



UNIVERSITAT DE
BARCELONA

Facultat de Matemàtiques
i Informàtica

GRAU DE MATEMÀTIQUES

Treball final de grau

Computabilidad de los conjuntos de Julia de polinomios hiperbólicos

Autora: Esther Martínez Borrallo

Directora: Dra. Leticia Pardo Simón

Realitzat a: Departament de
Matemàtiques i Informàtica

Barcelona, 10 de junio de 2026

Abstract

This work explores the intersection between dynamical systems and theoretical computer science, focusing on the computability of Julia sets. Due to the uncountable cardinality of the complex plane and the unstable, chaotic behavior that arises in holomorphic dynamics, the question arises whether these geometric structures of infinite complexity can be accurately represented by machines of a discrete, finite, and approximate nature.

The work details the affirmative answer to this question, proven by Mark Braverman: the Julia sets of hyperbolic polynomials are computable in polynomial time. To achieve this, we combine models of algorithmic complexity and complex geometry. The key to the resolution lies in the expansive property of these polynomials with respect to the hyperbolic metric, allowing us to construct control domains and linearly bound the escape time of the orbits.

Resumen

Este trabajo explora la intersección entre sistemas dinámicos e informática teórica, enfocándose en la computabilidad de los conjuntos de Julia. Debido a la cardinalidad no numerable del plano complejo y al comportamiento inestable y caótico que aparece en dinámica holomorfa, surge la duda de si estas estructuras geométricas de infinita complejidad pueden ser representadas de forma exacta mediante máquinas de naturaleza discreta, finita y aproximada.

El trabajo detalla una respuesta afirmativa a esta cuestión, demostrada por Mark Braverman: los conjuntos de Julia de polinomios hiperbólicos son computables en tiempo polinómico. Para ello, combinamos modelos de complejidad algorítmica y geometría compleja. La clave de la resolución reside en la propiedad expansiva de estos polinomios respecto a la métrica hiperbólica, lo que permite construir dominios de control y acotar linealmente el tiempo de escape de las órbitas.

Resum

Aquest treball explora la intersecció entre els sistemes dinàmics i la informàtica teòrica, centrant-se en la computabilitat dels conjunts de Julia. Donada la naturalesa contínua del pla complex i el comportament inestable i caòtic que apareix en dinàmica holomorfa, sorgeix el dubte de si aquestes estructures geomètriques d'infinita complexitat poden ser representades computacionalment amb una precisió garantida mitjançant màquines de naturalesa discreta, finita i aproximada.

El treball detalla la resposta afirmativa a aquesta qüestió, demostrada per Mark Braverman: els conjunts de Julia de polinomis hiperbòlics són computables en temps polinòmic. Per a això, combinem models de complexitat algorítmica i geometria complexa. La clau de la resolució resideix en la propietat expansiva d'aquests polinomis respecte a la mètrica hiperbòlica, fet que permet construir dominis de control i fitar linealment el temps d'escapament de les òrbites.

Agradecimientos

En primer lugar, me gustaría darle las gracias a mi tutora, Leticia Pardo, por su inestimable guía y paciencia a lo largo de todo este proyecto. Gracias por tu dedicación incondicional y por implicarte al máximo durante todo el camino. Tu empatía y tu disposición para reunirme conmigo, explicarme cada concepto y ayudarme a superar cualquier bloqueo han sido fundamentales para que este trabajo saliera adelante.

A mis padres y a mi hermano, mi mayor apoyo. Gracias por no dejar de confiar nunca en mí y por tener la certeza absoluta de que sería capaz de sacar esta carrera adelante. Vuestro cariño y apoyo constante no solo han hecho que este largo camino haya sido muchísimo más ameno, sino que me han transmitido la fuerza que necesitaba para seguir esforzándome día a día. Aunque en ocasiones nos haya separado la distancia, os he sentido siempre presentes a mi lado.

A mi abuela. Aunque a veces se te confunda un poco si me voy a la universidad o al trabajo, tus palabras de ánimo y tus buenos deseos jamás me han faltado. Gracias por alegrarte siempre de cada paso que doy.

A mi pareja, por acompañarme cada día de este trayecto. Gracias por tu infinita comprensión, por saber levantarme el ánimo cada vez que decaía y por ser ese pilar fundamental que no me ha fallado ni un solo instante. Me has recordado constantemente que yo era capaz de lograrlo, convirtiéndote en mi mayor refugio y aportándome la calma que tanto necesitaba en los momentos de mayor agobio.

A mis compañeras de piso, Emma y María. Gracias por haber sabido animarme en todo momento y por hacer que llegar a casa fuera siempre una alegría. Vuestra compañía ha sido, a veces, verdaderamente imprescindible para mí, aportándome muchísimos momentos de desconexión y las risas que tanto necesitaba para recargar las pilas.

A mis compañeros de la universidad, quienes mejor podían comprender los retos por los que estábamos pasando. Gracias por el apoyo mutuo, por las risas y por haberme hecho los días de estudio mucho más felices y llevaderos. Dentro de este grupo, quiero darle las gracias a Samanta Ruiz. En ti he encontrado mucho más que una compañera; me llevo a una amiga para toda la vida. Hemos formado un equipo increíble, apoyándonos mutuamente a lo largo de todo este camino. Gracias por tu inmensa empatía y por estar siempre ahí, acompañándome en mis momentos de estrés a pesar de tener que lidiar también con los tuyos. Me siento profundamente feliz y afortunada de poder concluir esta etapa a tu lado.

Finalmente, quiero darme las gracias a mí misma. Empecé esta etapa universitaria sintiéndome muy perdida y dudando de si sería capaz de superar este reto, pero siempre supe tirar hacia adelante. A día de hoy, terminando por fin la carrera, puedo decir que me siento profundamente orgullosa de todo lo que he logrado.

Índice general

1. Introducción	1
2. Dinámica holomorfa	6
2.1. Conceptos básicos de dinámica holomorfa	6
2.2. Conjuntos de Julia y Fatou	9
2.3. Dinámica en superficies hiperbólicas	13
2.4. Clasificación de las componentes de Fatou	15
2.5. Polinomios hiperbólicos	17
3. Computabilidad	21
3.1. Conceptos previos	21
3.2. Computabilidad de conjuntos reales	25
3.3. Clases de complejidad P y NP	35
4. Computabilidad de los conjuntos de Julia hiperbólicos	36
4.1. Información no uniforme y preprocesamiento	36
4.2. Construcción de dominios	38
4.2.1. Algoritmo 1 [Bra05, Sección 4.4.3]	43
4.2.2. Algoritmo 2 [Bra05, Sección 4.5.2]	44
4.2.3. Algoritmo 3 [Bra05, Sección 4.5.4]	48
A. Superficies de Riemann y métrica hiperbólica	50
A.1. Superficies de Riemann y aplicaciones conformes	50
A.1.1. Aplicaciones propias y mapas recubridores	50
A.2. Métricas conformes e isometrías	51
A.3. La métrica hiperbólica en el disco unidad	52
A.4. Uniformización y superficies hiperbólicas	53
A.5. Lema de Schwarz y Teorema de Schwarz-Pick	54
B. Relación entre la Ko-computabilidad y las clases P y NP	55

Capítulo 1

Introducción

La dinámica holomorfa se encarga de estudiar el comportamiento de las sucesivas iteradas de una función. En este trabajo, consideraremos un polinomio $P : \widehat{\mathbb{C}} \rightarrow \widehat{\mathbb{C}}$ de grado $d \geq 2$, donde $\widehat{\mathbb{C}}$ es la esfera de Riemann, definida como $\widehat{\mathbb{C}} = \mathbb{C} \cup \{\infty\}$. Al evaluar la sucesión de las iteradas $P^n(z)$ a partir de un punto inicial, la órbita resultante presenta, a grandes rasgos, dos comportamientos posibles: o bien tiende a infinito, o bien permanece acotada. En base a este comportamiento, el espacio se divide en dos regiones fundamentales:

Definición 1.0.1 (Conjuntos de Fatou y Julia). *Se define el conjunto lleno de Julia, denotado por $K(P)$, como el conjunto de todos los puntos cuya órbita bajo P permanece acotada, es decir, $K(P) := \{z \in \mathbb{C} \mid \sup_n |P^n(z)| < \infty\}$.*

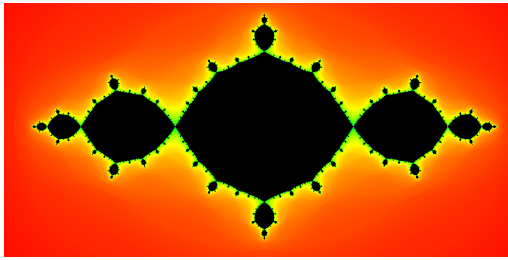
Su frontera topológica es el conjunto de Julia, $J(P) = \partial K(P)$.

El complementario de este último es el conjunto de Fatou, $F(P) = \widehat{\mathbb{C}} \setminus J(P)$, el cual representa la región de estabilidad del sistema.

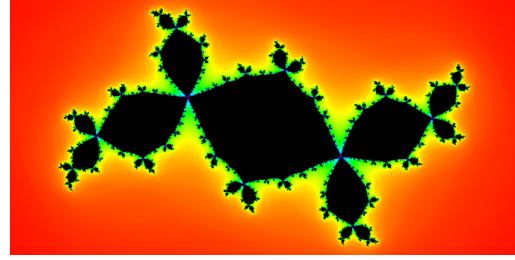
Los conjuntos de Julia sorprenden a la comunidad matemática por la infinita complejidad geométrica. Su estudio y visualización están ligados al desarrollo de la informática, lo que plantea una dicotomía profunda: la cardinalidad no numerable del plano complejo y el comportamiento caótico en dinámica holomorfa chocan frontalmente con la naturaleza discreta, finita y aproximada de las máquinas que utilizamos para explorarlos.

Para ilustrar esta infinita complejidad geométrica y visualizar el concepto central de este trabajo, el polinomio hiperbólico (entendido intuitivamente como aquel cuya dinámica empuja a todos sus puntos críticos hacia ciclos estables, y cuya definición formal daremos más adelante), resulta muy ilustrativo observar dos de los ejemplos canónicos más célebres de la dinámica holomorfa: la “Basílica de San Marcos” y el “Conejo de Douady”.

En el primer sistema ($P(z) = z^2 - 1$), el único punto crítico en el plano complejo es el origen, $z = 0$. Al iterar este punto, su órbita alterna indefinidamente entre dos valores ($0 \mapsto -1 \mapsto 0 \mapsto -1 \dots$), siendo capturado inmediatamente por un ciclo periódico atractor de periodo 2. Por otro lado, en el segundo sistema ($P(z) = z^2 + c$, con $c \approx -0,123 + 0,745i$), la órbita del punto crítico converge hacia un ciclo atractor de periodo 3, lo que se traduce visualmente en una configuración repetitiva de tres “orejas” que se transforman cíclicamente unas en otras.



“Basílica de San Marcos”



“Conejo de Douady”

Figura 1.1: **Conjuntos de Julia hiperbólicos.** En ambas figuras, la frontera que delimita las siluetas representa el conjunto de Julia $J(P)$, donde la dinámica es inestable. Las diferentes regiones interiores (cuencas de los ciclos periódicos finitos) junto con la gran región exterior oscura (la cuenca de atracción del infinito) conforman el conjunto de Fatou $F(P)$, donde las órbitas presentan un comportamiento atractor estable. (Imágenes generadas mediante el software <https://icefractal.com/julia/>).

Este proceso de aproximación algorítmica para generar estas representaciones gráficas formula una pregunta fundamental: cuando una máquina calcula y dibuja un conjunto de Julia, ¿está aproximando verdaderamente el conjunto matemático con una precisión garantizada, o la imagen resultante está distorsionada por la inevitable acumulación de errores de redondeo? Para poder responder a esta cuestión de forma rigurosa, necesitamos crear un puente formal entre dos mundos: la teoría de los sistemas dinámicos y la teoría de la computabilidad discreta.

El objetivo de este Trabajo Final de Grado es desarrollar el marco teórico necesario en ambos mundos para demostrar rigurosamente que una familia particular de estos conjuntos, los conjuntos de Julia hiperbólicos, pueden ser calculados con una precisión garantizada y en tiempo polinómico por un ordenador.

El conjunto de Julia es el lugar geométrico donde la dinámica presenta extrema inestabilidad, ya que pequeñas perturbaciones en las condiciones iniciales conducen a órbitas drásticamente diferentes. Para estudiar la geometría de estos conjuntos, nos interesamos en las *superficies de Riemann*, variedades complejas de dimensión uno, las cuales supondremos siempre conexas. Dentro de estas superficies, nos restringimos a las *superficies hiperbólicas*, que son superficies de Riemann tal que su recubridor universal es isomorfo al disco unidad abierto $\mathbb{D}$.

A lo largo del trabajo, demostramos que los dominios dinámicos con los que operamos pertenecen a esta categoría:

Proposición 1.0.2. *El conjunto de Fatou $F(P)$ es, por definición, un conjunto abierto (al ser el complementario del conjunto cerrado $J(P)$), y cualquier componente conexa U de $F(P)$ es un subconjunto abierto de $\widehat{\mathbb{C}}$ y, por tanto, una superficie hiperbólica.*

La demostración de esta proposición se apoya en el hecho de que el conjunto de Julia $J(P)$ contiene infinitos puntos, por lo que cualquier componente conexa de $F(P)$ omitirá esos puntos. Al omitir estrictamente más de tres puntos, garantizamos que el dominio es una superficie hiperbólica.

Identificar nuestros dominios como superficies hiperbólicas es crucial porque nos permite dotarlos de una métrica específica.

Definición 1.0.3. Definimos la métrica hiperbólica en el disco unidad $\mathbb{D}$ mediante la densidad $\lambda_{\mathbb{D}}(z)$, cuya expresión diferencial es:

$$ds = \frac{2|dz|}{1 - |z|^2}.$$

Mediante el recubrimiento universal, esta métrica se traslada a cualquier superficie hiperbólica S , definiendo una densidad λ_S que garantiza que la frontera del dominio se encuentre a una distancia infinita.

Una propiedad fundamental en este contexto es el *Lema de Schwarz-Pick*, que establece que las funciones holomorfas entre superficies de Riemann hiperbólicas actúan como contracciones débiles respecto a esta métrica.

En el trabajo construimos una serie de dominios encajados alrededor del conjunto de Julia. Aplicando el Teorema de Schwarz-Pick a las funciones inversas del polinomio (sus preimágenes), deducimos que estas actúan como una contracción estricta de las distancias hiperbólicas. Por consiguiente, si invertimos la perspectiva, el polinomio iterado hacia adelante actúa como una expansión estricta.

Si el espacio se expande en cada iteración, cualquier punto que no pertenezca exactamente al conjunto de Julia será expulsado de su vecindad. Sin embargo, para garantizar que esta dinámica expansiva se cumple uniformemente en todo el conjunto, exigimos una última condición al polinomio:

Definición 1.0.4. Un polinomio P de grado $d \geq 2$ es hiperbólico si la órbita de cualquier punto crítico de P converge a un ciclo periódico atractor (incluyendo el punto en el infinito).

Para caracterizar este comportamiento de forma global, es fundamental hacer un seguimiento de estas órbitas críticas; el conjunto que reúne a todas las imágenes futuras de los puntos críticos de P se denomina conjunto postcrítico y se denota por $\mathcal{P}$ (siendo $\overline{\mathcal{P}}$ su clausura).

La importancia de restringirnos a esta familia de polinomios está en el siguiente teorema, donde mostramos la equivalencia fundamental que demuestra este comportamiento:

Teorema 1.0.5. Sea P un polinomio de grado $d \geq 2$. Las siguientes afirmaciones son equivalentes:

- P es hiperbólico.
- La clausura del conjunto postcrítico es disjunta del conjunto de Julia, $\overline{\mathcal{P}} \cap J(P) = \emptyset$.
- P es expansivo en su conjunto de Julia respecto a una métrica conforme.

La propiedad expansiva asegura que la propia dinámica del polinomio “empuja” cualquier punto que no pertenezca a $J(P)$ hacia ciclos atractores. Como consecuencia directa de esta expansión, la norma de la derivada del polinomio evaluada en cualquier punto del conjunto de Julia es siempre estrictamente mayor que uno. Dado que en los puntos críticos la derivada se anula por definición, se deduce que estos no pueden pertenecer a $J(P)$. Todos ellos quedan en el interior del conjunto de Fatou $F(P)$, iterando de forma segura hacia los ciclos atractores. Es precisamente esta separación de las órbitas respecto a los puntos críticos la que permite acotar el tiempo de cálculo y preparar el terreno para el análisis algorítmico posterior.

Una vez caracterizado el comportamiento del conjunto en el plano complejo, debemos formalizar qué significa “calcularlo”. Los ordenadores no pueden operar con números reales arbitrarios debido a sus infinitos decimales. Para trasladar el problema al ámbito discreto, introducimos los números diádicos:

Definición 1.0.6. *El conjunto de los números diádicos $\mathcal{D}$ está formado por aquellos números con una representación binaria finita.*

$$\mathcal{D} = \left\{ \frac{m}{2^n} \mid m \in \mathbb{Z}, n \in \mathbb{N} \right\}.$$

Definimos $\mathcal{D}_k$ como el subconjunto de números diádicos con una precisión de, a lo sumo, k bits fraccionarios.

Para operar con puntos del plano utilizando representaciones finitas, utilizamos el modelo de la máquina de Turing. En particular, cuando la máquina necesita información sobre un número real x , hace uso de un oráculo ϕ :

Definición 1.0.7. *Decimos que una función $\phi : \mathbb{N} \rightarrow \mathcal{D}$ representa a un número real x si para cada precisión exigida $n \in \mathbb{N}$, devuelve un número diádico $\phi(n)$ que satisface la cota de error:*

$$|x - \phi(n)| < 2^{-n}.$$

A esta función ϕ la denominamos el oráculo de x .

Esta noción se extiende de manera natural a espacios de dimensión superior, sustituyendo el valor absoluto por la norma euclídea usual. En nuestro contexto, dado que los conjuntos de Julia habitan en el plano complejo (el cual identificamos de forma isomórfica con $\mathbb{R}^2$), un oráculo para un punto $x = (x_1, x_2) \in \mathbb{R}^2$ será una función $\phi : \mathbb{N} \rightarrow \mathcal{D}^2$ que, para cada precisión exigida n , devuelve un par de números diádicos $\phi(n) = (d_1, d_2)$ que satisfacen: $\|x - \phi(n)\| < 2^{-n}$.

