de um algoritmo para computar a fronteira da união de círculos. Naquela implementação, chamamos de I-coordenadas os números reais que podem ser coordenadas da intersecção de duas circunferências distintas com centros de coordenadas racionais e raios racionais. Aqui iremos generalizar este conceito levando em conta o subcorpo real $\mathbb{F}$, começando com a definição precisa do conjunto das I-coordenadas.

**Definição 52** — O ***conjunto das I-coordenadas*** é o conjunto

$$\mathbb{I} = \left\{a + b\sqrt{c} : a, b, c \in \mathbb{F} \wedge c \geq 0\right\}.$$

Este conjunto de números já foi considerado anteriormente na literatura por Berberich et al [8], que denominaram seus elementos no caso em que $\mathbb{F} = \mathbb{Q}$ por *one-root numbers*.

Outros trabalhos descrevem conjuntos mais gerais, como por exemplo [15]. A técnica de representar números algébricos computacionalmente, tal como implementada na biblioteca LEDA [3], é mais geral do que todas estas técnicas. Porém, o preço de generalidade é, em geral, uma perda de performance (mesmo que por um fator constante). Portanto, ficamos com a técnica mais específica que atende nossas demandas.

Na seção 1.5, mostramos como elementos em $\mathbb{F}$ são representados computacionalmente através de rotinas que os manipulam. Similarmente, mostramos agora como representamos e manipulamos I-coordenadas computacionalmente.

**Definição 53** — Uma I-coordenada $a + b\sqrt{c}$ com $a, b, c \in \mathbb{F}$ e $c \geq 0$ poderá ser computacionalmente representada pela tripla $(a, b, c) \in \mathbb{F}^3$.

Note, no entanto, que pode haver várias representações computacionais para uma mesma I-coordenada.

Talvez a primeira observação relevante que devemos fazer sobre este conjunto é que ele não possui nenhuma estrutura algébrica imediatamente aparente. Em particular a soma ou subtração de duas I-coordenadas pode não ser uma I-coordenada e o produto de duas I-coordenadas pode não ser uma I-coordenada. Porém, é possível realizar algumas operações com I-coordenadas e, na verdade, essas operações são suficientes para nossos propósitos. Inclusive, chega a ser impressionante como nossos algoritmos parecem requisitar precisamente o que as I-coordenadas podem nos oferecer.

A seguir mostramos quais operações úteis a nós podem ser feitas sobre I-coordenadas.

**Proposição 19** — Se $i \in \mathbb{I}$ é uma I-coordenada e $\alpha \in \mathbb{F}$ é um escalar real, então o número real $\alpha i$ é também uma I-coordenada, ou seja, $\alpha i \in \mathbb{I}$.
Ainda, existe um algoritmo que recebe como entrada uma I-coordenada $i$ e um elemento $\alpha$ de $\mathbb{F}$ e devolve a I-coordenada $\alpha i$ realizando no máximo um número constante de operações do corpo $\mathbb{F}$.

**Prova** — Elementar. □

Observe que as I-coordenadas na entrada e na saída são representadas como descrito anteriormente e, como podem existir várias representações computacionais possíveis para uma mesma I-coordenada, o algoritmo mencionado na proposição 19 está autorizado a produzir saídas diferentes para as mesmas entradas. Pedimos ao leitor que note que esta observação será válida para a maioria de nossos algoritmos que produzem I-coordenadas em suas saídas.

**Proposição 20** — Se $i \in \mathbb{I}$ é uma I-coordenada e $n \in \mathbb{N}$ é um número natural, então $i^n \in \mathbb{I}$.
Ainda, existe um algoritmo que recebe como entrada uma I-coordenada $i \in \mathbb{I}$ e um número natural $n \in \mathbb{N}$ e devolve a I-coordenada $i^n$ usando, para cada valor fixo de $n$, no máximo um número constante de operações do corpo $\mathbb{F}$.

**Prova** — Se $i = a + b\sqrt{c}$, onde $a, b, c \in \mathbb{F}$ e $c \geq 0$, então

$$\begin{aligned} i^n &= \left(a + b\sqrt{c}\right)^n \\ &= \sum_{i=0}^{n} \binom{n}{i} a^{n-i} \left(b\sqrt{c}\right)^i \\ &= \sum_{i=0}^{\lfloor n/2 \rfloor} \binom{n}{2i} a^{n-2i} \left(b\sqrt{c}\right)^{2i} + \sum_{i=0}^{\lfloor (n-1)/2 \rfloor} \binom{n}{2i+1} a^{n-2i-1} \left(b\sqrt{c}\right)^{2i+1} \\ &= \sum_{i=0}^{\lfloor n/2 \rfloor} \binom{n}{2i} a^{n-2i} b^{2i} c^i + \left( \sum_{i=0}^{\lfloor (n-1)/2 \rfloor} \binom{n}{2i+1} a^{n-2i-1} b^{2i+1} c^i \right) \sqrt{c}, \end{aligned}$$

o que não só demonstra que $i^n \in \mathbb{I}$ mas também dá um algoritmo para computar esta potência. □

**Proposição 21** — Existe um algoritmo que recebe como entrada uma I-coordenada $i \in \mathbb{I}$ e um número real $x \in \mathbb{F}$ e decide se $i < x$ fazendo uso de no máximo um número constante de operações do corpo $\mathbb{F}$.

**Prova** — Seja $i = a + b\sqrt{c}$ com $a, b, c \in \mathbb{F}$ e $c \geq 0$. Se $b = 0$, então claramente $i < x \iff a < x$ e a demonstração está concluída. Se $b > 0$, então

$$\begin{aligned} i < x &\iff a + b\sqrt{c} < x \\ &\iff b\sqrt{c} < x - a \\ &\iff \sqrt{c} < \frac{x-a}{b} \\ &\iff 0 < \frac{x-a}{b} \quad \wedge \quad c < \left(\frac{x-a}{b}\right)^2 \\ &\iff a < x \quad \wedge \quad b^2 c < (x-a)^2. \end{aligned}$$

Já se $b < 0$, temos que

$$\begin{aligned} i < x &\iff a + b\sqrt{c} < x \\ &\iff b\sqrt{c} < x - a \\ &\iff \sqrt{c} > \frac{x-a}{b} \\ &\iff \frac{x-a}{b} < 0 \quad \vee \quad c > \left(\frac{x-a}{b}\right)^2 \\ &\iff a < x \quad \vee \quad b^2 c > (x-a)^2, \end{aligned}$$

o que resulta no desejado algoritmo. □

**Proposição 22** — Existe um algoritmo que recebe como entrada duas I-coordenadas $i, j \in \mathbb{I}$ e decide se $i < j$ fazendo uso de no máximo um número constante de operações do corpo $\mathbb{F}$.

**Prova** — Sejam $i = a_0 + b_0\sqrt{c_0}$ e $j = a_1 + b_1\sqrt{c_1}$ com $a_0, a_1, b_0, b_1, c_0, c_1 \in \mathbb{F}$, $c_0 \geq 0$

e $c_1 \geq 0$. Se $b_0 = 0$, então $i < j \iff a_0 < j \iff -j < -a_0$. Já se $b_0 > 0$, então

$$
\begin{aligned}
i < j \quad &\iff \quad a_0 + b_0\sqrt{c_0} < a_1 + b_1\sqrt{c_1} \\
&\iff \quad b_0\sqrt{c_0} < a_1 - a_0 + b_1\sqrt{c_1} \\
&\iff \quad \sqrt{c_0} < \frac{a_1 - a_0}{b_0} + \frac{b_1}{b_0}\sqrt{c_1} \\
&\iff \quad \frac{a_1 - a_0}{b_0} + \frac{b_1}{b_0}\sqrt{c_1} > 0 \quad \wedge \quad c_0 < \Big(\frac{a_1 - a_0}{b_0} + \frac{b_1}{b_0}\sqrt{c_1}\Big)^2 \\
&\iff \quad -(a_1 - a_0) - b_1\sqrt{c_1} < 0 \quad \wedge \quad b_0^2 c_0 < \big(a_1 - a_0 + b_1\sqrt{c_1}\big)^2.
\end{aligned}
$$

Finalmente, se $b_0 < 0$, então

$$
\begin{aligned}
i < j \quad &\iff \quad a_0 + b_0\sqrt{c_0} < a_1 + b_1\sqrt{c_1} \\
&\iff \quad b_0\sqrt{c_0} < a_1 - a_0 + b_1\sqrt{c_1} \\
&\iff \quad \sqrt{c_0} > \frac{a_1 - a_0}{b_0} + \frac{b_1}{b_0}\sqrt{c_1} \\
&\iff \quad \frac{a_1 - a_0}{b_0} + \frac{b_1}{b_0}\sqrt{c_1} < 0 \quad \vee \quad c_0 > \Big(\frac{a_1 - a_0}{b_0} + \frac{b_1}{b_0}\sqrt{c_1}\Big)^2 \\
&\iff \quad -(a_1 - a_0) - b_1\sqrt{c_1} < 0 \quad \vee \quad b_0^2 c_0 > \big(a_1 - a_0 + b_1\sqrt{c_1}\big)^2.
\end{aligned}
$$

Em todos estes casos, a proposição 21 e a proposição 20 são suficientes para o projeto do algoritmo desejado. □

**Proposição 23** — Sejam

$$p = \big(a_{00} + b_{00}\sqrt{c}, a_{01} + b_{01}\sqrt{c}\big) \in \mathbb{I}^2$$

e

$$q = \big(a_{10} + b_{10}\sqrt{c}, a_{11} + b_{11}\sqrt{c}\big) \in \mathbb{I}^2$$

dois pontos no plano cujas coordenadas são I-coordenadas que podem ser expressas com um mesmo radical $c$. Então $p \times q$, o produto cruzado de $p$ com $q$, é um elemento de $\mathbb{I}$ que pode ser expresso na forma $a + b\sqrt{c}$ com o mesmo radical $c$. Ainda, existe um algoritmo que recebe como entrada representações $\big((a_{00}, b_{00}, c), (a_{01}, b_{01}, c)\big)$ e $\big((a_{10}, b_{10}, c), (a_{11}, b_{11}, c)\big)$ de dois pontos $p, q \in \mathbb{I}$ e devolve uma representação $(a, b, c)$ para a I-coordenada $p \times q$ fazendo uso de no máximo um número constante de operações do corpo $\mathbb{F}$.

**Prova** — Uma simples manipulação algébrica. □

**Proposição 24** — Seja $C$ um círculo cujas coordenadas do centro e cujo raio são elementos de $\mathbb{F}$ e seja $\ell = \{u + tv : t \in \mathbb{R}\}$ uma reta definida por dois vetores $u \in \mathbb{F}^2$ e $v \in \mathbb{F}^2 \setminus \{0\}$ cujas coordenadas são também elementos de $\mathbb{F}$. Então $\partial C \cap \ell \subseteq \big\{a + b\sqrt{c} : a, b \in \mathbb{F}\big\}^2 \subseteq \mathbb{I}^2$ para algum $c \in \mathbb{F}$, $c \geq 0$.
Ainda, existe um algoritmo que recebe como entrada três elementos de $\mathbb{F}$ representando as coordenadas do centro e o raio de um círculo $C$ e dois vetores $u, v \in \mathbb{F}^2$ com $v \neq 0$ e devolve o conjunto (finito) $\partial C \cap \ell$ através de uma representação em que todas as coordenadas dos pontos no conjunto são I-coordenadas representadas com um mesmo radical fazendo uso de no máximo um número constante de operações do corpo $\mathbb{F}$, sendo $\ell = \{u + tv : t \in \mathbb{R}\}$ uma reta.

**Prova** — Sejam $c$ e $r$ o centro e o raio, respectivamente, do círculo $C$. Considere então a função $\gamma$ dada por

$$\begin{array}{rccc} \gamma : & \mathbb{R} & \to & \mathbb{R}^2 \\ & t & \mapsto & u + tv \end{array}$$

e observe que $\ell = \mathrm{Im}(\gamma)$. Temos então que

$$\begin{aligned} \gamma(t) \in \partial C \quad & \Longleftrightarrow \quad \big(\gamma(t) - c\big)^2 = r^2 \\ & \Longleftrightarrow \quad (u + tv - c)^2 = r^2 \\ & \Longleftrightarrow \quad \big(tv + (u - c)\big)^2 = r^2 \\ & \Longleftrightarrow \quad ||v||^2 t^2 + 2\big(v \cdot (u - c)\big)t + \big(||u - c||^2 - r^2\big) = 0, \end{aligned}$$

que é uma equação de segundo grau em $t$. Assim, se o número

$$\Delta = 4\big(v \cdot (u - c)\big)^2 - 4||v||^2\big(||u - c||^2 - r^2\big) \in \mathbb{F}$$

for negativo, não há intersecções entre $\ell$ e $C$. Já se $\Delta \geq 0$, há uma ou duas raízes desta equação correspondendo a um ou dois pontos na intersecção de $\ell$ com $C$. Estas raízes são dadas por

$$\frac{-2\big(v \cdot (u - c)\big) + \sqrt{\Delta}}{2||v||^2} \quad \text{e} \quad \frac{-2\big(v \cdot (u - c)\big) - \sqrt{\Delta}}{2||v||^2},$$

de forma que há duas raízes distintas se, e somente se, $\Delta \neq 0$. Logo, as intersecções são

$$u + \left(\frac{-2\big(v \cdot (u - c)\big) + \sqrt{\Delta}}{2||v||^2}\right)v \quad \text{e} \quad u + \left(\frac{-2\big(v \cdot (u - c)\big) - \sqrt{\Delta}}{2||v||^2}\right)v,$$

que podem ser escritas como

$$\left(u - \frac{\big(v \cdot (u - c)\big)}{||v||^2}v\right) + \left(\frac{\sqrt{\Delta}}{2||v||^2}\right)v \quad \text{e} \quad \left(u - \frac{\big(v \cdot (u - c)\big)}{||v||^2}v\right) - \left(\frac{\sqrt{\Delta}}{2||v||^2}\right)v,$$

que claramente são vetores com coordenadas em $\mathbb{I}$, todas representáveis com um mesmo radical $\sqrt{\Delta}$. $\square$

**Proposição 25** — Sejam $C_0$ e $C_1$ dois círculos distintos cujas coordenadas dos centros e cujos raios são todos elementos de $\mathbb{F}$. Então $\partial C_0 \cap \partial C_1 \subseteq \{a + b\sqrt{c} : a, b \in \mathbb{F}\}^2 \subseteq \mathbb{I}^2$ para algum $c \in \mathbb{F}$, $c \geq 0$.
Ainda, existe um algoritmo que recebe como entrada seis elementos de $\mathbb{F}$ representando as coordenadas do centro e os raios de dois círculos $C_0$ e $C_1$ distintos e devolve o conjunto (finito) $\partial C_0 \cap \partial C_1$ através de uma representação em que todas as coordenadas dos pontos no conjunto são I-coordenadas representadas com um mesmo radical fazendo uso de no máximo um número constante de operações do corpo $\mathbb{F}$.

**Prova** — Como os círculos $C_0$ e $C_1$ são distintos, podemos assumir, sem perda de generalidade, que seus centros são distintos, pois do contrário seus raios não podem ser iguais e logo a intersecção de suas fronteiras deve ser vazia.

Sejam então $c_0, c_1 \in \mathbb{R}^2$ com $c_0 \neq c_1$ os centros dos círculos $C_0$ e $C_1$, respectivamente. Sejam também $r_0, r_1 \in \mathbb{R}$ os respectivos raios de $C_0$ e $C_1$. Considere então a reta $\ell \subseteq \mathbb{R}^2$ definida por

$$\ell = \{p \in \mathbb{R}^2 : 2(c_1 - c_0) \cdot p = ||c_1||^2 - ||c_0||^2 + r_0^2 - r_1^2\}.$$

Se um ponto $p \in \mathbb{R}^2$ está na intersecção $\partial C_0 \cap \partial C_1$ das fronteiras dos círculos $C_0$ e $C_1$, então vale que

$$\begin{aligned} ||p - c_0||^2 = r_0^2 \quad &\iff \quad ||p||^2 - 2p \cdot c_0 + ||c_0||^2 = r_0^2 \\ &\iff \quad 2c_0 \cdot p = ||p||^2 + ||c_0||^2 - r_0^2 \end{aligned}$$

e

$$\begin{aligned} ||p - c_1||^2 = r_1^2 \quad &\iff \quad ||p||^2 - 2p \cdot c_1 + ||c_1||^2 = r_1^2 \\ &\iff \quad 2c_1 \cdot p = ||p||^2 + ||c_1||^2 - r_1^2. \end{aligned}$$

Subtraindo estas duas últimas equações, temos que

$$2(c_1 - c_0) \cdot p = ||c_1||^2 - ||c_0||^2 + r_0^2 - r_1^2,$$

ou seja, que $p \in \ell$. Portanto, $\partial C_0 \cap \partial C_1 \subseteq \ell$.

Similarmente, pode-se mostrar que $\partial C_0 \cap \ell \subseteq \partial C_1$ e que $\partial C_1 \cap \ell \subseteq \partial C_0$. Assim, para obter os pontos na intersecção (finita) $\partial C_0 \cap \partial C_1$, basta intersectar a reta $\ell$ e a fronteira do círculo $C_0$ (ou $C_1$).

Sejam então $u \in \mathbb{R}^2$ um vetor dado por

$$u = \frac{||c_1||^2 - ||c_0||^2 + r_0^2 - r_1^2}{2||c_1 - c_0||^2}(c_1 - c_0)$$

e $v \in \mathbb{R}^2 \setminus \{0\}$ um vetor não-nulo qualquer paralelo ao vetor $c_1 - c_0$. É simples então perceber que $u \in \ell$ e que $u + tv \in \ell$ para todo $t \in \mathbb{R}$. Logo, podemos concluir, como $v \neq 0$, que $\ell = \{u + tv : t \in \mathbb{R}\}$.

Se as coordenadas do centro e os raios de $C_0$ e $C_1$ são todos elementos de $\mathbb{F}$, segue claramente que $u, v \in \mathbb{F}^2$. Assim, podemos usar o algoritmo obtido pela proposição 24 para computar estas intersecções. Mais ainda, por causa desta proposição, sabemos que todas as coordenadas dos pontos devolvidos são representadas como I-coordenadas partilhando um mesmo radical. □

**Proposição 26** — Existe um algoritmo que recebe como entrada três elementos de $\mathbb{F}$ representando as coordenadas do centro e o raio de um círculo $C$ e um ponto $p \in \mathbb{I}^2$ cujas coordenadas são I-coordenadas representadas com um mesmo radical e, fazendo uso de no máximo um número constante de operações do corpo $\mathbb{F}$, decide se $p$ é um elemento do interior de $C$, da fronteira de $C$ ou se $p$ não pertence a $C$.

**Prova** — Trata-se de um simples exercício de geometria plana. □

**Proposição 27** — Existe um algoritmo que recebe como entrada:

- Seis elementos de $\mathbb{F}$ representando as coordenadas do centro e os raios de dois círculos $C$ e $D$;
- Um ponto $u \in \mathbb{I}^2$ com coordenadas representadas usando o mesmo radical; e
- Um ponto $v \in \mathbb{I}^2$ com coordenadas representadas usando o mesmo radical, possivelmente distinto do radical das coordenadas de $u$,

satisfazendo $u, v \in \partial D$ e $\big(\alpha \setminus \{u, v\}\big) \cap \partial C = \{\}$, onde $\alpha = \operatorname{arc}(D, u, v)$, e decide se $(\alpha \setminus \{u, v\}) \subseteq C^\circ$ ou se $(\alpha \setminus \{u, v\}) \subseteq \mathbb{R}^2 \setminus C$ fazendo uso de um número no máximo constante de operações do subcorpo real $\mathbb{F}$.

**Prova** — Primeiramente, observe que como $C$ é um conjunto fechado e $\alpha$ é um conjunto conexo, de fato exatamente uma das alternativas $(\alpha \setminus \{u, v\}) \subseteq C^\circ$ ou $(\alpha \setminus \{u, v\}) \subseteq \mathbb{R}^2 \setminus C$ ocorre.

Note agora que, se $u \in C^\circ$ ou $v \in C^\circ$, então, pelo teorema do valor intermediário (teorema 1) e como $\big(\alpha \setminus \{u, v\}\big) \cap \partial C = \{\}$, devemos ter $(\alpha \setminus \{u, v\}) \subseteq C^\circ$. Similarmente, se $u \notin C$ ou se $v \notin C$, devemos ter $(\alpha \setminus \{u, v\}) \subseteq \mathbb{R}^2 \setminus C$. Portanto, podemos assumir, sem perda de generalidade, que $u, v \in \partial C$.

Seja agora $\gamma : [0, 1] \to \mathbb{R}^2$ a parametrização canônica de $\alpha$, de modo que $\gamma(0) = u$ e $\gamma(1) = v$, e seja também $\delta : \mathbb{R} \to \mathbb{R}^2$ a extensão periódica da parametrização canônica de $C$, de forma que existem valores reais $t_0, t_1 \in \mathbb{R}$ com $0 \leq t_0 < 1$ e $t_0 < t_1 < t_0 + 1$ tais que $\delta(t_0) = u$ e $\delta(t_1) = v$. Denote também as derivadas de $\gamma$ e $\delta$, respectivamente, por $\gamma'$ e $\delta'$.

Seja então $c = \gamma'(0) \times \delta'(t_0)$. Se $c = 0$, temos que $\alpha$ e $C$ são tangentes em $u$. Mas então $D$ e $C$ têm apenas uma intersecção em comum, contradizendo $u, v \in \partial C$. Já se $c > 0$ então, em uma vizinhança de $u$, o arco $\alpha$ deve estar fora de $C$. Mais precisamente, existe um $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ tal que, para todo $t \in \mathbb{R}$ com $0 < t < \varepsilon$, $\gamma(t) \notin C$. Deste modo somos forçados a concluir que $(\alpha \setminus \{u, v\}) \subseteq \mathbb{R}^2 \setminus C$. Similarmente, se $c < 0$, em uma vizinhança de $u$, o arco $\alpha$ deve estar em $C^\circ$, ou, mais precisamente, existe um $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ tal que, para todo $t \in \mathbb{R}$ com $0 < t < \varepsilon$, $\gamma(t) \in C^\circ$, de modo que $(\alpha \setminus \{u, v\}) \subseteq C^\circ$.

Resta então mostrar como computar $c = \gamma'(0) \times \delta'(t_0)$. Para isso, seja

$$\begin{array}{rccc} \rho_{\frac{\pi}{2}} : & \mathbb{R}^2 & \to & \mathbb{R}^2 \\ & (x, y) & \mapsto & (-y, x) \end{array}$$

a transformação de rotação do plano por 90 graus em sentido anti-horário. Como a derivada de uma curva em um ponto é o sentido de uma reta tangente à curva naquele ponto e como retas tangentes a círculos formam ângulos retos entre o ponto de tangência e o centro do círculo, então, sendo $A \in \mathbb{R}^2$ o centro de $C$ e $B \in \mathbb{R}^2$ o centro de $D$, temos que $\delta'(t_0) = \rho_{\frac{\pi}{2}}(u - A)$ e $\gamma'(0) = \rho_{\frac{\pi}{2}}(u - B)$, pois as parametrizações $\gamma$ e $\delta$ são em sentido anti-horário. Na verdade, tecnicamente, com o que dissemos, só podemos deduzir que $\gamma'(0)$ e $\delta'(t_0)$ são proporcionais a estes vetores. Porém, o cálculo efetivo destas derivadas, o qual deixamos como exercício ao leitor, irá confirmar estes valores.

Note agora que $\delta'(t_0)$ e $\gamma'(0)$ são ambos elementos de $\mathbb{I}^2$ (pois $u \in \mathbb{I}^2$) e todas as quatro coordenadas de $\delta'(t_0)$ e $\gamma'(0)$ têm o mesmo radical (o radical comum às coordenadas de $u$). Logo, pela proposição 23, podemos de fato computar $c = \gamma'(0) \times \delta'(t_0)$ fazendo o uso de um número no máximo constante de operações do corpo $\mathbb{F}$. $\square$

## 3.2 Enunciado do problema

Com o estudo da união de círculos realizado na seção 3.3, podemos finalmente enunciar o problema da união de círculos.

**Problema 1** (união de círculos) —

- **Entrada:** Uma lista $C_0 C_1 \cdots C_{n-1}$ de círculos cujas coordenadas do centro e cujos raios são todos elementos de $\mathbb{F}$. Para todo $i \in \{0, 1, \ldots, n-1\}$, se $(x_i, y_i) \in \mathbb{F}^2$ é o centro de $C_i$ e $r_i \in \mathbb{F}$ é o raio de $C_i$, então o círculo $C_i$ será representado pela tripla $(x_i, y_i, r_i) \in \mathbb{F}^3$.
- **Saída:** Uma sequência finita de arcos de circunferência distintos cujas orientações em sentido anti-horário formam uma boa representação para $C_0 \cup C_1 \cup \cdots \cup C_{n-1}$. Todos os arcos desta sequência devem ser da forma $\operatorname{arc}(C_i, p, q)$ onde $p, q \in \partial C_i \cap \mathbb{I}^2$ e $i \in \{0, 1, \ldots, n-1\}$. Ainda, um tal arco será representado pela tripla $(i, p, q)$.

Resta ainda mostrar que o problema 1 sempre tem uma solução. Porém, nossa prova depende de conceitos que serão vistos apenas na seção 3.3, de forma que adiamos esta prova até o final desta seção.

## 3.3 Uniões de círculos

O objetivo desta seção é realizar um estudo de algumas propriedades da união de círculos. Uma vez estabelecidos estes resultados, a derivação de um algoritmo para unir círculos é bastante simples, bem como é a prova de que o problema 1 sempre admite uma solução, que será apresentada ao final desta seção.

Começamos introduzindo o conceito de pontos chave de um conjunto de círculos.

**Definição 54** — Seja $\mathcal{C} = \{C_0, C_1, \ldots, C_{n-1}\}$ um conjunto finito de círculos (dois a dois distintos). Dizemos então que os ***pontos chave*** de $\mathcal{C}$ são os pontos no conjunto

$$\mathcal{K}(\mathcal{C}) = \left( \bigcup_{i=0}^{n-1} \{(x_i - r, y_i), (x_i + r, y_i)\} \right) \cup \left( \bigcup_{i=0}^{n-2} \bigcup_{j=i+1}^{n-1} \partial C_i \cap \partial C_j \right),$$

onde $(x_i, y_i)$ é o centro de $C_i$ e $r_i$ é o raio de $C_i$ para todo $i \in \{0, 1, \ldots, n-1\}$.

Se $\mathcal{C}$ é um conjunto finito de círculos, então os pontos chave de $\mathcal{C}$ particionam a fronteira de cada círculo $C$ em $\mathcal{C}$ em arcos de circunferência, que chamaremos de arcos candidatos. Veremos, embora seja relativamente claro, que arcos candidatos distintos não podem se intersectar exceto possivelmente em seus extremos.

A razão de nomearmos estes arcos desta forma é que, sendo $U$ a união dos círculos em $\mathcal{C}$, iremos mostrar que cada arco candidato está ou inteiramente contido na fronteira de $U$ ou não intersecta a fronteira de $U$ exceto possivelmente em seus extremos. Ainda, demonstraremos não só que a fronteira de $U$ pode ser expressa como a união de alguns arcos candidatos, mas também que estes arcos com sua orientação original formam uma boa representação para o conjunto $U$.

Formalizamos agora nossa discussão.

**Definição 55** — Seja $\mathcal{C}$ um conjunto finito de círculos. Para cada círculo $C \in \mathcal{C}$, seja $\{p_0, p_1, \ldots, p_{k-1}\} = \mathcal{K}(\mathcal{C}) \cap \partial C$, com $\widetilde{C}^{-1}(p_i) < \widetilde{C}^{-1}(p_j)$ para todos $i, j \in \mathbb{N}$ com $i < j < k$. Pela definição de $\mathcal{K}(\mathcal{C})$, temos claramente que $k \geq 2$. Denotando $p_0$ por $p_k$, definimos os ***arcos candidatos*** de $C$ em $\mathcal{C}$ como os arcos de circunferência orientados $\overrightarrow{\mathrm{arc}}(C, p_i, p_{i+1})$ com $i \in \{0, 1, \ldots, k-1\}$. Ainda, definimos os ***arcos candidatos*** de $\mathcal{C}$ como os arcos candidatos dos círculos $C$ em $\mathcal{C}$.

**Proposição 28** — Cada arco candidato de um conjunto finito de círculos $\mathcal{C}$ é um arco candidato de apenas um círculo em $\mathcal{C}$.

**Prova** — Um arco candidato de um círculo deve estar contido em sua fronteira. Assim, se dois círculos possuem um mesmo arco candidato, eles devem ter um número infinito de pontos comuns em suas fronteiras, o que mostra que eles devem ser, na verdade, o mesmo círculo. □

**Proposição 29** — Seja $\mathcal{C}$ um conjunto finito de círculos e sejam $\alpha_0 = \overrightarrow{\mathrm{arc}}(C_0, p_0, q_0)$ e $\alpha_1 = \overrightarrow{\mathrm{arc}}(C_1, p_1, q_1)$ dois arcos candidatos distintos de $\mathcal{C}$. Então

$$\mathrm{arc}(C_0, p_0, q_0) \cap \mathrm{arc}(C_1, p_1, q_1) \subseteq \{p_0, q_0, p_1, q_1\}.$$

**Prova** — Qualquer outro ponto em $\mathrm{arc}(C_0, p_0, q_0) \cap \mathrm{arc}(C_1, p_1, q_1)$ seria um ponto chave de $\mathcal{C}$ e logo um dos arcos $\alpha_0$ ou $\alpha_1$ não seria um arco candidato de $\mathcal{C}$. □

**Definição 56** — Um ponto chave $p \in \mathcal{K}(\mathcal{C})$ de um conjunto finito de círculos $\mathcal{C}$ é dito ***interno*** se existe algum círculo em $\mathcal{C}$ que possui $p$ em seu interior.

Mostramos agora nossos principais resultados sobre arcos candidatos.

**Lema 3** — Seja $\alpha = \overrightarrow{\mathrm{arc}}(C,p,q)$ um arco candidato de um conjunto finito de círculos $\mathcal{C}$ (com $C \in \mathcal{C}$ e $p,q \in \mathcal{K}(\mathcal{C})$) e seja $D \in \mathcal{C}$ um círculo em $\mathcal{C}$ tal que existe ponto em $\mathrm{arc}(C,p,q) \setminus \{p,q\}$ que também está no interior de $D$. Então $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq D^\circ$.

**Prova** — Note que o arco candidato $\alpha$ não é um arco candidato do círculo $D$, pois caso contrário ele deveria estar contido em $\partial D$. Logo, se existisse algum ponto em $\mathrm{arc}(C,p,q) \setminus \{p,q\}$ e ao mesmo tempo na fronteira de $D$, este ponto seria um ponto chave, contradizendo $\alpha$ ser um arco candidato. Assim, o teorema do valor intermediário (teorema 1) nos garante que $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq D^\circ$. $\square$

**Lema 4** — Seja $\mathcal{C}$ um conjunto finito de círculos cuja união é $U \subseteq \mathbb{R}^2$ e seja $p \in U^\circ$. Então existe um círculo $C \in \mathcal{C}$ com $p \in C^\circ$ ou existem três círculos distintos em $\mathcal{C}$ contendo $p$ em suas fronteiras.

**Prova** — Suponha que $p \notin C^\circ$ para todo círculo $C$ em $\mathcal{C}$. Seja então $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ um valor real positivo tal que $B_\varepsilon(p)$ não contenha pontos chave em $\mathcal{K}(\mathcal{C})$ diferentes de $p$ e não intersecte nenhum círculo em $\mathcal{C}$ que não contenha $p$. Assim, como $p \in U^\circ$, a união dos círculos em $\mathcal{C}$ que contém $p$ deve conter $B_\varepsilon(p)$. Porém, um tal círculo $C \in \mathcal{C}$ contendo $p$ deve conter $p$ em sua fronteira, pois $p$ não reside no interior de nenhum círculo em $\mathcal{C}$. Logo, devemos ter que

$$\operatorname{area}\big(C \cap B_\varepsilon(p)\big) < \frac{1}{2}\big(\operatorname{area} B_\varepsilon(p)\big).$$

Portanto, é necessário que existam em $\mathcal{C}$ ao menos três círculos distintos contendo $p$ em suas fronteiras. $\square$

**Proposição 30** — Seja $\alpha = \overrightarrow{\mathrm{arc}}(C,p,q)$ um arco candidato de um conjunto finito de círculos $\mathcal{C}$ (com $p,q \in \mathcal{K}(\mathcal{C})$) e seja $U$ a união dos círculos em $\mathcal{C}$. Então se $\big(\mathrm{arc}(C,p,q) \setminus \{p,q\}\big) \cap U^\circ \neq \{\}$, existe um círculo $D \in \mathcal{C}$ tal que $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq D^\circ$.