En un mundo ideal, nos gustaría poder decir si un punto pertenece a la frontera de un conjunto de manera exacta, pero es imposible en un tiempo finito. Por ello, permitimos que el algoritmo tenga un margen de incertidumbre rigurosamente acotado mediante el concepto de Ko-computabilidad:

Definición 1.0.8. *Decimos que un conjunto compacto $C \subset \mathbb{R}^2$ es Ko-computable si existe una máquina de Turing con oráculo $M^\phi(n)$ tal que, si ϕ representa un punto $x \in \mathbb{R}^2$, para una entrada n la salida es:*

- 1 si $x \in C$,
- 0 si $B(x, 2^{-n}) \cap C = \emptyset$,
- 0 o 1 en otro caso (zona de incertidumbre).

A lo largo del trabajo, demostramos la equivalencia entre las diferentes nociones de computabilidad de conjuntos. Sin embargo, que un conjunto sea computable no garantiza que podamos visualizarlo en la práctica. Para ello, necesitamos introducir la variable de la complejidad temporal.

Definición 1.0.9. *Diremos que un conjunto es Ko-computable en tiempo polinómico (Ko P-computable) si la máquina $M^\phi(n)$ se ejecuta en un número de pasos acotado por un polinomio $p(n)$ respecto a la precisión exigida.*

La resolución definitiva de este problema se alcanza al unificar las herramientas geométricas y dinámicas desarrolladas, dando así respuesta a la computabilidad en el contexto de los polinomios hiperbólicos.

Como ya hemos establecido que el polinomio actúa como una expansión uniforme respecto a la métrica hiperbólica en un determinado dominio abierto y acotado V que contiene al conjunto de Julia, es posible deducir analíticamente que el tiempo que tarda una órbita exterior en escapar hacia el infinito o hacia un ciclo atractor es lineal.

Proposición 1.0.10 (Cota lineal del tiempo de escape). *Si un punto $x \in \mathbb{C}$ cumple que su distancia euclídea al conjunto de Julia satisface $\text{dist}(x, J(P)) \geq 2^{-n}$, entonces su órbita abandonará dicho entorno V en un número de pasos s acotado linealmente por n ; es decir, $s = \mathcal{O}(n)$.*

Al requerir únicamente esta cantidad lineal de pasos, la complejidad total del algoritmo se reduce drásticamente, pasando de requerir un tiempo exponencial a un tiempo polinómico. El desarrollo de este proceso algorítmico culmina con la demostración rigurosa del resultado principal de esta memoria, probado originalmente por Mark Braverman [Bra05, Teorema 4.5.8].

Teorema 1.0.11 (Computabilidad de los Conjuntos de Julia Hiperbólicos). *Para cada polinomio hiperbólico $P(z)$, su conjunto de Julia $J(P)$ es computable en tiempo polinómico. Específicamente, la complejidad temporal de la máquina de Turing está acotada por $\mathcal{O}(n \cdot M(n))$, donde $M(n)$ es el tiempo necesario para multiplicar dos números de n bits.*

Estructura del trabajo

Este trabajo se estructura de la siguiente manera. El Capítulo 2 expone la teoría de la dinámica holomorfa, caracterizando los conjuntos de Julia y Fatou, así como las propiedades específicas de los polinomios hiperbólicos. En el Capítulo 3, construimos el puente hacia la informática teórica, definiendo los modelos de computabilidad sobre los números reales y las clases de complejidad P y NP. Finalmente, en el Capítulo 4, se combinan las cotas de la métrica hiperbólica con los modelos de máquinas de Turing para detallar los algoritmos que demuestran que los conjuntos de Julia hiperbólicos son computables en tiempo polinómico. Para complementar la rigurosidad teórica, en el Anexo A se introducen los fundamentos de la métrica hiperbólica, y en el Anexo B se recogen las demostraciones formales que vinculan la Ko-computabilidad con las clases de complejidad P y NP.

Capítulo 2

Dinámica holomorfa

La dinámica holomorfa tiene como objetivo estudiar el comportamiento de los puntos en el plano complejo $\widehat{\mathbb{C}}$ unidimensional bajo la iteración de una función holomorfa. Siguiendo la exposición de Milnor [Mil06], en este trabajo nos centraremos en el estudio de los polinomios de grado $d \geq 2$, y para poder estudiarlos, necesitaremos una serie de conceptos previos y resultados que veremos en esta sección.

2.1. Conceptos básicos de dinámica holomorfa

Definición 2.1.1. Un polinomio será una función $P : \widehat{\mathbb{C}} \rightarrow \widehat{\mathbb{C}}$ de forma que $P(z) = a_0 + a_1z + \dots + a_dz^d$, donde $a_i \in \mathbb{C}$ para todo $i = 0, \dots, d$. Denotaremos por $\deg(P)$ al grado del polinomio y $P^n(z)$ a la n -ésima iterada del polinomio P evaluada en el punto z para $n \in \mathbb{N}$, es decir:

$$P^n(z) = (P \circ \dots \circ P)(z).$$

A lo largo de este capítulo y en las secciones siguientes, a menos que se indique lo contrario, asumiremos que P denota siempre un polinomio de grado $d \geq 2$.

Definición 2.1.2. Dado un punto inicial $z_0 \in \mathbb{C}$, definimos como su órbita a la sucesión de imágenes bajo la iteración de P :

$$z_0, P(z_0), P^2(z_0), \dots, P^n(z_0) \dots$$

Definición 2.1.3. Un punto $z_0 \in \mathbb{C}$ será un punto crítico de P si su derivada se anula, es decir, $P'(z_0) = 0$.

Definición 2.1.4. Sea $z_0 \in \mathbb{C}$ un punto fijo de P . Definimos el multiplicador del punto fijo como la derivada del polinomio evaluada en el punto fijo:

$$\lambda = P'(z_0).$$

Determinamos el comportamiento dinámico local alrededor del punto fijo gracias al valor del multiplicador. El punto fijo puede ser:

1. Atractor: si $0 < |\lambda| < 1$.
2. Superatractor: si $\lambda = 0$.

3. Repulsor: si $|\lambda| > 1$.

4. Neutro: si $|\lambda| = 1$.

En el último caso, podemos escribir el multiplicador como $\lambda = e^{2\pi i\theta}$, y tenemos los siguientes subcasos:

- a) Si $\theta \in \mathbb{Q}$, lo llamaremos punto parabólico. En este caso, el multiplicador λ es una raíz de la unidad, por lo que existe un entero $q \geq 1$ tal que $\lambda^q = 1$. Pasando a la iterada P^q , el multiplicador se convierte exactamente en 1.
- b) Si $\theta \notin \mathbb{Q}$, podrá ser un punto de Siegel o un punto de Cremer (dependiendo de la dinámica local).

Observación 2.1.5. Para los puntos parabólicos, el comportamiento dinámico local se describe de manera precisa mediante el Teorema de la Flor de Leau-Fatou [Mil06, Cap 10]. Este teorema establece que el entorno del punto fijo se divide en una serie de regiones alternas en forma de pétalo; en unas la dinámica atrae hacia el punto fijo (pétalos atractores) y en otras lo repele (pétalos repulsores). Observemos que los pétalos atractores serán las regiones invariantes atractoras. Por otro lado, dentro de los puntos de tipo neutro, si nos encontramos en el caso irracional ($\theta \notin \mathbb{Q}$), la dinámica local puede formar una rotación irracional estable (Disco de Siegel) o presentar un comportamiento caótico (Punto de Cremer).

Definición 2.1.6. Un punto $z_0 \in \mathbb{C}$ es un punto periódico de periodo $k \geq 1$ si $P^k(z_0) = z_0$ y $P^j(z_0) \neq z_0$ para $1 \leq j < k$. Dado que z_0 es, por definición, un punto fijo para la iterada P^k , clasificamos el comportamiento dinámico de su órbita aplicando exactamente la misma clasificación de la Definición 2.1.4, evaluando el multiplicador:

$$\lambda = (P^k)'(z_0) = P'(z_{k-1}) \cdot P'(z_{k-2}) \cdots P'(z_0).$$

Definición 2.1.7. Dado un punto periódico z_0 atractor o superatractor de periodo k , definimos la cuenca de atracción de dicho punto, denotada como $A(z_0)$, como el conjunto de puntos del plano cuyas órbitas bajo la iteración de P^k convergen a z_0 :

$$A(z_0) = \{z \in \mathbb{C} \mid \lim_{n \rightarrow \infty} P^{nk}(z) = z_0\}.$$

Puesto que $A(z_0)$ es un conjunto abierto, este puede formarse por una unión de componentes conexas disjuntas. Definimos la cuenca inmediata de atracción, denotada como $A_0(z_0)$, como la componente conexa particular de $A(z_0)$ que contiene al propio punto z_0 . En el caso de un ciclo periódico de periodo k , la cuenca inmediata del ciclo será la unión de las componentes conexas que contienen a cada uno de los k puntos del ciclo.

Proposición 2.1.8. El punto en el infinito $\infty \in \hat{\mathbb{C}}$ es un punto fijo superatractor. Su cuenca de atracción será:

$$A(\infty) = \{z \in \mathbb{C} \mid \lim_{n \rightarrow \infty} P^n(z) = \infty\}.$$

Demostración. Para estudiar el comportamiento del polinomio en el punto en el infinito dentro de la esfera de Riemann $\hat{\mathbb{C}}$, realizamos el cambio de coordenadas local $w = \frac{1}{z}$, el cual envía el infinito al origen. Así, conjugamos la dinámica definiendo la función:

$$f(z) = \frac{1}{P\left(\frac{1}{z}\right)}.$$

Sustituyendo el polinomio en la definición de f tenemos:

$$f(z) = \frac{1}{a_d \left(\frac{1}{z}\right)^d + a_{d-1} \left(\frac{1}{z}\right)^{d-1} + \cdots + a_1 \left(\frac{1}{z}\right) + a_0}.$$

Multiplicando numerador y denominador por z^d , obtenemos una expresión racional para f :

$$f(z) = \frac{z^d}{a_d + a_{d-1}z + \cdots + a_1z^{d-1} + a_0z^d}.$$

Evalutando en el origen, observamos que:

$$f(0) = \frac{0^d}{a_d} = 0 \iff \frac{1}{P(\infty)} = 0 \iff P(\infty) = \infty.$$

Esto demuestra que el infinito es, efectivamente, un punto fijo de la esfera de Riemann.

Para clasificar la naturaleza de este punto fijo, analizamos su multiplicador calculando la derivada $f'(0)$. Derivando $f(z)$ mediante la regla del cociente obtenemos:

$$f'(z) = \frac{dz^{d-1}(a_d + a_{d-1}z + \cdots + a_0z^d) - z^d(a_{d-1} + 2a_{d-2}z + \cdots + da_0z^{d-1})}{(a_d + a_{d-1}z + \cdots + a_0z^d)^2}.$$

Dado que el grado es $d \geq 2$, tenemos que $d-1 \geq 1$. Esto implica que todos los términos del numerador están multiplicados por z elevado a una potencia estrictamente positiva. Por lo tanto, al evaluar en $z = 0$, el numerador se anula por completo, mientras que el denominador es $a_d^2 \neq 0$. Concluimos que:

$$f'(0) = 0.$$

Como la derivada se anula, el multiplicador es cero, lo que por definición convierte al infinito en un punto fijo superatractor. En consecuencia, por las propiedades de los puntos atractores, todos los puntos con módulo suficientemente grande iterarán inevitablemente hacia él, conformando su cuenca de atracción $A(\infty)$. La construcción explícita del radio a partir del cual se garantiza este escape hacia el infinito se formalizará analíticamente en la demostración de la Proposición 2.2.2. $\square$

Proposición 2.1.9. [Mil06, Teorema 8.6] *La cuenca inmediata de atracción asociada a cualquier ciclo atractor (o superatractor) contiene, en su interior, al menos un punto crítico de P .*

Proposición 2.1.10. *$P(z)$ tiene como mucho $\deg(P(z)) - 1$ órbitas periódicas atractoras.*

Demostración. Aplicando la Proposición 2.1.9 sabemos que cada ciclo atractor necesita al menos un punto crítico.

Por definición, la cuenca de atracción de un ciclo está formada por todos los puntos cuyas órbitas bajo la iteración de P convergen a dicho ciclo. Por la propiedad de unicidad del límite de una sucesión en $\mathbb{C}$, la órbita de un punto no puede converger a dos ciclos atractores distintos simultáneamente. Por consiguiente, las cuencas de atracción de ciclos distintos son conjuntos estrictamente disjuntos.

Al ser conjuntos disjuntos, sus respectivas cuencas inmediatas también son mutuamente disjuntas. Como cada cuenca inmediata contiene al menos un punto crítico propio, cada ciclo atractor contiene al menos un punto crítico distinto.

Por otro lado, como $\deg(P(z)) \geq 2$ tenemos que $\deg(P'(z))$ es $d - 1$, por lo tanto existen $d - 1$ puntos críticos en el plano complejo.

Por consiguiente, si asumimos la existencia de N ciclos atractores distintos, forzosamente requeriríamos al menos N puntos críticos en el plano, concluyendo que $N \leq d - 1$. $\square$

2.2. Conjuntos de Julia y Fatou

Definición 2.2.1. El conjunto lleno de Julia, denotado por $K(P)$, es el conjunto de todos los puntos cuya órbita es acotada. Es decir,

$$K(P) := \{z \in \mathbb{C} \mid \sup_n |P^n(z)| < \infty\}.$$

Observamos que $K(P) = \mathbb{C} \setminus A(\infty)$. El hecho geométrico de que las órbitas no acotadas pertenezcan necesariamente a la cuenca de atracción del infinito es una consecuencia directa de la Proposición 2.1.8 y quedará formalizado durante la demostración de la Proposición 2.2.2.

Proposición 2.2.2. El conjunto lleno de Julia es cerrado y acotado.

Demostración. Acotado: Sea el polinomio P . Sabemos que el término de grado mayor domina cuando $|z|$ es grande. Veremos que existe un radio R tal que si $|z| > R$, entonces $|P(z)| > c|z|$ (con $c > 1$). Por lo tanto, la órbita de estos puntos tiende a infinito, demostrando así que el exterior de este disco está contenido en $A(\infty)$.

Factorizando tenemos:

$$|P(z)| = |a_d z^d| \left| 1 + \frac{a_{d-1}}{a_d z} + \cdots + \frac{a_0}{a_d z^d} \right| \geq |a_d| |z|^d \left(1 - \left| \frac{a_{d-1}}{a_d z} + \cdots + \frac{a_0}{a_d z^d} \right| \right).$$

Como $\lim_{z \rightarrow \infty} \left(\frac{a_{d-1}}{a_d z} + \cdots + \frac{a_0}{a_d z^d} \right) = 0$, escogemos $R_1 > 0$ tal que $\left| \frac{a_{d-1}}{a_d z} + \cdots + \frac{a_0}{a_d z^d} \right| < \frac{1}{2}$ para todo $|z| > R_1$, entonces:

$$|P(z)| > \frac{|a_d| |z|^d}{2} = \left(\frac{|a_d|}{2} |z|^{d-1} \right) |z|.$$

Para demostrar que la órbita tiende a infinito, necesitamos ver que $\frac{|a_d|}{2} |z|^{d-1}$ sea mayor que algún $c > 1$. Por sencillez, busquemos que sea mayor o igual a 2:

$$\frac{|a_d|}{2} |z|^{d-1} \geq 2 \implies |z|^{d-1} \geq \frac{4}{|a_d|} \implies |z| \geq \left(\frac{4}{|a_d|} \right)^{\frac{1}{d-1}}.$$

Sea $R_2 = \left(\frac{4}{|a_d|} \right)^{\frac{1}{d-1}}$. Finalmente, definimos $R = \max(R_1, R_2)$. Entonces para los $|z| > R$, tenemos:

$$|P(z)| > 2|z| > 2R > R.$$

Como $|P(z)| > R$, sus iteraciones también cumplirán: $|P^n(z)| > 2^n |z|$. Haciendo el límite cuando $n \rightarrow \infty$ vemos que $|P^n(z)| \rightarrow \infty$, demostrando así que $K(P)$ está acotado y contenido en el disco cerrado $D_R = \{z \in \mathbb{C} : |z| \leq R\}$.

Cerrado: Sea D_R el disco cerrado de radio R fijado en el apartado anterior. Como $K(P)$ es el complementario de la cuenca de atracción del infinito $A(\infty)$, está formado por

el conjunto de puntos cuyas órbitas nunca escapan de D_R . Por lo tanto, podemos expresar $K(P)$ como la intersección de todas las preimágenes de D_R bajo las iteraciones de P :

$$K(P) = \bigcap_{n=0}^{\infty} P^{-n}(D_R).$$

Como los polinomios son funciones continuas y la preimagen de un conjunto cerrado es también un conjunto cerrado, concluimos que $K(P)$ es cerrado. Al ser cerrado y acotado, $K(P)$ es un conjunto compacto. $\square$

Proposición 2.2.3. $K(P)$ es totalmente invariante, es decir, $K(P) = P(K(P)) = P^{-1}(K(P))$.

Demostración. Sea $z \in K(P)$, sabemos que su órbita es acotada, por lo tanto la órbita de $P(z)$ también será acotada, ya que su órbita es la misma que la de z omitiendo el punto z . Con esto tenemos que $P(K(P)) \subset K(P)$.

Por otro lado, sea $z \in P^{-1}(K(P))$. Tenemos que $P(z) \in K(P)$, por lo que la órbita de $P(z)$ está acotada. La órbita de z es la misma que la de $P(z)$ pero añadiendo el punto z , por lo tanto también está acotada, entonces $z \in K(P)$. Con esto tenemos que $P^{-1}(K(P)) \subset K(P)$.

Como nuestro polinomio P es exhaustivo, deducimos la invarianza total del conjunto lleno de Julia:

$$K(P) = P(K(P)) = P^{-1}(K(P)). \quad \square$$

Definición 2.2.4. El conjunto de Julia, denotado por $J(P)$, es la frontera del conjunto lleno de Julia. Es decir,

$$J(P) = \partial K(P).$$

Por otro lado, el conjunto de Fatou, denotado por $F(P)$, es el complementario del conjunto de Julia.

$$F(P) = \widehat{\mathbb{C}} \setminus J(P).$$

A las componentes conexas de $F(P)$ las llamaremos componentes de Fatou.

Proposición 2.2.5. $J(P)$ es totalmente invariante, es decir, $J(P) = P(J(P)) = P^{-1}(J(P))$.