**Prova** — De fato, tome um ponto $x \in \big(\mathrm{arc}(C,p,q) \setminus \{p,q\}\big) \cap U^\circ$. Se existe algum círculo $D \in \mathcal{C}$ tal que $x \in D^\circ$, então, pelo lema 3, $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq D^\circ$, como queríamos. Já se tal círculo $D$ não existe, o lema 4 nos garante que existem círculos $D_0, D_1 \in \mathcal{C}$ distintos tal que $x \in \partial D_0 \cap \partial D_1$, de forma que $x \in \mathcal{K}(\mathcal{C})$, contradizendo $\alpha$ ser um arco candidato. $\square$

**Corolário 6** — Seja $\alpha = \overrightarrow{\mathrm{arc}}(C,p,q)$ um arco candidato de um conjunto finito de círculos $\mathcal{C}$ (com $p,q \in \mathcal{K}(\mathcal{C})$) e seja $U$ a união dos círculos em $\mathcal{C}$. Então ou $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq U^\circ$ ou $\mathrm{arc}(C,p,q) \subseteq \partial U$. Ainda, $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq U^\circ$ se, e somente se, $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq D^\circ$ para algum círculo $D \in \mathcal{C}$.

**Prova** — Se $\big(\mathrm{arc}(C,p,q) \setminus \{p,q\}\big) \cap U^\circ \neq \{\}$, então, pela proposição 30, existe um círculo $D \in \mathcal{C}$ tal que $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq D^\circ \subseteq U^\circ$. Do contrário, como $\mathrm{arc}(C,p,q) \subseteq U$, devemos ter $\mathrm{arc}(C,p,q) \subseteq \partial U$.

Para mostrar o resto do resultado, observe que se $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq D^\circ$ para algum círculo $D \in \mathcal{C}$, então, como $D^\circ \subseteq U^\circ$, claramente $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq U^\circ$. Ainda, se $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq U^\circ$, então certamente

$$\big(\mathrm{arc}(C,p,q) \setminus \{p,q\}\big) \cap U^\circ \neq \{\},$$

de forma que, novamente pela proposição 30, existe um círculo $D \in \mathcal{C}$ tal que $\mathrm{arc}(C,p,q) \setminus \{p,q\} \subseteq D^\circ$. $\square$

**Proposição 31** — Seja $\mathcal{C}$ um conjunto finito de círculos cuja união é $U \subseteq \mathbb{R}^2$. Então o conjunto dos arcos candidatos de $\mathcal{C}$ que estão contidos em $\partial U$ (com suas respectivas orientações em sentido anti-horário) é uma boa representação do conjunto $U$.

**Prova** — Mostraremos os três fatos a seguir, que juntos formam uma demonstração do resultado:

- Todo arco candidato $\alpha = \overrightarrow{\mathrm{arc}}(C,p,q)$ tal que $\mathrm{arc}(C,p,q) \subseteq \partial U$ é compatível com $U$;
- Dois arcos candidatos $\alpha_0 = \overrightarrow{\mathrm{arc}}(C_0,p_0,q_0)$ e $\alpha_1 = \overrightarrow{\mathrm{arc}}(C_1,p_1,q_1)$ distintos tais que $\mathrm{arc}(C_0,p_0,q_0) \subseteq \partial U$ e $\mathrm{arc}(C_1,p_1,q_1) \subseteq \partial U$ devem sempre satisfazer

$$\mathrm{arc}(C_0,p_0,q_0) \cap \mathrm{arc}(C_0,p_0,q_0) \subseteq \{p_0,q_0,p_1,q_1\}; \text{ e}$$

- Todo ponto na fronteira de $U$ está em algum arco candidato contido na fronteira de $U$.

Para mostrar o primeiro fato, note que $\alpha$ é trivialmente compatível com $C$. Tome então um $x \in \mathrm{arc}(C,p,q) \setminus \{p,q\}$. Como $\alpha$ é um arco candidato, $x$ não é um ponto chave. Ainda, nenhum outro círculo em $\mathcal{C}$ pode conter $C$, de modo que existe um $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ tal que $B_\varepsilon(x)$ intersecta apenas o círculo $C$ dentre os círculos em $\mathcal{C}$. Assim, $B_\varepsilon(x) \cap U = B_\varepsilon(x) \cap C$, de forma que este valor de $\varepsilon$ e a escolha de $x$ fazem $\alpha$ satisfazer a definição de compatibilidade com $U$.

O segundo fato é uma consequência direta da proposição 29.

Já o terceiro fato pode ser demonstrado da seguinte forma. Seja $x \in \partial U \setminus \mathcal{K}(\mathcal{C})$ um ponto na fronteira de $U$ que não é um ponto chave. Como a fronteira da união de conjuntos deve sempre estar contida na união das fronteiras dos conjuntos, existe algum círculo $C$ tal que $x \in \partial C$. Como $x \notin \mathcal{K}(\mathcal{C})$, existe um arco candidato da forma $\overrightarrow{\mathrm{arc}}(C,p,q)$ com $p,q \in \partial C \cap \mathcal{K}(\mathcal{C})$ tal que $x \in \mathrm{arc}(C,p,q)$. Pelo corolário 6, temos então que $\mathrm{arc}(C,p,q) \subseteq \partial U$, pois $x \notin U^\circ$.

Por outro lado, se $x \in \mathcal{K}(\mathcal{C})$, seja $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ um valor real tal que $B_\varepsilon(x) \cap \mathcal{K}(\mathcal{C}) = \{x\}$ e tal que se um arco candidato intersecta $B_\varepsilon(x)$, então ele tem $x$ como um de seus extremos (podemos escolher tal $\varepsilon$ porque claramente $x$, sendo um ponto chave, só pode pertencer a arcos candidatos dos quais ele é um extremo). Usamos então o fato de que a fronteira $\partial U$ da união de círculos em $\mathcal{C}$ não possui pontos isolados. Logo, deve existir um ponto $y \in B_\varepsilon(x)$ com $y \neq x$ e $y \in \partial U$. Como a fronteira de uma união deve estar contida na união das fronteiras, deve existir um círculo $C \in \mathcal{C}$ com $y \in \partial C$. Sendo assim $y$ deve estar em algum arco candidato $\operatorname{arc}(C, p, q)$ com $p, q \in \mathcal{K}(\mathcal{C}) \cap \partial C$. Porém, como $y \in B_\varepsilon(x)$, pela nossa escolha de $\varepsilon$, $x = p$ ou $x = q$. Mais ainda, como $y \notin \mathcal{K}(\mathcal{C})$, $y \neq p$ e $y \neq q$, de forma que, como $y \notin U^\circ$, o corolário 6 mostra que $\operatorname{arc}(C, p, q) \subseteq \partial U$, como desejado. □

Provamos agora que o problema 1 sempre admite uma solução.

**Proposição 32** — O problema 1 sempre admite uma solução.

**Prova** — Seja $\mathcal{C}$ o conjunto dos círculos em uma entrada para o problema 1. A proposição 31 mostra que um subconjunto dos arcos candidatos de $\mathcal{C}$ constitui uma boa representação da união dos círculos em $\mathcal{C}$ quando orientados em sentido anti-horário. Ainda, cada arco candidato é da forma $\operatorname{arc}(C, p, q)$ com $C \in \mathcal{C}$ e $p, q \in \partial C \cap \mathbb{I}^2$ pela definição 55 e pela proposição 25. □

## 3.4 Um algoritmo ingênuo

O corolário 6 e a proposição 31 tornam imediato o desenvolvimento do algoritmo 2, que computa uma boa representação para a fronteira da união de círculos. Nesta seção iremos explicar vários detalhes da implementação deste algoritmo, mostrar sua correção e analisar sua complexidade.

As seguintes considerações são importantes ao implementar o algoritmo 2:

- Na linha 1 do algoritmo, devemos remover círculos repetidos da lista $L$ obtendo um conjunto de círculos. Como os círculos na entrada são representados como triplas de elementos de $\mathbb{F}$, como triplas diferentes dão origem a círculos diferentes e como temos uma rotina para comparar elementos de $\mathbb{F}$, podemos fazer isso com uma simples ordenação lexicográfica;

- Ainda na linha 1, por motivos que ficarão claros adiante, é necessário manter, para cada círculo $C_i$ no conjunto $\mathcal{C}$ (com $i \in \{0, 1, \ldots, n-1\}$), um índice $\phi_i$ de uma posição em que $C_i$ estava na lista $L$. Mais precisamente, se, no início do algoritmo, $L = C'_0 C'_1 \cdots C'_{m-1}$, então, para todo $i \in \{0, 1, \ldots, n-1\}$, devemos ter $C_i = C'_{\phi_i}$. Como obter estes índices é um exercício padrão de ordenação;

- Na linha 5 inicializamos conjuntos $\mathcal{K}_i$ para cada $i \in \{0, 1, \ldots, n-1\}$. Devemos representar estes conjuntos com estruturas de dados que permitam inserção em tempo constante (amortizado) e que consumam espaço linear, como por exemplo listas ligadas ou vetores dinâmicos;

- Na linha 15 pedimos que o algoritmo realize uma ordenação relativamente complicada do conjunto $\mathcal{K}_i$ para algum $i \in \{0, 1, \ldots, n-1\}$. Para fazer isso, separamos nossa lista representando $\mathcal{K}_i$ nas listas $A$ e $B$ da seguinte forma: para cada ponto $(x, y)$ em nossa lista representando $\mathcal{K}_i$, se $y > y_i$ ou se $y = y_i$ e $x > x_i$, então inserimos $(x, y)$ em $A$ e, do contrário, em $B$. A razão disso é que podemos ordenar os pontos em $A$ (removendo eventuais duplicatas) de acordo com a ordem induzida pela função $\widetilde{C}_i|_{[0,1)}^{-1}$ simplesmente ordenando-os em ordem (estritamente)

**Algorithm 2** Naïve-Union-of-Circles $(L)$

`Pré-condição:` $L$ é uma entrada para o problema 1

`Efeito:` devolve a saída para o problema 1 correspondente à instância $L$ como uma co-rotina

Remova as duplicatas da lista $L$ obtendo um conjunto de círculos

$$\mathcal{C} = \{C_0, C_1, \ldots, C_{n-1}\}.$$

**for** $i$ **from** $0$ **to** $n-1$ **do**
    Seja $(x_i, y_i)$ o centro de $C_i$.
    Seja $r_i$ o raio de $C_i$.
    $\mathcal{K}_i \leftarrow \big\{(x_i - r, y_i), (x_i + r, y_i)\big\}$
**done**
**for** $i$ **from** $1$ **to** $n-1$ **do**
    **for** $j$ **from** $0$ **to** $i-1$ **do**
        $K \leftarrow \partial C_i \cap \partial C_j$
        $\mathcal{K}_i \leftarrow \mathcal{K}_i \cup K$
        $\mathcal{K}_j \leftarrow \mathcal{K}_j \cup K$
    **done**
**done**
**for** $i$ **from** $0$ **to** $n-1$ **do**
    Ordene $\mathcal{K}_i$ em $p_0 p_1 \cdots p_{k-1}$ com $\widetilde{C}_i|_{[0,1)}^{-1}(p_0) < \widetilde{C}_i|_{[0,1)}^{-1}(p_1) < \cdots < \widetilde{C}_i|_{[0,1)}^{-1}(p_{k-1})$.
    $p_k \leftarrow p_0$
    **for** $a$ **from** $0$ **to** $k-1$ **do**
        internal $\leftarrow$ `false`
        **for** $j$ **from** $0$ **to** $n-1$ **do**
            **if** $\big(\operatorname{arc}(C_i, p_a, p_{a+1}) \setminus \{p_a, p_{a+1}\}\big) \subseteq C_j^\circ$ **then**
                internal $\leftarrow$ `true`
                `break`
            **endif**
        **done**
        **if not** internal **then**
            `yield` $\operatorname{arc}(C_i, p_a, p_{a+1})$
        **endif**
    **done**
**done**

decrescente de abscissa. Similarmente os pontos em $B$ podem ser ordenados em ordem (estritamente) crescente de abscissa. Ao final podemos concatenar a ordenação de $A$ com a ordenação de $B$ e produzir a ordenação desejada para $\mathcal{K}_i$;

- O teste feito na linha 20 pode ser implementado através do algoritmo descrito na proposição 27; e
- Na linha 26, devemos produzir um arco de circunferência, mais precisamente o arco $\mathrm{arc}(C_i, p_a, p_{a+1})$. A representação computacional deste arco envolve um índice do círculo $C_i$ na lista da entrada, e por isso previamente computamos $\phi_i$ para $i \in \{0, 1, \ldots, n-1\}$. Assim podemos produzir a representação computacional $(\phi_i, p_a, p_{a+1})$.

Com estes detalhes esclarecidos, passamos então a demonstrar este algoritmo correto.

**Teorema 5** — O algoritmo 2 corretamente resolve o problema 1.

**Prova** — É simples mostrar que ao término do laço da linha 7, para cada $i \in \{0, 1, \ldots, n-1\}$, o conjunto $\mathcal{K}_i$ será exatamente o conjunto $\mathcal{K}(\mathcal{C}) \cap \partial C_i$.

Portanto, por construção, a linha 18 será executada exatamente uma vez para cada arco candidato e, em cada uma destas vezes, $\overrightarrow{\mathrm{arc}}(C_i, p_a, p_{a+1})$ será um arco candidato distinto.

No laço da linha 19, procuramos um círculo $C_j$ (com $j \in \{0, 1, \ldots, n-1\}$) tal que $\big(\mathrm{arc}(C_i, p_a, p_{a+1}) \setminus \{p_a, p_{a+1}\}\big) \subseteq C_j^\circ$. Ao final deste laço, a variável internal será verdadeira se, e somente se, um tal círculo existir. Na verdade, pelo corolário 6, a variável internal será falsa se, e somente se, $\mathrm{arc}(C_i, p_a, p_{a+1}) \subseteq \partial U$, onde $U$ é a união dos círculos em $\mathcal{C}$.

Assim, a linha 26 irá produzir como saída exatamente os arcos candidatos contidos na fronteira da união, cada um exatamente uma vez.

Portanto, pela proposição 31, o algoritmo produz como saída uma boa representação do conjunto $U$. $\square$

Resta analisar a complexidade do algoritmo 2. Como foi discutido na seção 1.5, se pudermos calcular a complexidade algébrica de nosso algoritmo e mostrar que ele é algebricamente tratável, imediatamente teremos um limite superior para a real complexidade dele.

**Lema 5** — O algoritmo 2, quando executado em uma lista de $m$ círculos com exatamente $n$ círculos distintos, possui complexidades algébricas $\mathcal{O}(m \log m + n^3)$ de tempo e $\mathcal{O}(m + n^2)$ de espaço.

**Prova** — A ordenação feita na primeira linha do algoritmo possui complexidades algébricas $\mathcal{O}(m \log m)$ de tempo e $\mathcal{O}(m)$ de espaço. Portanto, basta mostrar que o restante do algoritmo possui complexidades algébricas $\mathcal{O}(n^3)$ de tempo e $\mathcal{O}(n^2)$ de espaço.

De fato, as linhas 10 e 11 são executadas $\mathcal{O}(n^2)$ vezes, de modo que

$$\sum_{i=0}^{n-1} |\mathcal{K}_i| \in \mathcal{O}(n^2).$$

Como $\mathcal{K}_i$ (com $i \in \{0, 1, \ldots, n-1\}$) são as únicas estruturas de dados adicionais no restante deste algoritmo, sua complexidade algébrica de espaço é (a menos da primeira linha) $\mathcal{O}(n^2)$, como queríamos mostrar.

Ainda, pelo que acabamos de ver, existem $\mathcal{O}(n^2)$ arcos candidatos a serem testados no laço da linha 19. Este laço possui no máximo $n$ iterações, de forma que a complexidade algébrica de tempo do algoritmo (com exceção da primeira linha) é $\mathcal{O}(n^3)$, também como queríamos mostrar. □

**Lema 6** — O algoritmo 2 é algebricamente tratável.

**Prova** — Antes de começarmos propriamente esta demonstração, recomendamos ao leitor a leitura da seção 1.5, onde está definido o conceito de nível de complexidade algébrica. A razão disso é que esta demonstração consiste essencialmente em se inspecionar o "código fonte" do algoritmo, linha a linha, e mostrar que existe um limite superior para os níveis de complexidade algébrica gerados naquela linha.

Mais precisamente, dizemos que uma expressão do algoritmo é algebricamente tratável, ou simplesmente tratável, se existir um número natural maior do que todos os níveis de complexidade algébrica de valores correspondentes a elementos de $\mathbb{F}$ computados ou presentes naquela expressão em qualquer que seja a entrada do algoritmo. Também dizemos que uma linha do algoritmo é tratável se todas as expressões daquela linha são tratáveis. Observe que expressões envolvendo apenas elementos da entrada (e as operações básicas) são sempre tratáveis, pois o nível de complexidade algébrica de elementos da entrada é 0.

Como anunciamos anteriormente, inspecionamos agora cada linha do código do algoritmo 2:

- Linha 1: durante a ordenação feita nesta linha, nenhum novo valor em $\mathbb{F}$ é computado, pois os elementos da entrada são apenas comparados;
- Linha 2: não envolve elementos de $\mathbb{F}$;
- Linhas 3 e 4: apenas copiam elementos de $\mathbb{F}$;
- Linha 5: como $x_i$, $y_i$ e $r$ são tratáveis (pois estão na entrada), temos que $x_i - r$, $x_i + r$ e, obviamente, $y_i$ são todos tratáveis;
- Linha 6: possui apenas sintaxe;
- Linhas 7 e 8: não envolvem elementos de $\mathbb{F}$;
- Linha 9: como $C_i$ e $C_j$ são tratáveis (pois estão na entrada), a proposição 25 nos garante que o conjunto (finito) $K = \partial C_i \cap \partial C_j$ será tratável;
- Linhas 10 e 11: apenas inserem elementos de $\mathbb{F}$ em estruturas de dados. Porém, observe que como $K$ é tratável, por indução podemos provar que, ao final do laço das linhas 7 a 13, os conjuntos $\mathcal{K}_i$ serão tratáveis para todo $i \in \{0, 1, \ldots, n-1\}$;
- Linhas 12 e 13: possuem apenas sintaxe;
- Linha 14: não envolve elementos de $\mathbb{F}$;

- Linha 15: note que como descrevemos nos detalhes de implementação do algoritmo, nesta linha não é necessário utilizar operações de elementos de $\mathbb{F}$ além das comparações. Observe também que como cada $\mathcal{K}_i$ é tratável para $i \in \{0, 1, \ldots, n-1\}$, então $p_a$ é tratável para cada $a \in \{0, 1, \ldots, k\}$, pois $p_0, p_1, \ldots, p_{k-1}$ é uma permutação dos elementos de $\mathcal{K}_i$;
- Linha 16: apenas copia (define) um elemento de $\mathbb{F}$;
- Linhas 17, 18 e 19: não envolvem elementos de $\mathbb{F}$;
- Linha 20: como $C_i$ e $C_j$ são tratáveis por estarem na entrada e, como visto anteriormente, $p_a$ é tratável para todo $a \in \{0, 1, \ldots, k\}$, a proposição 27 nos garante que todos os valores de $\mathbb{F}$ calculados nesta linha são tratáveis;
- Linha 21: não envolve elementos de $\mathbb{F}$;
- Linhas 22, 23 e 24: possuem apenas sintaxe;
- Linha 25: não envolve elementos de $\mathbb{F}$;
- Linha 26: apenas copia (para a saída) elementos de $\mathbb{F}$; e
- Linhas 27, 28 e 29: possuem apenas sintaxe.

Assim, devemos concluir que todas as expressões no algoritmo são tratáveis e, logo, que existe um número natural maior do que todos os níveis de complexidade algébrica de valores computados no algoritmo em qualquer entrada. Portanto, o algoritmo 2 é algebricamente tratável. □

**Teorema 6** — Existem constantes $k_0, k_1 \in \mathbb{N}$ tais que o algoritmo 2, quando executado com uma lista de $m \geq 2$ círculos contendo exatamente $n$ círculos distintos e tal que o número máximo de *bits* em uma representação computacional de um elemento de $\mathbb{F}$ é $B$, possui complexidades $\mathcal{O}\big((m \log m + n^3)B^{k_0}\big)$ de tempo e $\mathcal{O}\big((m + n^2)B^{k_1}\big)$ de espaço.

**Prova** — Consequência direta dos lemas 5 e 6 e do que foi discutido na seção 1.5. □

## 3.5 O algoritmo de Brown

Em 1980, Brown apresentou em sua tese de doutorado [11] o primeiro algoritmo essencialmente ótimo da literatura para computar a união de círculos. Esta, no entanto, é apenas uma das muitas contribuições desta tese para a área de geometria computacional.

O tema central da tese de Brown é o uso de transformações geométricas para reduzir problemas de geometria computacional a outros problemas conhecidos. Portanto, não é surpreendente que o algoritmo lá desenvolvido use uma transformação geométrica para reduzir o problema de computar a união de círculos a outros problemas que já possuem solução eficiente.

Nesta seção fazemos um estudo breve do algoritmo de Brown, mas antes estudamos a transformação usada por ele para reduzir o problema.

**Definição 57** — Seja $d \in \mathbb{N}$ uma dimensão e seja $p \in \mathbb{R}^d$ um ponto no espaço real $d$-dimensional. Definimos então a ***transformação de inversão*** de centro $p$ em $\mathbb{R}^d$ como a função

$$\begin{array}{rccl} \phi_p: & \mathbb{R}^d \setminus \{p\} & \to & \mathbb{R}^d \\ & q & \mapsto & p + \dfrac{q-p}{||q-p||^2}. \end{array}$$

Estudamos agora algumas propriedades essenciais das transformações de inversão para o algoritmo de Brown.

**Proposição 33** — Seja $d \in \mathbb{N}$ uma dimensão e seja $p \in \mathbb{R}^d$ um ponto no espaço real $d$-dimensional. Então, para todo ponto $q \in \mathbb{R}^d \setminus \{p\}$, $\phi_p(q) \neq p$ e $\phi_p\big(\phi_p(q)\big) = q$.

**Prova** — Como $q \neq p$, temos que $q - p \neq 0$, de forma que

$$\phi_p(q) = p + \frac{q-p}{||q-p||^2} \neq p.$$

Ainda, temos que

$$\begin{aligned} \phi_p\big(\phi_p(q)\big) &= p + \frac{\phi_p(q) - p}{||\phi_p(q) - p||^2} \\ &= p + \frac{p + \dfrac{q-p}{||q-p||^2} - p}{\left|\left|p + \dfrac{q-p}{||q-p||^2} - p\right|\right|^2} \\ &= p + \frac{\dfrac{q-p}{||q-p||^2}}{\left|\left|\dfrac{q-p}{||q-p||^2}\right|\right|^2} \\ &= p + \frac{\dfrac{q-p}{||q-p||^2}}{1/||q-p||^2} \\ &= p + q - p \\ &= q, \end{aligned}$$

como queríamos. $\square$

**Corolário 7** — Seja $d \in \mathbb{N}$ uma dimensão e seja $p \in \mathbb{R}^d$ um ponto no espaço real $d$-dimensional. Então $\phi_p$ é uma bijeção de $\mathbb{R}^d \setminus \{p\}$ em $\mathbb{R}^d \setminus \{p\}$.

**Prova** — Sejam $x, y \in \mathbb{R}^d \setminus \{p\}$ tais que $\phi_p(x) = \phi_p(y)$. Então, pela proposição 33,

$$x = \phi_p\big(\phi_p(x)\big) = \phi_p\big(\phi_p(y)\big) = y,$$

o que mostra que $\phi_p$ é injetiva.

Escolha então um $y \in \mathbb{R}^d \setminus \{p\}$. Temos, ainda pela proposição 33, que

$$\phi_p\big(\phi_p(y)\big) = y,$$

o que mostra que $\phi_p$ é sobrejetiva e, portanto, bijetiva. $\square$

**Proposição 34** — Seja $d \in \mathbb{N}$ uma dimensão, seja $p \in \mathbb{R}^d$ um ponto no espaço real $d$-dimensional e seja $X \subseteq \mathbb{R}^d \setminus \{p\}$. Então

$$\phi_p\big(\partial X \setminus \{p\}\big) = \big(\partial \phi_p(X)\big) \setminus \{p\}.$$

**Prova** — Imediato do corolário 1 quando consideramos $\Omega_0 = \Omega_1 = \mathbb{R}^2 \setminus \{p\}$. $\square$

**Proposição 35** — Seja $d \in \mathbb{N}$ uma dimensão, seja $p \in \mathbb{R}^d$ um ponto no espaço real $d$-dimensional e seja $\Delta \in \mathbb{R}^d$ um vetor de deslocamento. Então, para todo ponto $q \in \mathbb{R}^d \setminus \{p\}$,

$$\phi_{p+\Delta}(q + \Delta) = \phi_p(q) + \Delta.$$

**Prova** — Temos que

$$\phi_{p+\Delta}(q+\Delta) = p + \Delta + \frac{(q+\Delta) - (p+\Delta)}{||(q+\Delta) - (p+\Delta)||^2} = p + \frac{q-p}{||q-p||^2} + \Delta = \phi_p(q) + \Delta,$$

como desejado. $\square$

**Proposição 36** — Seja $d \in \mathbb{N}$ uma dimensão, seja $p \in \mathbb{R}^d$ um ponto no espaço real $d$-dimensional e seja $T : \mathbb{R}^d \to \mathbb{R}^d$ uma transformação linear tal que, para todo ponto $x \in \mathbb{R}^d$, $||T(x)|| = ||x||$. Então, para todo ponto $q \in \mathbb{R}^d \setminus \{p\}$,

$$\phi_p\big(p + T(q-p)\big) = p + T\big(\phi_p(q) - p\big).$$

**Prova** — Temos que,

$$\begin{aligned}
\phi_p\big(p + T(q-p)\big) &= p + \frac{p + T(q-p) - p}{||p + T(q-p) - p||^2} \\
&= p + \frac{T(q-p)}{||T(q-p)||^2} \\
&= p + \frac{T(q-p)}{||q-p||^2} \\
&= p + T\left(\frac{q-p}{||q-p||^2}\right) \\
&= p + T\left(p + \frac{q-p}{||q-p||^2} - p\right) \\
&= p + T\big(\phi_p(q) - p\big),
\end{aligned}$$

como queríamos. $\square$

A transformação de inversão é bastante conhecida entre os cartógrafos, pois a imagem da fronteira de uma esfera (em três dimensões) menos um ponto por uma transformação de inversão centrada neste ponto é um plano que não contém o ponto. Este método pode ser usado para produzir mapas, isto é, mapear a superfície de uma esfera em um plano. A seguir mostramos uma variante deste resultado.

**Proposição 37** — Seja $d \in \mathbb{N}$ uma dimensão e seja $p \in \mathbb{R}^d$ um ponto no espaço real $d$-dimensional. Então, para toda bola aberta $B$ tal que $p \in \partial B$, existe um semi-espaço aberto $H \subseteq \mathbb{R}^d$ tal que $\phi_p(B) = H$ e $\phi_p\big(\partial B \setminus \{p\}\big) = \partial H$.

**Prova** — Sejam $c \in \mathbb{R}^d$ e $r \in \mathbb{R}$ o centro e o raio, respectivamente, da bola $B$, de forma que $B = B_r(c)$. Note que $2c - p \neq p$, pois do contrário $2c = 2p$ e $c = p$, contradizendo $p \in \partial B$. Logo, podemos definir o semi-espaço aberto

$$H = \Big\{q \in \mathbb{R}^d : \big(q - \phi_p(2c - p)\big) \cdot (c - p) > 0\Big\},$$

onde "$\cdot$" denota o produto interno de vetores. Observe que a restrição sobre $q \in \mathbb{R}^d$ que define $H$ é uma restrição linear e estrita, de forma que $H$ é de fato um semi-espaço aberto.

Começamos então calculando

$$\begin{aligned}
\phi_p(2c - p) &= p + \frac{2c - p - p}{||2c - p - p||^2} \\
&= p + \frac{2(c - p)}{||2(c - p)||^2} \\
&= p + \frac{2(c - p)}{4||c - p||^2} \\
&= p + \frac{c - p}{2r^2},
\end{aligned}$$

pois claramente a distância entre $p$ e $c$ é o raio $r$. Assim,

$$\begin{aligned}
H &= \left\{q \in \mathbb{R}^d : \left((q - p) - \frac{c - p}{2r^2}\right) \cdot (c - p) > 0\right\}, \\
&= \left\{q \in \mathbb{R}^d : (q - p) \cdot (c - p) - \frac{(c - p)(c - p)}{2r^2} > 0\right\}, \\
&= \left\{q \in \mathbb{R}^d : (q - p) \cdot (c - p) > \frac{1}{2}\right\},
\end{aligned}$$

pois $(c - p) \cdot (c - p)$ é igual a $||c - p||^2 = r^2$.

Note agora que $p \notin H$, pois $(p - p) \cdot (c - p) = 0$. Assim,

$$\begin{aligned}
\phi_p(H) &= \phi_p\left(\left\{q \in \mathbb{R}^d \setminus \{p\} : (q - p) \cdot (c - p) > \frac{1}{2}\right\}\right) \\
&= \left\{\phi_p(q) : q \in \mathbb{R}^d \setminus \{p\} \land (q - p) \cdot (c - p) > \frac{1}{2}\right\} \\
&= \left\{q \in \mathbb{R}^d \setminus \{p\} : \big(\phi_p(q) - p\big) \cdot (c - p) > \frac{1}{2}\right\},
\end{aligned}$$

pois a função $\phi_p$ é a inversa de si própria. Logo,

$$\begin{aligned}
\phi_p(H) &= \left\{q \in \mathbb{R}^d \setminus \{p\} : \left(p + \frac{q-p}{||q-p||^2} - p\right) \cdot (c-p) > \frac{1}{2}\right\} \\
&= \left\{q \in \mathbb{R}^d \setminus \{p\} : \frac{q-p}{||q-p||^2} \cdot (c-p) > \frac{1}{2}\right\} \\
&= \left\{q \in \mathbb{R}^d \setminus \{p\} : \frac{(q-c)+(c-p)}{\big|\big|(q-c)+(c-p)\big|\big|^2} \cdot (c-p) > \frac{1}{2}\right\} \\
&= \left\{q \in \mathbb{R}^d \setminus \{p\} : 2\big((q-c)+(c-p)\big) \cdot (c-p) > \big|\big|(q-c)+(c-p)\big|\big|^2\right\} \\
&= \left\{q \in \mathbb{R}^d \setminus \{p\} : 2(q-c) \cdot (c-p) + 2(c-p) \cdot (c-p) > \big|\big|(q-c)+(c-p)\big|\big|^2\right\} \\
&= \left\{q \in \mathbb{R}^d \setminus \{p\} : 2(q-c) \cdot (c-p) + 2r^2 > ||q-c||^2 + 2(q-c) \cdot (c-p) + ||c-p||^2\right\} \\
&= \left\{q \in \mathbb{R}^d \setminus \{p\} : 2r^2 > ||q-c||^2 + ||c-p||^2\right\} \\
&= \left\{q \in \mathbb{R}^d \setminus \{p\} : 2r^2 > ||q-c||^2 + r^2\right\} \\
&= \left\{q \in \mathbb{R}^d \setminus \{p\} : ||q-c||^2 < r^2\right\} \\
&= \left\{q \in \mathbb{R}^d \setminus \{p\} : ||q-c|| < r\right\} \\
&= B_r(c) \setminus \{p\} \\
&= B,
\end{aligned}$$

pois $p \in \partial B$ de forma que $p \notin B$. Em suma, $\phi_p(H) = B$ e, como $\phi_p$ é sua própria inversa, $H = \phi_p(B)$.

Para completar a demonstração, tome $\Omega_0 = \Omega_1 = \mathbb{R}^d \setminus \{p\}$ e aplique o corolário 1, obtendo

$$\phi_p(\partial B \setminus \{p\}) = \partial \phi_p(B) \setminus \{p\} = \partial H \setminus \{p\} = \partial H,$$

pois $p \notin \partial H$ já que $(p-p) \cdot (c-p) = 0$. □

Com isto já estamos aptos a descrever o algoritmo de Brown. Este algoritmo recebe como entrada uma lista de círculos e remove dela círculos repetidos obtendo um conjunto $\mathcal{C}$ de círculos. Antes de descrever o resto do algoritmo, no entanto, vamos nos concentrar no caso particular em que existe um ponto $p \in \mathbb{R}^2$ tal que $p \in \partial C$ para todo círculo $C \in \mathcal{C}$.

Primeiramente consideramos o conjunto $\mathcal{B}$ das bolas abertas correspondentes aos círculos em $\mathcal{C}$. Em vez de nos focarmos em obter a união dos círculos em $\mathcal{C}$, vamos nos focar em obter a união das bolas abertas em $\mathcal{B}$. Para isso, consideramos o conjunto $\mathcal{H}$ dos semiplanos abertos que são a imagem de bolas abertas em $\mathcal{B}$ através da transformação de inversão $\phi_p$.

Note que

$$\bigcup_{H \in \mathcal{H}} H = \bigcup_{B \in \mathcal{B}} \phi_p(B) = \phi_p\left(\bigcup_{B \in \mathcal{B}} B\right).$$

Portanto, se pudermos computar a união dos semiplanos em $\mathcal{H}$ ou, mais precisamente, quais subconjuntos (conexos) das fronteiras dos semiplanos em $\mathcal{H}$ fazem parte da fronteira desta união, podemos, pela proposição 34, saber quais partes das fronteiras das bolas em $\mathcal{B}$

fazem parte da fronteira da união destas bolas, obtendo uma boa representação para esta união, que é também uma boa representação para a união dos círculos.

Note agora que

$$\partial\left(\bigcup_{H\in\mathcal{H}} H\right) = \partial\left(\mathbb{R}^2 \setminus \left(\bigcap_{H\in\mathcal{H}} \mathbb{R}^2 \setminus H\right)\right) = \partial\left(\bigcap_{H\in\mathcal{H}} \mathbb{R}^2 \setminus H\right).$$

Como $\mathbb{R}^2 \setminus H$ é um semiplano (fechado), resta computarmos a intersecção de semiplanos fechados. O próprio Brown, em sua tese de doutorado [11], trata este problema e mostra que ele pode ser resolvido com complexidades algébricas $\mathcal{O}(n \log n)$ de tempo e $\mathcal{O}(n)$ de espaço, onde $n$ é o número de semiplanos e, logo, o número de círculos.

Mas e se não existir um ponto no plano comum a todas as fronteiras de círculos na entrada? Neste caso usa-se um truque interessante: considerar os círculos do conjunto como círculos em um plano no espaço. Mais precisamente, definimos o conjunto

$$\mathcal{C}' = \big\{\{(x, y, 0) : (x, y) \in C\} : C \in \mathcal{C}\big\}$$

das versões espaciais dos círculos na entrada do algoritmo e escolhemos um ponto qualquer $p \in \mathbb{R}^3$ cuja terceira coordenada seja diferente de 0, ou seja, um ponto fora do plano em que os "círculos" em $\mathcal{C}'$ residem.

Vamos mostrar agora que, para cada "círculo" $C'$ em $\mathcal{C}'$, existe uma bola aberta que contém o ponto $p$ em sua fronteira e cuja intersecção com o plano onde os elementos de $\mathcal{C}'$ residem é $C'$.

**Proposição 38** — Seja $C \subseteq \mathbb{R}^2$ um círculo, seja $p = (x_1, y_1, z_1) \in \mathbb{R}^3$ com $z_1 \neq 0$ um ponto no espaço e seja $C' = \big\{(x, y, 0) : (x, y) \in C\big\}$ a versão espacial do círculo $C$. Então existe uma bola aberta $B \subseteq \mathbb{R}^3$ tal que $p \in \partial B$ e tal que $\big\{(x, y, 0) : (x, y, 0) \in B\big\} = C'^{\circ}$.

**Prova** — Seja $(x_0, y_0) \in \mathbb{R}^2$ o centro do círculo $C$ e seja $r \in \mathbb{R}$ com $r > 0$ o raio do círculo $C$. Defina então $z \in \mathbb{R}$ por

$$z = \frac{(x_1 - x_0)^2 + (y_1 - y_0)^2 + z_1^2 - r^2}{2z_1}$$

(note que $z_1 \neq 0$) e seja $B \subseteq \mathbb{R}^3$ a bola aberta

$$B = B_{\sqrt{r^2+z^2}}\big((x_0, y_0, z)\big).$$

Note então que

$$\begin{aligned}
\big|\big|p - (x_0, y_0, z)\big|\big|^2 &= \big|\big|(x_1, y_1, z_1) - (x_0, y_0, z)\big|\big|^2 \\
&= \big|\big|(x_1 - x_0, y_1 - y_0, z_1 - z)\big|\big|^2 \\
&= (x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z)^2 \\
&= (x_1 - x_0)^2 + (y_1 - y_0)^2 + z_1^2 - 2z_1 z + z^2 \\
&= (x_1 - x_0)^2 + (y_1 - y_0)^2 + z_1^2 - 2z_1 \frac{(x_1 - x_0)^2 + (y_1 - y_0)^2 + z_1^2 - r^2}{2z_1} + z^2 \\
&= (x_1 - x_0)^2 + (y_1 - y_0)^2 + z_1^2 - (x_1 - x_0)^2 - (y_1 - y_0)^2 - z_1^2 + r^2 + z^2 \\
&= r^2 + z^2
\end{aligned}$$

(pois $z_1 \neq 0$), de forma que $p \in \partial B$.

Seja então $(x, y) \in \mathbb{R}^2$ um ponto qualquer do plano. Para concluir a demonstração, devemos mostrar que $(x, y) \in C^\circ$ se, e somente se, $(x, y, 0) \in B$. Porém,

$$\begin{aligned}
(x, y, 0) \in B & \qquad \iff \\
\big|\big|(x, y, 0) - (x_0, y_0, z)\big|\big|^2 < r^2 + z^2 & \qquad \iff \\
\big|\big|(x - x_0, y - y_0, -z)\big|\big|^2 < r^2 + z^2 & \qquad \iff \\
(x - x_0)^2 + (y - y_0)^2 + z^2 < r^2 + z^2 & \qquad \iff \\
(x - x_0)^2 + (y - y_0)^2 < r^2 & \qquad \iff \\
(x, y) \in C^\circ, &
\end{aligned}$$

como queríamos. □

A ideia é então definir $\mathcal{B}$ como o conjunto das bolas obtidas tal como na proposição 38 em relação aos "círculos" em $\mathcal{C}'$. Se pudermos, de alguma forma, computar a união das bolas em $\mathcal{B}$, podemos intersectá-la com o plano dos pontos de terceira coordenada 0 para obter a união dos interiores dos círculos em $\mathcal{C}$.

Sabemos que a descrição destas últimas etapas é um pouco confusa, pois não tratamos neste trabalho de uniões de bolas tridimensionais (ou mesmo esferas). Por isso, vamos deixar os detalhes de como representar esta tal união de lado e nos concentrar no fato de que, teoricamente, com esta união podemos obter a resposta do problema intersectando-a com o plano onde residem os elementos de $\mathcal{C}'$.

Similarmente ao caso anterior, definimos o conjunto $\mathcal{H} = \big\{\phi_p(B) : B \in \mathcal{B}\big\}$ dos semi-espaços associados pela transformação de inversão a cada uma das bolas em $\mathcal{B}$. Novamente temos

$$\bigcup_{H \in \mathcal{H}} H = \bigcup_{B \in \mathcal{B}} \phi_p(B) = \phi_p\left(\bigcup_{B \in \mathcal{B}} B\right).$$

Logo, considerando a nossa proposital desatenção aos detalhes, basta computar a união dos semiplanos em $\mathcal{H}$.

Também similarmente ao que fizemos antes, notamos que

$$\bigcup_{H \in \mathcal{H}} H = \mathbb{R}^3 \setminus \left(\bigcap_{H \in \mathcal{H}} \mathbb{R}^3 \setminus H\right).$$

Assim, este problema recai, também de forma similar, no problema de computar intersecções de semi-espaços tridimensionais fechados. Brown, ainda em sua tese de doutorado, demonstra que é possível resolver este problema novamente com complexidades algébricas $\mathcal{O}(n \log n)$ de tempo e $\mathcal{O}(n)$ de espaço, onde $n$ é o número de semi-espaços, ou seja, o número de círculos na entrada.

## 3.6 O algoritmo de Imai, Iri e Murota

Nesta seção descrevemos o algoritmo mais elegante deste trabalho, desenvolvido por Imai, Iri e Murota em [19] como uma melhoria do algoritmo descrito em [23]. Contudo, em contraste com a maior parte deste trabalho, não faremos uma descrição formal deste algoritmo, pois ela envolveria vários conceitos sofisticados e complicados de formalizar. Em vez disso apresentamos as principais ideias por trás deste algoritmo intuitivamente e direcionamos os leitores mais interessados aos trabalhos previamente citados.

Suponha então que nos é dado um conjunto $\mathcal{C} = \{C_0, C_1, \ldots, C_{n-1}\}$ de círculos no plano onde, para todo $i \in \{0, 1, \ldots, n-1\}$, $c_i = (x_i, y_i)$ é o centro e $r_i$ é o raio do círculo $C_i$. O algoritmo de Imai, Iri e Murota então computa a união

$$U = \bigcup_{i=0}^{n-1} C_i$$

dos círculos em $\mathcal{C}$ com complexidades algébricas $\mathcal{O}(n \log n)$ de tempo e $\mathcal{O}(n)$ de espaço.

Didaticamente falando, é melhor iniciar explicando este algoritmo no caso particular em que todos os raios são iguais, ou seja, no caso em que $r_0 = r_1 = \cdots = r_{n-1}$. Observe que, neste caso, todos os centros $c_0, c_1, \ldots, c_{n-1}$ dos círculos em $\mathcal{C}$ são distintos. Assim, a chave do algoritmo é considerar o diagrama de Voronoi [6] do conjunto de pontos $\{c_0, c_1, \ldots, c_{n-1}\}$.

Lembramos que este diagrama de Voronoi é, essencialmente, uma partição do plano em conjuntos $\mathcal{V}_0, \mathcal{V}_1, \ldots, \mathcal{V}_{n-1}$, onde, para cada $i \in \{0, 1, \ldots, n-1\}$, $\mathcal{V}_i$, também denominado a $i$-ésima célula de Voronoi, é o conjunto dos pontos no plano que possuem o ponto $c_i$ como um dos pontos mais próximos dentre os pontos $c_0, c_1, \ldots, c_{n-1}$. Mais formalmente,

$$\mathcal{V}_i = \left\{p \in \mathbb{R}^2 : ||p - c_i|| \leq ||p - c_j||, \forall j \in \{0, 1, \ldots, n-1\}\right\}.$$

Se olharmos a definição de $\mathcal{V}_i$ com mais cuidado, perceberemos que este é o conjunto dos pontos do plano que satisfazem $n-1$ restrições. Nominalmente, para cada $j \in \{0, 1, \ldots, n-1\} \setminus \{i\}$, um ponto $p \in \mathcal{V}_i$ deve satisfazer $||p - c_i|| \leq ||p - c_j||$. Esta restrição, no entanto, é uma restrição linear. A mediatriz do segmento $c_i c_j$ (note que $c_i \neq c_j$) é precisamente o conjunto de pontos no plano que distam o mesmo de $c_i$ e $c_j$, de forma que os pontos $p$ do plano que satisfazem a restrição são exatamente os pontos em cima desta mediatriz ou no semiplano determinado por ela que contém $c_i$.

O conjunto $\mathcal{V}_i$ é, portanto, uma intersecção finita de semiplanos fechados. Tecnicamente dizemos que $\mathcal{V}_i$ é um politopo em $\mathbb{R}^2$. Porém, como todos estes semiplanos devem conter $c_i$ em seu interior (pois os centros dos círculos são todos distintos), o politopo $\mathcal{V}_i$ nunca é severamente degenerado. Existem, na verdade, três casos para a "forma" de $\mathcal{V}_i$:

- $\mathcal{V}_i$ é o próprio plano $\mathbb{R}^2$ quando existe apenas um círculo na entrada;
- $\mathcal{V}_i$ é um polígono convexo; ou
- $\mathcal{V}_i$ é um conjunto ilimitado relativamente difícil de se descrever.

É por causa deste último caso que muitas vezes um retângulo (fechado) $R$ contendo todos os círculos em $\mathcal{C}$ em seu interior é considerado. Neste caso a intersecção $\mathcal{V}_i \cap R$ é sempre um polígono convexo (podendo ser o próprio retângulo $R$), o que é um fato muito mais simples que admite uma elementar demonstração por indução. Inclusive, quando consideramos, nesta seção, um algoritmo para computar este diagrama de Voronoi, usamos este truque para representar computacionalmente sua saída. Porém, em geral, esta não é a forma mais elegante de tratar este conjunto computacionalmente.

A principal ideia por trás do algoritmo de Imai, Iri e Murota é, traduzida para este caso particular que estamos tratando, observar que, na célula $\mathcal{V}_i$, a união dos círculos coincide com o círculo $C_i$ ou, mais precisamente, $U \cap \mathcal{V}_i = C_i \cap \mathcal{V}_i$. Para demonstrar este fato, primeiro observe que $C_i \subseteq U$ e que, logo, $C_i \cap \mathcal{V}_i \subseteq U \cap \mathcal{V}_i$. Resta então mostrar que $U \cap \mathcal{V}_i \subseteq C_i \cap \mathcal{V}_i$. De fato, seja $p \in U \cap \mathcal{V}_i$. Como $p \in U$, existe algum $j \in \{0, 1, \ldots, n-1\}$ tal que $p \in C_j$, ou seja, $||p - c_j|| \leq r_j$. Porém, como $p \in \mathcal{V}_i$, $||p - c_i|| \leq ||p - c_j|| \leq r_j = r_i$, de modo que $p \in C_i$. Ainda, como $p \in \mathcal{V}_i$, segue que $p \in C_i \cap \mathcal{V}_i$. Vale a pena notar que aqui usamos fortemente o fato de todos os raios dos círculos serem iguais.

Observe também que, se $R$ é um retângulo (fechado) contendo todos os círculos de $\mathcal{C}$ em seu interior, então, como $U \cap \mathcal{V}_i = C_i \cap \mathcal{V}_i$,

$$\begin{aligned}
U \cap (\mathcal{V}_i \cap R) &= (U \cap R) \cap (\mathcal{V}_i \cap R) \\
&= (U \cap \mathcal{V}_i) \cap R \\
&= (C_i \cap \mathcal{V}_i) \cap R \\
&= (C_i \cap R) \cap (\mathcal{V}_i \cap R) \\
&= C_i \cap (\mathcal{V}_i \cap R).
\end{aligned}$$

Com isso, segue que $\partial U \cap (\mathcal{V}_i \cap R)^\circ = \partial C_i \cap (\mathcal{V}_i \cap R)^\circ$. A proposição 31 nos mostra que $U$ pode ser bem representado por um número finito de arcos de circunferência orientados. Juntamente com o fato de que $\partial(\mathcal{V}_i \cap R)$ é a união de um número finito de segmentos de reta, segue que $\partial U \cap \partial(\mathcal{V}_i \cap R)$ é um conjunto finito. Assim, para obter uma boa representação para $U$, basta determinar os arcos de circunferência em $\partial C_i \cap (\mathcal{V}_i \cap R)^\circ$ para cada $i \in \{0, 1, \ldots, n-1\}$. Porém, isto pode ser feito em tempo linear no número de arestas do polígono convexo $\mathcal{V}_i \cap R$. Essencialmente basta achar a intersecção da fronteira de um círculo com a de um polígono convexo contendo seu centro, o que é relativamente simples.

Resta apenas uma questão: sendo a complexidade (algébrica de tempo) deste algoritmo proporcional às somas dos números de arestas (ou vértices) nos polígonos $\mathcal{V}_i \cap R$, como mostramos que esta soma é $\mathcal{O}(n \log n)$?

Na verdade, é um resultado padrão de diagramas de Voronoi que esta soma é inclusive $\mathcal{O}(n)$. Uma forma de demonstrar isso consiste em notar uma estrutura combinatória interessante que ainda não exploramos. Trata-se do grafo $G = (V, E)$ onde $V = \{0, 1, \ldots, n-1\}$ e $E$ é o conjunto das arestas $\{i, j\}$ tais que $\mathcal{V}_i \cap \mathcal{V}_j$ é um conjunto infinito. Alternativamente, podemos definir o conjunto $E$ como o conjunto das arestas $\{i, j\}$ tais que existe uma aresta de $\mathcal{V}_i \cap R$ contida em uma aresta de $\mathcal{V}_j \cap R$ ou vice-versa. De qualquer forma, o ponto importante é que o grafo $G$ é planar, e está inclusive fortemente relacionado com o que chamamos de uma triangulação de Delaunay [14, capítulo 9].

Para mostrar que $G$ é planar, precisamos apresentar um mapeamento ("embedding") de $G$ no plano. A cada vértice $i$ de $G$, associamos então o ponto $c_i$ e a cada aresta $\{i, j\}$ de $G$, associamos o segmento de reta $c_i c_j$. Resta então mostrar que dois tais segmentos nunca se cruzam ou estão sobrepostos. Esta demonstração é extremamente técnica, no entanto, e orientamos o leitor interessado a consultar [5].

O fato de $G$ ser planar é extremamente relevante, pois, em todo grafo planar com pelo menos três vértices, o número de arestas nunca é superior a três vezes o número de vértices menos seis. Em nosso caso particular, $|V| = n$ e, portanto, $|E| \in \mathcal{O}(n)$. Porém, a menos de possíveis degenerações, $|E|$ é justamente metade do total de arestas nos polígonos convexos $\mathcal{V}_i \cap R$ com $i \in \{0, 1, \ldots, n-1\}$ que não estão contidas em arestas do retângulo $R$. Se uma verificação mais detalhada for conduzida, mesmo com as mencionadas degenerações e incluindo as arestas contidas em arestas de $R$, conclui-se que este total de arestas é de fato $\mathcal{O}(n)$.

Assim, toda a parte que discutimos até então é na verdade linear. O fator $\mathcal{O}(n \log n)$ vem do algoritmo para computar o diagrama de Voronoi em si. Um algoritmo extremamente elegante para fazer isso é o algoritmo de Fortune [17].

Passamos agora a tratar do caso geral em que os raios dos círculos em $\mathcal{C}$ não são necessariamente idênticos. Neste caso não necessariamente é verdade que os centros dos círculos são todos distintos. Contudo, podemos, sem perda de generalidade, descartar da entrada círculos contidos em outros círculos na entrada, uma vez que a união é preservada. Assim, de forma similar ao que o algoritmo 2 faz para remover círculos repetidos da entrada, podemos ordenar os círculos em $\mathcal{C}$ lexicograficamente pelas coordenadas de seus centros e agrupar círculos com o mesmo centro, descartando todos exceto o de maior raio de cada agrupamento. Desta forma, os círculos restantes terão centros distintos e a mesma união que os círculos originais. Ainda, este procedimento possui complexidades algébricas $\mathcal{O}(n \log n)$ de tempo e $\mathcal{O}(n)$ de espaço, não atrapalhando os limites assintóticos que desejamos obter para este algoritmo. Sendo assim assumimos daqui em diante que os círculos no conjunto $\mathcal{C}$ possuem centros distintos.

Mesmo assim, no entanto, diagramas de Voronoi não são mais suficientes para este caso. Aqui precisamos de uma estrutura combinatória que leve em conta os raios dos círculos, chamada de diagrama de Laguerre [19]. O diagrama de Laguerre do conjunto de círculos $\mathcal{C}$ são os conjuntos $\mathcal{L}_0, \mathcal{L}_1, \ldots, \mathcal{L}_{n-1}$, também chamados de células de Laguerre, definidos por

$$\mathcal{L}_i = \left\{p \in \mathbb{R}^2 : ||p - c_i||^2 - r_i^2 \leq ||p - c_j||^2 - r_j^2, \forall j \in \{0, 1, \ldots, n-1\}\right\}.$$

A primeira coisa que percebemos sobre o diagrama de Laguerre é que ele generaliza o diagrama de Voronoi. De fato, se os raios dos círculos forem iguais, $\mathcal{L}_i = \mathcal{V}_i$ para todo $i \in \{0, 1, \ldots, n-1\}$. Porém, estes diagramas possuem diversas outras propriedades, que estudamos agora.

Uma propriedade que é chave para o algoritmo de Imai, Iri e Murota é o fato de que cada restrição da forma

$$||p - c_i||^2 - r_i^2 \leq ||p - c_j||^2 - r_j^2$$

com $i, j \in \{0, 1, \ldots, n-1\}$ e $i \neq j$ é uma restrição linear no sentido de que o conjunto dos pontos $p \in \mathbb{R}^2$ que a satisfazem é um semiplano fechado. Inclusive, antes da descoberta deste algoritmo, o melhor algoritmo no estado da arte para este problema era o descrito em [23]. Este algoritmo usava ideias extremamente similares às do algoritmo de Imai, Iri e Murota, mas, em vez de usar diagramas de Laguerre, ele usava um outro tipo de diagrama com propriedades similares, cujas restrições não eram lineares. Isso fez com que este algoritmo consumisse tempo $\mathcal{O}(n \log^2 n)$.

Vejamos então como demonstrar que estas restrições são de fato lineares, mas primeiro clarifiquemos a notação. Sejam $i, j \in \{0, 1, \ldots, n-1\}$ com $i \neq j$, seja $d \in \mathbb{R}$ com $d > 0$

a distância euclidiana $||c_j - c_i||$ entre $c_i$ e $c_j$, seja

$$\begin{array}{rccl} \mu: & \mathbb{R}^2 & \to & \mathbb{R} \\ & p & \mapsto & \big(||p-c_j||^2 - r_j^2\big) - \big(||p-c_i||^2 - r_i^2\big) \end{array}$$

uma função objetivo e seja

$$L = \big\{p \in \mathbb{R}^2 : \mu(p) \geq 0\big\}$$

o conjunto dos pontos que satisfazem a restrição. Considere então a reta $\ell$ que passa pelos centros $c_i$ e $c_j$ dos círculos $C_i$ e $C_j$ e a parametrize através da função

$$\begin{array}{rccl} \gamma: & \mathbb{R} & \to & \mathbb{R}^2 \\ & t & \mapsto & (1-t)c_i + tc_j. \end{array}$$

Nosso primeiro passo é estudar quais valores $t \in \mathbb{R}$ são tais que $\gamma(t) \in L$, ou seja, $\mu\big(\gamma(t)\big) \geq 0$. Temos que

$$\begin{aligned} \mu\big(\gamma(t)\big) \geq 0 &\iff \\ \big(||\gamma(t)-c_j||^2 - r_j^2\big) - \big(||\gamma(t)-c_i||^2 - r_i^2\big) \geq 0 &\iff \\ \Big(\big(d(1-t)\big)^2 - r_j^2\Big) - \big((dt)^2 - r_i^2\big) \geq 0 &\iff \\ \big(d^2(1-t)^2 - d^2t^2\big) - (r_j^2 - r_i^2) \geq 0 &\iff \\ (d^2 - 2d^2t + d^2t^2 - d^2t^2) - (r_j^2 - r_i^2) \geq 0 &\iff \\ (d^2 - 2d^2t) - (r_j^2 - r_i^2) \geq 0 &\iff \\ 2d^2t \leq d^2 - (r_j^2 - r_i^2) &\iff \\ t \leq \frac{d^2 - (r_j^2 - r_i^2)}{2d^2}. & \end{aligned}$$

Em outras palavras, $L \cap \ell$ é um "prefixo" fechado da reta $\ell$, o que mostra que, sobre a reta $\ell$, o conjunto $L$ é uma restrição linear.

Para mostrar isto para o plano, seja $A \in \mathbb{R}^2$ um ponto qualquer do plano e considere as seguintes definições:

- O ponto $B \in \ell$ é a projeção ortogonal de $A$ sobre a reta $\ell$;
- O número real não-negativo $h$ é a altura do triângulo (possivelmente degenerado) $c_iAc_j$;
- O número real não-negativo $d_0$ é o comprimento $||B - c_i||$ do segmento (possivelmente degenerado) $c_iB$;
- O número real não-negativo $d_1$ é o comprimento $||c_j - B||$ do segmento (possivelmente degenerado) $Bc_j$;
- O número real não-negativo $d_0'$ é o comprimento $||A - c_i||$ do segmento (possivelmente degenerado) $c_iA$; e
- O número real não-negativo $d_1'$ é o comprimento $||c_j - A||$ do segmento (possivelmente degenerado) $Ac_j$.

Assim,

$$
\begin{aligned}
\mu(A) - \mu(B) &= \Big( \big( ||A - c_j||^2 - r_j^2 \big) - \big( ||A - c_i||^2 - r_i^2 \big) \Big) \\
&- \Big( \big( ||B - c_j||^2 - r_j^2 \big) - \big( ||B - c_i||^2 - r_i^2 \big) \Big) \\
&= \big( ({d'_1}^2 - r_j^2) - ({d'_0}^2 - r_i^2) \big) - \big( ({d_1}^2 - r_j^2) - ({d_0}^2 - r_i^2) \big) \\
&= ({d'_1}^2 - d_1^2) - ({d'_0}^2 - d_0^2) \\
&= h^2 - h^2 \\
&= 0,
\end{aligned}
$$

devido às aplicações do teorema de Pitágoras aos triângulos retângulos $c_iBA$ e $c_jBA$. Portanto, $\mu(A) = \mu(B)$ e, consequentemente, $A \in L \iff B \in L$, o que mostra que $L$ é de fato uma restrição linear.

Contudo, as células de Laguerre, quando intersectadas com um retângulo (fechado) $R$ contendo todos os círculos de $\mathcal{C}$ em seu interior, não necessariamente são polígonos convexos. Nos diagramas de Voronoi, vimos que cada célula $\mathcal{V}_i$ continha um ponto que satisfazia estritamente todas as restrições definindo a célula, nominalmente o ponto $c_i$. No diagrama de Laguerre isto não necessariamente é verdade, e assim o conjunto $\mathcal{L}_i \cap R$ pode ser degenerado (ter área nula). Porém, do contrário, este conjunto será sim um polígono convexo.

O que resta mostrar para basicamente concluir nossa apresentação deste tópico é que a união dos círculos iguala cada círculo $C_i$ dentro da célula $\mathcal{L}_i$, ou seja, que $U \cap \mathcal{L}_i = C_i \cap \mathcal{L}_i$. Similarmente ao que vimos antes, $C_i \cap \mathcal{L}_i \subseteq U \cap \mathcal{L}_i$, pois $C_i \subseteq U$. Agora, para mostrar que $U \cap \mathcal{L}_i \subseteq C_i \cap \mathcal{L}_i$, tome um ponto $p \in U \cap \mathcal{L}_i$ qualquer. Como $p \in U$, existe algum $j \in \{0, 1, \ldots, n-1\}$ tal que $p \in C_j$. Assim, $||p - c_j|| \leq r_j$ ou, em outros termos, $||p - c_j||^2 - r_j^2 \leq 0$. Porém, como $p \in \mathcal{L}_i$, $||p - c_i||^2 - r_i^2 \leq ||p - c_j||^2 - r_j^2 \leq 0$, de modo que $||p - c_i|| \leq r_i$, o que mostra que $p \in C_i$. Logo, concluímos que $U \cap \mathcal{L}_i \subseteq C_i \cap \mathcal{L}_i$, como queríamos.

O algoritmo de Imai, Iri e Murota consiste então em computar o diagrama de Laguerre e calcular a intersecção entre a fronteira de cada círculo com sua célula de Laguerre, de forma extremamente similar ao que descrevemos com os diagramas de Voronoi. Por um argumento praticamente idêntico envolvendo a planaridade de um grafo análogo, se $R$ é um retângulo (fechado) contendo todos os círculos de $\mathcal{C}$ em seu interior, então o total de arestas em polígonos convexos da forma $\mathcal{L}_i \cap R$ com $i \in \{0, 1, \ldots, n-1\}$ é $\mathcal{O}(n)$. Logo, a complexidade do algoritmo é dominada pela complexidade de computar o diagrama de Laguerre.

Ainda em [19], há a descrição de um algoritmo com complexidades algébricas $\mathcal{O}(n \log n)$ de tempo e $\mathcal{O}(n)$ de espaço para computar diagramas de Laguerre. Este algoritmo difere substancialmente do algoritmo de Fortune [17] para diagramas de Voronoi, pois os autores notam que é difícil generalizar o algoritmo de Fortune. A solução encontrada por eles foi generalizar o primeiro algoritmo na literatura para computar diagramas de Voronoi, o algoritmo de Shamos [22], que funciona por divisão e conquista em vez de usar uma linha de varredura.

# Capítulo 4

# União de polígonos

## 4.1 Enunciado do problema

Enunciamos agora o problema da união de polígonos, que é o problema no qual iremos nos focar no resto deste trabalho.

**Problema 2** (união de polígonos) —

- **Entrada:** Uma lista de polígonos cujos vértices são todos elementos de $\mathbb{F}^2$. Cada polígono na entrada será representado por uma lista de seus vértices em sentido anti-horário. Cada um destes vértices será representado por um par com suas coordenadas.
- **Saída:** Uma sequência finita de segmentos de reta orientados que formam uma boa representação da união dos polígonos na entrada. As coordenadas dos extremos destes segmentos orientados devem ser todas elementos de $\mathbb{F}$. Ainda, cada um destes segmentos orientados deverá ser representado pelas quatro coordenadas de seus extremos.

Precisamos ainda mostrar que este problema sempre admite uma solução. Porém, esta prova fica bastante mais simples quando adiada ao final da seção 4.3, onde estudamos uniões de polígonos.

## 4.2 Estado da arte

Correndo o risco de repetir muito do que dissemos na introdução, observamos que a literatura em operações *booleanas* em polígonos é bastante extensa e compreende vários algoritmos eficientes para computar a união, intersecção, diferença, diferença simétrica e várias outras composições de dois polígonos. Dois destes algoritmos são apresentados em [20, seção 7.8] e [14, seção 2.4]. Ambos podem ser implementados de forma a possuírem complexidades algébricas $\mathcal{O}(M \log N)$ de tempo e $\mathcal{O}(N)$ de espaço, onde $N$ é o número total de vértices nos polígonos e $M$ é o número de pares de arestas dos polígonos que se intersectam.

Porém, conforme descritos, estes algoritmos funcionam apenas com dois polígonos. Generalizar estes algoritmos para mais polígonos não é exatamente simples, uma vez que a união de dois polígonos não necessariamente é um polígono, e assim uma aplicação do método de divisão e conquista não é imediata.

Também como mencionamos na introdução, existem técnicas para mesmo assim aplicar a divisão e conquista a este problema. As mais notáveis são as técnicas baseadas em polígonos de Nef [10] e as técnicas baseadas em estender o conceito de polígonos para aceitar os denominados "buracos". Ambas as técnicas definem estruturas mais gerais que são fechadas pela operação de união. Há implementações eficientes de ambas as técnicas na biblioteca CGAL [1] e uma implementação da união de polígonos com "buracos" na biblioteca LEDA [3].

Porém, todas estas implementações oferecem apenas uma análise de complexidade da operação de unir dois "polígonos", que custa, em todas elas, $\mathcal{O}(M \log N)$ de tempo e $\mathcal{O}(N)$ de espaço. Portanto, o que podemos fazer aqui é oferecer uma análise simples sobre o método de divisão e conquista implementado da forma usual.