Demostración. Como $J(P) = \partial K(P)$ y P es una aplicación continua y abierta, la preimagen de la frontera de un conjunto es igual a la frontera de su preimagen. Usando la Proposición 2.2.3 tenemos que:

$$P^{-1}(J(P)) = P^{-1}(\partial K(P)) = \partial P^{-1}(K(P)) = \partial K(P) = J(P).$$

Y aplicando P a ambos lados de la igualdad, como P es exhaustivo tenemos que:

$$P(P^{-1}(J(P))) = P(J(P)) \implies J(P) = P(J(P)). \quad \square$$

Dado que $K(P)$ es un conjunto cerrado, sabemos que su frontera está contenida en él ($J(P) \subset K(P)$). Además, gracias a las definiciones anteriores, podemos establecer que el conjunto de Fatou se descompone como $F(P) = A(\infty) \cup \mathring{K}(P)$. A partir de aquí, para poder entender y estudiar la estabilidad en el conjunto de Fatou utilizaremos el concepto de familia normal.

Definición 2.2.6. Sea $U \subset \mathbb{C}$ una superficie de Riemann. Una familia de funciones holomorfas $\mathcal{F}$ definidas en U es una familia normal si toda sucesión infinita de funciones en $\mathcal{F}$ contiene una subsucesión que converge uniformemente sobre compactos de U .

Teorema 2.2.7 (Teorema de Weierstrass, [Mil06, Teorema 1.4]). Si una sucesión de funciones holomorfas converge uniformemente sobre conjuntos compactos, entonces sus derivadas también convergen uniformemente, y la función límite es en sí misma holomorfa.

Observación 2.2.8 ([Mil06, Observación 2.6]). En nuestro contexto, al estudiar la familia de iteradas de un polinomio, consideramos funciones que pueden tomar valores en la esfera de Riemann $\widehat{\mathbb{C}}$. Por lo tanto, si extraemos una subsucesión que converge uniformemente sobre compactos, su límite solo podrá ser una función holomorfa (si la imagen se mantiene acotada) o, alternativamente, la sucesión divergirá uniformemente hacia infinito.

Definición 2.2.9. Retomando nuestro polinomio P , diremos que un punto $z \in \widehat{\mathbb{C}}$ es normal si existe algún entorno $U \subseteq \widehat{\mathbb{C}}$ del punto z donde la familia de iteradas $\{P^n\}_{n \in \mathbb{N}}$ constituya una familia normal.

De una manera intuitiva, $\{f^n\}_{n \in \mathbb{N}}$ será una familia normal en el punto z si todos los puntos en un entorno de z se comportan bajo iteración de f de forma análoga a z .

Para poder comprobar la normalidad usaremos el siguiente criterio:

Teorema 2.2.10 (Teorema de Montel). Sea $\mathcal{F} = \{f^n\}_{n \in \mathbb{N}}$ la familia de iteradas de una función holomorfa f definida en un dominio D , cuyos valores pertenecen a la esfera de Riemann $\widehat{\mathbb{C}}$. Si existen tres puntos distintos en la esfera de Riemann $\widehat{\mathbb{C}}$ que son omitidos por todas las funciones de la familia $\mathcal{F}$ en D , entonces $\mathcal{F}$ es una familia normal en D .

Teorema 2.2.11. El conjunto de Fatou $F(P) = \widehat{\mathbb{C}} \setminus J(P)$ coincide con el conjunto de puntos normales de P .

Demostración. Por definición, el conjunto de Fatou está formado por la unión del interior de $K(P)$ y $\widehat{\mathbb{C}} \setminus K(P) = A(\infty)$.

Por un lado, en el interior de $K(P)$, la familia de iteradas está uniformemente acotada por definición. Aplicando el Teorema 2.2.10 tenemos que $\{P^n\}_{n \in \mathbb{N}}$ forma una familia normal.

Por otro lado, en $A(\infty)$, la sucesión de iteradas converge uniformemente a infinito sobre compactos, por lo que también forma una familia normal.

Sin embargo, si consideramos cualquier punto en la frontera $J(P) = \partial K(P)$, cualquier entorno suyo intersecará tanto a $\overset{\circ}{K}(P)$ (donde las iteradas permanecen uniformemente acotadas) como a $A(\infty)$ (donde las iteradas tienden hacia infinito). Esta coexistencia de dinámicas extremas en todo vecindario impide extraer cualquier subsucesión que converja uniformemente, por lo que la familia no puede ser normal. $\square$

Proposición 2.2.12. Si $J(P)$ contiene un punto interior, entonces $J(P) = \widehat{\mathbb{C}}$.

Demostración. Supongamos que $\overset{\circ}{J}(P) \neq \emptyset$, por lo tanto existe un U abierto tal que $U \subset J(P)$.

Por definición, como $F(P)$ es el conjunto donde la familia de iteradas es normal, $U \subset J(P)$ será un conjunto donde la familia de iteradas P^n no es normal en U . Aplicando el Teorema 2.2.10, P^n no puede omitir tres valores distintos.

Definimos el siguiente conjunto $V = \bigcup_{n \in \mathbb{N}} P^n(U)$, que por su definición debe cubrir toda la esfera de Riemann, excepto como mucho dos puntos.

Como $P(J(P)) \subset J(P)$ tenemos que $P^n(U) \subset J(P)$ para todo $n \in \mathbb{N}$, entonces $V \subset J(P)$.

Como V cubre toda la esfera de Riemann, excepto como mucho dos puntos, y $V \subset J(P)$, tenemos que $J(P)$ cubre $\widehat{\mathbb{C}}$, excepto como mucho dos puntos y que es un conjunto cerrado, por lo tanto su clausura es $\widehat{\mathbb{C}}$. $\square$

Lema 2.2.13. *Para P se cumple que $J(P) \neq \emptyset$ y $\mathring{J}(P) = \emptyset$.*

Demostración. ■ $J(P) \neq \emptyset$: Primero, observemos que $K(P)$ no es vacío. La ecuación $P(z) - z = 0$ tiene al menos una solución en $\widehat{\mathbb{C}}$ por el Teorema Fundamental del Álgebra. Este punto fijo tiene una órbita constante y, por tanto, acotada, lo que garantiza que pertenece a $K(P)$. Por otro lado, la cuenca de atracción del infinito $A(\infty)$ tampoco es vacía en $\widehat{\mathbb{C}}$, ya que contiene al propio infinito y a los puntos con módulo suficientemente grande (visto en la Proposición 2.2.2). Como la esfera de Riemann $\widehat{\mathbb{C}}$ es conexa, no puede ser la unión de dos conjuntos abiertos disjuntos no vacíos. Dado que $A(\infty)$ es abierto y $K(P)$ es su complementario cerrado, si la frontera fuera vacía, $K(P)$ sería simultáneamente abierto y cerrado en $\widehat{\mathbb{C}}$. Por la conexidad de $\widehat{\mathbb{C}}$, esto obligaría a que $K(P)$ fuera el conjunto vacío o todo $\widehat{\mathbb{C}}$, pero sabemos que no es ninguno de los dos.

Por consiguiente, su frontera $J(P) = \partial K(P)$ debe ser estrictamente no vacía.

- Para ver que el interior de $J(P)$ es vacío, procedemos por reducción al absurdo. Supongamos que el interior de $J(P)$ no es vacío. Aplicando directamente la Proposición 2.2.12, esto implicaría necesariamente que $J(P) = \widehat{\mathbb{C}}$. Sin embargo, sabemos que el infinito es un punto fijo superatractor (Proposición 2.1.8), por lo que existe todo un entorno alrededor del infinito que pertenece a su cuenca de atracción y, por tanto, al conjunto de Fatou $F(P)$. Como $F(P)$ y $J(P)$ son disjuntos por definición, es imposible que $J(P)$ cubra toda la esfera de Riemann. Hemos llegado a una contradicción, por lo que el interior de $J(P)$ debe ser vacío. $\square$

Proposición 2.2.14. [Mil06, Corolario 3.10] *El conjunto de Julia $J(P)$ carece de puntos aislados. Puesto que es un conjunto cerrado, no vacío y sin puntos aislados, $J(P)$ es un conjunto perfecto y, en consecuencia, contiene una cantidad no numerable de puntos.*

Proposición 2.2.15. *Sea z_0 un punto periódico parabólico de un polinomio P . Entonces, $z_0 \in J(P)$. Además, si U es una componente de Fatou tal que las órbitas de sus puntos bajo la iterada correspondiente P^k convergen hacia z_0 (lo que llamamos pétalo atractor), entonces $z_0 \in \partial U$.*

La demostración de esta Proposición se encuentra dividida en dos partes en [Mil06]. En el Lema 3.4 nos demuestra que un punto periódico parabólico pertenece al conjunto de Julia, y el Corolario 7.5 nos demuestra la última parte. Aun así, dejamos una idea de la demostración:

Por la Observación 2.1.5, en el entorno del punto parabólico existen pétalos repulsores, donde la familia de iteradas $\{P^n\}$ no podrá ser normal en z_0 , ya que puntos arbitrariamente cercanos a z_0 se alejan de él en ese pétalo bajo iteración. Al no ser un punto normal, entonces $z_0 \in J(P)$. Por otro lado, como las órbitas de los puntos de U convergen hacia el punto $z_0 \in J(P)$, y dado que los conjuntos de Fatou y Julia son disjuntos, el límite

de dichas órbitas no puede encontrarse en el interior de U . Por lo tanto, este punto de acumulación z_0 debe pertenecer forzosamente a su frontera ∂U .

2.3. Dinámica en superficies hiperbólicas

Proposición 2.3.1. *Dado que el conjunto de Fatou $F(P)$ es, por definición, un conjunto abierto (al ser el complementario del conjunto cerrado $J(P)$), cualquier componente conexa U de $F(P)$ es un subconjunto abierto de $\widehat{\mathbb{C}}$ y, por tanto, una superficie hiperbólica.*

Demostración. Al ser U una componente conexa del conjunto de Fatou, tenemos que $U \subset \widehat{\mathbb{C}} \setminus J(P)$. Sabemos por la Proposición 2.2.14 que el conjunto de Julia $J(P)$ es un conjunto que contiene infinitos puntos.

Visto dentro de la esfera de Riemann $\widehat{\mathbb{C}}$, la componente U está omitiendo todos los infinitos puntos del conjunto de Julia $J(P)$. Al omitir estrictamente más de tres puntos de $\widehat{\mathbb{C}}$, aplicando el Teorema A.4.5 concluimos directamente que U es una superficie hiperbólica. $\square$

Teorema 2.3.2. [Mil06, Cap 5] *Sea X una superficie de Riemann hiperbólica y conexa y $f : X \rightarrow X$ una función holomorfa. Entonces, el comportamiento de las iteradas de f se clasifica exactamente en una de las siguientes cuatro categorías mutuamente excluyentes:*

1. **Caso atractor:** *Si f tiene un punto fijo atractor, entonces todas las órbitas bajo f convergen hacia este punto fijo. La convergencia será uniforme en los subconjuntos compactos de X .*
2. **Escape:** *Si alguna órbita bajo f no tiene un punto de acumulación en X , entonces ninguna órbita tiene un punto de acumulación. De hecho, para cualquier conjunto compacto $K \subset X$ existirá un natural n_K tal que $K \cap f^n(K) = \emptyset$ para $n \geq n_K$.*
3. **Orden finito:** *Si f tiene dos puntos periódicos distintos, entonces alguna iterada f^n es la aplicación identidad y cada punto de X es periódico.*
4. **Rotación irracional:** *En todos los demás casos, X es isomorfa ya sea a un disco $\mathbb{D}$, a un disco perforado $\mathbb{D} \setminus \{0\}$ o a un anillo $A_r = \{z \mid 1 < |z| < r\}$, y la dinámica de f es conjugada a una rotación irracional $z \rightarrow e^{2\pi i \alpha} z$ con $\alpha \notin \mathbb{Q}$.*

Proposición 2.3.3 (Convergencia a un punto fijo en la frontera, [Mil06, Lema 5.5]). *Sea U un subconjunto abierto hiperbólico de una superficie de Riemann compacta. Supongamos que la aplicación $f : U \rightarrow U$ se extiende continuamente a la frontera ∂U , con a lo sumo un número finito de puntos fijos en ∂U . Si alguna órbita de f en U no tiene ningún punto de acumulación dentro de U , entonces todas las órbitas en U deben converger dentro de la clausura $\overline{U}$ a un único punto fijo de la frontera. Esta convergencia es uniforme sobre subconjuntos compactos de U .*

Proposición 2.3.4 (Lema del Caracol, [Mil06, Teorema 16.2]). *Supongamos que existe un polinomio P y un camino $\gamma : [0, \infty] \rightarrow V \setminus \{0\}$, siendo V un entorno del origen, tal que $P(\gamma(t)) = \gamma(t+1)$ para todo $t \geq 0$. Supongamos además que el camino converge al origen, es decir, $\lim_{t \rightarrow \infty} \gamma(t) = 0$.*

Entonces, el origen es un punto fijo de P tal que su multiplicador λ cumple $|\lambda| < 1$ (el origen es un atractor), o bien $\lambda = 1$ (el origen es un punto parabólico).

Demostración. Por hipótesis, la órbita $\gamma(0) \rightarrow P(\gamma(0)) = \gamma(1) \rightarrow P(\gamma(1)) = \gamma(2) \rightarrow \dots$ converge hacia el origen, por lo tanto por continuidad el origen es un punto fijo para P .

Sea λ el multiplicador del origen, vamos a ver que no puede ser repulsor por reducción al absurdo.

Supongamos que el origen es un punto fijo repulsor, entonces existe un entorno U alrededor del origen donde P expande las distancias:

$$|P(z)| \geq c|z|, \text{ para todo } z \in U, \text{ con } c > 1.$$

Como la curva $\gamma(t)$ converge a 0 por hipótesis, existirá un tiempo T suficientemente grande tal que $\gamma(t) \in U$ para todo $t \geq T$. Pero en ese entorno tendríamos que:

$$|\gamma(t+1)| = |P(\gamma(t))| \geq c|\gamma(t)|.$$

Inductivamente obtenemos que $|\gamma(t+n)| \geq c^n|\gamma(t)|$ para $n \geq 1$. Cuando $n \rightarrow \infty$ tendríamos que la sucesión de puntos escapa de U , contradiciendo nuestra hipótesis de que $\lim_{t \rightarrow \infty} \gamma(t) = 0$.

Por lo tanto, el origen no puede ser un punto fijo repulsor y su multiplicador cumple $|\lambda| \leq 1$.

Tendríamos tres opciones:

1. $|\lambda| < 1$.
2. $\lambda = 1$.
3. $|\lambda| = 1$ pero $\lambda \neq 1$.

Asumamos que $|\lambda| = 1$ con $\lambda \neq 1$ y lleguemos a contradicción. Sabemos que P es una función entera y que el origen es un punto fijo. Si hacemos el desarrollo de Taylor alrededor del origen nos queda lo siguiente:

$$P(z) = P(0) + P'(0)z + \frac{P''(0)}{2}z^2 + \frac{P'''(0)}{6}z^3 + \dots = \lambda z + a_2 z^2 + a_3 z^3 + \dots \text{ para } a_i \in \mathbb{C}.$$

A medida que la órbita del punto $\gamma(0)$ (trazada por el camino $\gamma(t)$) se acerca cada vez más al origen, el término dominante será el lineal, es decir, $\gamma(t+1) = P(\gamma(t)) \approx \lambda\gamma(t)$ cuando $t \rightarrow \infty$. Como tenemos que $\lambda = e^{i\theta}$ con $\theta \neq 0$, multiplicar por λ equivale a aplicar una rotación, por lo tanto el camino irá acercándose al origen formando una espiral.

Consideremos un segmento rectilíneo E que conecte un punto de la espiral $\gamma(t_0)$ con el punto correspondiente en la iteración siguiente $\gamma(t_0+1)$. Definimos W como el interior de la región compacta delimitada conjuntamente por el arco de la espiral $\gamma([t_0, t_0+1])$ y el segmento E .

Como $\gamma(t+1) = P(\gamma(t))$, por la continuidad de P tenemos que el polinomio mapea W estrictamente dentro de sí mismo, es decir, $P(W) \subsetneq W$.

En este punto, queremos intentar poder aplicar el Lema de Schwarz. Observamos que W es un dominio abierto, simplemente conexo y estrictamente contenido en el plano complejo $\mathbb{C}$. Aplicando el Teorema A.4.2, garantizamos la existencia de un isomorfismo conforme $\phi : W \rightarrow \mathbb{D}$. Lo podemos escoger de manera que envíe el origen al centro del disco, es decir, tal que $\phi(0) = 0$.

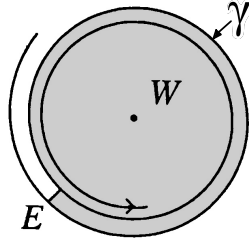


Figura 2.1: Dibujo del espiral y la región W . Adaptado de [Mil06]

Ahora, definimos la función holomorfa $f = \phi \circ P \circ \phi^{-1} : \mathbb{D} \rightarrow \mathbb{D}$. Como sabemos que $P(W) \subset W$ es una inclusión estricta, tenemos que f no puede ser biyectiva. Aplicando el Lema A.5.1 f no es una rotación y hemos llegado a una contradicción ya que es la única manera que $|\lambda| = 1$.

Por lo tanto, hemos descartado el caso $|\lambda| = 1$ con $\lambda \neq 1$, por lo que solo pueden darse los dos primeros casos: que el origen sea un punto fijo atractor ($|\lambda| < 1$) o un punto parabólico con $\lambda = 1$. $\square$

2.4. Clasificación de las componentes de Fatou

Como establecimos anteriormente, las componentes conexas del conjunto abierto $F(P)$ reciben el nombre de componentes de Fatou. En esta sección veremos el Teorema de Clasificación de las componentes de Fatou, el cual nos enseñará qué tipos de componentes invariantes (o periódicas) existen bajo la iteración de P .

Teorema 2.4.1 (Clasificación de las componentes de Fatou, adaptado de [Mil06, Teorema 16.1]). *Sea U una componente conexa de Fatou invariante bajo la iteración en P , entonces U pertenece a uno de los siguientes tres tipos:*

1. U es la cuenca de un punto fijo atractor.
2. U es una cuenca parabólica (o pétalo atractor) asociada a un punto fijo parabólico (donde el multiplicador es $\lambda = 1$).
3. U es un disco de Siegel, es decir, U es isomorfa al disco unidad $\mathbb{D}$ y la dinámica en esta componente es conjugada a una rotación irracional.