Para isso, considere quatro polígonos $P_0, P_1, P_2, P_3 \subseteq \mathbb{R}^2$ com $N$ vértices cada um tais que, para cada par de índices $i, j \in \{0, 1, 2, 3\}$ com $i \neq j$, existam $\Omega(M)$ pares de arestas nestes polígonos que se intersectam. Assim, o algoritmo consiste em obter os "polígonos" $P_{0,1} = P_0 \cup P_1$, $P_{2,3} = P_2 \cup P_3$ e finalmente $P_{0,1,2,3} = P_{0,1} \cup P_{2,3}$. As duas primeiras etapas custam cada uma $\Omega(M \log N)$ de tempo e $\Omega(N)$ de espaço. Porém, a última etapa, na verdade, requer tempo $\Omega(M \log N)$ mas precisa de espaço $\Omega(M)$, pois os "polígonos" $P_{0,1}$ e $P_{2,3}$ podem ter $\Omega(M)$ vértices. Assim, assintoticamente falando, estes algoritmos precisam de $\Omega(M \log N)$ de tempo e $\Omega(M)$ de espaço.

O algoritmo que projetamos possui a mesma complexidade de tempo, porém requer apenas $\mathcal{O}(N)$ de espaço. Embora este não seja um resultado extremamente significativo, esta é o que consideramos a real contribuição deste trabalho.

Vale dizer que o artigo [8] descreve um algoritmo muito mais geral. Porém, eles não fizeram menção às complexidades de seus algoritmos e estimamos que estes algoritmos tenham o mesmo consumo adicional de espaço.

Vale a pena perguntar qual a importância de economizar o espaço, e na verdade o que significa isso, uma vez que o próprio tamanho da saída é $\Omega(M)$. Acontece que há aplicações de união de polígonos em que não é necessário gravar a saída completa, mas sim processá-la conforme ela é gerada. Estas aplicações incluem não só computar a área, o perímetro e o centro de massa da união, mas também até renderizar a união em um *framebuffer*.

## 4.3 Uniões de polígonos

Esta seção tem um papel análogo ao da seção 3.3, introduzindo os conceitos de pontos chave e segmentos candidatos de uma lista de polígonos. Mais adiante, veremos também que o estudo destes conceitos resulta em um método para se obter uma boa representação da união de polígonos.

**Definição 58** — Dizemos que o conjunto dos ***pontos chave*** de uma lista $L$ de polígonos é o conjunto $\mathcal{K}(L)$ dos pontos em $\mathbb{R}^2$ que pertencem à uma intersecção finita de duas arestas de polígonos (não necessariamente distintos) na lista $L$.

Note que os pontos chave de uma lista de polígonos incluem todos os vértices de polígonos na lista.

**Definição 59** — Os ***segmentos candidatos*** de uma lista de polígonos $L$ são os segmentos de reta da forma $uv$ com $u, v \in \mathcal{K}(L)$ tais que $uv \cap \mathcal{K}(L) = \{u, v\}$ e que existe uma aresta em um polígono de $L$ que contém $uv$.

Note que devido à possibilidade de arestas de polígonos diferentes na lista se sobreporem, não é em geral possível assinalar uma direção a um segmento candidato.

**Proposição 39** — Dois segmentos candidatos distintos de uma lista de polígonos podem se intersectar apenas em seus extremos.

**Prova** — Sejam $u_0v_0$ e $u_1v_1$ dois segmentos candidatos distintos de uma lista $L$ de polígonos. Então, se a intersecção de $u_0v_0$ e $u_1v_1$ for infinita, como $u_0v_0 \neq u_1v_1$, devemos ter $\{u_0, v_0\} \cap \big(u_1v_1 \setminus \{u_1, v_1\}\big) \neq \{\}$ ou $\{u_1, v_1\} \cap \big(u_0v_0 \setminus \{u_0, v_0\}\big) \neq \{\}$. Porém, como $u_0, v_0, u_1, v_1 \in \mathcal{K}(L)$ e ambos $u_0v_0$ e $u_1v_1$ são segmentos candidatos, isto contradiz, respectivamente, $\big(u_1v_1 \setminus \{u_1, v_1\}\big) \cap \mathcal{K}(L) = \{\}$ ou $\big(u_0v_0 \setminus \{u_0, v_0\}\big) \cap \mathcal{K}(L) = \{\}$.

Já se esta intersecção for finita, ela pode ter zero pontos (caso no qual o resultado vale trivialmente) ou exatamente um ponto $p \in \mathbb{R}^2$. Se $p$ for um extremo de $u_0v_0$ ou de $u_1v_1$, o resultado também segue trivialmente. Porém, se $p \in \big(u_0v_0 \setminus \{u_0, v_0\}\big) \cap \big(u_1v_1 \setminus \{u_1, v_1\}\big)$, observamos que, sendo a intersecção finita de dois segmentos candidatos, $p$ deve ser a intersecção finita de duas arestas em polígonos de $L$ e assim um ponto chave de $L$, o que claramente contradiz $u_0v_0$ e $u_1v_1$ serem segmentos candidatos. $\square$

As propriedades dos arcos candidatos vistos na seção 3.3 são muito mais numerosas do que a de segmentos candidatos, de forma que os resultados correspondentes aos que seguem também o são.

**Lema 7** — Seja $L$ uma lista de polígonos cuja união é $U$, seja $p \in \partial U$ um ponto na fronteira de $U$, seja $\overrightarrow{uv}$ uma aresta de um polígono em $L$ que contém $p$ e seja $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ um valor real tal que $B_\varepsilon(p)$ não contenha pontos chave de $L$ e não intersecte nenhuma aresta de polígonos em $L$ que não contenha $p$. Então, para todo ponto $q \in B_\varepsilon(p)$, $q \in U$ se, e somente se, $(v - u) \times (q - p) \geq 0$.

**Prova** — Primeiramente, note que como $B_\varepsilon(p) \cap \mathcal{K}(L) = \{\}$, então $p \notin \mathcal{K}(L)$ e, em particular, $p \neq u$ e $p \neq v$. Assim, sendo $E$ o conjunto das arestas dos polígonos em $L$ que contém $p$, toda aresta $e \in E$ deve ter uma intersecção infinita com $uv$, pois do contrário, como $p$ está em $e$ e em $uv$, teríamos $p \in \mathcal{K}(L)$. Deste modo, toda aresta em $E$ é paralela a $\overrightarrow{uv}$.

Agora, se $P$ é um polígono em $L$ que possui $\overrightarrow{uv}$ como aresta, note que $\overrightarrow{uv}$ é a única aresta de $P$ que intersecta $B_\varepsilon(p)$, e assim, podemos usar um teste local para decidir se um ponto $q \in B_\varepsilon(p)$ está em $P$: $q \in P$ se, e somente se, $(v-u) \times (q-p) \geq 0$, ou seja, $q$ está "a esquerda" do segmento $\overrightarrow{uv}$.

Vamos então estudar a intersecção de outros polígonos em $L$ com $B_\varepsilon(p)$. Se $Q$ é um polígono em $L$ que não possui nenhuma aresta em $E$, então nenhuma aresta de $Q$ intersecta $B_\varepsilon(p)$, e assim ou $B_\varepsilon(p) \subseteq Q^\circ$ ou $B_\varepsilon(p) \cap Q = \{\}$. Porém, no primeiro caso, teríamos $p \in B_\varepsilon(p) \subseteq Q^\circ \subseteq U^\circ$, contradizendo $p \in \partial U$. Logo devemos ter $B_\varepsilon(p) \cap Q = \{\}$.

Já se $Q$ é um polígono em $L$ que contém uma aresta $\overrightarrow{u'v'}$ em $E$, já vimos que $\overrightarrow{u'v'}$ deve ser paralela a $\overrightarrow{uv}$, ou seja, deve existir um $\alpha \in \mathbb{R}$ com $v' - u' = \alpha(v-u)$. Mostramos agora que, além de paralelas, estas arestas devem ter o mesmo sentido, isto é, $\alpha > 0$. De fato, por um argumento similar ao feito anteriormente para o polígono $P$, um ponto $q \in B_\varepsilon(p)$ está em $Q$ se, e somente se, $(v'-u') \times (q-p) \geq 0$. Porém, se $\alpha < 0$, $(v-u) \times (q-p) < 0$ implica em $(v'-u') \times (q-p) = \alpha(v-u) \times (q-p) > 0$, de modo que todo ponto $q \in B_\varepsilon(p)$ está em $P$ ou em $Q$. Mas assim $B_\varepsilon(p) \subseteq (P \cup Q) \subseteq U$, contradizendo $p \in \partial U$.

Assim, devemos ter que todo $q \in B_\varepsilon(p)$ está em $Q$ se, e somente se, $q$ está em $P$. No entanto, a intersecção de $B_\varepsilon(p)$ com $U$ é a união da intersecção de $B_\varepsilon(p)$ com os polígonos em $L$. Vimos que para alguns destes polígonos esta intersecção é vazia e para outros (pelo menos o polígono $P$), ela é igual a $B_\varepsilon(p) \cap P$, de modo que $B_\varepsilon(p) \cap U = B_\varepsilon(p) \cap P$, e assim, como vimos antes, um ponto $q \in B_\varepsilon(p)$ está em $U$ se, e somente se, $(v-u) \times (q-p) \geq 0$. $\square$

**Proposição 40** — Seja $L$ uma lista de polígonos cuja união é $U$ e seja $uv$ um segmento candidato de $L$. Então ou $uv \subseteq \partial U$ ou $uv \setminus \{u, v\} \subseteq U^\circ$.

**Prova** — A demonstração será por contradição. Suponha então que $uv \not\subseteq \partial U$ e que $uv \setminus \{u, v\} \not\subseteq U^\circ$. Logo, deve haver pontos $p_0, p_1 \in uv \setminus \{u, v\}$ tais que $p_0 \in \partial U$ e $p_1 \in U^\circ$ (pois claramente $uv \subseteq U$). Seja agora $\gamma : [0, 1] \to \mathbb{R}^2$ a parametrização canônica de $uv$ e sejam $t_0, t_1 \in [0, 1]$ os únicos valores tais que $\gamma(t_0) = p_0$ e $\gamma(t_1) = p_1$. Assumimos, sem perda de generalidade, que $t_0 < t_1$, pois do contrário observamos que $vu$ é o mesmo segmento candidato, mas cuja parametrização canônica satisfaz este requisito.

Seja então $T = \{t \in [t_0, t_1] : \gamma(t) \in \partial U\}$. Como $t_0 \in T$ (pois $\gamma(t_0) = p_0 \in \partial U$), temos que $T \neq \{\}$. Ainda, obviamente $t_1$ é um limite superior para todo elemento de $T$. Logo existe o supremo $t = \sup T$. Defina então $p = \gamma(t)$.

Mostramos agora que $p \in \partial U$. De fato, se $t = t_0$, então obviamente

$$p = \gamma(t) = \gamma(t_0) = p_0 \in \partial U.$$

Já se $t > t_0$, definimos uma sequência (infinita) de pontos $x_0, x_1, \ldots$ em $\partial U$ da seguinte forma: para cada $k \in \mathbb{N}$, como $t = \sup T$, deve existir um elemento

$$t'_k \in \left[\max\left\{t_0, t - \frac{1}{k+1}\right\}, t\right) \cap T.$$

Assim, $\gamma(t'_k) \in \partial U$, e pomos $x_k = \gamma(t'_k)$. Note, no entanto, que $\lim_{k\to\infty} t'_k = t$. Logo, como $\gamma$ é uma função contínua, $\lim_{k\to\infty} x_k = \lim_{k\to\infty} \gamma(t'_k) = \gamma(t) = p$. Porém, $\partial U$ é um conjunto fechado, portanto devemos concluir que $p \in \partial U$.

Por outro lado, como $0 < t_0 < t_1 < 1$, devemos ter $p \notin \{u, v\}$, de modo que $p \notin \mathcal{K}(L)$ (pois os únicos pontos chave no segmento candidato $uv$ são $u$ e $v$). Seja então $E$ o conjunto de todas as arestas de polígonos em $L$ que contém o ponto $p$ e tome um $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ tal que $B_\varepsilon(p)$ não contenha pontos chave e não intersecte nenhuma aresta de polígonos em $L$ a menos das arestas em $E$ (este $\varepsilon$ existe pois arestas são conjuntos fechados). Note então que, pelo lema 7 um ponto $q \in B_\varepsilon(p)$ está em $U$ se, e somente se, $s \times (q - p) \geq 0$, onde $s$ é o sentido de uma aresta em $E$, (onde o sentido de um segmento orientado $AB$ é o vetor $\frac{B-A}{||B-A||}$). Logo, podemos concluir que $uv \cap B_\varepsilon(p) \subseteq \partial U$, pois as arestas em $E$ devem ser todas paralelas a $uv$.

Isto, no entanto, contradiz $t = \sup T$, pois, como $\gamma$ é contínua, existe um $\delta \in \mathbb{R}$ com $0 < \delta < \min\{t, 1-t\}$, tal que, para todo $t' \in (t - \delta, t + \delta)$, $\gamma(t') \in B_\varepsilon(p)$. Logo,

$$\gamma\left(t + \frac{\delta}{2}\right) \in \big(uv \cap B_\varepsilon(p)\big) \subseteq \partial U$$

e assim $t + \frac{\delta}{2} \in T$, de fato contradizendo $t = \sup T$.

Portanto, somos de fato forçados a conluir que $uv \subseteq \partial U$ ou $uv \setminus \{u, v\} \subseteq U^\circ$. □

**Proposição 41** — Seja $L$ uma lista de polígonos cuja união é $U$ e seja $uv$ um segmento candidato de $L$ tal que $uv \subseteq \partial U$. Então todas as arestas de polígonos em $L$ que contém $uv$ estão orientadas no mesmo sentido, isto é, para cada par $\overrightarrow{u_0v_0}$ e $\overrightarrow{u_1v_1}$ destas arestas,

$$\widetilde{u_0v_0}^{-1}(u) < \widetilde{u_0v_0}^{-1}(v) \iff \widetilde{u_1v_1}^{-1}(u) < \widetilde{u_1v_1}^{-1}(v).$$

**Prova** — Esta demonstração já foi essencialmente feita na prova do lema 7. Porém, vamos refazê-la aqui neste contexto, já com o auxílio deste lema.

Para fins de uma prova por contradição, suponha, sem perda de generalidade, que $\widetilde{u_0v_0}^{-1}(u) < \widetilde{u_0v_0}^{-1}(v)$ e $\widetilde{u_1v_1}^{-1}(u) > \widetilde{u_1v_1}^{-1}(v)$.

Defina então o ponto $p \in \mathbb{R}^2$ por $p = \frac{u+v}{2}$, de forma que $p \in uv \subseteq \partial U$. Como $p \neq u$ e $p \neq v$, devemos ter $p \notin \mathcal{K}(L)$, pois $uv$ é um segmento candidato.

Seja agora $E$ o conjunto de todas as arestas em polígonos de $L$ que contém $p$. Como $p \notin \mathcal{K}(L)$, cada aresta $e \in E$ deve ter uma intersecção infinita com $u_0v_0$ e $u_1v_1$ e, consequentemente, com $uv$.

Seja então $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ um valor real tal que $B_\varepsilon(p)$ não contenha nenhum ponto em $\mathcal{K}(L)$ e não intersecte nenhuma aresta em polígonos de $L$ que não esteja em $E$ (tal $\varepsilon$ existe porque arestas de polígonos são conjuntos fechados).

Sejam ainda $P_0$ e $P_1$ polígonos em $L$ contendo as arestas $u_0v_0$ e $u_1v_1$, respectivamente. Pela lema 7, temos que

- Um ponto $q \in B_\varepsilon(p)$ está em $P_0$ se, e somente se, $(v - u) \times (q - p) \geq 0$, pois $\widetilde{u_0v_0}^{-1}(u) < \widetilde{u_0v_0}^{-1}(v)$; e
- Um ponto $q \in B_\varepsilon(p)$ está em $P_1$ se, e somente se, $(v - u) \times (q - p) \leq 0$, pois $\widetilde{u_1v_1}^{-1}(u) > \widetilde{u_1v_1}^{-1}(v)$.

Mas assim $B_\varepsilon(p) \subseteq (P_0 \cup P_1) \subseteq U$, contradizendo $p \in \partial U$. □

**Definição 60** — Seja $L$ uma lista de polígonos cuja união é $U$ e seja $uv$ um segmento candidato de $L$ contido em $\partial U$. Seja ainda $\overrightarrow{u'v'}$ uma aresta de um polígono em $L$ tal que $uv \subseteq u'v'$. Dizemos então que a ***orientação*** de $uv$ é o segmento orientado $\overrightarrow{uv}$ se $\widetilde{u'v'}^{-1}(u) < \widetilde{u'v'}^{-1}(v)$ e o segmento orientado $\overrightarrow{vu}$ caso contrário.

Observe que a proposição 41 nos garante uma única orientação para um tal segmento candidato, tornando a definição consistente.

Agora finalmente mostramos o principal resultado desta seção.

**Proposição 42** — Seja $L$ uma lista de polígonos cuja união é $U \subseteq \mathbb{R}^2$. Então o conjunto das orientações dos segmentos candidatos de $L$ contidos em $\partial U$ é uma boa representação do conjunto $U$.

**Prova** — Vamos mostrar os seguintes três fatos, que em conjunto mostram que as orientações dos segmentos candidatos contidos em $\partial U$ formam uma boa representação para $U$:

- A orientação de um segmento candidato contido na fronteira de $U$ é sempre compatível com $U$;
- Dois segmentos candidatos nunca se intersectam exceto possivelmente em seus extremos; e
- Todo ponto na fronteira de $U$ está em um segmento candidato contido na fronteira de $U$.

O primeiro fato segue imediatamente do lema 7 e o segundo, da proposição 39.

Para demonstrar o terceiro fato, seja primeiro $p \in \partial U \setminus \mathcal{K}(L)$ um ponto na fronteira de $U$ que não é um ponto chave. Como a fronteira da união de conjuntos deve estar contida na união das fronteiras, devemos ter um polígono $P$ em $L$ com $p \in \partial P$. Como a fronteira de um polígono em $L$ é particionada em segmentos candidatos pelos pontos chave e como $p \notin \mathcal{K}(L)$, deve existir um segmento candidato $uv$ com $p \in uv \setminus \{u, v\}$. Logo, pela proposição 40, devemos ter $uv \subseteq \partial U$, como desejado.

Suponha agora que $p \in \partial U \cap \mathcal{K}(L)$ é um ponto chave na fronteira de $U$. Escolha então um $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ tal que $B_\varepsilon(p) \cap \mathcal{K}(L) = \{p\}$ e tal que $B_\varepsilon(p)$ não intersecte segmentos candidatos que não possuam $p$ como extremo (note que tal $\varepsilon$ existe porque $p$ não pertence a nenhum destes segmentos, que são conjuntos fechados). Observamos que a fronteira da união de polígonos jamais possui pontos isolados. Devemos ter então um ponto $q \in B_\varepsilon(p)$ com $q \in \partial U$. Porém, como $q \in B_\varepsilon(p)$, temos que $q \notin \mathcal{K}(L)$ e, como acabamos de ver, deve existir um segmento candidato contendo $q$ e contido na fronteira de $U$. No entanto, este segmento intersecta $B_\varepsilon(p)$ (ao menos no ponto $q$), e assim deve ter $p$ como um de seus extremos e, consequentemente, como um de seus pontos, também como desejado. $\square$

Com o que vimos até então, é trivial demonstrar que o problema 2 sempre admite uma solução, que é um resultado que estávamos devendo.

**Proposição 43** — O problema 2 sempre admite uma solução.

**Prova** — Imediata da proposição 42. □

## 4.4 Posição geral e cisalhamentos simbólicos

Existem várias complicações técnicas ao se tentar resolver o problema da união de polígonos e, na verdade, vários outros problemas em geometria computacional. Por isso, é comum inicialmente restringir as possíveis entradas do problema a entradas que possuem certas propriedades que entradas aleatórias quase sempre possuem. Entradas deste tipo são ditas em posição geral.

A seguir definimos precisamente o que é uma entrada do problema 2 estar em posição geral.

**Definição 61** — Uma entrada do problema 2 é dita estar em ***posição geral*** se as duas seguintes condições são satisfeitas:

- Dois vértices $u, v \in \mathbb{R}^2$ de polígonos na entrada tais que $u \neq v$ sempre possuem abscissas diferentes; e
- Arestas de polígonos com índices diferentes na lista de entrada nunca se intersectam em um número infinito de pontos.

Note que, em particular, uma entrada em posição geral para o problema 2 não pode possuir polígonos repetidos.

Seria extremamente útil se pudéssemos transformar uma entrada qualquer do problema em uma outra entrada em posição geral, resolver o problema para a segunda entrada e recuperar uma solução para a entrada original. Infelizmente, não sabemos como fazer isso. Porém, uma das duas restrições da posição geral pode ser removida com o método desenvolvido nesta seção, nominalmente a primeira.

Na verdade, em [16], uma técnica chamada de *simulation of simplicity* foi desenvolvida para ajudar a tratar de casos degenerados em problemas geométricos. O método que apresentamos aqui foi desenvolvido independentemente, mas parece ser um caso particular desta técnica.

Nosso método se baseia no fato de que podemos aplicar na entrada uma transformação linear bijetiva chamada de cisalhamento. A entrada resultante jamais terá dois vértices distintos que partilham a mesma abscissa se a transformação for bem escolhida. Ainda, a solução desta nova entrada, ou seja, uma boa representação para a união dos polígonos cisalhados, pode ser transformada pela transformação linear inversa, resultando numa boa representação para a união original.

No restante desta seção descrevemos esta ideia de forma mais formal e mostramos um truque bastante elegante para evitar fazer esta conversão mas ainda sim não precisar nos preocupar com a primeira restrição: o cisalhamento simbólico.

**Definição 62** — Dado um valor real $\delta \in \mathbb{R}$, dizemos que o ***cisalhamento*** por $\delta$ é a transformação linear

$$\begin{array}{rccc} \delta[\cdot]: & \mathbb{R}^2 & \to & \mathbb{R}^2 \\ & (x, y) & \mapsto & (x + \delta y, y). \end{array}$$

**Definição 63** — Se $\delta \in \mathbb{R}$ é um valor real, adotamos as seguintes convenções notacionais:

- Se $p \in \mathbb{R}^2$ é um ponto, denotamos $\delta[\cdot](p)$ por $\delta[p]$;
- Se $X \subseteq \mathbb{R}^2$ é um conjunto de pontos, denotamos o conjunto $\{\delta[x] : x \in X\}$ por $\delta[X]$;
- Se $\mathcal{C}$ é uma família de subconjuntos de $\mathbb{R}^2$, denotamos a família $\{\delta[X] : X \in \mathcal{C}\}$ por $\delta[\mathcal{C}]$;
- Se $L = X_0 X_1 \cdots X_{n-1}$ é uma lista de subconjuntos de $\mathbb{R}^2$, denotamos a lista

  $$\delta[X_0]\,\delta[X_1]\cdots\delta[X_{n-1}]$$

  por $\delta[L]$;
- Se $\overrightarrow{uv}$ é um segmento de reta orientado, denotamos o segmento de reta orientado $\overrightarrow{\delta[u]\,\delta[v]}$ por $\delta[\overrightarrow{uv}]$; e
- Se $\mathcal{R}$ é um conjunto de segmentos de reta orientados, denotamos o conjunto $\{\delta[s] : s \in \mathcal{R}\}$ por $\delta[\mathcal{R}]$.

**Proposição 44** — Seja $uv \subseteq \mathbb{R}^2$ um segmento de reta e seja $\delta \in \mathbb{R}$ um número real. Então $\delta[uv]$ é o segmento de reta $\delta[u]\,\delta[v]$.

**Prova** — Como $uv = \{(1-t)u + tv : t \in [0,1]\}$ e como $\delta[\cdot]$ é uma transformação linear, temos que

$$\begin{aligned} \delta[uv] &= \delta\left[\{(1-t)u + tv : t \in [0,1]\}\right] \\ &= \{\delta[(1-t)u + tv] : t \in [0,1]\} \\ &= \{(1-t)\delta[u] + t\delta[v] : t \in [0,1]\} \\ &= \delta[u]\,\delta[v], \end{aligned}$$

como queríamos. □

**Proposição 45** — Para todo valor real $\delta \in \mathbb{R}$, o cisalhamento por $\delta$ é uma função bijetiva cuja inversa é o cisalhamento por $-\delta$.

**Prova** — Para mostrar que a função $\delta[\cdot]$ é injetiva, sejam $u = (x_0, y_0) \in \mathbb{R}^2$ e $v = (x_1, y_1) \in \mathbb{R}^2$ dois pontos no plano tais que $\delta[u] = \delta[v]$, ou seja, tais que

$$(x_0 + \delta y_0, y_0) = (x_1 + \delta y_1, y_1).$$

Claramente então $y_0 = y_1$ e, como $x_0 + \delta y_0 = x_1 + \delta y_1$, segue que $x_0 = x_1$. Logo $u = v$ e a função deve ser injetiva.

Já para mostrar a sobrejetividade de $\delta[\cdot]$, considere um ponto $(x, y) \in \mathbb{R}^2$ e observe que

$$\delta[-\delta[(x, y)]] = \delta[(x - \delta y, y)] = (x - \delta y + \delta y, y) = (x, y).$$

Com o que vimos também fica claro que $-\delta[\cdot]$ é a função inversa de $\delta[\cdot]$. $\square$

**Lema 8** — Seja $\delta \in \mathbb{R}$ um valor real e sejam $u, v \in \mathbb{R}^2$ dois vetores. Então $\delta[u] \times \delta[v] = u \times v$.

**Prova** — Sejam $u$ e $v$ escritos como $u = (x_0, y_0)$ e $v = (x_1, y_1)$. Então

$$\begin{aligned}
\delta[u] \times \delta[v] &= (x_0 + \delta y_0, y_0) \times (x_1 + \delta y_1, y_1) \\
&= (x_0 + \delta y_0)y_1 - y_0(x_1 + \delta y_1) \\
&= x_0y_1 + \delta y_0y_1 - y_0x_1 - \delta y_0y_1 \\
&= x_0y_1 - y_0x_1 \\
&= u \times v,
\end{aligned}$$

como desejado. $\square$

**Lema 9** — Para todos os valores reais $\delta, \varepsilon_1 \in \mathbb{R}$ com $\varepsilon_1 > 0$, existe um valor real $\varepsilon_0 \in \mathbb{R}$ com $\varepsilon_0 > 0$ tal que, para todo ponto $p \in \mathbb{R}^2$, $\delta[B_{\varepsilon_0}(p)] \subseteq B_{\varepsilon_1}\big(\delta[p]\big)$.

**Prova** — Mostramos que escolher

$$\varepsilon_0 = \frac{\varepsilon_1}{(|\delta| + 1)\sqrt{2}}$$

é suficiente. Para isso escreva $p = (x_0, y_0)$ e considere um ponto $q = (x_1, y_1) \in \delta[B_{\varepsilon_0}(p)]$ qualquer. Devemos mostrar que $q \in B_{\varepsilon_1}\big(\delta[p]\big)$.

Como $q \in \delta[B_{\varepsilon_0}(p)]$ e como a função $-\delta[\cdot]$ é a função inversa de $\delta[\cdot]$, temos que $-\delta[q] \in B_{\varepsilon_0}(p)$, ou seja, $(x_1 - \delta y_1, y_1) \in B_{\varepsilon_0}(p)$, de modo que $|x_1 - \delta y_1 - x_0| < \varepsilon_0$ e $|y_1 - y_0| < \varepsilon_0$. Logo,

$$\begin{aligned}
\varepsilon_0 &> |x_1 - \delta y_1 - x_0| \\
&= |x_1 - \delta y_1 + \delta y_0 - \delta y_0 - x_0| \\
&= |x_1 - \delta(y_1 - y_0) - \delta y_0 - x_0| \\
&= |x_1 - (x_0 + \delta y_0) - \delta(y_1 - y_0)| \\
&\geq |x_1 - (x_0 + \delta y_0)| - |\delta(y_1 - y_0)| \\
&= |x_1 - (x_0 + \delta y_0)| - |\delta||y_1 - y_0| \\
&\geq |x_1 - (x_0 + \delta y_0)| - |\delta|\varepsilon_0,
\end{aligned}$$

de forma que

$$|x_1 - (x_0 + \delta y_0)| < (|\delta| + 1)\varepsilon_0 = (|\delta| + 1)\frac{\varepsilon_1}{(|\delta| + 1)\sqrt{2}} = \frac{\varepsilon_1}{\sqrt{2}}.$$

Ainda, note que $\varepsilon_0 < {}^{\varepsilon_1}/\sqrt{2}$, de forma que $|y_1 - y_0| < \varepsilon_0 < {}^{\varepsilon_1}/\sqrt{2}$. Portanto,

$$\begin{aligned}
||q - \delta[p]|| &= ||(x_1, y_1) - (x_0 + \delta y_0, y_0)|| \\
&= \sqrt{\big(x_1 - (x_0 + \delta y_0)\big)^2 + (y_1 - y_0)^2} \\
&< \sqrt{\left(\frac{\varepsilon_1}{\sqrt{2}}\right)^2 + \left(\frac{\varepsilon_1}{\sqrt{2}}\right)^2} \\
&= \sqrt{\frac{\varepsilon_1^2}{2} + \frac{\varepsilon_1^2}{2}} \\
&= \sqrt{\varepsilon_1^2} \\
&= \varepsilon_1
\end{aligned}$$

e assim temos que $q \in B_{\varepsilon_1}(p)$. □

**Proposição 46** — Seja $P \subseteq \mathbb{R}^2$ um polígono e seja $\delta \in \mathbb{R}$ um valor real. Então $\delta[P]$ é um polígono tal que $V\big(\delta[P]\big) = \delta[V(P)]$ e

$$E\big(\delta[P]\big) = \left\{\overrightarrow{\delta[u]\,\delta[v]} : \overrightarrow{uv} \in E(P)\right\}.$$

**Prova** — Seja $P$ definido em termos de uma curva de Jordan poligonal $\gamma : [0,1] \to \mathbb{R}^2$, de forma que $P = \mathrm{Im}(\gamma) \cup \gamma^\circ$. Seja então

$$\begin{array}{rccc}
\gamma' : & [0,1] & \to & \mathbb{R}^2 \\
& t & \mapsto & \delta[\gamma(t)]
\end{array}$$

a composição da transformação linear bijetiva $\delta[\cdot]$ com a curva $\gamma$. Pela injetividade de $\delta[\cdot]$ e $\gamma|_{[0,1)}$, a função $\gamma'|_{[0,1)}$ é injetiva e, como $\gamma(0) = \gamma(1)$, vale que $\gamma'(0) = \gamma'(1)$. Mais ainda, pela continuidade de $\delta[\cdot]$, tanto $\delta[\gamma^\circ]$ quanto $\delta[\mathbb{R}^2 \setminus \mathrm{Im}(\gamma) \setminus \gamma^\circ]$ são conjuntos conexos, sendo que $\delta[\gamma^\circ]$ é também limitado. Logo, como $\mathbb{R}^2 = \delta[\gamma^\circ] \cup \mathrm{Im}(\gamma') \cup \delta[\mathbb{R}^2 \setminus \mathrm{Im}(\gamma) \setminus \gamma^\circ]$, segue que $\mathbb{R}^2 \setminus \mathrm{Im}(\gamma')$ possui exatamente duas componentes conexas, exatamente uma das quais é limitada. Segue portanto que $\gamma'$ é uma curva de Jordan (com $\gamma'^\circ = \delta[\gamma^\circ]$) e é poligonal (pois $\delta[\cdot]$ é injetiva). Ainda, claramente $\delta[P] = \mathrm{Im}(\gamma') \cup \gamma'^\circ$, de forma que $\delta[P]$ é um polígono.

Devemos mostrar também que as extensões periódicas de $\gamma$ e $\gamma'$ são diferenciáveis exatamente nos mesmos parâmetros. Para isso, observe que a imagem por $\delta[\cdot]$ de um segmento de reta é ainda um segmento de reta. Logo, se $a, b, c \in \mathbb{R}^2$ são pontos distintos, então eles são colineares se, e somente se, $\delta[a]$, $\delta[b]$ e $\delta[c]$ o são. Porém, uma curva poligonal estendida periodicamente é diferenciável em um parâmetro $t \in \mathbb{R}$ se, e somente se, as imagens dos parâmetros em uma vizinhança de $t$ são todas colineares.