Demostración. Como U es una componente conexa de $F(P)$, aplicando la Proposición 2.3.1 tenemos que U es una superficie hiperbólica. Entonces, por el Teorema 2.3.2 tenemos cuatro posibilidades:

1. Caso atractor. U contiene un punto fijo atractor y todas las órbitas en U convergen al punto, es exactamente la definición del primer caso.
2. Escape. Ninguna órbita tiene puntos de acumulación dentro de U . Aplicando la Proposición 2.3.3, todas las órbitas convergen uniformemente a un único punto fijo $p \in \partial U$. Como este punto fijo $p \in \partial U$, entonces $p \in J(P)$.

Por un lado, no puede ser atractor, ya que sino estaría en el interior de $F(P)$, pero por el otro lado, no puede ser repulsor, ya que atrae todas las órbitas de U . Por lo

tanto, p es un punto fijo con $|\lambda| = 1$. Para ver que $\lambda = 1$ queremos poder aplicar la Proposición 2.3.4.

Dado que U es una componente conexa de un conjunto abierto en $\mathbb{C}$, es por definición un dominio abierto y conexo, lo que implica que topológicamente es conexo por caminos. Por tanto, podemos construir el camino continuo: tomamos un punto $z_0 \in U$ y sea $\gamma : [0, 1] \rightarrow U$ una curva continua que une z_0 con su imagen $P(z_0)$. Después, extendemos la curva γ para todo $t \geq 0$ definiendo iterativamente $c(t) = P^n(\gamma(t-n))$ para $t \in [n, n+1]$. De esta manera hemos construido un camino continuo $c : [0, \infty] \rightarrow U \setminus \{p\}$, tal que $P(c(t)) = c(t+1)$ para todo $t \geq 0$. Como tenemos que las órbitas en U convergen hacia p , el camino que hemos construido converge a p cuando $t \rightarrow \infty$.

Al cumplirse todas las hipótesis de la Proposición 2.3.4 y saber que el punto no es atractor ($|\lambda| \neq 1$) concluimos necesariamente que $\lambda = 1$ como queríamos ver.

3. Orden finito. P^n es la aplicación identidad para algún $n \in \mathbb{N}$. Pero P es una función racional de grado $d \geq 2$, si es la Id en algun abierto U , aplicando el Principio de Identidad de funciones holomorfas, sería la identidad en toda la esfera de Riemann, que no puede ser ya que una función racional de grado $d \geq 2$ tiene un número finito de puntos periódicos de periodo n .
4. Rotación irracional. U debe ser isomorfo a $\mathbb{D}, \mathbb{D} \setminus \{0\}$ o a un anillo $A_r = \{z \mid 1 < |z| < r\}$.

a) $\mathbb{D} \setminus \{0\}$. Si U fuese isomorfo a un disco “perforado”, el origen tendría que ser un punto fijo aislado en la frontera de U . Como U es una componente del conjunto de Fatou, la familia de iteradas forma una familia normal, por lo tanto, las órbitas a su alrededor están acotadas y 0 sería una singularidad evitable. Si eso pasase, entonces $0 \in F(P)$ y por lo tanto, la singularidad es evitable y el dominio no está perforado, sino que U es isomorfo a un disco completo. Esto nos llevaría al tercer tipo de componente de la clasificación: un Disco de Siegel.

b) A_r . Sería un Anillo de Herman, pero en el caso de los polinomios es imposible. Supongamos, por reducción al absurdo, que es posible. Sea P un polinomio y U una componente de Fatou conformemente equivalente a un anillo circular. La dinámica de P dentro del anillo es isomorfa a la de un anillo estándar donde actúa como una rotación irracional. Es decir, existe una curva simple cerrada invariante γ estrictamente contenida en el interior de la componente U tal que $P(\gamma) = \gamma$, por lo tanto, γ divide el plano en la componente acotada D (donde $\partial D = \gamma$) y la componente no acotada. Como P es una función entera, no puede tener polos dentro de D .

Escojamos un punto $a \in D$ y veamos si tiene preimágenes en el interior de D .

Definimos la función $g(z) = P(z) - a$. Como $P(\gamma) = \gamma$ y $a \in \overset{\circ}{D}$, tenemos que $P(z) \neq a$, para todo $z \in \gamma$, entonces g no tiene ceros en γ .

$P(\gamma) = \gamma$ da una vuelta entera sobre a , por lo tanto su índice es 1. Aplicando el Principio del Argumento, g tiene al menos un 0. Eso nos dice que $\forall a \in D, \exists z \in D$ tal que $P(z) = a$, por lo tanto $D \subset P(D)$.

Además $P(\partial D) = P(\gamma) = \gamma = \partial D$. Dado que P es una función holomorfa en el dominio acotado D y continua en su frontera, el Principio del Módulo Máximo establece que el máximo absoluto de $|P(z)|$ se alcanza en la frontera $\partial D = \gamma$. Como sabemos que $P(\partial D) = \partial D$, se concluye que la imagen del interior D no puede escapar de la región acotada por γ . Al ser P una función abierta, mapea interiores en interiores, por lo que concluimos que $P(D) = D$.

Pero si $P(D) = D$, D es topológicamente un disco y p lo mapea sobre sí mismo, aplicando el Teorema del punto fijo de Brouwer tenemos que hay un punto fijo en el interior de D , con lo cual sería imposible tener una dinámica conjugada a una rotación irracional alrededor de dicho punto fijo, contradiciendo el supuesto de que actuaba como un anillo estándar.

c) $\mathbb{D}$. Sería un disco de Siegel, P actuaría como una rotación irracional. $\square$

2.5. Polinomios hiperbólicos

A partir de las definiciones anteriores, deducimos algunas propiedades topológicas y dinámicas de los conjuntos, pero antes debemos definir los polinomios sobre los que trabajaremos (para un desarrollo riguroso de los conceptos de métricas conformes y espacios tangentes utilizados en esta sección, véase el Anexo A):

Definición 2.5.1. Diremos que un polinomio P es expansivo en su conjunto de Julia si existe una métrica conforme $\lambda(z)|dz|$ definida en un entorno de $J(P)$, tal que la derivada en cada $z \in J(P)$, denotada como DP_z , cumple:

$$\|DP_z(v)\|_\lambda > \|v\|_\lambda.$$

para cualquier vector tangente $v \in T_z\mathbb{C}$.

Es decir, el operador lineal DP_z estrictamente aumenta la longitud de cualquier vector tangente v medido con la métrica λ .

Observemos que esa desigualdad la podemos reescribir como el cociente:

$$\frac{\|DP_z(v)\|_\lambda}{\|v\|_\lambda} > 1$$

Gracias a la Observación A.2.2 sabemos que ese cociente no depende de la elección del vector tangente. Es decir, el valor del cociente depende únicamente de la elección del punto $z \in J(P)$.

Por lo tanto, podemos definir la norma del operador derivada (que cuantifica el factor de expansión en el punto z) como: $\|DP_z\|_\lambda = \sup_{v \neq 0} \frac{\|DP_z(v)\|_\lambda}{\|v\|_\lambda}$. Dado que P es continuamente diferenciable y la densidad de la métrica conforme es continua, esta función norma es continua respecto a z en el conjunto compacto $J(P)$. Como el conjunto de Julia $J(P)$ es compacto, aplicando el Teorema de Weierstrass tenemos que la función alcanzará su mínimo absoluto en algún punto del compacto. Sea $z_{min} \in J(P)$ ese punto, como sabemos que en todos los puntos de este compacto la derivada es mayor que 1, tenemos que:

$$k = \|DP_{z_{min}}\|_\lambda > 1.$$

Por lo tanto, podemos decir que para cualquier otro punto $z \in J(P)$ se cumplirá que $\|DP_z\|_\lambda \geq k > 1$.

Además, por la continuidad de la función, si tenemos que en el compacto $J(P)$ la derivada es $\geq k > 1$, podremos encontrar un entorno abierto U muy ajustado al conjunto de Julia, de modo que $\|DP_z\|_\lambda \geq k' > 1$ para todo $z \in U$, donde $1 < k' \leq k$.

Definición 2.5.2. Definimos el conjunto postcrítico de P como la colección de todas las iteradas futuras de sus puntos críticos. Es decir, si denotamos por $C(P) = \{z \in \mathbb{C} \mid P'(z) = 0\}$ al conjunto de puntos críticos, entonces:

$$\mathcal{P} = \bigcup_{n \geq 1} P^n(C(P)).$$

Teorema 2.5.3. [Mil06, Teorema 11.17] Sea P un polinomio que posee un disco de Siegel U , entonces la frontera de U está contenida en la clausura del conjunto postcrítico. Es decir, $\partial U \subset \overline{\mathcal{P}}$. Esto implica que, la órbita de al menos un punto crítico se acumula en la frontera del dominio de rotación.

Definición 2.5.4. Diremos que P es hiperbólico si la órbita de cualquier punto crítico converge a un ciclo periódico atractor, incluyendo el infinito.

Teorema 2.5.5 (adaptado de [Mil06, Teorema 19.1]). Las siguientes afirmaciones son equivalentes:

1. P hiperbólico.
2. La clausura del conjunto postcrítico es disjunta del conjunto de Julia, $\overline{\mathcal{P}} \cap J(P) = \emptyset$.
3. P es expansivo en su conjunto de Julia.

Demostración. Lo demostraremos mediante una cadena de implicaciones.

■ $1 \implies 2$:

Supongamos que P es hiperbólico. Por definición, la órbita de cada punto crítico converge a un ciclo periódico atractor (ya sea finito o el infinito). Sabemos que estos ciclos atractores y sus cuencas de atracción están contenidos en el interior del conjunto de Fatou $F(P)$ (como se deduce de la descomposición del Teorema 2.2.11 y el Teorema de Clasificación 2.4.1).

Como las cuencas de atracción son conjuntos abiertos y el conjunto de Julia $J(P)$ es compacto, existe una distancia estrictamente positiva entre los ciclos atractores (que son los puntos de acumulación de las órbitas críticas) y $J(P)$. Por lo tanto, el conjunto de todas las iteradas futuras de los puntos críticos $\mathcal{P}$, así como su clausura $\overline{\mathcal{P}}$, están contenidos en $F(P)$. Como $F(P) = \widehat{\mathbb{C}} \setminus J(P)$, concluimos directamente que $\overline{\mathcal{P}} \cap J(P) = \emptyset$.

■ $2 \implies 3$:

Supongamos que $\overline{\mathcal{P}} \cap J(P) = \emptyset$, por lo tanto, $J(P) \subset \widehat{\mathbb{C}} \setminus \overline{\mathcal{P}}$.

Definimos los siguientes conjuntos: $V = \widehat{\mathbb{C}} \setminus \overline{\mathcal{P}}$ y $W = P^{-1}(V)$. Veamos que W es un subconjunto de V . Si $z \notin V$, entonces $z \in \overline{\mathcal{P}}$, y por la invariancia del conjunto postcrítico tenemos que $P(z) \in \overline{\mathcal{P}}$, por lo tanto, $P(z) \notin V$. Por la elección inicial de z , podemos concluir que $W \subset V$.

Como V no tiene valores críticos de P , su preimagen W tampoco puede tener ningún punto crítico. Por lo tanto tenemos que la restricción $P : W \rightarrow V$ es un isomorfismo local en cualquier punto $z \in W$. Como el polinomio P es una aplicación propia de grado d , aplicando el Lema A.1.7 deducimos que $P : W \rightarrow V$ es un mapa recubridor de d hojas.

Para poder emplear la métrica hiperbólica y analizar si el polinomio P actúa como una expansión (aumentando la distancia entre puntos) o una contracción respecto a dicha métrica, es condición necesaria que los dominios considerados (en este caso, V y W) sean superficies hiperbólicas. El Teorema A.4.1 nos dice que, para conseguir un espacio hiperbólico tenemos que quitarle al menos tres puntos a $\hat{\mathbb{C}}$. Los polinomios 'triviales' ($z \mapsto z^d$) tienen únicamente dos puntos en su conjunto postcrítico, el 0 y el ∞ , por lo tanto no podríamos usar la métrica hiperbólica. Por esa misma razón, excluiríamos estos casos y supondremos que nuestro conjunto postcrítico está formado por al menos tres puntos. Así conseguimos que cada componente conexa de V y W sea una superficie hiperbólica.

Queremos ver que W está estrictamente contenido en V ($W \subsetneq V$). Suponiendo lo contrario tendríamos que V se mapea sobre sí mismo, pero entonces aplicando el Teorema 2.2.10 tendríamos que $\{P^n\}_{n \in \mathbb{N}}$ es normal en todo V (ya que estamos suponiendo que el postcrítico tiene al menos tres puntos). Pero entonces, por definición de ser normal, $V \subset F(P)$ y llegamos a contradicción.

Por la invariancia, sabemos que $P^{-1}(J(P)) = J(P)$. Como, por hipótesis, $J(P) \subset V$, se cumple que $J(P) \subset P^{-1}(V) = W$. En resumen, tenemos la cadena de inclusiones: $J(P) \subset W \subsetneq V$.

Como $W \subsetneq V$, la aplicación inclusión $i : W \hookrightarrow V$ es una función holomorfa cuya imagen está estrictamente contenida en V , por lo tanto no puede ser un isomorfismo global. Por el Teorema de Schwarz-Pick (Teorema A.5.2), toda función holomorfa entre superficies hiperbólicas que no sea un isomorfismo global es una contracción estricta de la métrica. Por tanto, para cualquier vector tangente $v \in T_z W$, se cumple que su norma en la métrica de V es estrictamente menor que en la de W :

$$\|i(v)\|_V < \|v\|_W \implies \|v\|_V < \|v\|_W.$$

Por otro lado, dado que $P : W \rightarrow V$ es un isomorfismo local entre dos superficies hiperbólicas, aplicando el Teorema A.5.2, tenemos que se cumple el caso de igualdad; es decir, la aplicación es una isometría local. Es decir, para todo $z \in W$ y todo vector tangente $v \in T_z W$, se cumple la igualdad:

$$\|DP_z(v)\|_V = \|v\|_W.$$

Combinando ambos resultados, obtenemos que $\|DP_z(v)\|_V > \|v\|_V$, lo que demuestra que el polinomio P actúa como una expansión de la métrica hiperbólica de V en el conjunto W . Como el conjunto de Julia estaba contenido en W , hemos encontrado un entorno donde el polinomio P es expansivo respecto a esta métrica.

- **3 $\implies$ 1:** Supongamos que P es expansivo en su conjunto de Julia y escogemos λ su métrica conforme con factor de expansión $k > 1$. Por la continuidad de la derivada podemos encontrar un entorno $N_\epsilon(J(P))$ lo suficientemente ajustado a $J(P)$ donde la derivada seguirá siendo mayor que uno.

Como los puntos críticos cumplen que la derivada en esos puntos vale cero, es imposible que pertenezcan al conjunto de Julia, ya que al ser P expansivo, la derivada es siempre mayor que uno. Por lo tanto, todos los puntos críticos están en el conjunto de Fatou $F(P)$.

Tomemos un punto $z \in N_\epsilon(J(P)) \setminus J(P)$. Dado que $J(P)$ es totalmente invariante ($P(J(P)) = J(P)$), la distancia del punto a dicho conjunto bajo la iteración de P , medida respecto a la métrica expansiva λ (es decir, $d_\lambda(z, J(P)) =$

$\inf_{w \in J(P)} d_\lambda(z, w)$), se incrementará estrictamente en cada paso por un factor mínimo de $k' > 1$. Consecuentemente, tras un número finito de iteraciones, la órbita acabará abandonando el entorno $N_\epsilon(J(P))$.

Sea z_0 un punto crítico arbitrario de P . Como hemos visto, $z_0 \in F(P)$. Analicemos su órbita $z_n = P^n(z_0)$. Afirmamos que esta órbita solo puede contener un número finito de puntos dentro del entorno expansivo $N_\epsilon(J(P))$. Si la órbita de este punto interseca $N_\epsilon(J(P))$, es imposible que entre en $J(P)$ debido a la invarianza total de ambos conjuntos ($F(P)$ y $J(P)$ son totalmente invariantes). Por tanto, si la órbita entra en la franja $N_\epsilon(J(P)) \setminus J(P)$, la propiedad expansiva descrita anteriormente obligará a que la distancia euclídea respecto a $J(P)$ aumente iterativamente por un factor mínimo $k' > 1$. Consecuentemente, tras un número finito de iteraciones, la órbita será expulsada de dicho entorno. Esto implica que los puntos de acumulación de la órbita crítica deben localizarse forzosamente en el interior de $F(P)$, a una distancia estrictamente positiva de $J(P)$.

Sea U la componente conexa de $F(P)$ que contiene dicho punto de acumulación. Entonces, alguna iterada P^i mapeará U en U y aplicando el Teorema 2.4.1 tenemos que U será:

1. Una cuenca de atracción.
2. Una cuenca parabólica.

Este caso no es posible. Por la Proposición 2.2.15, si U fuera una cuenca parabólica, todas las órbitas en su interior (incluyendo la órbita de nuestro punto crítico) convergerían hacia un punto fijo parabólico situado en la frontera ∂U . Dado que la frontera de cualquier componente de Fatou está siempre contenida en el conjunto de Julia ($\partial U \subset J(P)$), esto implicaría que la órbita del punto crítico se acercaría arbitrariamente a $J(P)$. Sin embargo, esto contradice lo que acabamos de demostrar: que los puntos de acumulación de la órbita del punto crítico deben mantenerse a una distancia estrictamente positiva de $J(P)$.

3. Un dominio de rotación.

Este caso tampoco es posible. Por el Teorema 2.5.3, si existiera un disco de Siegel U , entonces $\partial U \subset \overline{P}$. Como la frontera de cualquier componente de Fatou está contenida en el conjunto de Julia, tendríamos que $\overline{P} \cap J(P) \neq \emptyset$. Pero, bajo nuestra hipótesis de que P es expansivo en el conjunto de Julia, hemos demostrado que las órbitas de los puntos críticos se mantienen alejadas de la frontera y convergen hacia el interior del conjunto de Fatou. Por lo tanto, hemos llegado a una contradicción, por lo que U no puede ser un dominio de rotación.

Por descarte, la órbita debe converger a una órbita periódica atractora asociada a la cuenca de atracción. Como hemos visto que los puntos críticos están contenidos en $F(P)$, tenemos que la órbita de cualquier punto crítico debe converger a una órbita periódica atractora, lo cual corresponde exactamente a la definición de ser hiperbólico. $\square$

Capítulo 3

Computabilidad

Hasta ahora, hemos estudiado los conjuntos de Julia desde una perspectiva puramente dinámica y geométrica. Sin embargo, para responder a la pregunta de si estos conjuntos pueden ser “dibujados” o “calculados” mediante un algoritmo formal, necesitamos un marco teórico que conecte la naturaleza continua de los números reales con la naturaleza discreta de las máquinas de Turing. En este capítulo, introducimos los conceptos y teoremas necesarios para definir formalmente qué significa que un conjunto de Julia sea computable en tiempo polinómico. El desarrollo de esta sección sigue de cerca la exposición de [Bra05].