Resta mostrar ainda que os sentidos das arestas são preservados. Para isso, recomendamos ao leitor que a definição 33 seja revisitada. Assumimos, sem perda de generalidade, que $\gamma$ está em sentido anti-horário. Devemos então mostrar que $\gamma'$ também está. Para isso, considere o ponto $p \in \mathbb{R}^2$ de maior abscissa dentre os pontos em $\mathrm{Im}(\gamma)$ de menor ordenada. Observe então que $\delta[p]$ é o ponto de maior abscissa dentre os pontos em $\mathrm{Im}(\gamma')$ de menor ordenada, pois $\delta[\cdot]$ preserva as ordenadas dos pontos e a ordem entre as abscissas.

Seja $t$ o único valor em $[0,1)$ tal que $\gamma(t) = p$. Como $\gamma$ está em sentido anti-horário, existe $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ tal que, para todos $u, v \in \mathbb{R}$ com $t - \varepsilon < u < t < v < t + \varepsilon$,

$$\big(\gamma(v) - p\big) \times \big(\gamma(u) - p\big) > 0.$$

Mas, pelo lema 8,

$$\delta\,[\gamma(v) - p] \times \delta\,[\gamma(u) - p] = \big(\gamma(v) - p\big) \times \big(\gamma(u) - p\big)$$

e logo

$$\big(\delta\,[\gamma(v)] - \delta\,[p]\big) \times \big(\delta\,[\gamma(u)] - \delta\,[p]\big) > 0,$$

ou seja

$$\big(\gamma'(v) - \delta\,[p]\big) \times \big(\gamma'(u) - \delta\,[p]\big) > 0.$$

Logo, $\gamma'$ também está em sentido anti-horário, concluindo a demonstração. $\square$

**Proposição 47** — Seja $X \subseteq \mathbb{R}^2$ um conjunto de pontos, seja $\mathcal{R}$ um conjunto finito de segmentos de reta orientados e seja $\delta \in \mathbb{R}$ um valor real. Então $\mathcal{R}$ é uma boa representação para $X$ se, e somente se, $\delta\,[\mathcal{R}]$ é uma boa representação para $\delta\,[X]$.

**Prova** — Primeiramente mostramos que se $\mathcal{R}$ é uma boa representação para $X$, então $\delta\,[\mathcal{R}]$ é uma boa representação para $\delta\,[X]$. Para isso devemos mostrar que quatro fatos são verdadeiros:

- Todo segmento de reta orientado em $\delta\,[\mathcal{R}]$ é compatível com $\delta\,[X]$;
- Dois segmentos de reta em $\delta\,[\mathcal{R}]$ jamais se intersectam, exceto possivelmente em seus extremos;
- Todo segmento de reta em $\delta\,[\mathcal{R}]$ está contido em $\partial\delta\,[X]$; e
- Todo ponto em $\partial\delta\,[X]$ está em um segmento de reta em $\delta\,[\mathcal{R}]$.

Para mostrar o primeiro fato, seja $\overrightarrow{\delta\,[u]\,\delta\,[v]}$ um segmento de reta orientado em $\delta\,[\mathcal{R}]$, onde $\overrightarrow{uv} \in \mathcal{R}$ e considere um ponto $q \in \delta\,[u]\,\delta\,[v] \backslash \{\delta\,[u]\,, \delta\,[v]\}$. Assim, pelas proposições 44 e 45, temos que o ponto $p = -\delta\,[q]$ é tal que $\delta\,[p] = q$ e $p \in uv \setminus \{u, v\}$. Logo, como $\mathcal{R}$ é uma boa representação de $X$, temos que $uv$ é compatível com $X$ e, como $p \in uv \setminus \{u, v\}$, temos que existe um $\varepsilon_0 \in \mathbb{R}$ com $\varepsilon_0 > 0$ tal que, para todo $x \in B_{\varepsilon_0}(p)$, vale que $x \in X$ se $(v - u) \times (x - p) > 0$ e $x \notin X$ se $(v - u) \times (x - p) < 0$.

Pelo lema 9, temos então que existe um valor $\varepsilon_1 \in \mathbb{R}$ com $\varepsilon_1 > 0$ tal que

$$-\delta\,[B_{\varepsilon_1}(q)] \subseteq B_{\varepsilon_0}\big(-\delta\,[q]\big) = B_{\varepsilon_0}(p).$$

Considere então um ponto $y \in B_{\varepsilon_1}(q)$. Temos então, que $x = -\delta\,[y] \in B_{\varepsilon_0}(p)$ e, pelo lema 8, que

$$\begin{aligned}(\delta\,[v] - \delta\,[u]) \times (y - q) &= (\delta\,[v] - \delta\,[u]) \times (\delta\,[x] - \delta\,[p]) \\ &= \delta\,[v - u] \times \delta\,[x - p] \\ &= (v - u) \times (x - p).\end{aligned}$$

Deste modo, se $(\delta[v] - \delta[u]) \times (y - q) > 0$, então $(v - u) \times (x - p) > 0$ e, como $x \in B_{\varepsilon_0}(p)$, devemos ter $x \in X$. Mas então $y = \delta[x] \in \delta[X]$. Similarmente, se $(\delta[v] - \delta[u]) \times (y - q) < 0$, então $(v - u) \times (x - p) < 0$ e, como $x \in B_{\varepsilon_0}(p)$, devemos ter $x \notin X$ e, logo, $y = \delta[x] \notin \delta[X]$.

Desta forma, para todo $y \in B_{\varepsilon_1}(q)$, temos que $y \in \delta[X]$ se $(\delta[v] - \delta[u]) \times (y - q) > 0$ e $y \notin \delta[X]$ se $(\delta[v] - \delta[u]) \times (y - q) < 0$. Como nossas escolhas de $q$ e $y$ foram arbitrárias, temos então que o segmento $\overrightarrow{\delta[u]\,\delta[v]}$ é de fato compatível com $\delta[X]$.

Como $\delta[\cdot]$ é uma função contínua, temos que $\partial\delta[X] = \delta[\partial X]$. Assim, o segundo, o terceiro e o quarto fato seguem imediatamente da proposição 44. Note, contudo, que para mostrar estes três fatos fazemos forte uso de que $\mathcal{R}$ é uma boa representação para $X$.

Até aqui mostramos que se $\mathcal{R}$ é uma boa representação para $X$, então $\delta[\mathcal{R}]$ é uma boa representação para $\delta[X]$. Para mostrar a recíproca, simplesmente observe que $\mathcal{R} = -\delta[\delta[\mathcal{R}]]$ e que $X = -\delta[\delta[X]]$ e assim, na verdade, já demonstramos isto. □

Considere agora os seguintes resultados.

> **Proposição 48** — Seja $X \subseteq \mathbb{R}^2$ um conjunto finito de pontos no plano. Então existe um valor real positivo $\delta_0 \in \mathbb{R}$ tal que, para todo valor real $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$ e todo par de pontos $(x_0, y_0), (x_1, y_1) \in X$ no conjunto $X$, a abscissa de $\delta[(x_0, y_0)]$ será menor do que a abscissa de $\delta[(x_1, y_1)]$ se, e somente se, $x_0 < x_1 \vee (x_0 = x_1 \wedge y_0 < y_1)$.

**Prova** — Seja $R \subseteq \mathbb{R}$ o conjunto

$$\bigcup_{\substack{(x_0,y_0)\in X\\(x_1,y_1)\in X\\y_0\neq y_1}} \left\{\frac{x_0 - x_1}{y_1 - y_0}\right\}$$

e seja

$$\delta_0 = \min\bigl(\{1\} \cup \{r \in R : r > 0\}\bigr).$$

Considere então um valor real $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$ e dois pontos $(x_0, y_0), (x_1, y_1) \in X$.

Se a abscissa de $\delta[(x_0, y_0)]$ é menor do que a abscissa de $\delta[(x_1, y_1)]$, ou seja, se $x_0 + \delta y_0 < x_1 + \delta y_1$, vamos mostrar que $x_0 < x_1$ ou $x_0 = x_1$ e $y_0 < y_1$. De fato, se $x_0 > x_1$, então note que devemos ter $y_1 > y_0$, pois $\delta > 0$ e $\delta(y_1 - y_0) > x_0 - x_1$. Assim,

$$\delta > \frac{x_0 - x_1}{y_1 - y_0} > 0,$$

contradizendo $\delta$ ser menor do que todo elemento positivo de $R$. Por outro lado, se $x_0 = x_1$, então $\delta(y_1 - y_0) > x_0 - x_1 = 0$ e, como $\delta > 0$, devemos ter $y_0 < y_1$, como queríamos.

Conversamente, vamos mostrar que se $x_0 < x_1$ ou se $x_0 = x_1$ e $y_0 < y_1$, então a abscissa de $\delta[(x_0, y_0)]$ é menor do que a abscissa de $\delta[(x_1, y_1)]$, ou seja, $x_0 + \delta y_0 < x_1 + \delta y_1$. De fato, se $x_0 = x_1$ e $y_0 < y_1$, então, como $\delta > 0$, claramente $x_0 + \delta y_0 < x_1 + \delta y_1$. Por outro lado, se $x_0 < x_1$, há dois casos. Se $y_0 > y_1$, então devemos ter

$$0 < \delta < \frac{x_0 - x_1}{y_1 - y_0},$$

pois $\delta$ é menor do que todo elemento positivo de $R$. Assim, como $y_1 - y_0 < 0$, temos que $\delta(y_1 - y_0) > x_0 - x_1$ e assim que $x_0 + \delta y_0 < x_1 + \delta y_1$. Finalmente, se $x_0 < x_1$ e $y_0 \leq y_1$, então, como $\delta > 0$, obviamente $x_0 + \delta y_0 < x_1 + \delta y_1$, concluindo a demonstração. $\square$

**Corolário 8** — Seja $X \subseteq \mathbb{R}^2$ um conjunto finito de pontos no plano. Então existe um valor real positivo $\delta_0 \in \mathbb{R}$ tal que, para todo valor real $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$, o conjunto $\delta[X]$ não possui pontos distintos de mesma abscissa.

**Prova** — Sejam $(x_0, y_0), (x_1, y_1) \in X$ dois pontos distintos no conjunto $X$. Se as duas condições $x_0 < x_1 \vee (x_0 = x_1 \wedge y_0 < y_1)$ e $x_1 < x_0 \vee (x_1 = x_0 \wedge y_1 < y_0)$ fossem simultaneamente insatisfeitas, então certamente teríamos $(x_0, y_0) = (x_1, y_1)$ (verifique), o que seria uma contradição.

Assim, tomando $\delta_0$ como na proposição 48 e considerando um $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$, por esta mesma proposição e como $(x_0, y_0)$ e $(x_1, y_1)$ são distintos, ou a abscissa de $\delta[(x_0, y_0)]$ é menor do que a abscissa de $\delta[(x_1, y_1)]$, ou a abscissa de $\delta[(x_1, y_1)]$ é menor do que a abscissa de $\delta[(x_0, y_0)]$. $\square$

Estes resultados nos inspiram a definir uma ordem sobre os pontos do plano: um ponto precede o outro se, para um cisalhamento positivo suficientemente pequeno, a imagem do primeiro ponto possui abscissa menor do que a do segundo ponto.

**Definição 64** — Dizemos que um ponto $(x_0, y_0) \in \mathbb{R}^2$ ***precede*** um ponto $(x_1, y_1) \in \mathbb{R}^2$ se $x_0 < x_1 \vee (x_0 = x_1 \wedge y_0 < y_1)$, e denotamos isto por $(x_0, y_0) < (x_1, y_1)$. Também definimos as relações $\leq$, $>$ e $\geq$ sobre pontos do plano da forma padrão.

**Proposição 49** — A ordem de precedência de pontos em $\mathbb{R}^2$ é uma ordem total sobre os pontos de $\mathbb{R}^2$.

**Prova** — Primeiro mostramos que a relação de precedência é uma ordem parcial demonstrando para ela as duas propriedades de ordens parciais:

- Irreflexividade: um ponto $p = (x_0, y_0) \in \mathbb{R}^2$ nunca é tal que $p < p$, pois obviamente $x_0 = x_0$ mas $y_0$ não é menor do que $y_0$; e
- Transitividade: sejam $p_0 = (x_0, y_0), p_1 = (x_1, y_1), p_2 = (x_2, y_2) \in \mathbb{R}^2$ pontos no plano tais que $p_0 < p_1$ e $p_1 < p_2$. Devemos demonstrar que $p_0 < p_2$, ou seja, que $x_0 < x_2$ ou $x_0 = x_2$ e $y_0 < y_2$. Como $p_0 < p_1$ e $p_1 < p_2$, claramente temos $x_0 \leq x_1 \leq x_2$. Logo, se $x_0 < x_1$ ou $x_1 < x_2$ temos $x_0 < x_2$. Já se $x_0 = x_1 = x_2$, como $p_0 < p_1$ e $p_1 < p_2$, devemos ter $y_0 < y_1 < y_2$.

Para verificar que $<$ é uma ordem total sobre os pontos em $\mathbb{R}^2$, sejam $(x_0, y_0), (x_1, y_1) \in \mathbb{R}^2$ dois pontos distintos no plano. Se $x_0 < x_1$ ou se $x_0 > x_1$, estes dois pontos são claramente comparáveis. Já se $x_0 = x_1$, então ou $y_0 < y_1$ ou $y_0 > y_1$, pois $(x_0, y_0) \neq (x_1, y_1)$, de forma que estes pontos são ainda comparáveis. Assim, dois pontos distintos em $\mathbb{R}^2$ são sempre comparáveis por $<$, fazendo de $<$ uma ordem total. $\square$

**Proposição 50** — Seja $X \subseteq \mathbb{R}^2$ um conjunto finito de pontos que pode ser escrito como $\{p_0, p_1, \ldots, p_{n-1}\}$ com $p_0 < p_1 < \cdots < p_{n-1}$. Então existe um valor real positivo $\delta_0 \in \mathbb{R}$ tal que, para todo valor real $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$, as abscissas de $\delta[p_0], \delta[p_1], \ldots, \delta[p_{n-1}]$ estão em ordem (estritamente) crescente.

**Prova** — Segue imediatamente da proposição 48, do corolário 8 e da proposição 49. □

Perceba que, com estes resultados, podemos ordenar os pontos em um conjunto finito de pontos do plano por abscissa assumindo um cisalhamento positivo suficientemente pequeno sendo aplicado aos pontos. Note, no entanto, que em momento algum escolhemos um número real positivo para determinar o cisalhamento e em momento algum efetivamente computamos o cisalhamento ou mesmo fazemos referência a ele. Isso é o primeiro passo para atingir o que chamamos de cisalhamento simbólico, onde operamos na imagem de um conjunto (finito) de pontos por um cisalhamento suficientemente pequeno sem ter que computar ou determinar este cisalhamento.

Como veremos adiante, retas verticais têm um papel importante em nosso algoritmo para o problema da união de polígonos. O que mostramos a seguir nos possibilitará tratar de retas verticais na imagem de um cisalhamento simbólico sem se referir a ele. A chave é perceber que uma tal reta vertical é determinada não mais por uma abscissa mas por um ponto.

**Definição 65** — Denotamos a reta vertical que passa por um ponto $p \in \mathbb{R}^2$ por $VL(p)$.

**Definição 66** — Seja $uv$ um segmento de reta e seja $p \in \mathbb{R}^2$ um ponto. Dizemos que $uv$ ***passa*** por $p$ se $u \leq p \leq v$ ou se $v \leq p \leq u$.

**Proposição 51** — Seja $uv$ um segmento de reta e seja $p \in \mathbb{R}^2$ um ponto. Então existe um valor real positivo $\delta_0 \in \mathbb{R}$ tal que, para todo valor real $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$, a reta $VL(\delta[p])$ intersecta o segmento $\delta[uv]$ se, e somente se, o segmento $uv$ passa pelo ponto $p$.

**Prova** — Seja $X = \{u, p, v\} \subseteq \mathbb{R}^2$ um conjunto de pontos. Pela proposição 50, temos que existe um valor real $\delta_0 \in \mathbb{R}$ com $\delta_0 > 0$ tal que, para todo $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$, a abscissa de um ponto $\delta[a]$ com $a \in X$ é menor ou igual do que a abscissa de um ponto $\delta[b]$ com $b \in X$ se, e somente se, $a \leq b$ (verifique porque o resultado vale para a relação $\leq$ também).

Porém, a reta $VL(\delta[p])$ intersecta o segmento de reta $\delta[u]\,\delta[v] = \delta[uv]$ se, e somente se:

- A abscissa de $\delta[u]$ for menor ou igual a abscissa de $\delta[p]$ e a abscissa de $\delta[p]$ for menor ou igual a abscissa do ponto $\delta[v]$; ou
- A abscissa de $\delta[v]$ for menor ou igual a abscissa de $\delta[p]$ e a abscissa de $\delta[p]$ for menor ou igual a abscissa do ponto $\delta[u]$.

Como $X = \{u, p, v\}$, isto na verdade é equivalente a $u \leq p \leq v$ ou $v \leq p \leq u$, que por sua vez é a definição de $uv$ passar por $p$. $\square$

**Definição 67** — Seja $uv$ um segmento de reta que passa por um ponto $p \in \mathbb{R}^2$. Dizemos que a ***altura*** do segmento $uv$ na reta $VL(p)$ através de um cisalhamento simbólico é a ordenada do único ponto de mesma abscissa que $p$ que intersecta $uv$ se $u$ e $v$ possuem abscissas diferentes e a ordenada do próprio ponto $p$ se $u$ e $v$ possuem as mesmas abscissas. Esta altura será denotada por $\mathrm{height}_p(uv)$.

**Proposição 52** — Seja $uv$ um segmento de reta que passa por um ponto $p \in \mathbb{R}^2$. Então existe um valor real positivo $\delta_0 \in \mathbb{R}$ tal que, para todo valor real $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$, o segmento $\delta[uv]$ intersecta a reta $VL(\delta[p])$ em exatamente um ponto. Ainda, se, para cada $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$, denotarmos por $y_\delta$ a ordenada do único ponto em $\delta[uv] \cap VL(\delta[p])$, então

$$\lim_{\delta \to 0} y_\delta = \mathrm{height}_p(uv)$$

e o valor $\mathrm{sign}\big(y_\delta - \mathrm{height}_p(uv)\big)$ é o mesmo para todo $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$.

**Prova** — Primeiramente, podemos assumir sem perda de generalidade que $u$ precede $v$, ou seja, $u < v$, pois, pela proposição 49, $<$ é uma ordem total sobre os pontos de $\mathbb{R}^2$.

Em conjunto, as proposições 51 e 48 (tome $X = \{u, p, v\}$ para a proposição 48 e considere o menor valor de $\delta_0$ obtido pelas duas proposições) nos garantem a existência de um $\delta_0 \in \mathbb{R}$ com $\delta_0 > 0$ tal que, para todo $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$, o segmento $\delta[u]\,\delta[v]$ intersecta a reta vertical $VL(\delta[p])$ e a abscissa de $\delta[u]$ é menor do que a abscissa de $\delta[v]$, pois $u \leq p \leq v$, já que $uv$ passa por $p$ e $u < v$. Na verdade, sendo a abscissa de $\delta[u]$ menor do que a abscissa de $\delta[v]$, a intersecção $\delta[u]\,\delta[v] \cap VL(\delta[p])$ deve possuir exatamente um ponto, pois o segmento $\delta[u]\,\delta[v]$ não pode ser paralelo à reta vertical $VL(\delta[p])$.

Se $u$ e $v$ possuem a mesma abscissa, então $u$, $v$ e $p$ possuem a mesma abscissa e, como $u \leq p \leq v$, temos que $p \in uv$. Logo, $\delta[p] \in \delta[uv] = \delta[u]\,\delta[v]$, e, como obviamente $\delta[p] \in VL(\delta[p])$, devemos ter $\delta[u]\,\delta[v] \cap VL(\delta[p]) = \big\{\delta[p]\big\}$. Porém, a ordenada de $\delta[p]$ é a própria ordenada de $p$, que neste caso é o valor $\mathrm{height}_p(uv)$, pois $u$ e $v$ têm a mesma abscissa, de forma que o resultado segue trivialmente.

Já se $u$ e $v$ possuem abscissas diferentes, então o segmento $uv$ intersecta a reta vertical $VL(p)$ em exatamente um ponto $q$, cuja ordenada é $\mathrm{height}_p(uv)$. Escreva então $u = (u_x, u_y)$, $v = (v_x, v_y)$, $p = (p_x, p_y)$ e $q = (q_x, q_y)$. Como $u \leq p \leq v$ e $q \in VL(p)$, devemos ter $u_x \leq p_x = q_x \leq v_x$. Como $q \in uv \cap VL(p)$ e $uv = \big\{(1-t)u + tv : t \in [0,1]\big\}$, devemos ter um $t \in [0,1]$ com $(1-t)u + tv = q$ e $(1-t)u_x + tv_x = p_x$. Assim $(v_x - u_x)t = p_x - u_x$ e, como $u_x < v_x$ já que $u$ e $v$ possuem abscissas diferentes e $u < v$, $t = \dfrac{p_x - u_x}{v_x - u_x}$, de forma que

$$
\begin{aligned}
\operatorname{height}_p(uv) &= q_y \\
&= (1-t)u_y + tv_y \\
&= u_y + t(v_y - u_y) \\
&= u_y + \frac{p_x - u_x}{v_x - u_x}(v_y - u_y).
\end{aligned}
$$

De uma maneira similar, podemos também calcular a ordenada $y_\delta$ da intersecção de $\delta[u]\,\delta[v]$ com $VL(\delta[p])$. Observamos que $\delta[u]\,\delta[v] = \big\{(1-t)\delta[u] + t\delta[v] : t \in [0,1]\big\}$ e que existe um único $t \in [0,1]$ tal que $(1-t)\delta[u] + t\delta[v] \in VL(\delta[p])$, ou seja, a abscissa de $(1-t)\delta[u] + t\delta[v]$ é $p_x + \delta p_y$, a abscissa de $\delta[p]$. Temos então

$$(1-t)(u_x + \delta u_y) + t(v_x + \delta v_y) = p_x + \delta p_y,$$

de forma que $\big((v_x + \delta v_y) - (u_x + \delta u_y)\big)t = (p_x + \delta p_y) - (u_x + \delta u_y)$. Assim, como a abscissa de $\delta[u]$ é menor do que a abscissa de $\delta[v]$, devemos ter $u_x + \delta u_y < v_x + \delta v_y$. Portanto,

$$t = \frac{(p_x + \delta p_y) - (u_x + \delta u_y)}{(v_x + \delta v_y) - (u_x + \delta u_y)} = \frac{(p_x - u_x) + \delta(p_y - u_y)}{(v_x - u_x) + \delta(v_y - u_y)}$$

e a ordenada da intersecção é

$$y_\delta = (1-t)u_y + tv_y = u_y + t(v_y - u_y) = u_y + \frac{(p_x - u_x) + \delta(p_y - u_y)}{(v_x - u_x) + \delta(v_y - u_y)}(v_y - u_y).$$

Logo,

$$
\begin{aligned}
& y_\delta - \operatorname{height}_p(uv) \\
={}& u_y + \frac{(p_x - u_x) + \delta(p_y - u_y)}{(v_x - u_x) + \delta(v_y - u_y)}(v_y - u_y) - u_y - \frac{p_x - u_x}{v_x - u_x}(v_y - u_y) \\
={}& \left(\frac{(p_x - u_x) + \delta(p_y - u_y)}{(v_x - u_x) + \delta(v_y - u_y)} - \frac{p_x - u_x}{v_x - u_x}\right)(v_y - u_y) \\
={}& \frac{(p_x - u_x)(v_x - u_x) + \delta(p_y - u_y)(v_x - u_x) - (v_x - u_x)(p_x - u_x) - \delta(v_y - u_y)(p_x - u_x)}{\big((v_x - u_x) + \delta(v_y - u_y)\big)(v_x - u_x)}(v_y - u_y) \\
={}& \frac{\delta(p_y - u_y)(v_x - u_x) - \delta(v_y - u_y)(p_x - u_x)}{\big((v_x - u_x) + \delta(v_y - u_y)\big)(v_x - u_x)}(v_y - u_y) \\
={}& \delta\left(\frac{(p_y - u_y)(v_x - u_x) - (v_y - u_y)(p_x - u_x)}{(v_x - u_x)^2 + \delta(v_y - u_y)(v_x - u_x)}\right)(v_y - u_y).
\end{aligned}
$$

Então, como $u_x < v_x$, claramente

$$\lim_{\delta \to 0}\big(y_\delta - \operatorname{height}_p(uv)\big) = 0,$$

de forma que

$$\lim_{\delta \to 0} y_\delta = \operatorname{height}_p(uv).$$

Ainda, se impusermos a restrição adicional de que

$$|v_y - u_y||v_x - u_x|\delta_0 \le (v_x - u_x)^2,$$

teremos que

$$(v_x - u_x)^2 + \delta(v_y - u_y)(v_x - u_x) > 0$$

e, como $\delta > 0$, que

$$\operatorname{sign}\big(y_\delta - \operatorname{height}_p(uv)\big) = \operatorname{sign}\Big(\big((p_y - u_y)(v_x - u_x) - (v_y - u_y)(p_x - u_x)\big)(v_y - u_y)\Big),$$

que independe da escolha de $\delta \in \mathbb{R}$ com $0 < \delta < \delta_0$. $\square$

## 4.5 Ideia do algoritmo

Nesta seção descrevemos os dois principais conceitos por trás de nosso algoritmo para computar a união de polígonos: como decidir se um segmento candidato está contido na fronteira da união e como obter todos os segmentos candidatos.

O primeiro passo é separar as arestas dos polígonos em dois tipos: arestas cujo início precede o fim e arestas cujo fim precede o início.

**Definição 68** — Seja $e = \overrightarrow{uv}$ uma aresta de um polígono. Definimos o ***extremo esquerdo*** de $e$ como

$$\operatorname{left}(e) = \begin{cases} u, & u < v \\ v, & u > v. \end{cases}$$

Similarmente, definimos o ***extremo direito*** de $e$ como

$$\operatorname{right}(e) = \begin{cases} v, & u < v \\ u, & u > v. \end{cases}$$

**Definição 69** — Seja $e = \overrightarrow{uv}$ uma aresta de um polígono. Dizemos que $e$ é uma ***aresta de ida*** se $\operatorname{left}(e) = u$ e que $e$ é uma ***aresta de volta*** se $\operatorname{left}(e) = v$.

**Definição 70** — Definimos a ***direção*** de uma aresta $e$ de um polígono como o símbolo $\rightarrow$ se $e$ é uma aresta de ida e como o símbolo $\leftarrow$ se $e$ é uma aresta de volta. Ainda, denotamos esta direção por $\operatorname{dir}(e)$, de forma que $\operatorname{dir}(e) \in \{\rightarrow, \leftarrow\}$.

O segundo passo é definir, para cada ponto no plano, uma ordem total sobre um subconjunto das arestas de polígonos em uma lista de polígonos.

**Definição 71** — Dizemos que um segmento de reta $uv$ ou uma aresta de um polígono correspondente a tal segmento é ***comparável*** em um ponto $p \in \mathbb{R}^2$ se $u < p \le v$ ou se $v < p \le u$.

**Definição 72** — Seja $L$ uma lista de polígonos, seja $p \in \mathbb{R}^2$ um ponto no plano, sejam $e_0$ e $e_1$ duas arestas de polígonos em $L$ comparáveis em $p$ e sejam $i, j \in \mathbb{N}$ os índices dos polígonos na lista $L$ que possuem as arestas $e_0$ e $e_1$, respectivamente. Sejam ainda $u = \mathrm{right}(e_0) - \mathrm{left}(e_0) \in \mathbb{R}^2$ e $v = \mathrm{right}(e_1) - \mathrm{left}(e_1) \in \mathbb{R}^2$. Dizemos então que $e_0$ ***precede*** $e_1$ em $L$ no ponto $p$ se alguma das seguintes condições for satisfeita:

- $\mathrm{height}_p(e_0) < \mathrm{height}_p(e_1)$;
- $\mathrm{height}_p(e_0) = \mathrm{height}_p(e_1)$ e $u \times v < 0$;
- $\mathrm{height}_p(e_0) = \mathrm{height}_p(e_1)$, $u \times v = 0$ e $\big(\mathrm{dir}(e_0), \mathrm{dir}(e_1)\big) = (\rightarrow, \leftarrow)$; ou
- $\mathrm{height}_p(e_0) = \mathrm{height}_p(e_1)$, $u \times v = 0$, $\mathrm{dir}(e_0) = \mathrm{dir}(e_1)$ e $i < j$.

Denotamos que $e_0$ precede $e_1$ em $L$ no ponto $p$ por $e_0 <_p^L e_1$.

Tentamos agora providenciar uma noção intuitiva do que significa uma aresta $e_0$ preceder uma aresta $e_1$ em $L$ no ponto $p$. O primeiro critério é a altura onde as arestas cruzam a reta vertical contendo $p$ (lembrando que se $e_0$ ou $e_1$ forem verticais esta altura é a ordenada de $p$). Se houver empate, o critério utilizado é o comportamento em uma vizinhança do cruzamento com a reta vertical, sendo que a aresta que "vinha mais de baixo" precede a aresta que "vinha mais de cima". Se novamente houver empate, os próximos dois critérios são técnicos e servem para auxilar os resultados subsequentes. O primeiro critério remanescente é a direção da aresta e o segundo o identificador do polígono de origem. Claramente, se houver empate em todos estes critérios, $e_0$ e $e_1$ são a mesma aresta.

**Proposição 53** — Para toda lista $L$ de polígonos e todo ponto $p \in \mathbb{R}^2$, a ordem $<_p^L$ é uma ordem total sobre as arestas de polígonos em $L$ que são comparáveis em $p$.

**Prova** — Elementar. □

**Definição 73** — Para toda lista $L$ de polígonos e todo todo ponto $p \in \mathbb{R}^2$, definimos os operadores $>_p^L$, $\leq_p^L$, $\geq_p^L$ e $=_p^L$ da forma convencional, isto é, para todas as arestas de polígonos $e_0$ e $e_1$ comparáveis em $p$:

- $e_0 >_p^L e_1 \iff e_1 <_p^L e_0$;
- $e_0 \leq_p^L e_1 \iff \neg(e_1 <_p^L e_0)$;
- $e_0 \geq_p^L e_1 \iff \neg(e_0 <_p^L e_1)$; e
- $e_0 =_p^L e_1 \iff \neg(e_0 <_p^L e_1) \land \neg(e_1 <_p^L e_0)$.

É importante observar que duas arestas $e_0$ e $e_1$ de dois polígonos de índices diferentes na lista $L$, quando ambas comparáveis em um ponto $p$, sempre ocorrerá ou que $e_0 <_p^L e_1$ ou que $e_1 <_p^L e_0$, mesmo quando estas arestas representam o mesmo segmento de reta no plano, quando elas possuem a mesma orientação e quando seus polígonos de origem representam o mesmo conjunto de pontos no plano.

A motivação para incluir o produto cruzado $u \times v$ na definição da ordem $<_p^L$ é que desejamos, como veremos bem mais adiante, que esta ordem se mantenha a mesma imediatamente à esquerda do ponto $p$. Tornamos esta noção mais precisa com o resultado a seguir.

> **Proposição 54** — Seja $L$ uma lista de polígonos e sejam $p_0, p_1 \in \mathbb{R}^2$ dois pontos no plano com $p_0 < p_1$. Sejam ainda $e_0$ e $e_1$ duas arestas de polígonos em $L$ comparáveis em ambos os pontos $p_0$ e $p_1$ tais que não existe um ponto chave $p \in \mathcal{K}(L)$ na intersecção de $e_0$ com $e_1$ satisfazendo $p_0 \leq p < p_1$. Então $e_0 <_{p_0}^L e_1$ se, e somente se, $e_0 <_{p_1}^L e_1$.

**Prova** — Observe que a definição da ordem $<_p^L$ envolve quatro fatores: a altura das arestas, o produto cruzado $u \times v$, as direções das arestas e os índices dos polígonos de origem das arestas. Destes, apenas as alturas das arestas dependem do ponto $p$. Assim, se $e_0$ e $e_1$ têm uma intersecção infinita, então $\text{height}_p(e_0) = \text{height}_p(e_1)$ para todo ponto $p \in \mathbb{R}^2$ no qual $e_0$ e $e_1$ são comparáveis, de forma que, neste caso, o resultado segue trivialmente. Assim, podemos assumir, sem perda de generalidade, que $e_0$ e $e_1$ possuem uma intersecção finita.