3.1. Conceptos previos

Definición 3.1.1. [Bra05, Definición 2.1.1] El conjunto de los números diádicos se define como $\mathcal{D} = \left\{ \frac{m}{2^n} \mid m \in \mathbb{Z}, n \in \mathbb{N} \right\}$.

Definición 3.1.2. Dado un número natural $k \in \mathbb{N}$, definimos el conjunto $\mathcal{D}_k$ como el subconjunto de los números diádicos cuya representación en base binaria tiene, a lo sumo, k dígitos a la derecha de la coma (lo que llamaremos precisión de k bits fraccionarios). Formalmente,

$$\mathcal{D}_k = \left\{ \frac{m}{2^k} \mid m \in \mathbb{Z} \right\}.$$

Para introducir el concepto de máquina de Turing, daremos primero su definición formal rigurosa y, a continuación, ofreceremos una explicación intuitiva para facilitar su comprensión.

Definición 3.1.3. [Sip12, Cap 3] Una máquina de Turing es un modelo computacional que realiza una lectura/escritura de manera automática sobre una entrada llamada cinta, generando una salida en esta misma. Una máquina de Turing con una sola cinta puede definirse como una 7-tupla:

$$M = (Q, \Sigma, \Gamma, s, b, F, \delta)$$

donde:

- Q es el conjunto finito de estados en los que puede estar la máquina.
- Σ es el alfabeto de entrada, un conjunto finito de símbolos distinto del espacio en blanco.
- Γ es el alfabeto de cinta, un conjunto finito de símbolos de cinta ($\Sigma \subset \Gamma$).

- $s \in Q$ es el estado inicial.
- $b \in \Gamma$ es el símbolo blanco, el único símbolo que se puede repetir un número infinito de veces.
- $F \subset Q$ es el conjunto de estados finales de aceptación.
- $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ es una función parcial denominada función de transición, donde L es un movimiento a la izquierda y R es un movimiento a la derecha.

El funcionamiento del modelo se basa en la función de transición, que recibe un estado inicial y una cadena de caracteres (la cinta, la cual puede ser infinita) perteneciente al alfabeto de entrada. La máquina va leyendo una celda de la cinta en cada paso, borrando el símbolo donde se encuentra su cabezal y escribiendo un nuevo símbolo perteneciente al alfabeto de cinta. Seguidamente desplaza el cabezal a la izquierda o a la derecha. Estos pasos se van repitiendo según se indique en la función de transición y se detendrá cuando llegue a un estado de aceptación siendo esa su salida. Por ejemplo, $\delta(q, a) = (r, b, L)$ significa que, cuando la máquina está en el estado q y lee el símbolo a en la cinta, pasa al estado r , reemplaza a por el símbolo b y mueve el cabezal hacia la izquierda.

Informalmente, una máquina de Turing consta de:

- Una cinta dividida por celdas. Cada celda contiene un símbolo de algún alfabeto finito, el cual contiene un símbolo especial llamado blanco y uno o mas símbolos adicionales. La máquina será provista de tanta cinta como necesite para su computación. Las celdas que no se hayan escrito previamente se asume que están rellenas con el símbolo blanco. En algunos modelos la cinta tendrá un extremo izquierdo marcado con un símbolo especial, pero en todos será indefinidamente extensible hacia la derecha.
- Un cabezal que puede leer y escribir símbolos en la cinta y mover la cinta a la izquierda o a la derecha.
- Un registro de estado que almacena el estado de la máquina de Turing, uno de los estados finitos. Se inicia con un estado inicial.
- Una tabla finita de instrucciones (función de transición), que dado el estado actual de la máquina y el símbolo que está leyendo en la cinta le indica a la máquina hacer lo siguiente en secuencia:
 - Borra o escribe un símbolo.
 - Mueve el cabezal (derecha, izquierda o permanecer en el mismo lugar).
 - Asume el mismo o el nuevo estado como prescrito.

Observación 3.1.4. Los números diádicos son los números que tienen una representación binaria finita, lo que permite que una máquina de Turing clásica pueda leerlos y escribirlos en su cinta sin necesidad de precisión infinita.

En lo sucesivo, utilizaremos M para denotar una máquina de Turing. La notación $M(n)$ o $M(d, n)$ representará la ejecución (o la salida) de dicha máquina al recibir como entrada el valor n o el par (d, n) , respectivamente.

Definición 3.1.5. [Bra05, Definición 2.1.2] Un número real $x \in \mathbb{R}$ es computable si y solo si existe una máquina de Turing $M(n)$, tal que para una entrada n , $M(n)$ emite un número diádico $d \in \mathcal{D}$ tal que $|x - d| < 2^{-n}$.

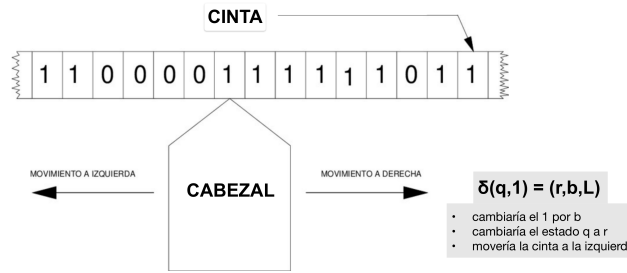


Figura 3.1: Esquema conceptual de la cinta de una máquina de Turing procesando un número diádico. (Imagen modificada a partir de un diagrama original de ResearchGate).

Ejemplo 3.1.6. Para podernos imaginar el funcionamiento de una máquina de Turing, construiremos una que aproxima el número $x = 1/3$. En binario, $1/3$ tiene una expansión periódica infinita: $0,010101\dots$

Supongamos que queremos calcular una representación de x con precisión $n = 3$. Según nuestra definición, la máquina debe devolver un número diádico d tal que $|1/3 - d| < 2^{-3} = 1/8$.

Para lograrlo, instanciamos los elementos de la 7-tupla de nuestra máquina:

- El alfabeto de entrada es $\Sigma = \{1\}$. Representaremos la precisión $n = 3$ en formato unario (tres unos consecutivos).
- El alfabeto de la cinta requiere símbolos adicionales para el cálculo y el formato de salida. Por tanto, $\Gamma = \{0, 1, ., X, b\}$, donde X se usará para marcar los bits ya procesados y b es el espacio en blanco.
- El conjunto de estados Q incluye el estado inicial s , el estado final de aceptación $f \in F$, y estados intermedios para buscar y escribir dígitos.

Proceso de cálculo guiado por δ : La función de transición $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ dicta cada paso de la máquina. El proceso será:

1. **Cinta inicial:** La cinta comienza con la entrada rodeada de blancos: **b 1 1 1 b** b... La máquina arranca en el estado s .
2. **Preparación del formato:** Las primeras instrucciones de δ ordenan a la máquina moverse a la derecha de la entrada, escribir un 0 seguido de un . (pertenecientes a Γ) para empezar la parte decimal del número diádico, y cambiar a un estado de “búsqueda”. La cinta queda: **b 1 1 1 b 0 .** b...
3. **Bucle de lectura/escritura:** A partir de aquí, la máquina entra en un ciclo lógico para ir generando el patrón $0101\dots$:
 - El cabezal retrocede (L) hasta leer el primer 1 de la entrada y lo sobrescribe con X (marcando así que ya se usó un nivel de precisión), cambia de estado (a uno que sería “debo escribir un cero”) y se mueve a la derecha.
 - Guiada por el anterior estado, la máquina avanza (R) hasta llegar al final de la cadena de salida, donde escribe un 0. La cinta queda: **b X 1 1 b 0 . 0** b...
 - El cabezal retrocede nuevamente, encuentra el siguiente 1 disponible, lo sobrescribe con X y la función δ la hace cambiar a otro estado (que sería “debo escribir

un uno"). Viaja al final de la salida y escribe un 1. Cinta actual: b X X 1 b 0 . 0 1 b...

- El ciclo se repite para el tercer y último 1. Se tacha con X y se escribe el siguiente dígito de la secuencia, otro 0. Cinta actual: b X X X b 0 . 0 1 0 b...

4. **Parada:** El cabezal retrocede buscando más unos. Como no hay, la función de transición hace que la máquina entre en el estado de aceptación $f \in F$ y el cómputo finaliza.

Resultado: En la cinta ha quedado escrito el número binario finito 0.010. Si lo convertimos a base diez, obtenemos el número diádico $d = \frac{0}{2} + \frac{1}{4} + \frac{0}{8} = 0,25$.

Verificamos el margen de error:

$$\left| \frac{1}{3} - 0,25 \right| = |0,333\ldots - 0,25| = 0,0833\ldots$$

Puesto que $0,0833\ldots < 0,125$ (que equivale a 2^{-3}), hemos conseguido tener una máquina de Turing que cumple con los requisitos de la definición.

Se deduce de la Definición 3.1.5 que solo puede haber una cantidad numerable de números reales computables. Esto se debe a que cualquier máquina de Turing puede codificarse unívocamente como una cadena finita de caracteres sobre un alfabeto finito (describiendo sus estados y transiciones), y el conjunto de todas las cadenas finitas sobre un alfabeto finito es numerable.

Por otro lado, muchos números reales “de uso común” son computables (los números racionales, muchos números trascendentales, etc.) y podemos obtener aproximaciones diádicas arbitrariamente buenas.

Definición 3.1.7. [Bra05, Definición 2.1.5] Sea $\phi : \{0,1\}^* \rightarrow \{0,1\}^*$ una función arbitraria (donde el superíndice $*$ denota el conjunto de todas las palabras o cadenas finitas formadas por el alfabeto $\{0,1\}$). Una máquina de Turing con oráculo, denotada por M^ϕ , es una máquina de Turing que, además de la cinta de trabajo ordinaria, está equipada con una cinta de consulta y dos estados: el estado de consulta y el estado de respuesta. Cuando la máquina entra en el estado de consulta, reemplaza el contenido actual de la cinta de consulta, digamos una cadena s , por su imagen $\phi(s)$, mueve el cabezal a la primera celda de la cinta de consulta y pone la máquina en el estado de respuesta. A efectos de evaluar la complejidad temporal, el coste de consultar al oráculo no es constante, sino que es proporcional a la precisión exigida (lo explicaremos más adelante).

Definición 3.1.8. [Bra05, Definición 2.1.6] Decimos que una función $\phi : \mathbb{N} \rightarrow \mathcal{D}$ representa al número real x si para cada $n \in \mathbb{N}$, se cumple que:

$$|x - \phi(n)| < 2^{-n}.$$

A esta función ϕ , que nos proporciona las aproximaciones garantizadas, la denominamos el oráculo de x .

Esta definición se extiende de manera natural a espacios de dimensión superior. En nuestro contexto, dado que los conjuntos de Julia habitan en el plano complejo (identificable isomórficamente con $\mathbb{R}^2$), un oráculo para un punto $x = (x_1, x_2) \in \mathbb{R}^2$ será una función $\phi : \mathbb{N} \rightarrow \mathcal{D}^2$ que, para cada precisión exigida n , devuelve un par de números diádicos $d = (d_1, d_2)$ que satisfacen la cota de distancia euclídea $\|x - d\| < 2^{-n}$.

Observación 3.1.9. Para un número computable podemos sustituir el oráculo $\phi(n)$ por una máquina $M(n)$ que lo calcule.

Definición 3.1.10. [Bra05, Definición 2.1.7] Una función $f : S \rightarrow \mathbb{R}$, donde $S \subset \mathbb{R}^2$, es computable si existe una máquina de Turing con oráculo $M^\phi(n)$ tal que, cada vez que el oráculo ϕ representa a un punto $x \in S$ (en el sentido bidimensional descrito anteriormente), $M^\phi(n)$ emite un número diádico $y_n \in \mathcal{D}$ tal que $|f(x) - y_n| < 2^{-n}$.

Teorema 3.1.11. [Bra05, Teorema 2.1.8] Sea $f : S \rightarrow \mathbb{R}$ una función computable en $S \subset \mathbb{R}^2$, entonces f es continua en S .

3.2. Computabilidad de conjuntos reales

En lo sucesivo, dado que trabajaremos sobre $\mathbb{R}^2$ (o subconjuntos suyos), utilizaremos la distancia euclídea $d(x, y) = \|x - y\|$. Además, definimos la distancia de un punto $x \in \mathbb{R}^2$ a un conjunto cerrado $A \subset \mathbb{R}^2$ como el ínfimo de las distancias a los puntos del conjunto:

$$d(x, A) = \inf_{a \in A} d(x, a).$$

Definición 3.2.1. Dado un conjunto compacto $C \subset \mathbb{R}^2$ y un radio $\epsilon > 0$, definimos la bola de radio ϵ centrada en C como el conjunto de puntos cuya distancia a C es menor que ϵ :

$$B(C, \epsilon) = \{x \in \mathbb{R}^2 \mid d(x, C) < \epsilon\}.$$

De ahora en adelante, fijaremos C como un subconjunto compacto de $\mathbb{R}^2$. Nuestro objetivo es saber si una máquina de Turing puede computar el conjunto C en su totalidad con una precisión garantizada y rigurosamente acotada por 2^{-n} .

Vamos a usar la intuición. Para cada precisión exigida $n \in \mathbb{N}$, queremos generar un conjunto U_n que sea la unión finita de discos cerrados que cumpla dos requisitos:

- U_n cubre C , es decir $C \subset U_n$.
- U_n no contiene puntos muy alejados de C . Para cada $u \in U_n$ hay un $c \in C$ tal que $\|u - c\| < 2^{-n}$. Es decir, $U_n \subset B(C, 2^{-n})$.

Con esta intuición, los centros de los círculos podrían ser números no computables, así que para formalizarlo vamos a obligar a que el cálculo de C genere U_n como una unión finita de bolas cerradas con centros y radios diádicos. Denotemos:

$$\mathcal{C}_2 = \left\{ \bigcup_{i=1}^l \overline{B(x_i, r_i)} \subset \mathbb{R}^2 \mid l \in \mathbb{N}, x_i \in \mathcal{D}^2, r_i \in \mathcal{D} \right\}.$$

Definición 3.2.2. [Bra05, Definición 2.2.2] Un conjunto compacto $C \subset \mathbb{R}^2$ se dice que es globalmente computable si existe una máquina de Turing $M(n)$ que, con la entrada n , emite una lista finita compuesta por los centros y radios diádicos que definen un conjunto $U_n \in \mathcal{C}_2$ tal que $C \subset U_n \subset B(C, 2^{-n})$. En este caso decimos que $M(n)$ calcula globalmente C .

Observemos que no podemos esperar que una máquina de Turing con oráculo decida si un punto está en el conjunto C o no. Si fuese así, la máquina estaría calculando la función característica χ_C , que no es continua y por tanto no puede ser computable por el Teorema 3.1.11. Como en la definición de globalmente computable no pedimos respuestas precisas, sino que permitimos tener algunas “áreas grises”, vamos a definir:

Definición 3.2.3. [Bra05, Definición 2.2.7] Decimos que el conjunto compacto $C \subset \mathbb{R}^2$ es localmente computable, si hay una máquina de Turing $M(d, n)$ tal que, para cada punto diádico $d \in \mathcal{D}^2$ y cada número natural $n \in \mathbb{N}$ recibidos como entrada, computa el valor de una función $f(d, n)$ que satisface las siguientes condiciones:

$$f(d, n) = \begin{cases} 1 & \text{si } B(d, 2^{-n}) \cap C \neq \emptyset \\ 0 & \text{si } B(d, 2 \cdot 2^{-n}) \cap C = \emptyset \\ 0 \text{ o } 1 & \text{en otro caso} \end{cases} \quad (3.2.1)$$

Teorema 3.2.4. [Bra05, Teorema 2.2.8] Las definiciones 3.2.2 y 3.2.3 son equivalentes. Es decir, un conjunto es globalmente computable si y solo si es localmente computable.

Demostración. ($\Rightarrow$) Supongamos que C es globalmente computable y busquemos poder definir una función una función $f(d, n)$ que satisface las condiciones de 3.2.1 para cada par (d, n) .

Lo primero que haremos será computar la imagen $U_{n+1} \subset \mathcal{C}_2$ de C usando la máquina de la Definición 3.2.2. Esta imagen tiene precisión $2^{-(n+1)}$. Como U_{n+1} está formado por una unión finita de discos con centros y radios diádicos, y la bola $B(d, 2^{-n})$ también posee parámetros diádicos, determinar si su intersección es vacía se reduce a un número finito de comparaciones entre distancias de centros y radios. Dado que las operaciones aritméticas y las comparaciones entre números diádicos son computables en un número finito de pasos por una máquina de Turing, podemos afirmar que existe un procedimiento algorítmico para verificar si $B(d, 2^{-n}) \cap U_{n+1} = \emptyset$. Entonces, si $B(d, 2^{-n}) \cap U_{n+1} = \emptyset$ devolveremos 0, de lo contrario, 1. Esta será nuestra $f(d, n)$. Veamos si cumple las condiciones 3.2.1:

- Sabemos que $C \subset U_{n+1}$, por lo tanto, si $B(d, 2^{-n}) \cap C \neq \emptyset$, entonces $B(d, 2^{-n}) \cap U_{n+1} \neq \emptyset$ y devolveremos 1.
- Sabemos que $U_{n+1} \subset B(C, 2^{-(n+1)})$. Si estamos en la situación que $B(d, 2 \cdot 2^{-n}) \cap C = \emptyset$, esto significa que $d(d, C) \geq 2 \cdot 2^{-n}$. Para cualquier $u \in U_{n+1}$, dado que $U_{n+1} \subset B(C, 2^{-(n+1)})$, existe al menos un $c \in C$ tal que $d(u, c) \leq 2^{-(n+1)}$. Aplicando la desigualdad triangular, obtenemos:

$$d(d, u) \geq d(d, c) - d(u, c) \geq d(d, C) - d(u, c) \geq 2 \cdot 2^{-n} - 2^{-(n+1)} = 3 \cdot 2^{-(n+1)}.$$

Dado que $3 \cdot 2^{-(n+1)} > 2 \cdot 2^{-(n+1)} = 2^{-n}$, deducimos que la distancia euclídea desde d a cualquier punto de U_{n+1} es estrictamente mayor que 2^{-n} . Por consiguiente, $B(d, 2^{-n}) \cap U_{n+1} = \emptyset$, y devolveremos 0.

($\Leftarrow$) Ahora supondremos que C es localmente computable y queremos poder codificar un conjunto U_n que cumpla los requisitos de ser globalmente computable.

Tenemos que C es acotado, por lo tanto existe un $M > 0$ tal que $C \subset [-M, M]^2$. Definimos G_n como una cuadrícula de puntos a distancia 2^{-n} en $[-(M+1), M+1]^2$.

Dada nuestra función $f(d, n)$, definimos U_n como la unión de bolas cerradas centradas en los puntos de una cuadrícula G_{n+i} , con $i \in \mathbb{Z}$, donde la función f es igual a 1 con precisión $n + 2$:

$$U_n = \bigcup \left\{ \overline{B(d, 2^{-(n+2)})} \mid d \in G_{n+i} \text{ y } f(d, n+2) = 1 \right\}$$

El objetivo de la constante i es hacer que la cuadrícula sea lo suficientemente “fina” para garantizar que cualquier punto del conjunto quede cubierto por las bolas que estamos colocando. Como U_n está formado por bolas de radio $2^{-(n+2)}$ centrados en los puntos de la cuadrícula G_{n+i} y nosotros queremos asegurar que $C \subset U_n$, necesitamos que para cualquier punto $x \in C$, exista un punto en la cuadrícula $d \in G_{n+i}$ tal que $d(x, d) < 2^{-(n+2)}$. La distancia entre los puntos de nuestra cuadrícula es de $2^{-(n+i)}$.

Imaginémonos la cuadrícula dividida por cuadrados formados por los puntos. Si cogemos un $x \in C$, el peor caso sería cuando x se encuentra exactamente en el centro de un cuadrado. La distancia del centro del cuadrado a cualquier esquina (cualquier punto de la cuadrícula) es la mitad de su diagonal. Si denotamos la diagonal del cuadrado por h , por Pitágoras tenemos que:

$$h^2 = (2^{-(n+i)})^2 + (2^{-(n+i)})^2.$$

Entonces la distancia máxima será $\frac{h}{2} = \frac{1}{2} \cdot \sqrt{2} \cdot 2^{-(n+i)}$. Queremos que se cumpla que:

$$\begin{aligned} \frac{1}{2} \cdot \sqrt{2} \cdot 2^{-(n+i)} &< 2^{-(n+2)}. \\ \sqrt{2} &< 2^{i-1}. \end{aligned}$$

Como estamos trabajando específicamente en $\mathbb{R}^2$, basta con elegir un número natural i que satisfaga esta desigualdad. Si tomamos $i = 2$, obtenemos $\sqrt{2} < 2^1$, lo cual es cierto. Por lo tanto, podemos definir directamente nuestra aproximación utilizando la cuadrícula G_{n+2} .

Es obvio que $U_n \in \mathcal{C}_2$. Ahora veamos que cumple los requisitos:

- Para cualquier $x \in C$, existe un $d \in G_{n+2}$, punto de la cuadrícula, tal que $d(x, d) < 2^{-(n+2)}$. Por lo tanto, tenemos que $B(d, 2^{-(n+2)}) \cap C \neq \emptyset$ y entonces $f(d, n+2) = 1$. Eso significa que $x \in B(d, 2^{-(n+2)}) \subset U_n$. Como pasa para cualquier punto de C , hemos visto que $C \subset U_n$.
- Para cualquier $y \in U_n$, existe un $d \in G_{n+2}$ tal que $d(y, d) < 2^{-(n+2)}$. Por lo tanto, tenemos que $f(d, n+2) = 1$, lo que significa que $B(d, 2^{-(n+2)}) \cap C \neq \emptyset$. Si la imagen de $f(d, n+2)$ es 1, no podemos estar en la situación de que $B(d, 2 \cdot 2^{-(n+2)}) \cap C = \emptyset$, por lo tanto, $d(d, C) < 2 \cdot 2^{-(n+2)} = 2^{-(n+1)}$. Aplicando la desigualdad triangular, obtenemos que

$$d(y, C) \leq d(y, d) + d(d, C) < 2^{-(n+2)} + 2^{-(n+1)} < 2^{-n}.$$

Y así hemos visto que $U_n \subset B(C, 2^{-n})$. □

La próxima definición de computabilidad se basa en máquinas de Turing con oráculo. Con un oráculo ϕ que representa un número real x , nuestro objetivo es poder devolver 1 si x está en C o 0 si x está alejado de C como mínimo a una distancia de 2^{-n} . Una vez más, existirá un “área gris” donde $0 < d(C, x) \leq 2^{-n}$ y cualquier respuesta será válida. Formalmente, definimos:

Definición 3.2.5. [Bra05, Definición 2.2.10] Decimos que un conjunto C es Ko computable si existe una máquina de Turing con oráculo $M^\phi(n)$, tal que si ϕ representa un punto $x \in \mathbb{R}^2$, entonces para una entrada n la salida de $M^\phi(n)$ es

$$M^\phi(n) = \begin{cases} 1 & \text{si } x \in C \\ 0 & \text{si } B(x, 2^{-n}) \cap C = \emptyset \\ 0 \text{ o } 1 & \text{en otro caso} \end{cases} \quad (3.2.2)$$

Observación 3.2.6. $M^\phi(n)$ no puede ser una función, porque los valores de $M^\phi(n)$ en el “área gris” no dependen solo del valor de la entrada x , sino también del oráculo particular ϕ que representa x .

Para poder ver con más claridad esta dependencia del oráculo, consideremos un ejemplo explícito donde la salida de la máquina falla al intentar definir una función.

Ejemplo 3.2.7. Sea $C = \{0\}$. Definimos el siguiente algoritmo para que $M^\phi(n)$ Ko compute el conjunto C :

1. Se consulta al oráculo el valor $d = \phi(n+1)$, el cual por definición garantiza que $|x - d| < 2^{-(n+1)}$.
2. Si $|d| < 2^{-(n+1)}$, el algoritmo devuelve 1; en caso contrario, devuelve 0.

Este algoritmo satisface las condiciones de la Ko computabilidad. Sin embargo, miremos que pasaría para la entrada específica $x = 2^{-(n+1)}$.

Para este valor de x , el oráculo tiene que devolver un valor d que cumpla:

$$|2^{-(n+1)} - d| < 2^{-(n+1)}.$$

Si resolvemos la desigualdad, vemos que el oráculo ϕ puede devolver cualquier número diádico d que se encuentre en el intervalo abierto $(0, 2^{-n})$. Entonces podríamos encontrarnos con las siguientes situaciones:

- Si el oráculo devuelve un d tal que $d < 2^{-(n+1)}$, la salida es 1.
- Si el oráculo devuelve un d tal que $d \geq 2^{-(n+1)}$ (lo cual sigue siendo válido si $d < 2^{-n}$), la salida es 0.

Por lo tanto, para el mismo número real de entrada $x = 2^{-(n+1)}$, la máquina $M^\phi(n)$ puede devolver 0 o 1 en función de la aproximación que elija el oráculo. Por lo tanto, hemos visto que no computa una función bien definida en dicho punto.

Antes de enunciar y demostrar el próximo teorema, es conveniente introducir una herramienta fundamental que usaremos en la demostración, proveniente de la Teoría de Grafos. Recordemos que, un árbol se define como un grafo que es conexo (todos los nodos están conectados) y acíclico (sin ciclos o caminos cerrados).

Proposición 3.2.8. [Bra05, Cap 2][Lema de König] Si cada vértice en un árbol tiene un grado finito, entonces el árbol es infinito si y solo si tiene una rama infinita.

Teorema 3.2.9. [Bra05, Teorema 2.2.11] Las definiciones 3.2.3 y 3.2.5 son equivalentes. Es decir, un conjunto es localmente computable si y solo si es Ko computable.

Demostración. Para simplificar la notación y facilitar la comprensión geométrica, presentaremos esta demostración para el caso unidimensional ($C \subset \mathbb{R}$).

($\Rightarrow$) Supongamos que C es localmente computable y que tenemos una máquina de Turing que computa una $f(d, n)$ de la familia 3.2.1. Para Ko computar haremos los siguientes pasos:

- Para el x que queramos darle una respuesta, consultamos su oráculo con precisión $n + 2$, $d = \phi(n + 2)$.
- Devolvemos $f(d, n + 2)$

Para ver que Ko computa, necesitamos ver que se satisface 3.2.2.

- Si $x \in C$, como $d(x, d) < 2^{-(n+2)}$ tenemos que $x \in B(d, 2^{-(n+2)}) \cap C \neq \emptyset$ y entonces $f(d, n + 2) = 1$.
- Si $B(x, 2^{-n}) \cap C = \emptyset$, como $d(x, d) < 2^{-(n+2)}$, podemos asegurar que $B(d, 2^{-n} - 2^{-(n+2)}) \subset B(x, 2^{-n})$. Para comprobar esto formalmente, tomamos cualquier punto $y \in B(d, 2^{-n} - 2^{-(n+2)})$ y aplicamos la desigualdad triangular:

$$d(y, x) \leq d(y, d) + d(d, x) < (2^{-n} - 2^{-(n+2)}) + 2^{-(n+2)} = 2^{-n}.$$

Por lo tanto, $y \in B(x, 2^{-n})$ para todo $y \in B(d, 2^{-n} - 2^{-(n+2)})$. Puesto que la bola con centro en x no interseca C , ningún subconjunto suyo lo hará, por lo que $B(d, 2^{-n} - 2^{-(n+2)}) \cap C = \emptyset$.

A continuación, sabiendo que: $2^{-n} = 2^2 \cdot 2^{-(n+2)} = 4 \cdot 2^{-(n+2)}$ y aplicándolo al radio de la bola, tenemos:

$$B(d, 2 \cdot 2^{-(n+2)}) \subset B(d, 2^{-n} - 2^{-(n+2)}).$$

Por consiguiente, $B(d, 2 \cdot 2^{-(n+2)}) \cap C \subset B(d, 2^{-n} - 2^{-(n+2)}) \cap C = \emptyset$ y entonces $f(d, n + 2) = 0$.

($\Leftarrow$) Esta implicación requiere una cierta preparación.

Por conveniencia supondremos que $C \subset [\frac{1}{4}, \frac{3}{4}]$, por lo tanto en 3.2.1 solo necesitaremos considerar puntos diádicos $d \in \mathcal{D} \cap [\frac{1}{4}, \frac{3}{4}]$. Además, asumiremos sin pérdida de generalidad que la precisión exigida es $n \geq 2$. De esta forma, el radio máximo de las bolas evaluadas será $2^{-2} = 0,25$, garantizando que $B(d, 2^{-n}) \subset [0, 1]$. De esta manera, nos aseguramos de que no le pedimos a la máquina que busque valores fuera del intervalo $[0, 1]$, que es donde están definidos los nodos de nuestro árbol (los casos triviales para $n = 1$ pueden resolverse algorítmicamente en tiempo constante con una comprobación finita). La demostración se extiende trivialmente a intervalos más grandes, ya que C estará siempre acotado por ser compacto.

Construiremos un árbol infinito T , donde cada vértice es un número diádico. Los números en el nivel l son de la forma $m \cdot 2^{-l}$ y la raíz, que está en el nivel 1 será $\frac{1}{2}$. Los vértices estarán etiquetados con el número en notación binaria.

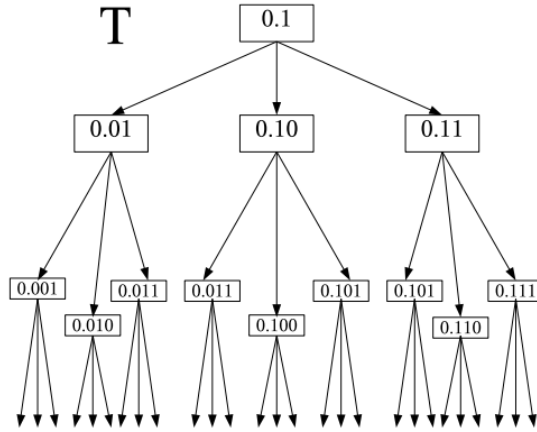


Figura 3.2: Los tres primeros niveles del árbol infinito T . Adaptado de [Bra05, Figura 2.6]

Observamos que cada vértice v en un nivel l tiene 3 hijos. Si la etiqueta de v es $m \cdot 2^{-l}$, las etiquetas de sus hijos serán

- $m \cdot 2^{-l} - 2^{-(l+1)} = (2m - 1) \cdot 2^{-(l+1)}$.
- $m \cdot 2^{-l} = 2m \cdot 2^{-(l+1)}$.
- $m \cdot 2^{-l} + 2^{-(l+1)} = (2m + 1) \cdot 2^{-(l+1)}$.

Podemos afirmar que los números a lo largo de cualquier camino p convergen a un valor $x_p \in [0, 1]$. Vamos a justificarlo usando las propiedades de las sucesiones de Cauchy y la acotación de la serie geométrica asociada. Veamos la justificación:

Sea $p(l)$ el valor numérico del nodo en el nivel l del camino p . Por construcción, al pasar de un nodo padre en el nivel l a cualquiera de sus tres hijos en el nivel $l + 1$, pueden pasar tres cosas: que $p(l + 1) = p(l)$, que $p(l + 1) = p(l) + 2^{-(l+1)}$, o que $p(l + 1) = p(l) - 2^{-(l+1)}$. Por tanto, la distancia entre un nivel y el siguiente está acotada por:

$$|p(l + 1) - p(l)| \leq 2^{-(l+1)}.$$

Para demostrar la convergencia, tomamos dos niveles cualesquiera n y m , con $m > n$. Si aplicamos iterativamente la desigualdad triangular tenemos:

$$\begin{aligned} |p(m) - p(n)| &= |p(m) - p(m - 1) + p(m - 1) - p(m - 2) + \cdots + p(n + 1) - p(n)| \\ &\leq \sum_{k=n}^{m-1} |p(k + 1) - p(k)| \leq \sum_{k=n}^{m-1} 2^{-(k+1)}. \end{aligned}$$

Tenemos una serie geométrica de razón $\frac{1}{2}$. Su valor exacto es $2^{-n} - 2^{-m}$. Por lo tanto:

$$|p(m) - p(n)| \leq 2^{-n} - 2^{-m} < 2^{-n}.$$

Podemos decir que para cualquier $\epsilon > 0$ podemos encontrar un nivel N suficientemente grande tal que $2^{-N} < \epsilon$ y así tener una sucesión de Cauchy. Al estar en $\mathbb{R}$, que es un espacio métrico completo, toda sucesión de Cauchy converge a un límite real, que denotamos como x_p .

Por último, veamos por qué $x_p \in [0, 1]$. Sabemos que la raíz tiene el valor $p(1) = \frac{1}{2}$. Calculemos el valor más grande (el camino que siempre suma $2^{-(l+1)}$) y el valor mínimo:

$$x_{max} = \frac{1}{2} + \sum_{l=1}^{\infty} 2^{-(l+1)} = 1, \quad x_{min} = \frac{1}{2} - \sum_{l=1}^{\infty} 2^{-(l+1)} = 0.$$

Por lo tanto, cualquier camino convergerá a un límite $x_p \in [0, 1]$.

Por otro lado, por como se ha construido el árbol, tenemos todas las combinaciones posibles en la notación binaria. Por lo tanto, para cualquier $x \in [0, 1]$ existirá un camino p tal que $x_p = x$ (podríamos escoger p como los prefijos de la expansión binaria de x). Además, si vamos por el camino p que converge a x_p , tenemos que en el nivel n , la distancia entre x_p y $p(n)$ está acotada:

$$|x_p - p(n)| \leq 2^{-n} < 2^{-(n-1)}.$$

Esta cota se deduce del hecho de que la máxima distancia entre $p(n)$ y x_p ocurrirá cuando siempre escojamos el camino que suma (o siempre el que resta) $2^{-(n+1)}$. Esta distancia sería:

$$\sum_{k=n+1}^{\infty} 2^{-k} = \frac{2^{-(n+1)}}{1 - 1/2} = 2^{-n}.$$

Por lo tanto, x_p siempre estará contenido en el intervalo $[p(n) - 2^{-n}, p(n) + 2^{-n}]$, lo que garantiza que $|x_p - p(n)| \leq 2^{-n}$. Por lo tanto, $\phi(n) = p(n+1)$ es un oráculo válido para x_p (cogemos $p(n+1)$ porque el oráculo exige que la precisión sea estricta). Para que quede más claro, acabamos de ver que cada camino que baja desde la raíz hasta el infinito será un oráculo distinto.

Ahora nuestro objetivo es buscar nuestra $f(d, n)$ de 3.2.1.

Dada una entrada (d, n) , lo que queremos es poder saber si $B(d, 2^{-n})$ interseca con C . Intuitivamente lo que haremos será buscar en todos los puntos cerca de d si alguno está en C , y para ello utilizaremos la máquina que Ko computa C . Explicaremos paso a paso:

- Paso 1, localizar la zona de “búsqueda”. Dado que los nodos del nivel n distribuyen los números diádicos en intervalos consecutivos de longitud 2^{-n} , cualquier valor d caerá forzosamente entre dos de estos puntos. Por tanto, encontraremos dos nodos v_1 y v_2 en el nivel n tales que sus valores diádicos asociados, que denotaremos como $val(v_1)$ y $val(v_2)$, cumplan $val(v_1) \leq d \leq val(v_2)$ y además $|val(v_1) - d| < 2^{-n}$ y $|val(v_2) - d| < 2^{-n}$. Los subárboles que tienen como raíz estos dos nodos, representan a todos los números x que viven cerca de d . Denotemos por p_1 y p_2 los caminos que empiezan en la raíz del árbol T y terminan en los nodos v_1 y v_2 , y $M^\phi(n)$ la máquina que Ko computa C .
- Paso 2, utilizar la máquina $M^\phi(n)$. Simularemos el procedimiento con el subárbol con raíz v_1 , se hará de la misma manera para el subárbol con raíz v_2 . Cada consulta que haga la máquina al oráculo, si le pide una precisión $m < n$ (un nivel superior) le devolveremos $p_1(m+1)$. Pero, si por lo contrario pidiese un dato del nivel $n+1$, como f quiere comprobar toda la zona, pausaremos la máquina y haremos tres clones de ella. Cada clon le dará a su máquina un oráculo distinto (los tres oráculos posibles). Si le pide una precisión $m > n$, consideraremos todos los descendientes de v_1 en el nivel $m+1$, clonaremos 3^{m-n+1}

máquinas y le daremos a cada una un oráculo distinto. Y así, sucesivamente. Esto crea una avalancha de máquinas ejecutándose en paralelo.

- Paso 3, respuesta final. Si alguna máquina ha devuelto 1, devolveremos 1, sino devolveremos 0.