Defina agora $u_0 = \text{left}(e_0)$, $v_0 = \text{right}(e_0)$, $u_1 = \text{left}(e_1)$ e $v_1 = \text{right}(e_1)$. Recorremos então novamente aos cisalhamentos, aplicando a proposição 50 ao conjunto finito $X = \{u_0, v_0, u_1, v_1, p_0, p_1\} \cup (u_0v_0 \cap u_1v_1)$ e as proposições 51 e 52 às quatro combinações de pontos e arestas envolvendo $e_0$ e $e_1$ como arestas e $p_0$ e $p_1$ como pontos, obtendo assim um valor $\delta \in \mathbb{R}$ com $\delta > 0$ tal que:

- Para todos os pontos $a, b \in X$, a abscissa de $\delta[a]$ é menor do que a abscissa de $\delta[b]$ se, e somente se, $a$ precede $b$ ($a < b$);
- A ordenada $y_{00}$ da intersecção de $\delta[u_0v_0]$ com $VL(\delta[p_0])$ é menor do que a ordenada $y_{10}$ da intersecção de $\delta[u_1v_1]$ com $VL(\delta[p_0])$ se, e somente se, $\text{height}_{p_0}(e_0) < \text{height}_{p_0}(e_1)$; e
- A ordenada $y_{01}$ da intersecção de $\delta[u_0v_0]$ com $VL(\delta[p_1])$ é menor do que a ordenada $y_{11}$ da intersecção de $\delta[u_1v_1]$ com $VL(\delta[p_1])$ se, e somente se, $\text{height}_{p_1}(e_0) < \text{height}_{p_1}(e_1)$.

Novamente lembramos que a ordem $<_p^L$ depende do ponto $p$ apenas para determinar as alturas. Assim, continuamos esta demonstração separando-a em dois casos: $\text{height}_{p_1}(e_0) < \text{height}_{p_1}(e_1)$ e $\text{height}_{p_1}(e_0) = \text{height}_{p_1}(e_1)$. Lembramos que podemos assumir, sem perda de generalidade, que $\text{height}_{p_1}(e_0) \leq \text{height}_{p_1}(e_1)$, pois do contrário trocamos $e_0$ com $e_1$.

O caso mais fácil é o caso em que $\text{height}_{p_1}(e_0) < \text{height}_{p_1}(e_1)$, onde claramente ocorre que $e_0 <_{p_1}^L e_1$. Com o que vimos, temos que $y_{01} < y_{11}$. Suponha então, para fins de um argumento por contradição, que $y_{00} \geq y_{10}$. Deve haver então um ponto $q \in \mathbb{R}^2$ tal que a abscissa de $\delta[p_0]$ é menor ou igual à abscissa de $\delta[q]$, tal que a abscissa de $\delta[q]$ é menor do que a abscissa de $\delta[p_1]$ e tal que $\delta[q] = \delta[u_0v_0] \cap \delta[u_1v_1]$, isto é, $\delta[u_0v_0]$ deve intersectar $\delta[u_1v_1]$ entre os pontos $\delta[p_0]$ e $\delta[p_1]$ (mas não em $\delta[p_1]$). Porém, devemos ter que $q \in X$, de forma que $p_0 \leq q < p_1$. Agora, como $q$ é a intersecção finita das duas arestas $e_0$ e $e_1$, devemos ter também que $q \in \mathcal{K}(L)$, uma contradição. Logo devemos concluir que $y_{00} < y_{10}$ e assim que $\text{height}_{p_0}(e_0) < \text{height}_{p_0}(e_1)$ e logo que $e_0 <_{p_0}^L e_1$, como queríamos.

Resta então tratarmos do caso em que $\text{height}_{p_1}(e_0) = \text{height}_{p_1}(e_1)$, ou seja, $y_{01} = y_{11}$. Antes de começarmos propriamente, no entanto, vamos mostrar um fato importante. Para isso, seja $x_1$ a abscissa de $\delta[p_1]$, de forma que $b = (x_1, y_{01}) = (x_1, y_{11}) \in \mathbb{R}^2$ é a intersecção de $\delta[u_0v_0]$ (ou $\delta[u_1v_1]$) com a reta vertical $VL(\delta[p_1])$. Seja ainda $x_0$ a abscissa de $\delta[p_0]$ e sejam $a_0, a_1 \in \mathbb{R}^2$ os pontos dados por $a_0 = (x_0, y_{00})$ e $a_1 = (x_0, y_{10})$. Em outras palavras, sejam $a_0$ e $a_1$ as respectivas intersecções de $\delta[u_0v_0]$ e $\delta[u_1v_1]$ com a reta vertical $VL(\delta[p_0])$. Claramente então, os vetores $\delta[v_0] - \delta[u_0]$ e $b - a_0$ têm o mesmo sentido, do mesmo modo que os vetores $\delta[v_1] - \delta[u_1]$ e $b - a_1$ têm o mesmo sentido. Assim, temos que

$$\text{sign}\big((\delta[v_0] - \delta[u_0]) \times (\delta[v_1] - \delta[u_1])\big) = \text{sign}\big((b - a_0) \times (b - a_1)\big).$$

Porém, pelo lema 8,

$$(\delta[v_0] - \delta[u_0]) \times (\delta[v_1] - \delta[u_1]) = \delta[v_0 - u_0] \times \delta[v_1 - u_1] = (v_0 - u_0) \times (v_1 - u_1).$$

Logo, o sinal de $(v_0 - u_0) \times (v_1 - u_1)$ é o sinal de

$$\begin{aligned}
(b - a_0) \times (b - a_1) &= \big((x_1, y_{01}) - (x_0, y_{00})\big) \times \big((x_1, y_{01}) - (x_0, y_{10})\big) \\
&= (x_1 - x_0, y_{01} - y_{00}) \times (x_1 - x_0, y_{01} - y_{10}) \\
&= (x_1 - x_0)(y_{01} - y_{10}) - (y_{01} - y_{00})(x_1 - x_0) \\
&= (x_1 - x_0)(y_{01} - y_{10} - y_{01} + y_{00}) \\
&= (x_1 - x_0)(y_{00} - y_{10})
\end{aligned}$$

e, como $x_0 < x_1$,

$$\text{sign}\big((v_0 - u_0) \times (v_1 - u_1)\big) = \text{sign}(y_{00} - y_{10}).$$

Voltamos então ao que estávamos fazendo. Se $\text{height}_{p_1}(e_0) = \text{height}_{p_1}(e_1)$ e logo $y_{01} = y_{11}$, temos dois sub-casos: $(v_0 - u_0) \times (v_1 - u_1) < 0$ e $(v_0 - u_0) \times (v_1 - u_1) = 0$. Note que podemos assumir, sem perda de generalidade, que $(v_0 - u_0) \times (v_1 - u_1) \leq 0$, pois do contrário trocamos $e_0$ e $e_1$ (como $\text{height}_{p_1}(e_0) = \text{height}_{p_1}(e_1)$, não precisamos realizar a troca de $e_0$ com $e_1$ anterior e logo $e_0$ e $e_1$ aqui são realmente arbitrários).

Se $(v_0 - u_0) \times (v_1 - u_1) < 0$, então claramente $e_0 <^L_{p_1} e_1$ e também, como acabamos de ver $y_{00} < y_{10}$. Mas isto mostra que $\text{height}_{p_0}(e_0) < \text{height}_{p_0}(e_1)$ e logo que $e_0 <^L_{p_0} e_1$, como queríamos.

Por outro lado, se $(v_0 - u_0) \times (v_1 - u_1) = 0$, temos apenas que $y_{00} = y_{10}$ e logo que $\text{height}_{p_0}(e_0) = \text{height}_{p_0}(e_1)$. Mas então $\text{height}_{p_0}(e_0) = \text{height}_{p_0}(e_1)$ e $\text{height}_{p_1}(e_0) = \text{height}_{p_1}(e_1)$, de forma que $e_0 <^L_{p_0} e_1$ se, e somente se, $e_0 <^L_{p_1} e_1$, pois, novamente, o papel exercido pelo ponto nesta ordem é apenas o de determinar a altura. □

Iremos agora estabelecer um resultado que consiste em uma melhora (embora não atingindo ainda a perfeição) do processo de *ray casting* descrito na seção 2.2.

**Definição 74** — Seja $uv$ um segmento de reta comparável em um ponto $p$ tal que $p \notin uv$. Assim sendo, o segmento $uv$ não pode ser vertical, ou seja, a abscissa de $u$ deve diferir da abscissa de $v$. Logo, temos que a intersecção $uv \cap VL(p)$ do segmento $uv$ com a reta vertical passando por $p$ possui exatamente um ponto com ordenada $\text{height}_p(uv)$. Dizemos então que $uv$ está ***abaixo*** de $p$ se $\text{height}_p(uv)$ for menor do que a ordenada de $p$ e que $uv$ está ***acima*** de $p$ caso contrário.

**Lema 10** — Seja $P$ um polígono e seja $p \in \mathbb{R}^2 \setminus \partial P$ um ponto no plano fora da fronteira de $P$. Então:

- Se $p \notin P$, dentre as arestas de $P$ comparáveis em $p$ e abaixo de $p$, o número de arestas de ida é igual ao número de arestas de volta; e
- Se $p \in P$, dentre as arestas de $P$ comparáveis em $p$ e abaixo de $p$, o número de arestas de ida é igual a um mais número de arestas de volta.

**Prova** — Primeiramente iremos demonstrar este lema assumindo que os vértices do polígono $P$ têm abscissas duas a duas distintas e que o ponto $p \in \mathbb{R}^2 \setminus \partial P$ tem uma abscissa diferente das abscissas dos vértices do polígono $P$. Esta prova é bastante similar à demonstração da proposição 8.

Seja

$$\begin{array}{rccl} \gamma : & (0, \infty) & \to & \mathbb{R}^2 \\ & t & \mapsto & p - t(0,1), \end{array}$$

a parametrização de uma semi-reta contida em $VL\,(p)$ com origem em $p$ e seja $T = \{t \in \mathbb{R} : t > 0 \wedge \gamma(t) \in \partial P\}$ um conjunto de números reais. Como a abscissa de $u$ é diferente da abscissa de $v$ para toda aresta $\overrightarrow{uv} \in E(P)$, temos que $uv \cap VL\,(p)$ é uma intersecção finita, de forma que $T$ é um conjunto finito, e pode portanto ser escrito como $T = \{t_0, t_1, \ldots, t_{n-1}\}$ com $0 < t_0 < t_1 < \cdots < t_{n-1}$.

Como a abscissa de $p$ é diferente da abscissa de $v$ para todo vértice $v \in V(P)$, temos que, para cada $i \in \{0, 1, \ldots, n-1\}$, existe exatamente uma aresta $e_i$ que contém contém o ponto $\gamma(t_i)$. Ainda, como cada aresta de $P$ pode intersectar $\operatorname{Im}(\gamma) \subseteq VL\,(p)$ apenas uma vez, as arestas $e_0, e_1, \ldots, e_{n-1}$ são duas a duas distintas.

Observe então que as arestas $e_0, e_1, \ldots, e_{n-1}$ são precisamente as arestas de $P$ comparáveis em $p$ e abaixo de $p$. De fato, como as abscissas de $p$ e dos vértices de $P$ são duas a duas distintas, uma aresta $\overrightarrow{uv} \in E(P)$ é tal que $uv$ é comparável em $p$ se, e somente se, $uv \cap VL\,(p) \neq \{\}$. Ainda, uma tal aresta está abaixo de $p$ se, e somente se, a ordenada $\operatorname{height}_p(uv)$ desta intersecção é menor do que a ordenada de $p$.

Mostraremos agora, por indução em $n$, que:

- Se $p \notin P$, então o número de arestas de ida em $\{e_0, e_1, \ldots, e_{n-1}\}$ é igual ao número de arestas de volta em $\{e_0, e_1, \ldots, e_{n-1}\}$; e
- Se $p \in P$, então o número de arestas de ida em $\{e_0, e_1, \ldots, e_{n-1}\}$ é igual a um mais o número de arestas de volta em $\{e_0, e_1, \ldots, e_{n-1}\}$.

O caso base ocorre quando $n = 0$. Neste caso, como $\operatorname{Im}(\gamma)$ não intersecta $\partial P$, devemos ter $p \notin P$. Assim, a hipótese indutiva vale trivialmente.

Já se $n > 0$, definimos $t' \in \mathbb{R}$ como $t' = t_0 + 1$ se $n = 1$ e $t' = \dfrac{t_0 + t_1}{2}$ se $n > 1$ (de modo que $0 < t_0 < t' < t_1$). Definimos também o ponto $p' \in \mathbb{R}^2$ como $p' = \gamma(t')$. Note que a abscissa de $p'$ é a mesma abscissa de $p$, de forma que $VL\,(p') = VL\,(p)$ e que as abscissas dos vértices de $P$ e a abscissa de $p'$ são duas a duas distintas. Observe ainda que as arestas de $P$ que intersectam $VL\,(p')$ em um ponto de ordenada menor do que a ordenada de $p'$ são precisamente as arestas $e_1, e_2, \ldots, e_{n-1}$, sendo estas também precisamente as arestas de $P$ comparáveis em $p'$ e abaixo de $p'$. Também devemos ter $p' \notin \partial P$, pois as intersecções de arestas de $P$ com $\operatorname{Im}(\gamma)$ são os pontos no conjunto $\{\gamma(t_0), \gamma(t_1), \ldots, \gamma(t_{n-1})\}$, do qual $p' = \gamma(t')$ não faz parte.

Temos então que a hipótese indutiva vale para $p'$ e o polígono $P$, de forma que:

- Se $p' \notin P$, então o número de arestas de ida em $\{e_1, e_2, \ldots, e_{n-1}\}$ é igual ao número de arestas de volta em $\{e_1, e_2, \ldots, e_{n-1}\}$; e
- Se $p' \in P$, então o número de arestas de ida em $\{e_1, e_2, \ldots, e_{n-1}\}$ é igual a um mais o número de arestas de volta em $\{e_1, e_2, \ldots, e_{n-1}\}$.

Observe agora que devemos ter que $p \in P$ se, e somente se, $p' \notin P$, pois há precisamente uma aresta de $P$ que intersecta (e cruza), o segmento $pp'$: a aresta $e_0$. Por causa disto, como as arestas de $P$ estão orientadas em sentido anti-horário, devemos ter que $e_0$ é uma aresta de ida se, e somente se, $p \in P$ e $p' \notin P$. De fato, se $e_0 = \overrightarrow{uv}$, devemos ter que $p \in P$ se, e somente se, $(p' - p) \times (v - u) > 0$ e, como $p' - p$ é um segmento vertical, também devemos ter que $\overrightarrow{uv}$ é uma aresta de ida se, e somente se, $(p - p') \times (v - u) > 0$.

Assim, se $p \in P$, então $p' \notin P$ e, pela hipótese indutiva, há o mesmo número de arestas de ida e de arestas de volta em $\{e_1, e_2, \ldots, e_{n-1}\}$. Também vimos que, neste caso, $e_0$ deve ser uma aresta de ida, e assim o número de arestas de ida em $\{e_0, e_1, \ldots, e_{n-1}\}$ deve exceder em um o número de arestas de volta em $\{e_0, e_1, \ldots, e_{n-1}\}$. Por outro lado, se $p \notin P$, então $p' \in P$ e, novamente pela hipótese indutiva, o número de arestas de ida em $\{e_1, e_2, \ldots, e_{n-1}\}$ é um mais do que o número de arestas de volta em $\{e_1, e_2, \ldots, e_{n-1}\}$. Porém, vimos que $e_0$ neste caso deve ser uma aresta de volta, e assim há o mesmo número de arestas de ida e arestas de volta em $\{e_0, e_1, \ldots, e_{n-1}\}$, concluindo esta prova por indução.

Até aqui mostramos que o lema é válido desde que o ponto $p$ e os vértices do polígono $P$ tenham abscissas distintas duas a duas. Gostaríamos de estender este resultado de forma simples ao caso geral, e para isso fazemos uso do que desenvolvemos sobre cisalhamentos.

Começamos aplicando a proposição 50 ao conjunto $X = V(P) \cup \{p\}$ e as proposições 51 e 52 a cada aresta de $P$, obtendo um valor $\delta \in \mathbb{R}$ com $\delta > 0$ tal que:

- Para todos os pontos $a, b \in X$, a abscissa de $\delta[a]$ é menor do que a abscissa de $\delta[b]$ se, e somente se, $a$ precede $b$ ($a < b$);
- Para toda aresta $\overrightarrow{uv} \in E(P)$, $\delta[uv]$ intersecta $VL(\delta[p])$ se, e somente se, $uv$ é comparável em $p$;
- Para toda aresta $\overrightarrow{uv} \in E(P)$ comparável em $p$ e abaixo de $p$, o segmento $\delta[uv]$ intersecta $VL(\delta[p])$ em exatamente um ponto $(x, y)$ com $y < \operatorname{height}_p(uv)$; e
- Para toda aresta $\overrightarrow{uv} \in E(P)$ comparável em $p$ e acima de $p$, o segmento $\delta[uv]$ intersecta $VL(\delta[p])$ em exatamente um ponto $(x, y)$ com $y > \operatorname{height}_p(uv)$.

Note que não existem arestas de $P$ contendo $p$ porque $p \notin \partial P$.

Observe agora que, pela proposição 46, $\delta[P]$ é um polígono e que, pelo corolário 1 (com $\Omega_0 = \Omega_1 = \mathbb{R}^2$ e $f = \delta[\cdot]$), $\partial\delta[P] = \delta[\partial P]$, de forma que $\delta[p] \notin \partial\delta[P]$. Ainda, obviamente o ponto $p$ está em $P$ se, e somente se, o ponto $\delta[p]$ está em $\delta[P]$. Porém, o ponto $\delta[p]$ possui abscissa diferente das abscissas dos vértices de $\delta[P]$ e os vértices de $\delta[P]$ possuem abscissas duas a duas distintas. Logo, o lema, como acabamos de mostrar, vale para $\delta[P]$ e $\delta[p]$. Assim, para terminar de provar o lema por completo, devemos mostrar apenas que toda aresta $\overrightarrow{uv} \in E(P)$ é tal que $uv$ é comparável em $p$ e está abaixo de $p$ se, e somente se, $\delta[uv]$ é comparável em $\delta[p]$ e está abaixo de $\delta[p]$.

De fato, se $uv$ é comparável em $p$ e está abaixo de $p$, então podemos assumir, sem perda de generalidade, que $u < p < v$, pois $p \notin V(P) \subseteq \partial P$. Logo, as abscissas de $\delta[u]$, $\delta[p]$ e $\delta[v]$ devem estar em ordem estritamente crescente, de forma que $\delta[u] < \delta[p] < \delta[v]$ e logo que $\delta[uv]$ é comparável em $\delta[p]$. Ainda, como $uv$ está abaixo de $p$, devemos ter que $uv$ intersecta $VL(p)$ em exatamente um ponto com ordenada menor do que a ordenada de $p$. Logo, pela proposição 52 e por nossa escolha de $\delta$, temos que $\delta[uv]$ intersecta $VL(\delta[p])$ em exatamente um ponto cuja ordenada deve ser também menor do que a ordenada de $p$, que é exatamente a ordenada de $\delta[p]$. Assim, devemos concluir que $\delta[uv]$ está de fato abaixo de $\delta[p]$.

Conversamente, se $\delta[uv]$ é comparável em $\delta[p]$ e está abaixo de $\delta[p]$, podemos assumir, sem perda de generalidade, que $\delta[u] < \delta[p] < \delta[v]$, pois

$$\delta[p] \notin V\big(\delta[P]\big) = \delta[V(P)] \subseteq \delta[\partial P] = \partial\delta[P].$$

No entanto, as abscissas de $\delta[u]$, $\delta[p]$ e $\delta[v]$ devem ser duas a duas distintas, e assim devemos ter que as abscissas de $\delta[u]$, $\delta[p]$ e $\delta[v]$ devem estar em ordem estritamente crescente, de forma que $u < p < v$, pois $p \notin V(P) \subseteq \partial P$. Logo, $uv$ deve ser comparável em $p$. Porém, $p \notin uv \subseteq \partial P$, e assim $uv \cap VL(p)$ deve possuir exatamente um ponto com ordenada diferente da ordenada de $p$. No entanto, como $\delta[uv]$ está abaixo de $\delta[p]$, o segmento $\delta[uv]$ intersecta a reta vertical $VL(\delta[p])$ em exatamente um ponto cuja ordenada é menor do que a ordenada de $\delta[p]$, que é a ordenada de $p$. Novamente então a proposição 52 e a nossa escolha de $\delta$ nos garantem que o único ponto em que $uv$ intersecta $VL(p)$ tem ordenada menor do que a ordenada de $p$. Assim, temos que de fato $uv$ está abaixo de $p$, finalmente concluindo esta demonstração. □

Em uma entrada genérica para o problema 2, arestas de polígonos diferentes podem se sobrepor, e assim cada segmento candidato pode estar em várias arestas.

> **Definição 75** — Seja $L$ uma lista de polígonos, $e = \overrightarrow{uv}$ a $j$-ésima aresta do $i$-ésimo polígono em $L$ e sejam $p, q \in \mathcal{K}(L) \cap uv$ tais que $p < q$ e que o segmento $pq$ é um segmento candidato de $L$. Dizemos então que a quádrupla $(i, j, p, q)$ é uma ***representação*** do segmento candidato $pq$.

Note que cada segmento candidato admite exatamente uma representação para cada polígono na lista que possui uma aresta que contém o segmento candidato.

A seguinte definição e o resultado que a sucede contém uma das principais ideias de nosso algoritmo: como descobrir se um segmento candidato está contido na fronteira da união de polígonos.

**Definição 76** — Seja $L$ uma lista de polígonos, seja $(i, j, u, v)$ uma representação de um segmento candidato $uv$ ($u < v$) de $L$ e seja $e$ a $j$-ésima aresta do $i$-ésimo polígono de $L$. Dizemos então que a representação $(i, j, u, v)$ é uma ***boa representação*** do segmento candidato $uv$ se, dentre as arestas de polígonos em $L$ comparáveis em $v$, o número de arestas de ida que precedem $e$ em $L$ no ponto $v$ for igual ao número de arestas de volta que não são precedidas por $e$ em $L$ no ponto $v$.

**Proposição 55** — Seja $L$ uma lista de polígonos cuja união é $U \subseteq \mathbb{R}^2$ e seja $uv$ um segmento candidato de $L$. Assim,

- Se $uv \nsubseteq \partial U$, então o segmento candidato $uv$ não possui uma boa representação; e
- Se $uv \subseteq \partial U$, então existe exatamente uma boa representação de $uv$.

**Prova** — Começamos definindo quatro conjuntos de arestas de polígonos em $L$:

- $E$ é o conjunto das arestas de polígonos em $L$ comparáveis em $v$;
- $E_=$ é o conjunto das arestas $e$ de polígonos em $L$ correspondentes a um segmento $ab$ comparável em $v$ e tal que $\text{height}_v(ab)$ é igual à ordenada de $v$ e $\big(\text{right}(e) - \text{left}(e)\big) \times (v - u) = 0$;
- $E_<$ é o conjunto das arestas $e$ de polígonos em $L$ correspondentes a um segmento $ab$ comparável em $v$ tal que $\text{height}_v(ab)$ é menor do que a ordenada de $v$ ou tal que $\text{height}_v(ab)$ é igual à ordenada de $v$ e $\big(\text{right}(e) - \text{left}(e)\big) \times (v - u) < 0$; e
- $E_>$ é o conjunto das arestas $e$ de polígonos em $L$ correspondentes a um segmento $ab$ comparável em $v$ tal que $\text{height}_v(ab)$ é maior do que a ordenada de $v$ ou tal que $\text{height}_v(ab)$ é igual à ordenada de $v$ e $\big(\text{right}(e) - \text{left}(e)\big) \times (v - u) > 0$.

É importante observar que duas arestas oriundas de polígonos de índices diferentes na lista $L$ são sempre consideradas elementos distintos nestes conjuntos, mesmo que elas correspondam ao mesmo segmento de reta, mesmo que elas possuam a mesma direção e mesmo que seus polígonos de origem correspondam ao mesmo conjunto de pontos. Isto é extremamente importante pois faremos contagens de elementos destes conjuntos posteriormente. Ainda, observamos que $E = E_< \cup E_= \cup E_>$.

Uma das maiores dificuldades em mostrar este resultado diretamente é a presença de arestas verticais. Contornamos este problema fazendo uso de um cisalhamento. Mais precisamente, aplicamos a proposição 50 ao conjunto $X = \mathcal{K}(L)$ e as proposições 51 e 52 às arestas de polígonos em $L$ em relação ao ponto $v$, obtendo um valor $\delta \in \mathbb{R}$ com $\delta > 0$ tal que:

- Para todos os pontos $a, b \in X = \mathcal{K}(L)$, a abscissa de $\delta[a]$ é menor do que a abscissa de $\delta[b]$ se, e somente se, $a$ precede $b$ ($a < b$); e
- Para toda aresta de polígonos em $L$ correspondente a um segmento de reta $ab$ comparável em $v$, $\delta[ab]$ é comparável em $\delta[v]$ e o sinal de $y - \text{height}_v(ab)$ é igual ao sinal de $y - \text{height}_{\delta[v]}(\delta[ab])$, onde $y$ é a ordenada de $v$ e logo de $\delta[v]$.

O próximo passo é localizar um ponto apropriado sobre o segmento $\delta[uv]$. Para isso, observe que existe ao menos um ponto chave de $L$ que precede $v$, nominalmente o ponto $u$. Assim, existe um único ponto chave predecessor de $v$, ou seja, existe um único ponto chave $q_0 \in \mathcal{K}(L)$ tal que $q_0 < v$ e tal que não existe outro ponto chave $q \in \mathcal{K}(L)$ com $q_0 < q < v$. Observe então que $u \leq q_0 < v$, de forma que, como $q_0 \in \mathcal{K}(L) = X$, a abscissa de $\delta[u]$ é menor ou igual à abscissa de $\delta[q_0]$ e a abscissa de $\delta[q_0]$ é menor do que a abscissa de $\delta[v]$. Assim, a abscissa de $\dfrac{\delta[q_0] + \delta[v]}{2}$ está estritamente entre as abscissas de $\delta[u]$ e $\delta[v]$, de forma que existe um único ponto $p \in \mathbb{R}^2$ tal que $\delta[p]$ está na intersecção de $\delta[uv]$ com a reta vertical $VL\left(\dfrac{\delta[q_0] + \delta[v]}{2}\right)$. Este ponto $p$ está contido no segmento $uv$ e é tal que as abscissas de $\delta[u]$, $\delta[p]$ e $\delta[v]$ estão em ordem estritamente crescente. Ainda, para todo ponto chave $q \in \mathcal{K}(L)$ com $q < v$, as abscissas de $\delta[q]$, $\delta[p]$ e $\delta[v]$ estão em ordem estritamente crescente, embora em geral não valha que $q < p < v$ porque $p \notin \mathcal{K}(L) = X$, de forma que a nossa aplicação da proposição 50 não levou em conta o ponto $p$.

Relacionamos agora os conjuntos $E$, $E_<$, $E_=$ e $E_>$ com o ponto $p$ mostrando os quatro seguintes fatos:

- Uma aresta $e$ de um polígono em $L$ correspondente a um segmento de reta $ab$ está em $E$ se, e somente se, $\delta[ab]$ é comparável em $\delta[p]$:
  - ($\Rightarrow$): Se $e \in E$, então $ab$ é comparável em $v$ e podemos assumir, sem perda de generalidade, que $a < v \leq b$. Assim, como $a \in \mathcal{K}(L)$, as abscissas de $\delta[a]$, $\delta[p]$ e $\delta[v]$ estão em ordem estritamente crescente, de modo que $\delta[ab]$ é comparável em $\delta[p]$ porque a abscissa de $\delta[v]$ é menor ou igual à abscissa de $\delta[b]$;
  - ($\Leftarrow$): Se $\delta[ab]$ é comparável em $\delta[p]$, então, como $a, b \in \mathcal{K}(L)$, podemos assumir, sem perda de generalidade, que as abscissas de $\delta[a]$, $\delta[p]$ e $\delta[b]$ estão em ordem estritamente crescente. Porém, $v$ é o menor ponto chave cuja abscissa pelo cisalhamento por $\delta$ é maior do que abscissa de $\delta[p]$, de forma que a abscissa de $\delta[v]$ é menor ou igual à abscissa de $\delta[b]$. Logo, temos que $a < v \leq b$, que $ab$ é comparável em $v$ e que $e \in E$;
- Uma aresta $e$ de um polígono em $L$ correspondente a um segmento de reta $ab$ está em $E_=$ se, e somente se, $\delta[ab]$ contém $\delta[p]$:
  - ($\Rightarrow$): Se $e \in E_=$, então $ab$ intersecta $uv$ no ponto $v$, de forma que $\delta[ab]$ intersecta $\delta[uv]$ no ponto $\delta[v]$. Mais ainda, temos que $(b - a) \times (v - u) = 0$, de forma que $ab$ e $uv$ são colineares e que, por consequência, $\delta[ab]$ e $\delta[uv]$ são colineares. Porém, como vimos, $e \in E_= \subseteq E$ implica em $\delta[ab]$ ser comparável em $\delta[p] \in \delta[uv]$, de forma que devemos ter $\delta[p] \in \delta[ab]$;
  - ($\Leftarrow$): Se $\delta[ab]$ contém $\delta[p]$, então $ab$ contém $p$, que por sua vez está em $uv$. Assim, a intersecção de $ab$ com $uv$ é infinita, pois do contrário $p$ seria um ponto chave. Logo $ab$ e $uv$ são colineares. Porém, $ab$ conter $p$ implica em $\delta[ab]$ ser comparável em $\delta[p]$, de forma que, como vimos antes, $ab$ é comparável em $v$. Logo, como $ab$ e $uv$ são colineares, devemos ter $\text{height}_v(ab)$ igual à ordenada de $v$ e $(b - a) \times (v - u) = 0$, de forma que $e \in E_=$;
- Uma aresta $e$ de um polígono em $L$ correspondente a um segmento de reta $ab$ está em $E_<$ se, e somente se, o segmento $\delta[ab]$ é comparável em $\delta[p]$, não contém o

ponto $\delta[p]$ e está abaixo de $\delta[p]$:

- ($\Rightarrow$): Se $e \in E_< \subseteq E$, então $ab$ é comparável em $v$ e $\delta[ab]$ é comparável em $\delta[p]$. Ainda, $\text{height}_v(ab)$ é menor ou igual à ordenada $y$ do ponto $v$. Se $\text{height}_v(ab) = y$, então temos que $\text{height}_{\delta[v]}(\delta[ab]) = y$. Porém, também temos, assumindo $a < b$ sem perda de generalidade, que $(b-a) \times (v-u) < 0$ e, pelo lema 8, que $(\delta[b] - \delta[a]) \times (\delta[v] - \delta[u]) < 0$, de forma que $ab$ não pode conter $p$ (pois $ab$ e $uv$ não são colineares) e $ab$ deve estar abaixo de $p$. Já se $\text{height}_v(ab) < y$ e logo $\text{height}_{\delta[v]}(\delta[ab]) < y$, observe que $\delta[ab]$ não pode intersectar $\delta[uv]$ em um ponto com abscissa maior ou igual à de $\delta[p]$ e menor do que a de $\delta[v]$ (pois não há pontos chave com imagens por $\delta[\cdot]$ com estas abscissas e a intersecção de $ab$ e $uv$ deve ser finita). Logo, devemos também ter que $\delta[ab]$ não contém $\delta[p]$ e está abaixo de $\delta[p] \in \delta[uv]$;
- ($\Leftarrow$) Suponha que $\delta[ab]$ é comparável em $\delta[p]$, não contém $\delta[p]$ e está abaixo de $\delta[p]$. Já vimos que $ab$ é comparável em $v$, de forma que, assumindo $a < b$ sem perda de generalidade, a abscissa de $\delta[a]$ é menor do que a abscissa de $\delta[p]$ e a abscissa de $\delta[b]$ é maior ou igual à abscissa de $\delta[v]$. Vimos também que $\delta[ab]$ não pode intersectar $\delta[uv]$ em um ponto com abscissa maior ou igual à de $\delta[p]$ e menor do que a de $\delta[v]$. Assim, devemos concluir que $\delta[ab]$ intersecta $VL(\delta[v])$ em um ponto de ordenada menor ou igual à ordenada de $v$, que é a mesma que a ordenada de $\delta[v]$. Se, no entanto, estas ordenadas forem iguais, precisamos ter $(\delta[b] - \delta[a]) \times (\delta[v] - \delta[u]) < 0$, que, pelo lema 8, equivale a $(b-a) \times (v-u) < 0$, pois do contrário $\delta[ab]$ não estaria abaixo de $\delta[p]$. Logo, temos de fato que $e \in E_<$; e

- Uma aresta $e$ de um polígono em $L$ correspondente a um segmento de reta $ab$ está em $E_>$ se, e somente se, o segmento $\delta[ab]$ é comparável em $\delta[p]$, não contém o ponto $\delta[p]$ e está acima de $\delta[p]$:
  - Na prova desta proposição não iremos usar este fato e nem, na verdade, faremos menção futura ao conjunto $E_>$, tendo introduzido este conjunto apenas por completude. Assim, observamos que esta demonstração é completamente análoga à demonstração anterior e a deixamos como exercício ao leitor.

Passando ao próximo passo, como arestas são conjuntos fechados e como $p \notin \mathcal{K}(L)$, existe um valor real $\varepsilon \in \mathbb{R}$ com $\varepsilon > 0$ tal que $B_\varepsilon(\delta[p])$ não intersecta $\delta[\mathcal{K}(L)]$ e nem a imagem por $\delta[\cdot]$ de um segmento correspondente a uma aresta de um polígono em $L$, exceto as arestas em $E_=$.

Escolha então dois pontos $p_0, p_1 \in \mathbb{R}^2$ tais que $\delta[p_0], \delta[p_1] \in B_\varepsilon(\delta[p])$, tais que as abscissas de $\delta[p_0]$ e $\delta[p_1]$ sejam iguais à abscissa de $\delta[p]$ e tais que $p_0$ possua uma ordenada menor do que a ordenada de $p$ e $p_1$ possua uma ordenada maior do que a ordenada de $p$. (Note que as ordenadas das imagens de pontos por $\delta[\cdot]$ são iguais às dos pontos originais). Por esta construção, temos que um segmento de reta $ab$ correspondente a uma aresta em um polígono em $L$ é sempre tal que $\delta[ab]$ é comparável em $\delta[p]$ se, e somente se, $\delta[ab]$ for comparável em $\delta[p_0]$ (ou $\delta[p_1]$). O que muda são as relações destas arestas com $p_0$ e com $p_1$. De fato, como $\delta[p_0]$, $\delta[p]$ e $\delta[p_1]$ compartilham as mesmas abscissas e como $\delta[u]$ não possui a mesma abscissa que $\delta[v]$, devemos ter que $\delta[p_0], \delta[p_1] \notin \delta[uv]$, ou seja, $p_0, p_1 \notin uv$, de forma que não existem arestas em polígonos de $L$ contendo $p_0$ ou $p_1$ (lembre que apenas arestas em $E_=$ podem intersectar $-\delta[B_\varepsilon(\delta[p])]$). Ainda, temos que:

- As arestas de polígonos em $L$ correspondentes a segmentos de reta $ab$ tais que $\delta[ab]$ é comparável em $\delta[p_0]$ e está abaixo de $\delta[p_0]$ são precisamente as arestas em $E_<$; e
- As arestas de polígonos em $L$ correspondentes a segmentos de reta $ab$ tais que $\delta[ab]$ é comparável em $\delta[p_1]$ e está abaixo de $\delta[p_1]$ são precisamente as arestas em $E_< \cup E_=$.

O truque é então observar que, como já vimos antes, não existem arestas de polígonos em $L$ contendo $p_0$ ou $p_1$, de forma $p_0$ e $p_1$ não estão na fronteira de nenhum polígono em $L$, o que nos possibilita aplicar o lema 10 aos pontos $\delta[p_0]$ e $\delta[p_1]$ juntamente com a imagem por $\delta[\cdot]$ de cada polígono em $L$ obtendo os dois seguintes fatos:

- O número de polígonos em $L$ cuja imagem por $\delta[\cdot]$ contém $\delta[p_0]$ é $\#\overrightarrow{E_<} - \#\overleftarrow{E_<}$; e
- O número de polígonos em $L$ cuja imagem por $\delta[\cdot]$ contém $\delta[p_1]$ é

$$\left(\#\overrightarrow{E_=} + \#\overrightarrow{E_<}\right) - \left(\#\overleftarrow{E_=} + \#\overleftarrow{E_<}\right),$$

onde $\#\overrightarrow{A}$ denota o número de arestas de ida em um conjunto de arestas $A$ e $\#\overleftarrow{A}$ denota o número de arestas de volta em um conjunto de arestas $A$. Em outras palavras

- O número de polígonos em $L$ contendo $p_0$ é $\#\overrightarrow{E_<} - \#\overleftarrow{E_<}$; e
- O número de polígonos em $L$ contendo $p_1$ é $\left(\#\overrightarrow{E_=} + \#\overrightarrow{E_<}\right) - \left(\#\overleftarrow{E_=} + \#\overleftarrow{E_<}\right)$.

Observe então que o segmento $\delta[uv]$ particiona $B_\varepsilon\big(\delta[p]\big) \setminus \delta[uv]$ em duas partes

$$A_0 = \Big\{q \in B_\varepsilon\big(\delta[p]\big) : (\delta[v] - \delta[u]) \times \big(q - \delta[p]\big) > 0\Big\}$$

e

$$A_1 = \Big\{q \in B_\varepsilon\big(\delta[p]\big) : (\delta[v] - \delta[u]) \times \big(q - \delta[p]\big) < 0\Big\},$$

cada uma das quais está (pelo teorema do valor intermediário) inteiramente contida em $\delta[U^\circ]$ ou em $\delta[\mathbb{R}^2 \setminus U]$ (lembre que $B_\varepsilon\big(\delta[p]\big)$ não pode intersectar a imagem por $\delta[\cdot]$ de arestas de polígonos em $L$ não colineares com $uv$). Porém ambas estas partes não podem estar contidas em $\delta[\mathbb{R}^2 \setminus U] = \mathbb{R}^2 \setminus \delta[U]$ porque $uv$ está contido em alguma aresta de algum polígono $P$ em $L$, de forma que ou $A_0 \subseteq \delta[P]^\circ$ ou $A_1 \subseteq \delta[P]^\circ$. Ainda, temos que $\delta[p_0] \in A_0$ e $\delta[p_1] \in A_1$. Logo, temos que $p \in \partial U$ se, e somente se, $p_0 \in U$ e $p_1 \notin U$ ou $p_0 \notin U$ e $p_1 \in U$. Assim, também segue que $p \in U^\circ$ se, e somente se, $p_0, p_1 \in U$.

Agora, como $E_=$ é um conjunto finito, podemos escrever $E_= = \{e_0, e_1, \ldots, e_{n-1}\}$ com $e_0 <_v^L e_1 <_v^L \cdots <_v^L e_{n-1}$. Note, no entanto, que toda aresta em $E_=$ deve conter o segmento $uv$ (pois $uv$ é colinear com uma tal aresta, que deve ser comparável em $v$, e não pode existir um ponto chave entre $u$ e $v$ contido em $uv$). Ainda, como todas as arestas em $E_=$ são duas a duas colineares, as arestas de ida em $E_=$ devem preceder as arestas de volta em $E_=$ em relação à ordem $<_v^L$, de forma que existe um número natural $k \in \{0, 1, \ldots, n\}$ tal que $e_0, e_1, \ldots, e_{k-1}$ são arestas de ida e $e_k, e_{k+1}, \ldots, e_{n-1}$ são arestas de volta.

Além da observação de que todas as arestas em $E_=$ contém $uv$, podemos adicionar que toda aresta em polígonos em $L$ que contém $uv$ está em $E_=$. Assim, toda representação do segmento candidato $uv$ deve ser em relação a uma aresta $e_i$ com $i \in \{0, 1, \ldots, n-1\}$ e cada aresta $e_i$ com $i \in \{0, 1, \ldots, n-1\}$ produz uma representação para $uv$.

Suponha agora que $e_i$ e $uv$ formam uma boa representação de $uv$ e que $i \in \{k, k+1, \ldots, n-1\}$. Temos então que, dentre as arestas de polígonos em $L$ comparáveis em $v$ (ou seja, as arestas em $E$), as arestas que precedem $e_i$ em $L$ no ponto $v$ são precisamente as arestas $e_0, e_1, \ldots, e_{i-1}$ juntamente com as arestas em $E_<$ e que as arestas que não são precedidas por $e_i$ em $L$ no ponto $v$ são precisamente $e_0, e_1, \ldots, e_i$ mais as arestas em $E_<$. Logo, as arestas de ida em $E$ que precedem $e_i$ em $L$ no ponto $v$ são precisamente as arestas de ida em $E_<$ e as arestas $e_0, e_1, \ldots, e_{k-1}$ (pois $k \leq i$), totalizando $\#\overrightarrow{E_<} + k$ arestas. Similarmente, as arestas de volta em $E$ que não precedem $e_i$ em $L$ no ponto $v$ são precisamente as arestas de volta em $E_<$ e $e_k, e_{k+1}, \ldots, e_i$, totalizando $\#\overleftarrow{E_<} + i - k + 1$ arestas. Porém, estamos assumindo que $e_i$ e $uv$ constituem uma boa representação, e assim devemos ter $\#\overrightarrow{E_<} + k = \#\overleftarrow{E_<} + i - k + 1$, de forma que $\#\overrightarrow{E_<} - \#\overleftarrow{E_<} = i - 2k + 1$. No entanto, o número de polígonos em $L$ contendo $p_1$ é

$$\begin{aligned}\left(\#\overrightarrow{E} + \#\overrightarrow{E_<}\right) - \left(\#\overleftarrow{E} + \#\overleftarrow{E_<}\right) &= k + \#\overrightarrow{E_<} - (n-k) - \#\overleftarrow{E_<} \\ &= \#\overrightarrow{E_<} - \#\overleftarrow{E_<} + 2k - n,\end{aligned}$$

e assim $\#\overrightarrow{E_<} - \#\overleftarrow{E_<} + 2k - n \geq 0$, de forma que $\#\overrightarrow{E_<} - \#\overleftarrow{E_<} \geq n - 2k$. Portanto, devemos ter que $i - 2k + 1 \geq n - 2k$, ou seja, que $i \geq n - 1$. Assim, das arestas $e_k, e_{k+1}, \ldots, e_{n-1}$, apenas $e_{n-1}$ pode participar de uma boa representação de $uv$.

Fazemos então o raciocínio análogo para as arestas $e_0, e_1, \ldots, e_{k-1}$, isto é, assumimos que $e_i$ participa de uma boa representação de $uv$ e $i \in \{0, 1, \ldots, k-1\}$. Novamente, dentre as arestas em $E$, temos que precisamente as arestas de $E_<$ e $e_0, e_1, \ldots, e_{i-1}$ precedem $e_i$ em $L$ no ponto $v$ e que precisamente as arestas de $E_<$ e $e_0, e_1, \ldots, e_i$ não são precedidas por $e_i$ em $L$ no ponto $v$. Assim, as arestas de ida em $E$ que precedem $e_i$ em $L$ no ponto $v$ são precisamente as arestas de ida em $E_<$ e as arestas $e_0, e_1, \ldots, e_{i-1}$, que devem ser todas de ida já que $i < k$, em um total de $\#\overrightarrow{E_<} + i$ arestas. Similarmente, as arestas de volta em $E$ que não são precedidas por $e_i$ em $L$ no ponto $v$ são as arestas de volta em $E_<$ apenas (como $i < k$ temos que não há arestas de volta em $e_0, e_1, \ldots, e_i$), em um total de $\#\overleftarrow{E_<}$ arestas. Logo, como $e_i$ participa de uma boa representação de $uv$, devemos ter que $\#\overrightarrow{E_<} + i = \#\overleftarrow{E_<}$. Porém, existem exatamente $\#\overrightarrow{E_<} - \#\overleftarrow{E_<}$ polígonos em $L$ contendo $p_0$, de forma que $\#\overrightarrow{E_<} \geq \#\overleftarrow{E_<}$, obrigando $i$ a ser 0. Concluímos então que dentre as arestas $e_0, e_1, \ldots, e_{k-1}$, apenas a aresta $e_0$ pode participar de uma boa representação de $uv$.

Em resumo, o segmento $uv$ pode ser bem representado apenas em conjunto com as arestas $e_0$ e $e_{n-1}$, o que limita nossa busca por boas representações a verificar a apenas estas duas arestas. Juntando agora o essencial do que vimos anteriormente, temos que $e_0$ participa de uma boa representação de $uv$ se, e somente se, $\#\overrightarrow{E_<} = \#\overleftarrow{E_<}$, ou seja, se nenhum polígono de $L$ contiver $p_0$ ou, em outras palavras, se $p_0 \notin U$. Similarmente, temos que $e_{n-1}$ participa de uma boa representação de $uv$ se, e somente se,

$$\#\overrightarrow{E_<} - \#\overleftarrow{E_<} = n - 1 - 2k + 1 = n - 2k,$$

que por sua vez ocorre se, e somente se,

$$\#\overrightarrow{E_<} + k = \#\overleftarrow{E_<} + (n-k) \iff \#\overrightarrow{E_<} + \#\overrightarrow{E_=} = \#\overleftarrow{E_<} + \#\overleftarrow{E_=},$$

ou seja, se não existirem polígonos em $L$ contendo $p_1$ ou, alternativamente, se $p_1 \notin U$. Sendo mais sucinto ainda, $e_0$ representa bem $uv$ se, e somente se, $p_0 \notin U$ e $e_{n-1}$ representa bem $uv$ se, e somente se, $p_1 \notin U$.

Passamos agora a mostrar o resultado diretamente. Se $uv \nsubseteq \partial U$, temos, pela proposição 40, que $p \in U^\circ$, de forma que $p_0, p_1 \in U^\circ$ e, logo, que nem $e_0$ nem $e_{n-1}$ participam de boas representações de $uv$. Vimos, no entanto, que estas eram as únicas arestas capazes de formarem uma boa representação de $uv$, de forma que $uv$ não possui uma boa representação.

Por outro lado, se $uv \subseteq \partial U$, então obviamente $p \in \partial U$ e ou $p_0 \in U$ e $p_1 \notin U$ ou $p_0 \notin U$ e $p_1 \in U$. Em outras palavras, ou $e_{n-1}$ participa de uma boa representação de $uv$ e $e_0$ não participa ou $e_0$ participa de uma boa representação de $uv$ e $e_{n-1}$ não participa. Em qualquer caso, temos que, como apenas $e_0$ e $e_{n-1}$ podem formar boa representações de $uv$, que $uv$ possui exatamente uma boa representação, mostrando o resultado. $\square$

Em face deste resultado, fica claro que se tivermos um algoritmo para descobrir todas as representações de segmentos candidatos em uma lista de polígonos, podemos obter todos os segmentos candidatos contidos na fronteira da união destes polígonos e consequentemente uma boa representação para esta união. Portanto não deve ser surpresa que a segunda ideia por trás de nosso algoritmo é um método para fazer justamente isto.

A principal ideia por trás deste método é o uso de uma ***linha de varredura***. Explicamos esta ideia primeiro assumindo que a instância $L$ do problema que estamos tentando resolver está em posição geral. Neste caso consideramos uma reta vertical, a linha de varredura, que percorre o plano da esquerda para a direita. Conforme a linha se desloca, mantemos informações sobre a interação da linha com a instância do problema, e estas informações são chamadas de o **status** da linha de varredura. Contudo, durante o curso do algoritmo, o *status* da linha de varredura pode se modificar. Abscissas em que o *status* muda são chamadas de ***pontos evento***. Dessa forma, o algoritmo funciona avançando a linha de varredura e, sempre que um ponto evento for atingido, alterando o *status*. Como veremos adiante, em posição geral, os pontos evento são as abscissas dos pontos chave.

Na verdade iremos descrever o algoritmo para uma entrada genérica de uma vez só. A principal diferença é que a linha vertical de varredura não interage com a instância $L$, mas com a instância $\delta[L]$, onde $\delta$ é um valor real positivo suficientemente pequeno. Trata-se do cisalhamento simbólico descrito na seção 4.4. A principal diferença é que a linha de varredura não é mais unicamente determinada por uma abscissa, mas por um ponto. Assim, a linha de varredura passa por pontos, e os pontos evento são, na verdade, pontos, não abscissas. Eles são inclusive os pontos chave.

O *status* da linha de varredura em um ponto $p \in \mathbb{R}^2$ é o conjunto das arestas de polígonos em $L$ que são comparáveis em $p$ junto com o registro, para cada uma destas arestas, de qual foi o último ponto chave percorrido pela linha de varredura que estava contido na aresta.

Durante a passagem por um ponto evento $p \in \mathcal{K}(L)$, o algoritmo tem três tarefas:

- Testar segmentos candidatos descobertos, como veremos adiante;

- Remover do *status* arestas $e$ tais que $\mathrm{right}(e) = p$; e
- Inserir no *status* arestas $e$ tais que $\mathrm{left}(e) = p$.

Como é usual nestes algoritmos de geometria computacional, o *status* da linha de varredura é armazenado em uma árvore de busca balanceada, que descreveremos na seção 4.7. Esta árvore tem como papel tornar os testes de segmentos candidatos extremamente eficientes.

Por último, para identificar pontos chave e as arestas que os contém, nosso algoritmo faz uso do algoritmo de Bentley e Ottmann [7] em um formato especial descrito na seção 4.6. Usamos este formato para não nos preocuparmos com os detalhes do algoritmo deles e nos focarmos no nosso.

## 4.6 O algoritmo de Bentley e Ottmann

Considere as duas seguintes definições.

> **Definição 77** — Seja $L$ uma lista de segmentos de reta. Então o ***conjunto dos pontos chave*** de $L$ é o conjunto $\mathcal{K}(L) \subseteq \mathbb{R}^2$ dos pontos do plano que pertencem a intersecções finitas de segmentos de $L$.

> **Definição 78** — Seja $L$ uma lista de segmentos de reta e seja $p \in \mathcal{K}(L)$ um ponto chave de $L$. Dizemos então que os ***índices de incidência*** de $L$ em $p$ são os elementos do conjunto $\mathcal{I}_L(p)$ dos índices de entradas em $L$ correspondentes a segmentos que contém $p$.

O algoritmo de Bentley e Ottmann [7] é um algoritmo clássico da literatura de geometria computacional. Para uma entrada $L$, uma lista de segmentos de reta, este algoritmo produz como saída todos os pares da forma $\big(p, \mathcal{I}_L(p)\big)$ com $p \in \mathcal{K}(L)$ em ordem estritamente crescente de $p$. Este formato no qual descrevemos o algoritmo de Bentley e Ottmann corresponde basicamente ao formato apresentado em [14, seção 2.1]. No restante desta seção, detalhamos este algoritmo.

Primeiramente, a entrada para o algoritmo de Bentley e Ottmann é a própria lista $L$. Cada segmento $uv$ em $L$ será representado por quatro elementos de $\mathbb{F}$ correspondendo às coordenadas de $u$ e de $v$.

A saída do algoritmo, no entanto, é mais complicada de descrever. Modelamos o algoritmo de Bentley e Ottmann como uma co-rotina, de modo que uma leitura da seção 1.6 é necessária para compreender o que vem adiante.

O algoritmo de Bentley e Ottmann é uma co-rotina Bentley-Ottmann. Desta forma, se $L$ é uma lista de segmentos de reta representada como anteriormente descrito, a execução de

$$b \leftarrow \textsc{Bentley-Ottmann}\ (L)$$

atribui à variável $b$ uma instância da co-rotina. Podemos então executar $b()$ repetidamente. O resultado de cada uma destas execuções é ou um par $\big(p, \mathcal{I}_L(p)\big)$ com $p \in \mathcal{K}(L)$

ou o valor especial $\perp$. Antes que uma destas execuções produza o valor $\perp$ pela primeira vez, as execuções anteriores terão produzido todos os pares $\big(p, \mathcal{I}_L(p)\big)$ com $p \in \mathcal{K}(L)$ exatamente uma vez e em ordem estritamente crescente de $p$. Depois desta primeira chamada produzindo $\perp$, as demais também retornarão este valor.

Um par $\big(p, \mathcal{I}_L(p)\big)$ com $p \in \mathcal{K}(L)$ produzido na saída do algoritmo será representado por dois elementos de $\mathbb{F}$ denotando as coordenadas de $p$ e por uma lista de números naturais representando a lista $\mathcal{I}_L(p)$.

Quanto às complexidades, defina primeiro $n$ como o número de entradas em $L$ (o tamanho de $L$) e $M$ como o número de pares de entradas em $L$ que correspondem à segmentos de reta que se intersectam. Assim, a execução de

$$b \leftarrow \textsc{Bentley-Ottmann}\ (L)$$

tem complexidades algébricas $\mathcal{O}\big((M+n)\log n\big)$ de tempo e $\mathcal{O}(n)$ de espaço. Ainda, executar $b()$ até a primeira vez em que $\perp$ é retornado também tem complexidades algébricas $\mathcal{O}\big((M+n)\log n\big)$ de tempo e $\mathcal{O}(n)$ de espaço. Futuras chamadas de $b()$ rodam em tempo constante consumindo uma quantidade constante de espaço adicional, mas estas chamadas nunca serão feitas em nosso algoritmo.

É importante ressaltar que iremos assumir que o algoritmo de Bentley e Ottmann é algebricamente tratável. Mostrar isto iria requerer uma análise mais a fundo do algoritmo deles e não temos espaço neste trabalho para fazê-la. Incentivamos o leitor interessado que já tenha tido contato com este algoritmo a refletir sobre este fato.

Finalmente, cabe explicar o motivo de pedirmos os índices na lista na saída e não os segmentos em si. Como o leitor pode imaginar, pretendemos, em nosso algoritmo para calcular a união de polígonos, chamar o algoritmo de Bentley e Ottmann com uma lista de segmentos correspondente à todas as arestas de polígonos na entrada do nosso problema. Neste caso, arestas de polígonos diferentes podem corresponder ao mesmo segmento de reta no plano, e assim pode haver vários segmentos repetidos na lista. Em nosso algoritmo, é crucial recuperar cada aresta contendo um determinado ponto chave, e podemos fazer isso com o índice do segmento na lista, pois existe exatamente uma aresta que deu origem à entrada correspondente ao índice.

## 4.7 Árvores rubro-negras aumentadas

Em nosso algoritmo, decidimos organizar o *status* da linha de varredura como uma árvore rubro-negra [13, capítulo 13]. As chaves nesta árvore são arestas de polígonos em uma lista $L$ de polígonos que intersectam a linha de varredura.

Como é usual em uma árvore rubro-negra, devemos especificar um ordenamento de suas chaves. Porém, a ordem de nossa árvore depende da posição da linha de varredura. Em primeiro lugar, apenas arestas que intersectam a linha de varredura podem estar na árvore. Mais precisamente, se a linha de varredura está em um ponto $p \in \mathbb{R}^2$, apenas arestas comparáveis em $p$ podem estar na árvore e, como é de se esperar, a ordem destas arestas na árvore é dada pela relação de precedência $<_p^L$.

Lembramos que a relação de ordenamento em uma árvore rubro-negra deve ser uma ordem estrita fraca, como é a ordem $<_p^L$ ($<_p^L$ é inclusive uma ordem total). Assim, duas arestas iguais (cujos polígonos de origem têm o mesmo índice na lista $L$) são consideradas equivalentes pela ordem $<_p^L$. Nesta implementação, exigimos que estas arestas jamais estejam simultaneamente na árvore, ou seja, não permitimos a inserção de um elemento repetido na árvore.

Por causa desta dependência da ordem da árvore com a posição $p \in \mathbb{R}^2$ da linha de varredura, as rotinas relativas a árvores rubro-negras que fazem uso da ordem recebem como parâmetro adicional esta posição $p$. Quando estas rotinas são chamadas, é necessário que a árvore esteja ***consistente*** com $p$, ou seja, que todas as entradas na árvore correspondam a arestas comparáveis em $p$ e que estas entradas estejam ordenadas segundo a ordem $<_p^L$. Assim, se duas destas rotinas são sucessivamente chamadas com argumentos $p_0, p_1 \in \mathbb{R}^2$, respectivamente, para a posição da linha de varredura, ao final da primeira rotina, a árvore deve conter apenas arestas comparáveis em ambos $p_0$ e $p_1$ e, restritas às arestas da árvore, as ordens $<_{p_0}^L$ e $<_{p_1}^L$ devem ser iguais.

Em [13, capítulo 14], um método de "aumentar" árvores rubro-negras é descrito. Usando esta ideia, nós podemos modificar nossa estrutura de dados para, dada uma chave $e$ (não necessariamente na árvore), uma direção $d \in \{\rightarrow, \leftarrow\}$ e uma posição $p \in \mathbb{R}^2$ da linha de varredura com a qual a árvore é consistente, devolver o número de chaves $e'$ na árvore tais que $e' <_p^L e$ (ou também $e' \leq_p^L e$) e $\mathrm{dir}(e') = d$. Note que isto é exatamente o que precisamos para decidir se uma representação de um segmento candidato é uma boa representação, como discutido na seção 4.5.

Descrevemos agora quais rotinas assumimos implementadas para tratar com árvores rubro-negras:

- A rotina RB-Init recebe uma referência para uma lista de polígonos $L$ e devolve uma árvore rubro-negra vazia que aceita como chaves arestas de polígonos em $L$;
- A rotina RB-Insert recebe uma árvore rubro-negra $T$, uma chave $e$ que não está na árvore $T$ e uma posição $p \in \mathbb{R}^2$ da linha de varredura tal que a árvore $T$ é consistente com $p$ e tal que $e$ é comparável em $p$. Ela então insere a chave $e$ na árvore $T$ na sua devida posição de acordo com a ordem $<_p^L$.
- A rotina RB-Remove recebe uma árvore rubro-negra $T$ e uma chave $e$ presente na árvore $T$ e a remove da árvore $T$, mantendo as chaves restantes em sua ordem relativa anterior. Note que a posição da linha de varredura (e consequentemente a ordem da árvore) não é usada nesta rotina;
- A rotina RB-Query-Strict recebe uma árvore rubro-negra $T$, uma chave $e$ (não necessariamente na árvore), uma direção $d \in \{\rightarrow, \leftarrow\}$ e uma posição $p \in \mathbb{R}^2$ para a linha de varredura tal que a árvore $T$ é consistente com $p$ e tal que $e$ é comparável em $p$. Ela então devolve o número de chaves $e'$ na árvore $T$ tais que $e' <_p^L e$ e $\mathrm{dir}(e') = d$; e
- Similarmente, a rotina RB-Query recebe uma árvore rubro-negra $T$, uma chave $e$ (não necessariamente na árvore), uma direção $d \in \{\rightarrow, \leftarrow\}$ e uma posição $p \in \mathbb{R}^2$ para a linha de varredura tal que a árvore $T$ é consistente com $p$ e tal que $e$ é comparável em $p$. Ela então devolve o número de chaves $e'$ na árvore $T$ tais que $e' \leq_p^L e$ e $\mathrm{dir}(e') = d$.

As rotinas RB-Insert, RB-Query-Strict e RB-Query todas têm complexidades algébricas $\mathcal{O}(\log n)$ de tempo e $\mathcal{O}(1)$ de espaço, onde $n$ é o número de chaves atualmente na árvore $T$. A rotina RB-REMOVE, por não usar a ordem e consequentemente nenhuma operação algébrica sobre o subcorpo real $\mathbb{F}$, possui complexidades $\mathcal{O}(\log n)$ de tempo e $\mathcal{O}(1)$ de espaço. Finalmente, a rotina RB-INIT consome tempo e espaço constantes. De fato, a lista $L$ em si jamais é usada diretamente na estrutura de dados. O argumento para a rotina RB-INIT é um mero formalismo, pois a árvore precisa saber apenas dos índices dos polígonos de origem das arestas e nada sobre a lista em si.

Observe que duas arestas que correspondem a um mesmo segmento de reta e possuem a mesma direção são consideradas distintas desde que não sejam arestas do mesmo polígono, isto é, a aresta carrega consigo informação sobre o seu polígono de origem. Esta informação é não sobre o polígono enquanto um conjunto de pontos, mas sobre ele enquanto estrutura de dados (mais precisamente, seu índice na lista $L$). Assim, se uma lista de polígonos possui duas cópias do mesmo polígono (como na entrada para o problema 2), duas arestas, uma de cada cópia, são consideradas distintas. Isso é importante não só por causa da rotina de inserção e remoção em uma árvore rubro-negra, mas também para que as rotinas de contagem devolvam o valor correto. Este detalhe será discutido novamente, sobre uma outra perspectiva, quando a implementação do algoritmo for detalhada na seção 4.8.

## 4.8 O algoritmo

Finalmente estamos prontos para apresentar o algoritmo 3, que calcula a união de polígonos.

No restante desta seção, explicamos como o algoritmo funciona, detalhamos sua implementação, mostramos que ele é correto e analisamos suas complexidades.

A primeira coisa que devemos observar é que o número $N$ usado no algoritmo é o número total de vértices nos polígonos de $L$. Mais precisamente, se $L = P_0 P_1 \cdots P_{n-1}$, então

$$N = \sum_{i=0}^{n-1} |V(P_i)|.$$

Consequentemente, note que se $i, j \in \{0, 1, \dots, N-1\}$ e $i \neq j$, então $e_i \neq e_j$, mesmo que $e_i$ e $e_j$ correspondam aos mesmos segmentos de reta no plano, tenham as mesmas orientações e tenham sido originadas de polígonos na lista que representam o mesmo conjunto de pontos no plano (mas que claramente devem possuir índices distintos na lista). Este ponto é muito importante para o algoritmo, pois $e_i$ e $e_j$ devem, se necessário, estar ao mesmo tempo na árvore rubro-negra.

Note agora que, no laço da linha 6, a variável $p$ itera exatamente sobre os pontos chave de $L$ em ordem estritamente crescente por causa do algoritmo de Bentley e Ottmann. O *status* da linha de varredura é mantido na árvore rubro-negra $T$ (as arestas que intersectam a linha de varredura) e nas variáveis $\ell_i$ com $i \in \{0, 1, \dots, N-1\}$, que registra o último ponto chave percorrido contido na aresta $e_i$. Sendo assim, o algoritmo tem o dever de atualizar o *status* na passagem pelo evento $p$ nas linhas de 7 a 34.

De fato, nas linhas de 24 a 28, o algoritmo remove arestas terminando em $p$ do *status* e, nas linhas de 7 a 9, insere arestas começando em $p$ no *status*. Já nas linhas de 10 a 23, ele

**Algorithm 3** UNION-OF-POLYGONS $(L)$

`Pré-condição:` $L$ é uma entrada para o problema 2

`Efeito:` devolve uma saída para o problema 2 correspondente à instância $L$ como uma co-rotina

```
Sejam e_0, e_1, ..., e_{N-1} as arestas dos polígonos em L.
Seja S = s_0 s_1 ··· s_{N-1} uma lista de segmentos de reta onde s_i é o segmento de reta
  corresponde à aresta e_i para todo i ∈ {0, 1, ..., N − 1}.
Para todo i ∈ {0, 1, ..., N − 1}, seja ℓ_i ∈ F^2 uma variável não inicializada.
T ← RB-INIT (L)
I ← {}
for (p, I_S(p)) in BENTLEY-OTTMANN (S) do
  for each i ∈ I do
    RB-INSERT (T, e_i, p)
  done
  for i in I_S(p) do
    if left(e_i) ≠ p then
      a ← RB-QUERY-STRICT (T, e_i, →, p)
      b ← RB-QUERY (T, e_i, ←, p)
      if a = b then
        if dir(e_i) = → then
          yield →(ℓ_i p)
        else
          yield →(p ℓ_i)
        endif
      endif
    endif
    ℓ_i ← p
  done
  for i in I_S(p) do
    if left(e_i) ≠ p then
      RB-REMOVE (T, e_i)
    endif
  done
  I ← {}
  for i in I_S(p) do
    if right(e_i) ≠ p then
      I ← I ∪ {i}
    endif
  done
done
```

identifica representações de segmentos candidatos que terminam em $p$, as testa de acordo com a definição 76, as coloca na saída se necessário (de acordo com a proposição 55) e finalmente atualiza os valores de $\ell_i$ com $i \in \mathcal{I}_S(p)$.