Necesitamos comprobar dos cosas:

- 1. El algoritmo que hemos construido termina.
- 2. La respuesta que da satisface 3.2.1.

Supongamos que el algoritmo nunca termina. Todas las computaciones que se van haciendo las podemos ver como un árbol donde cada nodo es una de ellas. Una computación C_1 será el padre de las 3^{as} subcomputaciones que se hacen a partir de él. Si no terminan, entonces pueden pasar dos cosas: o una de las computaciones C' no llama a subcomputaciones pero no termina, o el árbol de computaciones a realizar es infinito.

En el primer caso, si denotamos por P' el camino en el árbol inicial T que lleva al oráculo de C' , entonces P' corresponde a un número diádico d' el cuál tendrá su oráculo ϕ' . Es decir, podemos decir que computar C' sería ejecutar $M^{\phi'}(n)$. Entonces $M^{\phi'}(n)$ no termina en este caso, contradicción.

En el segundo caso, por el Lema 3.2.8, el árbol tiene una rama infinita. Denotemos la rama infinita por $C_1, C_2, C_3, \dots$. Es decir, C_1 llama a C_2 , C_2 llama a C_3 , etc. Ya sabemos que cada C_i trabaja con un camino p_i de T , y que p_{i+1} extiende p_i para cada i . Por lo tanto, la secuencia infinita de C_i corresponde a un camino infinito p de T . Sabemos que ese camino converge a un número real $x \in [0, 1]$, y p da lugar a un oráculo ϕ para x . En ese caso, la secuencia infinita de C_i simula la computación de $M^\phi(n)$. Por lo tanto $M^\phi(n)$ no termina, contradicción. Esto demuestra que el algoritmo termina.

Ahora vamos a ver el punto dos.

- Caso 1: $B(d, 2^{-n}) \cap C \neq \emptyset$. En este caso, existe un real $x \in C \cap B(d, 2^{-n})$. Por construcción, los subárboles con raíces en v_1 y v_2 recubren los intervalos $[val(v_1) - 2^{-n}, val(v_1) + 2^{-n}]$ y $[val(v_2) - 2^{-n}, val(v_2) + 2^{-n}]$. Puesto que $val(v_1) \leq d \leq val(v_2)$ y la distancia de ambos vértices a d es menor que 2^{-n} , la unión de estos intervalos cubre topológicamente por completo la bola $B(d, 2^{-n})$. En consecuencia, el número real x coincide necesariamente con el límite de algún camino descendente p contenido en uno de estos dos subárboles. Este camino p , tendrá su oráculo ϕ con el que una de las ramas de computación del algoritmo será una simulación de $M^\phi(n)$, y devolverá 1.
- Caso 2: $B(d, 3 \cdot 2^{-n}) \cap C = \emptyset$. En este caso, todos los puntos descendientes de v_1 o v_2 están a una distancia de, a lo sumo, $2 \cdot 2^{-n}$ de distancia de d . Esto se debe a que para cualquier punto y de los subárboles, tenemos que:

$$|y - d| = |(y - v_1) + (v_1 - d)| \leq |y - v_1| + |v_1 - d| < 2^{-n} + 2^{-n} = 2 \cdot 2^{-n}.$$

Lo que acabamos de ver, es lo mismo que decir que están al menos a distancia 2^{-n} de C , ya que para cualquier $c \in C$ tenemos que:

$$|c - y| \geq |c - d| - |y - d| \geq (3 \cdot 2^{-n}) - (2 \cdot 2^{-n}) \geq 2^{-n}.$$

Por lo tanto, cualquier computación en sus subárboles devolverá 0.

Con esto, tenemos una f tal que:

$$f(d, n) = \begin{cases} 1 & \text{si } B(d, 2^{-n}) \cap C \neq \emptyset \\ 0 & \text{si } B(d, 3 \cdot 2^{-n}) \cap C = \emptyset \\ 0 \text{ o } 1 & \text{en otro caso} \end{cases}$$

Pero, es muy sencilla usar f para computar una función que cumpla 3.2.1. Definamos:

$$g(d, n) = f(d - 2^{-(n+1)}, n + 1) \vee f(d, n + 1) \vee f(d + 2^{-(n+1)}, n + 1).$$

Veamos que g sí que cumple 3.2.1.

- Si $B(d, 2^{-n}) \cap C \neq \emptyset$, tenemos que $|d - c| < 2^{-n}$ para algún $c \in C$, entonces estamos en uno de estos tres casos:
 - $|d - c + 2^{-(n+1)}| < 2^{-(n+1)}$.
 - $|d - c| < 2^{-(n+1)}$.
 - $|d - c - 2^{-(n+1)}| < 2^{-(n+1)}$.

Por lo tanto, alguna de las tres f devolverá 1 y entonces g devolverá 1.

Comprobemos que se cumple obligatoriamente al menos uno de los tres casos sabiendo que:

$$-2 \cdot 2^{-(n+1)} = -2^{-n} < d - c < 2^{-n} = 2 \cdot 2^{-(n+1)}.$$

Si desarrollamos las inecuaciones de los tres casos anteriores, observamos que equivalen a que el valor $d - c$ pertenezca, respectivamente, a los intervalos $(-2 \cdot 2^{-(n+1)}, 0)$, $(-2^{-(n+1)}, 2^{-(n+1)})$ y $(0, 2 \cdot 2^{-(n+1)})$. Como la unión de estos tres subintervalos recubre por completo el intervalo original $(-2 \cdot 2^{-(n+1)}, 2 \cdot 2^{-(n+1)})$, el valor $d - c$ caerá forzosamente en alguno de ellos, garantizando que se satisfaga al menos una de las condiciones.

- Si $B(d, 2 \cdot 2^{-n}) \cap C = \emptyset$, entonces se cumple que:
 - $B(d - 2^{-(n+1)}, 3 \cdot 2^{-(n+1)}) \cap C = \emptyset$.
 - $B(d, 3 \cdot 2^{-(n+1)}) \cap C = \emptyset$.
 - $B(d + 2^{-(n+1)}, 3 \cdot 2^{-(n+1)}) \cap C = \emptyset$.

Por lo tanto, las tres f devuelven 0 y g devuelve 0.

Comprobemos que los tres casos se cumplen sabiendo que $|d - c| > 2 \cdot 2^{-n}$.

- En el primer caso, usando la desigualdad $|A - B| \geq |A| - |B|$, tenemos que:

$$|d - 2^{-(n+1)} - c| \geq |d - c| - 2^{-(n+1)} > 2 \cdot 2^{-n} - 2^{-(n+1)} = \frac{3}{2} \cdot 2^{-n}.$$

- En el segundo caso es sencillo ver que:

$$|d - c| > 2 \cdot 2^{-n} > \frac{3}{2} \cdot 2^{-n}.$$

- En el último caso, usando la desigualdad $|A + B| \geq |A| - |B|$, tenemos que:

$$|d + 2^{-(n+1)} - c| \geq |d - c| - 2^{-(n+1)} > 2 \cdot 2^{-n} - 2^{-(n+1)} = \frac{3}{2} \cdot 2^{-n}.$$

□

Observación 3.2.10. La demostración anterior se ha expuesto en dimensión uno por claridad expositiva. La generalización a $\mathbb{R}^2$ se obtiene de forma análoga sustituyendo los intervalos unidimensionales por cuadrados diádicos en el plano, y ajustando el número de hijos en el árbol de búsqueda.

Aclaración de notación: En el contexto de la complejidad computacional, $|d|$ no denota la norma euclídea del vector d , sino la longitud de su representación como cadena de símbolos en la cinta de la máquina de Turing. Al trabajar en $\mathbb{R}^2$, el punto diádico $d = (d_1, d_2)$ se codifica concatenando las representaciones de ambas coordenadas con un símbolo separador, por lo que su longitud total es $|d| = |d_1| + |d_2| + 1$.

Definición 3.2.11. [Bra05, Definición 3.1.6] Sea $T : \mathbb{N} \rightarrow \mathbb{N}$ una función de complejidad temporal. Diremos que el conjunto compacto $C \subset \mathbb{R}^2$ es localmente computable en tiempo $T(n)$, si existe una máquina de Turing $M(d, n)$ que se ejecuta en un tiempo a lo sumo de $T(n + |d|)$ y que computa una función de la familia 3.2.1.

Diremos que C es localmente computable en tiempo polinómico, o simplemente computable en tiempo polinómico, si es computable en un tiempo $p(n)$, para algún polinomio p .

Definición 3.2.12. [Bra05, Definición 3.1.7] Sea $T : \mathbb{N} \rightarrow \mathbb{N}$ una función de complejidad temporal. Diremos que el conjunto compacto $C \subset \mathbb{R}^2$ es Ko-computable en tiempo $T(n)$, si existe una máquina de Turing con oráculo $M^\Phi(n)$ que se ejecuta en un tiempo máximo de $T(n)$, y que computa una función de la familia 3.2.2.

Diremos que C es Ko-computable en tiempo polinómico, o simplemente Ko P-computable, si es computable en un tiempo $p(n)$ para algún polinomio p .

Observación 3.2.13. En este modelo, se contabiliza una unidad de tiempo por cada bit consultado al oráculo. Esto es crucial para la complejidad: cuando la máquina solicita al oráculo un número con precisión k , el oráculo le devuelve un número diádico que aproxima el valor real con un error menor que 2^{-k} . Para representar y transmitir este número en binario, el oráculo debe proporcionar al menos k bits. Como la máquina de Turing lee la cinta celda por celda, procesar esa respuesta le llevará k pasos de su tiempo disponible. Por lo tanto, si la máquina dispone de un tiempo total de ejecución acotado por $T(n)$, el número máximo de bits que puede leer del oráculo está estrictamente acotado por $T(n)$. Es decir, el coste temporal de lectura ya está implícito en $T(n)$, lo que impide matemáticamente que la máquina solicite precisiones arbitrariamente altas (mayores a $T(n)$) sin exceder su límite de tiempo.

Teorema 3.2.14. [Bra05, Teorema 3.2.2] Sea $C \subset \mathbb{R}^2$ un conjunto compacto. Si C es computable en tiempo polinómico (localmente computable en tiempo polinómico), entonces C es Ko P-computable.

Demostración. Existe una máquina de Turing $M(d, n)$ que computa una función de la familia 3.2.1 en un tiempo polinómico $p(n)$. Por el Teorema 3.2.9, tenemos que C es Ko computable. Nos falta ver si la Ko computación se produce en tiempo polinómico. Vamos por pasos:

- Leer la respuesta del oráculo con precisión $n + 2$ toma $n + 2$ unidades de tiempo, por lo tanto toma un tiempo lineal $\mathcal{O}(n)$.
- La ejecución de la función local $f(d, n + 2)$ toma un tiempo polinómico acotado por un polinomio $p(n + 2)$.

El tiempo total de ejecución de la máquina $M^\phi(n)$ es $\mathcal{O}(n) + p(n+2)$, lo cual sigue siendo un tiempo polinómico respecto a n , por lo tanto, tenemos que C es Ko P-computable. $\square$

3.3. Clases de complejidad P y NP

Antes de relacionar la computabilidad de conjuntos con su complejidad temporal, es necesario presentar las clases de complejidad. Dado que el enfoque principal de este trabajo es el análisis algorítmico de los conjuntos de Julia, presentaremos aquí únicamente las definiciones funcionales fundamentales y los resultados principales. El desarrollo formal, la introducción del problema SAT y las demostraciones detalladas de esta sección se han relegado al Anexo B.

Definición 3.3.1. *La clase de complejidad P contiene a todos los problemas de decisión que pueden ser resueltos por una máquina de Turing en tiempo polinómico respecto al tamaño de la entrada.*

De manera intuitiva, P representa el conjunto de problemas para los cuales es posible computar una respuesta de manera eficiente.

Definición 3.3.2. *La clase de complejidad NP contiene a los problemas de decisión para los cuales, dada una posible solución (frecuentemente llamada certificado), es posible verificar su validez utilizando una máquina de Turing en tiempo polinómico.*

Una de las cuestiones fundamentales de la informática teórica es saber si $P = NP$. La inmensa mayoría de la comunidad científica asume que $P \neq NP$. La estrecha relación entre la computabilidad de conjuntos geométricos y estas clases de complejidad queda establecida por los siguientes dos teoremas, cuyas demostraciones formales se encuentran en el Anexo B:

Teorema 3.3.3. [Bra05, Teorema 3.2.2] *Sea $C \subset \mathbb{R}^2$ un conjunto compacto. Si todo conjunto Ko P-computable es también computable en tiempo polinómico (localmente), entonces $P = NP$.*

Teorema 3.3.4. [Bra05, Teorema 3.2.2] *Si $P = NP$, entonces todo conjunto Ko P-computable es localmente computable en tiempo polinómico.*

Sin embargo, independientemente de si $P = NP$ o no, siempre podemos garantizar una cota algorítmica superior para cualquier conjunto Ko P-computable, un resultado que será esencial para el análisis del tiempo de ejecución de nuestros algoritmos en el siguiente capítulo:

Teorema 3.3.5. [Bra05, Teorema 3.2.2] *Una versión más débil del recíproco del Teorema 3.2.14 es: si C es Ko P-computable, entonces es computable en tiempo exponencial. Además, si la máquina que Ko P-computa C lee a lo sumo $p(n)$ bits de la entrada, entonces C es computable en tiempo $n^{\mathcal{O}(1)}2^{\mathcal{O}(p(n)+n)} = 2^{\mathcal{O}(p(n)+n)}$.*

Demostración. Esta demostración se obtiene inmediatamente aplicando fuerza bruta para evaluar la función local de acuerdo con simulaciones de la máquina con oráculo (ver la demostración del Teorema B.0.4 en el Anexo B), lo que produce un algoritmo exponencial para calcular C . $\square$

Capítulo 4

Computabilidad de los conjuntos de Julia hiperbólicos

En este capítulo uniremos los conceptos dinámicos desarrollados en el Capítulo 2 y la geometría del Anexo A con la teoría de computabilidad del Capítulo 3. Nuestro objetivo es demostrar, probado originalmente por Mark Braverman en [Bra05], que los conjuntos de Julia para polinomios hiperbólicos son computables en tiempo polinómico.

El algoritmo depende de evaluar la órbita de un punto bajo la iteración del polinomio, y la clave de su rapidez reside en la propiedad expansiva de la métrica hiperbólica. De aquí en adelante, P denotará un polinomio hiperbólico de grado m . Toda esta sección sigue la tesis [Bra05], a partir de la sección 4.4.1.

4.1. Información no uniforme y preprocesamiento

El algoritmo que construiremos utilizará información no uniforme (información que depende de P , pero no depende de la precisión n). El algoritmo no calcula estas constantes sobre la marcha; asumimos que, dado el polinomio P , podemos precalcularlas y dárselas a la máquina.

Sabemos por la Proposición 2.1.10 que el polinomio P posee, a lo sumo, $m - 1$ órbitas periódicas atractoras finitas. Denotaremos el número de estas órbitas por $o \leq m - 1$. Sea $X = \{O_1, \dots, O_o\}$ el conjunto de estas órbitas, donde cada órbita O_i tiene un periodo l_i y está formada por los puntos $O_i = (x_{i1}, x_{i2}, \dots, x_{il_i})$.

Dado que la máquina de Turing operará con números diádicos, no podemos proporcionarle los puntos exactos x_{ij} , sino que le daremos una aproximación diádica $A = \{(a_{i1}, \dots, a_{il_i})\}_{i=1}^o$ acotada por un error máximo $\epsilon_o > 0$, de tal forma que la distancia entre el punto aproximado y el real cumpla $\|a_{ij} - x_{ij}\| \leq \epsilon_o$ para todo punto del ciclo. Para garantizar que el algoritmo no falle al trabajar con estos “centros desplazados”, necesitamos acotar entornos alrededor de estas órbitas y determinar esta cota de error ϵ_o .

La existencia de las siguientes cotas se deduce directamente de las propiedades de los ciclos atractores estudiadas en el Capítulo 3. Dado que el multiplicador del ciclo satisface $|\lambda| < 1$ (Definiciones 2.1.4 y 2.1.6), el polinomio actúa localmente como una contracción estricta. Esto garantiza analíticamente que, dentro de la cuenca de atracción (Definición 2.1.7), podemos definir vecindades exactas donde la órbita queda confinada y converge de

forma controlada hacia el ciclo:

Proposición 4.1.1. *Existen radios $r_{ij} > 0$ y factores de contracción $s_i \in (0, 1)$ para cada $i \in \{1, \dots, o\}$ y $j \in \{1, \dots, l_i\}$ tales que:*

- Si $\|z - x_{ij}\| < r_{ij}$, entonces la órbita de z converge a la órbita atractora O_i .
- $P(B(x_{ij}, r_{ij})) \subset B(x_{i,j+1}, s_i \cdot r_{i,j+1})$ (donde el índice $j + 1$ se toma módulo l_i).

De forma análoga a los ciclos finitos, el infinito actúa como un punto fijo superatractor (Proposición 2.1.8). Necesitamos definir una frontera a partir de la cual sepamos que la órbita de un punto acabará divergiendo a infinito.

Proposición 4.1.2. *Tal y como se demostró explícitamente durante la construcción del conjunto acotado en la Proposición 2.2.2, existen constantes reales $R' > R > 0$ tales que:*

- Si $\|z\| > R$, la órbita de z diverge a infinito.
- Para todo z con $\|z\| > R$, se cumple que $\|P(z)\| > R'$.

Definimos la distancia mínima de la dinámica como:

$$\tilde{d}_l = \min \left(R' - R, \min_{i,j} \{r_{ij} - s_i \cdot r_{ij}\} \right) > 0.$$

Observación 4.1.3 (Interpretación geométrica del margen $\tilde{d}_l$). Si hacemos que el error $\epsilon_o = \tilde{d}_l/4$ tendremos lo siguiente:

- El margen de seguridad de la dinámica real: Sabemos que en el plano complejo, cuando iteramos un entorno de radio r_{ij} centrado en x_{ij} mediante el polinomio P , la imagen resultante no llena por completo el siguiente entorno de radio $r_{i,j+1}$. Al estar contraída por un factor $s_i < 1$, el disco imagen se sitúa estrictamente en el interior del disco destino. Geométricamente, queda una franja entre los dos radios cuyo grosor es, al menos, $r_{i,j+1} - s_i \cdot r_{i,j+1} > 0$. Un fenómeno análogo ocurre en la cuenca del infinito, donde el espacio entre la frontera de divergencia R y la imagen R' genera un margen equivalente. La constante $\tilde{d}_l$ representa el grosor mínimo garantizado de todas estas franjas.
- La perturbación computacional: Una máquina de Turing, por su propia definición, no puede operar aritméticamente con números reales arbitrarios, sino que está limitada a trabajar con aproximaciones finitas mediante números diádicos. Esto introduce dos perturbaciones geométricas insalvables:
 - Desplazamiento del origen: La máquina no “apunta” al verdadero centro del ciclo atractor x_{ij} , sino a una aproximación diádica que actúa como un centro falso, ligeramente desplazado.
 - Error de redondeo: Al evaluar las operaciones aritméticas del polinomio P , la máquina trunca los resultados, generando una pequeña bola de incertidumbre en cada iteración.