Existem vários detalhes de implementação deste algoritmo aos quais gostaríamos de chamar a atenção, talvez o mais importante dos quais é o que são, computacionalmente falando, as arestas $e_0, e_1, \ldots, e_{N-1}$. Em outras palavras, se entendemos $e_0, e_1, \ldots, e_{N-1}$ como um vetor $E$, o que tem em cada entrada de $E$?

Para todo $i \in \{0, 1, \ldots, N-1\}$, a $i$-ésima entrada de $E$ deve conter ao menos os extremos $\text{left}(e_i), \text{right}(e_i) \in \mathbb{F}^2$ como pontos em $\mathbb{F}^2$, a direção $\text{dir}(e_i)$ da aresta $e_i$, o índice em $L$ do polígono que possui a aresta (dependendo da implementação) e, um pouco menos óbvio, o nó da árvore rubro-negra em que a aresta $e_i$ se encontra.

Se formos um pouco mais fundo nos detalhes de implementação, veremos que a árvore rubro-negra é uma estrutura de dados que usa um registro de tamanho fixo para cada um de seus nós e mais apenas uma quantidade fixa de memória. Assim, se passarmos uma referência para um registro desocupado destes na rotina de inserção na árvore, podemos completamente evitar alocações dinâmicas de memória pela árvore rubro-negra. Com isso, o mais lógico a se fazer é alocar um registro em cada entrada do vetor $E$. Ainda, nas linhas 8, 12, 13 e 26, onde o algoritmo passa arestas para as rotinas de árvores rubro-negras, é conveniente entender que um ponteiro para uma entrada em $E$ está sendo passado para a rotina. Com ele a árvore pode usar o registro alocado na entrada para gerenciar a estrutura de dados, mas, mais importante que isso, saber qual nó da árvore está sendo removido na rotina de remoção.

Outro detalhe é o vetor $\mathcal{I}_S(p)$. Como o tamanho máximo deste vetor é $N$, podemos pré-alocar um vetor de tamanho $N$ de índices (elementos de $\{0, 1, \ldots, N-1\}$) e entender que a instância da co-rotina Bentley-Ottmann recebe um ponteiro para um vetor de índices onde ela possa depositar o vetor $\mathcal{I}_S(p)$ em sua saída e que ela retorna quantos elementos foram escritos.

Podemos implementar de forma similar o vetor $I$ que guarda quais índices de arestas serão inseridas na árvore no início da próxima iteração. Alocamos um vetor de $N$ índices (elementos de $\{0, 1, \ldots, N-1\}$) e implementamos uma estrutura de dados elementar, como uma pilha, em cima deste vetor. Isto é apropriado porque, como incentivamos o leitor a verificar, jamais inserimos no conjunto $I$ elementos repetidos na linha 32. Ainda, todos os elementos inseridos nesta linha são claramente números naturais menores do que $N$.

Finalmente, o algoritmo de Bentley e Ottmann pode ser implementado usando uma árvore rubro-negra também. A árvore rubro-negra neste algoritmo não é uma árvore rubro-negra aumentada no sentido de que ela não possui as funções de contagem que nossa árvore possui. Porém, nossa árvore é na verdade mais geral do que a do algoritmo deles, de forma que podemos manter, se a implementação for feita com cuidado, apenas uma árvore rubro-negra para o algoritmo inteiro. Se esta otimização não for feita, o algoritmo estará na verdade mantendo duas árvores rubro-negras, uma das quais possui todas as informações da outra.

Tendo discutido estes detalhes, passamos agora a mostrar que o algoritmo é correto.

**Teorema 7** — O algoritmo 3 corretamente resolve o problema 2.

**Prova** — Seja $E = \{e_0, e_1, \ldots, e_{N-1}\}$ o conjunto das arestas de polígonos em $L$, de forma que $|E| = N$, ou seja, duas arestas originadas de polígonos de índices distintos em $L$ correspondem a elementos diferentes em $E$ mesmo que elas representem o mesmo segmento de reta no plano, mesmo que elas possuam a mesma direção e mesmo que seus polígonos de origem correspondam ao mesmo conjunto de pontos no plano. Seja ainda, para cada ponto $p \in \mathbb{R}^2$, $E(p)$ o conjunto das arestas em $E$ comparáveis em $p$ e $C(p)$ o conjunto das arestas em $E$ contendo $p$.

Vamos mostrar por uma invariante de laço que, toda vez que o algoritmo termina o laço das linhas 7 a 9, a árvore rubro-negra $T$ não só está consistente com o ponto $p$, mas contém como chaves precisamente as arestas em $E(p)$.

O caso base desta invariante de laço é a primeira iteração do laço das linhas 6 a 35. Nesta iteração, a árvore $T$ claramente começa vazia e o conjunto $I$ também está vazio. Assim, o laço das linhas 7 a 9 não faz nada e a árvore rubro-negra permanece vazia. Porém, sendo a primeira iteração, $p$ é o menor ponto chave, pois o algoritmo de Bentley e Ottmann devolve os pontos chave em ordem estritamente crescente. Logo, toda aresta $e \in E$ deve ser tal que $\text{left}(e) \geq p$, de forma que $e$ não é comparável em $p$. Logo devemos ter $E(p) = \{\}$, de forma que a árvore $T$ não só está consistente com $p$, mas contém exatamente as arestas em $E(p)$.

Mostramos agora o passo indutivo, ou seja, que a invariante sempre se mantém entre duas iterações do laço das linhas 6 a 35. Para isso, considere que em uma destas iterações, o laço das linhas 7 a 9 acabou de ser completado e que, nesta iteração, a variável $p$ está assinalada a um ponto chave $p_0 \in \mathcal{K}(L)$. Considere ainda que a hipótese indutiva está valendo, ou seja, que a árvore $T$ está consistente com $p_0$ e possui como chaves exatamente as arestas em $E(p_0)$. Suponha também que esta não é a última iteração do laço das linhas 6 a 35. Desta forma, $p_0$ não pode ser o maior ponto chave e deve haver exatamente um ponto chave sucessor de $p_0$, ou seja, um ponto chave $p_1 \in \mathcal{K}(L)$ com $p_0 < p_1$ tal que não existe um ponto chave $q \in \mathcal{K}(L)$ com $p_0 < q < p_1$. Assim, na iteração seguinte, a variável $p$ irá carregar o valor $p_1$ e devemos mostrar que, imediatamente após o laço das linhas 7 a 9 ser completado nesta próxima iteração, a árvore $T$ estará consistente com $p_1$ e terá como chaves exatamente as arestas em $E(p_1)$.

Observe então que as linhas de 10 a 23 não modificam a árvore rubro-negra, fazendo nela apenas consultas. Estas consultas envolvem apenas arestas $e$ em $C(p_0)$ tais que $\text{left}(e) \neq p_0$, que estão portanto todas em $E(p_0)$. Assim, como estas consultas são feitas com $p_0$ como a posição da linha de varredura, elas são todas legais. Desta forma, podemos ignorar as linhas de 10 a 23 ao mostrar esta invariante de laço.

O laço das linhas 24 a 28, por sua vez, remove da árvore as arestas em $C(p_0)$ que não possuem $p$ como seu extremo esquerdo, ou seja, as arestas em $C(p_0) \cap E(p_0)$. Como obviamente todas estas arestas estão em $E(p_0)$, elas estavam todas na árvore, de modo que todas estas operações de remoção são legais. Assim, após a execução deste laço, a árvore tem como arestas exatamente as arestas em $E(p_0) \setminus C(p_0)$.

Mostramos agora que, após a execução deste laço das linhas 24 a 28, a árvore rubro-negra $T$ está consistente com o ponto $p_1$. De fato, vimos que a árvore, neste momento, possui como chaves exatamente as arestas em $E(p_0) \setminus C(p_0)$, de forma que uma aresta $e$ na árvore deve ser tal que $\text{left}(e) < p_0 < \text{right}(e)$, já que $\text{right}(e) \neq p_0$ porque $e \notin C(p_0)$. Assim, como $\text{right}(e) \in \mathcal{K}(L)$, temos que $\text{left}(e) < p_0 < p_1 \leq \text{right}(e)$, de forma que $e$ é comparável em $p_1$. Logo, a árvore contém então apenas arestas comparáveis em $p_0$ e em $p_1$. Se $e_0$ e $e_1$ são então duas arestas na árvore, observe que $p_0$ não pode estar na intersecção de $e_0$ e $e_1$, pois $p_0$ não pode estar nem em $e_0, e_1 \notin C(p_0)$. Assim, não existe um ponto chave $q \in \mathcal{K}(L)$ contido em $e_0$ e $e_1$ tal que $p_0 \leq q < p_1$. Logo a proposição 54 nos garante que $e_0 <^L_{p_0} e_1$ se, e somente se, $e_0 <^L_{p_1} e_1$. Em outras palavras, dentre as arestas da árvore, as ordens $<^L_{p_0}$ e $<^L_{p_1}$ são equivalentes. Agora, como vimos, a árvore rubro-negra $T$ está consistente com $p_0$. Porém, dado que todas as arestas na árvore estão em $E(p_0) \cap E(p_1)$ e dada a equivalência entre estas duas ordens, temos que a árvore também está consistente com $p_1$.

Agora, após as linhas de 29 a 34, claramente o conjunto $I$ contém exatamente os índices $i \in \{0, 1, \ldots, N-1\}$ tais que $e_i \in C(p_0)$ e $\text{right}(e_i) \neq p_0$. Vamos mostrar agora que estas arestas são precisamente as arestas no conjunto $C(p_0) \cap E(p_1)$. De fato, se uma aresta $e \in E$ é tal que $e \in C(p_0)$ e $\text{right}(e) \neq p_0$, temos que $\text{left}(e) \leq p_0 < p_1 \leq \text{right}(e)$ (pois $\text{right}(e) \in \mathcal{K}(L)$), de forma que $e \in C(p_0) \cap E(p_1)$. Conversamente, se uma aresta $e$ está em $C(p_0) \cap E(p_1)$, então $\text{left}(e) \leq p_0 < p_1 \leq \text{right}(e)$ (pois $\text{left}(e) \in \mathcal{K}(L)$), de forma que $e$ está em $C(p_0)$ e $\text{right}(e) \neq p_0$.

Desta forma, o laço das linhas 7 a 9 na iteração seguinte irão inserir na árvore (na posição $p_1$), precisamente as arestas em $C(p_0) \cap E(p_1)$. A árvore, antes da execução deste laço, tem como chave precisamente as arestas em $E(p_0) \setminus C(p_0)$, e assim nenhuma aresta repetida será inserida. Ainda, vimos que, antes da execução deste laço, a árvore está consistente com $p_1$, de forma que todas as inserções realizadas são legais. Observe então que, ao final deste laço a árvore terá como chaves precisamente as arestas em

$$
\begin{aligned}
& \big(E(p_0) \setminus C(p_0)\big) \cup \big(C(p_0) \cap E(p_1)\big) \\
= \; & \Big(\big(E(p_0) \setminus C(p_0)\big) \cup C(p_0)\Big) \cap \Big(\big(E(p_0) \setminus C(p_0)\big) \cup E(p_1)\Big) \\
= \; & \big(E(p_0) \cup C(p_0)\big) \cap E(p_1) \\
= \; & E(p_1).
\end{aligned}
$$

De fato, já vimos que toda aresta em $E(p_0) \setminus C(p_0)$ está em $E(p_1)$. Por outro lado, se $e \in E(p_1)$ é uma aresta, então $\text{left}(e) \leq p_0 < p_1 \leq \text{right}(e)$ (pois $\text{left}(e) \in \mathcal{K}(L)$), de forma que $e \in E(p_0)$ se $\text{left}(e) \neq p_0$ e que $e \in C(p_0)$ se $\text{left}(e) = p_0$. Assim $E(p_1) \subseteq E(p_0) \cup C(p_0)$.

Concluímos então que, após o término do laço das linhas 7 a 9, a árvore estará consistente com $p_1$ e terá como arestas precisamente as arestas em $E(p_1)$, provando a nossa invariante de laço correta. Na verdade, provamos até mais do que esta invariante: mostramos que todas as operações realizadas sobre a árvore rubro-negra $T$ neste algoritmo são legais e estabelecemos quais arestas estão na árvore em cada momento em uma iteração.

A argumentação até aqui estabeleceu completamente o comportamento da árvore rubro-negra, que é uma das duas estruturas de dados necessárias para a operação do laço das linhas 10 a 23, que constitui o cerne do algoritmo. A outra estrutura de dados é mais simples: são os valores de $\ell_i$ com $i \in \{0, 1, \ldots, N-1\}$. Iremos agora mostrar correta uma outra invariante de laço, que diz respeito a estes valores. Mais precisamente, iremos mostrar que, no início de cada iteração do laço das linhas 6 a 35, para todo índice $i \in \{0, 1, \ldots, N-1\}$ tal que $\text{left}(e_i) < p$, $\ell_i$ é um ponto chave de $L$ que está em $e_i$, precede $p$ e é tal que não existe um ponto chave $q \in \mathcal{K}(L)$ em $e_i$ com $\ell_i < q < p$.

O caso base é a primeira iteração deste laço. Nela, como o algoritmo de Bentley e Ottmann devolve os pontos chave em ordem estritamente crescente, temos que $p$ é o menor ponto chave, ou seja, não existe um ponto chave $q \in \mathcal{K}(L)$ com $q < p$. Por isso, toda aresta $e \in E$ é tal que $\text{left}(e) \geq p$, já que $\text{left}(e) \in \mathcal{K}(L)$. Assim, nossa invariante de laço vale por vacuidade.

Suponha agora que o algoritmo está no início de uma iteração do laço das linhas 6 a 35 e que a variável $p$ guarda um ponto chave $p_0 \in \mathcal{K}(L)$ que não é o maior ponto chave, de forma que irá haver uma próxima iteração. Não sendo $p_0$ o maior ponto chave, deve existir um único ponto chave $p_1 \in \mathcal{K}(L)$ com $p_0 < p_1$ tal que não existe um ponto chave $q \in \mathcal{K}(L)$ com $p_0 < q < p_1$. Este ponto chave $p_1$ é o sucessor de $p_0$ em $\mathcal{K}(L)$ e será o valor armazenado na variável $p$ na próxima iteração do laço. Devemos mostrar então que, nesta próxima iteração, todo valor de $i \in \{0, 1, \ldots, N-1\}$ com $\text{left}(e_i) < p_1$ é tal que $\ell_i$ é um ponto chave em $e_i$ que precede $p_1$ tal que não existe um ponto chave $q \in \mathcal{K}(L)$ em $e_i$ com $\ell_i < q < p_1$. Para isso, podemos assumir a invariante de laço valendo na iteração atual, ou seja, que todo índice $i \in \{0, 1, \ldots, N-1\}$ com $\text{left}(e_i) < p_0$ é tal que $\ell_i$ é um ponto chave em $e_i$ que precede $p_0$ tal que não existe um ponto chave $q \in \mathcal{K}(L)$ em $e_i$ com $\ell_i < q < p_0$.

Verificaremos isto para cada valor de $i \in \{0, 1, \ldots, N-1\}$ tal que $\text{left}(e_i) < p_1$ individualmente. Se um tal valor de $i$ é tal que $p_0$ não está na aresta $e_i$, então $e_i \notin C(p_0) \iff i \notin \mathcal{I}_S(p_0)$, de forma que o laço das linhas 10 a 23 não altera o valor de $\ell_i$. Como esta é a única parte do laço das linhas 6 a 35 que poderia alterar este valor, segue que, até a próxima iteração, o valor de $\ell_i$ permanece o mesmo. Porém, como $p_0$ não está em $e_i$, temos que $\text{left}(e_i) < p_0$, pois $\text{left}(e_i) \in \mathcal{K}(L)$, $\text{left}(e_i) < p_1$ e $\text{left}(e_i) \neq p_0$. Assim, vale a invariante de laço, ou seja, $\ell_i$ é um ponto chave em $e_i$ que precede $p_0$ e não existe um ponto chave $q \in \mathcal{K}(L)$ em $e_i$ com $\ell_i < q < p_0$. Assim, como $p_0$ não está em $e_i$, na verdade não há um ponto chave $q \in \mathcal{K}(L)$ em $e_i$ com $\ell_i < q < p_1$, mostrando a manutenção da invariante de laço para o índice $i$ (note que claramente $\ell_i$ precede $p_1$).

Por outro lado, se $i \in \{0, 1, \ldots, N-1\}$, $\text{left}(e_i) < p_1$ e $p_0$ está em $e_i$, então novamente observamos que $e_i \in C(p_0) \iff i \in \mathcal{I}_S(p_0)$, de forma que a linha 22 altera o valor de $\ell_i$ para $p_0$. Como o restante do algoritmo não modifica este valor de $\ell_i$, temos que, na iteração seguinte, valerá que $\ell_i = p_0$. Assim, claramente $\ell_i$ será um ponto chave em $e_i$ que precede $p_1$ e tal que não existe um ponto chave $q \in \mathcal{K}(L)$ com $\ell_i = p_0 < q < p_1$ ($p_1$ é o sucessor de $p_0$). Assim a invariante de laço também se mantém para este valor de $i$. Tendo esgotado as possibilidades dos índices $i \in \{0, 1, \ldots, N-1\}$ com $\text{left}(e_i) < p_1$, provamos correta mais esta invariante de laço.

O que resta fazer agora é estudar o laço das linhas 10 a 23 sob a luz destas duas invariantes de laço. Primeiramente, observe que, quando o algoritmo está prestes a executar a linha 12, $e_i$ é uma aresta em $C(p)$ com $\text{left}(e_i) \neq p$, de forma que $\text{left}(e_i) < p$ e, pela nossa segunda invariante de laço, $\ell_i$ é um ponto chave de $L$ contido na aresta $e_i$ que precede $p$ e tal que não há um ponto chave $q \in \mathcal{K}(L)$ também contido em $e_i$ tal que $\ell_i < q < p$. Desta forma, temos que $\ell_i p$ é um segmento candidato de $L$ e que, contendo $\ell_i p$, a aresta $e_i$

participa de uma representação deste segmento candidato (isto é, se $e_i$ é a $b$-ésima aresta do $a$-ésimo polígono em $L$, então $(a, b, \ell_i, p)$ é uma representação de $\ell_i p$). Em resumo, cada vez que o algoritmo passa pela linha 12, a aresta $e_i$, juntamente com $\ell_i p$, forma uma representação de $\ell_i p$. Ainda, esta representação nunca será repetida em outra passagem pela linha 12, pois sempre devemos ter que $\ell_i < p$.

Por outro lado, seja $(a, b, u, v)$ uma representação de um segmento candidato $uv$ de $L$ com $u < v$, onde a $b$-ésima aresta do $a$-ésimo polígono em $L$, que deve conter o segmento $uv$, é a aresta $e_i$, onde $i \in \{0, 1, \ldots, N-1\}$. Assim, na iteração do laço das linhas 6 a 35 em que $p$ vale $v$, teremos que $e_i \in C(p) \iff i \in \mathcal{I}_S(p)$, de forma que o laço das linhas 10 a 23 realizará exatamente uma iteração com este valor de $i$. Nesta iteração, teremos $p = v$ e deveremos ter também que $\ell_i = u$, pois este é o único ponto chave que pode estar armazenado em $\ell_i$ para satisfazer a segunda invariante de laço que demonstramos. Assim, a representação considerada nesta iteração é exatamente a representação $(a, b, u, v)$. Em outras palavras, nas suas passagens pela linha 12, o algoritmo considera toda representação de segmentos candidatos de $L$ e, como vimos, ele considera cada tal representação exatamente uma vez.

Vejamos agora o que o restante das linhas 12 a 20 faz com uma tal representação. A primeira invariante de laço que mostramos estabelece que, na árvore rubro-negra $T$ neste momento, há chaves correspondentes a exatamente as arestas em $E(p)$, que são as arestas de polígonos em $L$ comparáveis em $p$. Então, na linha 14, de acordo com a definição das rotinas RB-Query-Strict e RB-Query, ele testa se o número de arestas de ida em polígonos em $L$ comparáveis em $p$ que precedem $e_i$ em $L$ no ponto $p$ é igual ao número de arestas de volta em polígonos em $L$ comparáveis em $p$ que não são precedidas por $e_i$ em $L$ no ponto $p$. Assim, o teste tem um resultado positivo se, e somente se, $e_i$ e $\ell_i p$ formam uma boa representação do segmento candidato $\ell_i p$.

Portanto, as linhas de 15 a 19 são executadas precisamente uma vez para cada boa representação de segmentos candidatos de $L$, estando uma tal boa representação codificada na aresta $e_i$ e no segmento $\ell_i p$. É simples verificar que estas linhas então produzem na saída um segmento de reta orientado correspondente à orientação do segmento candidato $\ell_i p$.

Assim, pela proposição 55, a saída do algoritmo é constituída precisamente das orientações dos segmentos candidatos de $L$ contidos na fronteira da união dos polígonos em $L$, cada um destes segmentos orientados estando presente na saída exatamente uma vez. Logo, pela proposição 42, o algoritmo realmente produz uma boa representação da união dos polígonos em $L$, estando correto. □

Analisamos agora as complexidades deste algoritmo. Lembramos ao leitor que, como foi discutido na seção 1.5, se pudermos determinar as complexidades algébricas de nosso algoritmo e se pudermos mostrar que ele é algebricamente tratável, então teremos imediatamente um limite superior para suas complexidades.

> **Lema 11** — O algoritmo 3, quando executado em uma lista de polígonos cujo total de vértices é $N$ e cujo número de pares de arestas distintas que se intersectam é $M$, possui complexidades algébricas $\mathcal{O}(M \log N)$ de tempo e $\mathcal{O}(N)$ de espaço.

**Prova** — Analisamos primeiro a complexidade algébrica de tempo do algoritmo. Claramente as linhas de 1 a 5 são executadas em tempo $\mathcal{O}(N)$. Ainda, o custo algébrico de simplesmente iterar o laço das linhas de 6 a 35 é $\mathcal{O}\big((M+N)\log N\big) = \mathcal{O}(M \log N)$

(observar que $N \leq M$), de acordo com o que vimos na seção 4.6.

Observando agora as linhas de 29 a 34, temos que $|I| \leq |\mathcal{I}_S(p)| = |C(p)|$. Também é claro que a árvore rubro-negra $T$ tem sempre no máximo $N$ chaves. Assim, o custo algébrico total do laço das linhas de 7 a 9 é $\mathcal{O}\big(|C(p)| \log N\big)$, sendo $p$ o ponto chave na iteração anterior (se houver uma). Ainda, sendo o custo de todos eles dominado pelo custo das operações na árvore rubro-negra, os demais sub-laços do laço das linhas 6 a 35 também possuem custos algébricos $\mathcal{O}\big(|C(p)| \log N\big)$, agora sendo $p$ o ponto chave da iteração atual.

Portanto, o custo algébrico deste laço como um todo é assintoticamente limitado por

$$\sum_{p \in \mathcal{K}(L)} |C(p)| \log N = \left( \sum_{p \in \mathcal{K}(L)} |C(p)| \right) \log N \leq \left( \sum_{p \in \mathcal{K}(L)} \binom{|C(p)|}{2} \right) \log N \leq M \log N,$$

pois:

- $M$ é, por definição, o número de pares de arestas de polígonos em $L$ que se intersectam;
- Existem exatamente $\binom{|C(p)|}{2}$ tais pares que possuem, como intersecção finita, um dado ponto chave $p \in \mathcal{K}(L)$; e
- Sendo $p \in \mathcal{K}(L)$ um ponto chave, devemos sempre ter $|C(p)| \geq 2$.

Assim, a complexidade algébrica de tempo do algoritmo como um todo é também $\mathcal{O}(M \log N)$.

Para calcular agora a complexidade algébrica de espaço deste algoritmo, limitamos o espaço ocupado pelas estruturas de dados das quais ele faz uso:

- As representações $e_0, e_1, \ldots, e_{N-1}$ das arestas de polígonos em $L$ tem claramente um custo de espaço algébrico da ordem de $\mathcal{O}(N)$;
- O mesmo pode claramente ser dito da lista $S$ de segmentos de reta;
- Idem para as variáveis $\ell_i$ com $i \in \{0, 1, \ldots, N-1\}$;
- As estruturas de dados usadas pelo algoritmo de Bentley e Ottmann, sejam lá quais forem elas, também ocupam um espaço algébrico de no máximo $\mathcal{O}(N)$ de acordo com o que foi discutido na seção 4.6;
- Como vimos, o conjunto $I$ tem sempre no máximo $N$ elementos e o espaço ocupado por ele (espaço real, não algébrico) é $\mathcal{O}(N)$; e
- A árvore rubro-negra $T$ tem sempre no máximo $N$ chaves e não necessita do armazenamento de elementos de $\mathbb{F}$, ocupando um espaço real limitado por $\mathcal{O}(N)$.

Logo, o algoritmo como um todo tem um custo algébrico de espaço $\mathcal{O}(N)$. □

**Lema 12** — O algoritmo 3 é algebricamente tratável.

**Prova** — Esta prova é extremamente similar a da demonstração do lema 6 e recomendamos a leitura daquela prova antes da desta. Ainda, como a recomendação existente naquela prova, recomendamos que o leitor também tenha lido a seção 1.5, onde o conceito de tratabilidade algébrica e o conceito de níveis de complexidade algébrica são introduzidos.

Como a demonstração do lema 6, a demonstração deste lema consiste em fazer uma inspeção do código fonte do algoritmo 3 linha a linha para mostrar que existe um limite superior para os níveis de complexidade algébrica de valores que podem ser gerados nestas linhas.

Mais especificamente, novamente como na prova do lema 6, dizemos que uma expressão no código do algoritmo é algebricamente tratável, ou simplesmente tratável, se existir um número natural maior do que todos os níveis de complexidade algébrica de valores correspondentes a elementos de $\mathbb{F}$ computados ou presentes naquela expressão em qualquer que seja a entrada do algoritmo. Dizemos também que uma linha do algoritmo é tratável se todas as suas expressões também o forem. Observe que operações que envolvem apenas elementos da entrada e as operações básicas são sempre tratáveis, uma vez que elementos na entrada têm nível de complexidade algébrica igual a 0.

Um outro detalhe importante que assumimos e iremos deixar que o leitor verifique é que as ordens $<_p^L$ com $p \in \mathcal{K}(L)$ podem ser implementadas de forma algebricamente tratável. A ideia é que as alturas e os produtos cruzados envolvidos nestas ordens podem ser computados com um número finito de operações.

Fazemos então a anunciada inspeção do código do algoritmo 2:

- Linhas 1 e 2: apenas copia valores em $\mathbb{F}$;
- Linha 3: apenas declara variáveis;
- Linhas 4 e 5: não trabalha com valores em $\mathbb{F}$;
- Linha 6: o algoritmo de Bentley e Ottmann é algebricamente tratável;
- Linha 7: não envolve elementos de $\mathbb{F}$;
- Linha 8: não cria novos valores em $\mathbb{F}$, apenas faz as comparações algebricamente tratáveis mencionadas anteriormente na inserção na árvore;
- Linha 9: apenas sintaxe;
- Linha 10: não envolve elementos de $\mathbb{F}$;
- Linha 11: apenas faz uma comparação;
- Linhas 12 e 13: não criam novos elementos em $\mathbb{F}$, apenas fazem comparações (algebricamente tratáveis) nas rotinas da árvore rubro-negra;
- Linhas 14 e 15: não envolvem elementos de $\mathbb{F}$;
- Linha 16 e 18: apenas colocam elementos de $\mathbb{F}$ na saída;
- Linha 17, 19, 20 e 21: apenas sintaxe;
- Linha 22: apenas copia elementos de $\mathbb{F}$;
- Linha 23: apenas sintaxe;
- Linha 24: não envolve elementos de $\mathbb{F}$;

- Linha 25: faz apenas uma comparação;
- Linha 26: a rotina de remoção da árvore não trabalha com elementos de $\mathbb{F}$;
- Linha 27 e 28: apenas sintaxe;
- Linha 29 e 30: não envolvem elementos de $\mathbb{F}$;
- Linha 31: faz apenas uma comparação;
- Linha 32: não envolve elementos de $\mathbb{F}$; e
- Linhas 33 a 35: apenas sintaxe.

Como na prova do lema 6, devemos concluir que todas as expressões no algoritmo são algebricamente tratáveis e logo que existe um número natural maior do que todos os níveis de complexidade algébrica de todos os valores computados pelo algoritmo em qualquer que seja sua entrada. Assim, temos que o algoritmo 3 é de fato algebricamente tratável. □

**Teorema 8** — Existem constantes $k_0, k_1 \in \mathbb{N}$ tais que o algoritmo 3, quando executado em uma entrada cujo número total de vértices é $N$, cujo número de pares de arestas distintas que se intersectam é $M$ e cujo número máximo de *bits* em uma representação computacional de um elemento de $\mathbb{F}$ é $B$, possui complexidades $\mathcal{O}(M \log NB^{k_0})$ de tempo e $\mathcal{O}(NB^{k_1})$ de espaço.

**Prova** — Consequência direta dos lemas 11 e 12 e do que foi discutido na seção 1.5. □

# Bibliografia


[1] CGAL – the computational geometry algorithms library. `https://www.cgal.org`, 2014. [Online; accessed 30-Sep-2014].

[2] GMP – the GNU multiple precision arithmetic library. `https://www.gmplib.org`, 2014. [Online; accessed 26-Aug-2014].

[3] LEDA – library of efficient data types and algorithms. `http://www.algorithmic-solutions.com/leda/index.htm`, 2014. [Online; accessed 30-Sep-2014].

[4] A. M. Andrew. Another efficient algorithm for convex hulls in two dimensions. *Information Processing Letters*, 1979.

[5] F. Aurenhammer. Improved algorithms for discs and balls using power diagrams. *Journal of Algorithms*, 1988.

[6] F. Aurenhammer. Voronoi diagrams – a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 1991.

[7] J. L. Bentley and T. A. Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Transactions on Computers*, 1979.

[8] E. Berberich, A. Eigenwillig, M. Hemmer, S. Hert, K. Mehlhorn, and E. Schömer. A computational basis for conic arcs and boolean operations on conic polygons. 2002.

[9] G. O. Berg, W. Julian, R. Mines, and F. Richman. The constructive jordan curve theorem. *Rocky Mountain Journal of Mathematics*, 5(2):225–236, 1975.

[10] H. Bieri. Nef polyhedra: a brief introduction. *Computing Supplementum*, 10:43–60, 1995.

[11] K. Q. Brown. *Geometric transforms for fast geometric algorithms.* PhD thesis, Pittsburgh, 1980.

[12] B. Chazelle. Triangulating a simple polygon in linear time. *Discrete & Computational Geometry*, 6:485–524, 1991.

[13] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to Algorithms.* The MIT Press, third edition, 2009.

[14] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications.* Springer, third edition, 2008.

[15] O. Devillers, A. Fronville, B. Mourrain, and M. Teillaud. Algebraic methods and arithmetic filtering for exact predicates on circle arcs. *Computational Geometry: Theory and Applications*, 22:119–142, 2002.

[16] H. Edelsbrunner and E. P. Mücke. Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Trans. Graph.*, 9(1):66–104, January 1990.

[17] S. Fortune. A sweep-line algorithm for Voronoi diagrams. *Proceedings of the Second Annual Symposium on Computational Geometry*, 1986.

[18] E. Gurari. *An Introduction to the Theory of Computation*. Computer Science Press, Ohio State University, 1989.

[19] Imai H., Iri M., and Murota K. Voronoi diagrams in the Laguerre geometry and its applications. 1985.

[20] J. O'Rourke. *Computational Geometry in C*. Cambridge University Press, second edition, 1988.

[21] M. I. Shamos. *Computational geometry*. PhD thesis, New Haven, CT, USA, 1978.

[22] M. I. Shamos and D. Hoey. Closest-point problems. *Proceedings of the Sixteenth Annual IEEE Symposium on Foundations of Computer Science*, 1975.

[23] M. Sharir. Intersection and closest-pair problems for a set of planar discs. *SIAM Journal on Computing*, 1985.

[24] M. Sipser. *Introduction to the theory of computation*. PWS publishing company, 1997.

[25] P. G. Spirakis. *Very fast algorithms for the area of the union of many circles*. Courant Institute of Mathematical Sciences, New York University, 1983.

[26] S. Tatham. Coroutines in C. `http://www.chiark.greenend.org.uk/~sgtatham/coroutines.html`, 2000. [Online; accessed 26-Aug-2014].

# Índice