La exigencia de limitar el error de la aproximación a una cuarta parte del margen de seguridad ($\epsilon_o = \tilde{d}_l/4$) es el mecanismo que compatibiliza ambos mundos. Al imponer esta cota, garantizamos que la suma del error topológico por el desplazamiento del centro y

el error aritmético de la iteración sea estrictamente menor que el grosor $\tilde{d}_l$ de la corona circular.

En consecuencia, este margen absorbe completamente la inexactitud de la máquina de Turing: por mucho que la órbita diádica simulada se desvíe del valor continuo ideal, caerá inevitablemente dentro de la corona de seguridad y, por tanto, la órbita computada jamás podrá escapar accidentalmente de la verdadera cuenca de atracción.

Sea $Y = \{y_1, \dots, y_{m-1}\}$ el conjunto de los puntos críticos de P . Como P es hiperbólico, sabemos por la Definición 2.5.4 que la órbita de cada uno de estos puntos críticos converge a una de las órbitas atractoras finitas o al infinito.

Proposición 4.1.4. *Existe un radio $r_c > 0$ unificado y un tiempo máximo de iteraciones $q \in \mathbb{N}$ tales que, si un punto z está cerca de un punto crítico ($\|z - y_k\| < r_c$ para algún k), entonces tras q iteraciones su imagen ya ha entrado en una zona de atracción segura. Es decir, cumple una de las dos condiciones:*

- $\|P^q(z) - x_{ij}\| < r_{ij}$ para algún punto de un ciclo atractor.
- $\|P^q(z)\| > R'$ (ha escapado hacia el infinito).

Finalmente, para que el algoritmo pueda acotar el error propagado y aplicar expansiones métricas (como veremos usando la métrica hiperbólica), necesitamos controlar el módulo de la derivada.

El control del módulo de esta derivada es una consecuencia geométrica de la hiperbolicidad del polinomio y de su propiedad expansiva (Definición 2.5.4 y Teorema 2.5.5). Dado que la derivada es continua y evaluamos sobre conjuntos acotados alejados de los puntos críticos, podemos asegurar que:

Proposición 4.1.5. *Existen constantes reales $d > 0$ y $D \geq 1$ tales que:*

- (Cota inferior): $|P'(z)| > d$ para todo punto z que se encuentre dentro del disco $\|z\| \leq R'$, pero fuera de los entornos críticos (es decir, $\|z - y_k\| > r_c/2$ para todo k).
- (Cota superior): $|P'(P(z))| < D$ y $|P'(z)| < D/2$ para todo punto z tal que $\|z\| < R'$.

Gracias a toda esta información precalculada (los periodos, las aproximaciones A de las órbitas, las cotas R , R' , q , d y D), la máquina de Turing poseerá un “mapa” exacto de las zonas seguras del plano complejo, lo que le permitirá decidir la dinámica de cualquier punto en tiempo polinómico.

4.2. Construcción de dominios

El objetivo central de nuestro algoritmo será determinar en qué momento podemos detener la iteración de un punto para decidir, con absoluta certeza, si pertenece o no al conjunto de Julia $J(P)$. Para ello, nos apoyamos en la métrica hiperbólica definida en el Anexo A.

Proposición 4.2.1. [Bra05, Lema 4.4.1] *Definimos el siguiente conjunto:*

$$\tilde{U} = B(0, R') \setminus \bigcup_{i=1}^o \bigcup_{j=1}^{l_i} \overline{B(x_{ij}, s_i \cdot r_{ij})}.$$

Si definimos su preimagen bajo el polinomio como $\tilde{V} = P^{-1}(\tilde{U})$, se cumplen las siguientes propiedades:

- $\tilde{V} \subset \left(B(0, R) \setminus \bigcup_{i=1}^o \bigcup_{j=1}^{l_i} \overline{B(x_{ij}, r_{ij})} \right) \subset \tilde{U}$.
- La distancia euclídea entre el dominio interior $\tilde{V}$ y el complementario del dominio exterior ($\tilde{U}^c = \mathbb{C} \setminus \tilde{U}$) está acotada inferiormente por $\text{dist}(\tilde{V}, \tilde{U}^c) \geq \tilde{d}_l$.

Demostración. Para facilitar la exposición, denotaremos al conjunto intermedio como W :

$$W = B(0, R) \setminus \bigcup_{i=1}^o \bigcup_{j=1}^{l_i} \overline{B(x_{ij}, r_{ij})}.$$

- Sea $z \in \tilde{V}$. Sabemos que $P(z) \in \tilde{U}$, con lo cual, $\|P(z)\| < R'$ y $P(z) \notin \overline{B(x_{ij}, s_i \cdot r_{ij})}$ para cualquier par de índices i, j . Demostraremos que $z \in W$ por reducción al absurdo:

Supongamos que $z \notin B(0, R)$, lo que equivale a decir que $\|z\| \geq R$. Por la Proposición 4.1.2, $\|P(z)\| \geq R'$. Sin embargo, esto contradice directamente el hecho de que $P(z) \in \tilde{U}$. Por lo tanto, la suposición es falsa y debe cumplirse que $z \in B(0, R)$.

Supongamos que existe algún par de índices i, j tales que $z \in \overline{B(x_{ij}, r_{ij})}$. Por la Proposición 4.1.1, $P(z) \in \overline{B(x_{i,j+1}, s_i \cdot r_{ij})}$. Sin embargo, por definición, el conjunto $\tilde{U}$ excluye explícitamente todas estas bolas. Llegamos nuevamente a una contradicción, por lo tanto, $z \notin \overline{B(x_{ij}, r_{ij})}$ para todo i, j .

Al cumplirse ambas condiciones, queda demostrado que $z \in W$ y, por ende, que $\tilde{V} \subset W$.

Sea ahora un punto $z \in W$. Por la definición del conjunto W , sabemos que $\|z\| < R$, lo que implica que $\|z\| < R'$ y por lo tanto $z \in B(0, R')$. También tenemos que $\|z - x_{ij}\| > r_{ij}$ para todo ciclo, y como $s_i \in (0, 1)$, sabemos que $s_i \cdot r_{ij} < r_{ij}$, por lo que $z \notin \overline{B(x_{ij}, s_i \cdot r_{ij})}$ para cualquier i, j .

Verificadas ambas propiedades, concluimos que $z \in \tilde{U}$.

- Como hemos visto que $\tilde{V} \subset W$, se cumple la siguiente desigualdad:

$$\text{dist}(\tilde{V}, \tilde{U}^c) \geq \text{dist}(W, \tilde{U}^c).$$

Por otro lado, el complementario de $\tilde{U}$ en el plano complejo se escribe como:

$$\tilde{U}^c = \{w \in \mathbb{C} : |w| \geq R'\} \cup \left(\bigcup_{i=1}^o \bigcup_{j=1}^{l_i} \overline{B(x_{ij}, s_i \cdot r_{ij})} \right).$$

Evaluemos la distancia entre un punto $z \in W$ y cualquier punto $w \in \tilde{U}^c$ bajo los dos escenarios posibles para w :

- (Frontera exterior): Si $\|w\| \geq R'$, como $z \in W$ implica que $\|z\| < R$, aplicamos la desigualdad triangular inversa para obtener:

$$\|w - z\| \geq \|w\| - \|z\| > R' - R$$

Sabemos que $R' - R \geq \tilde{d}_l$. Por tanto, $\|w - z\| \geq \tilde{d}_l$.

- (Fronteras interiores): Si $\|w - x_{ij}\| \leq s_i \cdot r_{ij}$ para algún i, j , como $z \in W$ cumple que $\|z - x_{ij}\| > r_{ij}$, aplicamos la desigualdad triangular inversa:

$$\|z - w\| \geq \|z - x_{ij}\| - \|w - x_{ij}\| > r_{ij} - s_i \cdot r_{ij}$$

Nuevamente, se cumple que $r_{ij} - s_i \cdot r_{ij} \geq \tilde{d}_l$. Por tanto, $\|z - w\| \geq \tilde{d}_l$.

Al quedar acotados inferiormente todos los casos por $\tilde{d}_l$, se concluye que $\text{dist}(W, \tilde{U}^c) \geq \tilde{d}_l$, lo que implica por transitividad métrica que $\text{dist}(\tilde{V}, \tilde{U}^c) \geq \tilde{d}_l$. $\square$

Ahora ya hemos construido los dominios base $\tilde{U}$ y $\tilde{V}$, los cuales recubren y aíslan al conjunto de Julia, pero aún pueden contener puntos críticos en su interior. Para poder aplicar las herramientas de la geometría hiperbólica, necesitamos una región donde el polinomio actúe como un mapa recubridor sin ramificaciones. Para ello, iteramos hacia atrás estos dominios un número de veces suficiente para garantizar la expulsión de dichos puntos críticos.

Proposición 4.2.2. [Bra05, Lema 4.4.2] Sea $q \in \mathbb{N}$ el tiempo de escape de los puntos críticos definido en la Proposición 4.1.4, y sea D la cota superior global de la derivada. Definimos los dominios de trabajo como las preimágenes:

$$U = P^{-q-1}(\tilde{U}), \quad V = P^{-q-1}(\tilde{V}).$$

Estos dominios satisfacen las siguientes propiedades:

1. El conjunto de Julia está estrictamente contenido en el interior: $J(P) \subset V \subset U$.
2. No existen puntos críticos del polinomio en U (y por tanto, tampoco en V). Es decir, para todo punto crítico y_i , $B(y_i, r_c) \cap U = \emptyset$.
3. La distancia euclídea entre V y el complementario de U está acotada inferiormente por $d_l = \frac{\tilde{d}_l}{D^{q+1}} > 0$, donde $\tilde{d}_l$ es el margen inicial entre $\tilde{V}$ y $\tilde{U}$.

Demostración. 1. Sabemos que el conjunto de Julia es totalmente invariante. Dado que $J(P) \subset \tilde{V} \subset \tilde{U}$, al tomar $q+1$ preimágenes se preservan las inclusiones, obteniendo $P^{-q-1}(J(P)) \subset P^{-q-1}(\tilde{V}) \subset P^{-q-1}(\tilde{U})$, lo que equivale a $J(P) \subset V \subset U$.

2. Por la definición de r_c y q , cualquier punto dentro del entorno crítico $B(y_i, r_c)$, tras ser iterado q veces (y por extensión, $q+1$ veces), escapa del dominio acotado $\tilde{U}$. Por tanto, su preimagen de orden $q+1$ jamás puede pertenecer a U .
3. Sean $z \in V$ y $w \in U^c$, y sea γ cualquier curva diferenciable por partes que conecte z con w . Por definición, la longitud de la curva imagen al aplicarle el polinomio P viene dada por la integral $L(P(\gamma)) = \int_\gamma |P'(x)| |dx|$. Dado que la derivada está acotada globalmente por $|P'(x)| \leq D$, podemos acotar la integral:

$$L(P(\gamma)) \leq \int_\gamma D |dx| = D \cdot L(\gamma).$$

Aplicando este mismo razonamiento de forma inductiva, la longitud de la curva imagen tras aplicar P^{q+1} cumple la cota:

$$L(P^{q+1}(\gamma)) \leq D^{q+1} \cdot L(\gamma).$$

Por la definición de los dominios ($V = P^{-q-1}(\tilde{V})$ y $U = P^{-q-1}(\tilde{U})$), sabemos que el inicio de la curva $P^{q+1}(z)$ pertenece a $\tilde{V}$, y su final $P^{q+1}(w)$ pertenece a $\tilde{U}^c$. Como la distancia euclídea mínima garantizada entre los conjuntos base $\tilde{V}$ y $\tilde{U}^c$ es $\tilde{d}_l$, la longitud de cualquier curva que los conecte debe ser mayor o igual a dicha distancia. Es decir:

$$\tilde{d}_l \leq L(P^{q+1}(\gamma)).$$

Encadenando ambas desigualdades, obtenemos que $\tilde{d}_l \leq D^{q+1} \cdot L(\gamma)$, lo que despejando implica:

$$L(\gamma) \geq \frac{\tilde{d}_l}{D^{q+1}}.$$

Dado que esta cota inferior se cumple para absolutamente cualquier curva γ que conecte V con U^c , tomamos el ínfimo sobre todas las curvas posibles y concluimos que la distancia mínima entre los conjuntos verifica $\text{dist}(V, U^c) \geq \frac{\tilde{d}_l}{D^{q+1}} = d_l > 0$. $\square$

Por el Teorema A.4.5, sabemos que los conjuntos U y V son superficies de Riemann hiperbólicas. Por lo tanto, sobre ellas están definidas las métricas hiperbólicas con densidades $p_U(z)$ y $p_V(z)$ respectivamente.

Proposición 4.2.3. [Bra05, Lema 4.4.4] *Para la métrica hiperbólica en el dominio U , se cumplen las siguientes cotas euclídeas:*

- (Cota inferior global): $p_U(z) > \frac{2}{R'}$ para todo $z \in U$.
- (Cota superior interior): $p_U(z) < \frac{2}{d_l}$ para todo $z \in V$.

Demostración. ■ Cota inferior: Por el Corolario A.5.3 aplicado al conjunto U y a la bola $B(0, R')$, la densidad $p_U(z)$ es estrictamente mayor que la densidad de la bola $B(0, R')$ evaluada en ese mismo punto. Si denotamos la densidad de dicha bola como $p_B(z)$, afirmamos que $p_U(z) > p_B(z)$ para todo $z \in U$. La densidad exacta de un disco de radio R' en el punto z es $\frac{2R'}{(R')^2 - \|z\|^2}$. El denominador es como máximo $(R')^2$, por lo tanto, la densidad siempre está acotada inferiormente por $\frac{2}{R'}$.

- Vamos a usar un resultado clásico que nos dice que, la densidad métrica de un dominio en un punto z está acotada superiormente de forma universal por la inversa de la distancia euclídea desde dicho punto hasta la frontera del dominio (con un factor de 2), es decir, $p_U(z) \leq \frac{2}{\text{dist}(z, \partial U)}$. Este resultado podemos encontrarlo en [BM06][Teorema 8.2]. Aplicando ese resultado y sustituyendo esta distancia por el segundo punto de la Proposición 4.2.1 obtenemos que $p_U(z) < \frac{2}{d_l}$ para cualquier punto del dominio interior V . $\square$

Proposición 4.2.4. [Bra05, Lema 4.4.5] *Existe una constante explícita $c > 1$ tal que $p_V(z) \geq c \cdot p_U(z)$ para todo $z \in V$.*

Demostración. Sea la función cociente $f(z) = \frac{p_V(z)}{p_U(z)}$. Como $V \subset U$, aplicando directamente el Corolario A.5.3 sabemos que $p_U(z) < p_V(z)$, por lo que la función cociente es estrictamente mayor que 1 en todo el dominio V . Analicemos el comportamiento en la frontera: una propiedad fundamental de la métrica hiperbólica es que su densidad tiende a infinito al aproximarse a los límites del dominio. A medida que un punto z se acerca a la frontera de V , $p_V(z) \rightarrow \infty$, mientras que $p_U(z)$ se mantiene finito (ya que la frontera de V está a una distancia d_l del exterior de U). Esto provoca que el cociente tienda a infinito

en los bordes de V . Al ser una función continua, definida en un conjunto acotado, que tiende a infinito en la frontera, el Teorema de Weierstrass nos garantiza que alcanzará su mínimo absoluto dentro de V . Dado que la función es estrictamente mayor que 1 en todo el interior, este mínimo c será mayor que 1. $\square$

Proposición 4.2.5. [Bra05, Lema 4.4.9] *Para cualquier par de puntos x, y en la misma componente conexa de V , tenemos que $d_U(x, y) < M_W = \frac{128(R')^2}{d_1^2}$.*

Proposición 4.2.6. [Bra05, Lema 4.4.11] *Gracias a la cuidadosa construcción de los dominios $V \subset U$, podemos aplicar el Teorema de Schwarz-Pick (Teorema A.5.2) a las ramas inversas de P y combinarlo con la cota de la Proposición 4.2.4. Como resultado, el polinomio P expande de forma uniforme las distancias hacia el conjunto de Julia en la métrica hiperbólica. Es decir, para todo $x \in V$, la distancia hiperbólica d_U satisface:*

$$d_U(J(P), P(x)) \geq c \cdot d_U(J(P), x) \quad \text{con } c > 1$$

Proposición 4.2.7 (Cota lineal del tiempo de escape). *Si un punto $x \in \mathbb{C}$ cumple que $\text{dist}(x, J(P)) \geq 2^{-n}$, entonces su órbita bajo el polinomio P abandonará V en un número de pasos s acotado linealmente por n , es decir, $s = \mathcal{O}(n)$.*

Demostración. Utilizando la Proposición 4.2.3, sabemos que si la distancia euclídea inicial al conjunto de Julia es mayor o igual que 2^{-n} , la distancia hiperbólica satisface $d_U(J(P), x) \geq \frac{2 \cdot 2^{-n}}{R'}$.

Al aplicar el polinomio s veces, por la Proposición 4.2.6 tenemos que:

$$d_U(J(P), P^s(x)) \geq c^s \cdot d_U(J(P), x) \geq c^s \cdot \frac{2 \cdot 2^{-n}}{R'}.$$

Por otra parte, sabemos por la Proposición 4.2.5 que el diámetro hiperbólico máximo de cualquier componente conexa de V está acotado superiormente por M_W . Para que la órbita del punto acabe saliendo de V se debe cumplir que:

$$c^s \cdot \frac{2 \cdot 2^{-n}}{R'} > M_W \implies c^s > \frac{M_W \cdot R'}{2} 2^n.$$

Aplicando logaritmos en base c a ambos lados de la desigualdad, aislamos el número de iteraciones s :

$$s > \log_c \left(\frac{M_W \cdot R'}{2} \right) + n \cdot \log_c(2).$$

Dado que tanto M_W , R' como $c > 1$ son constantes fijas que dependen únicamente del polinomio P , el término dominante de la cota para s es lineal respecto a n . Concluimos, por tanto, que el tiempo necesario para que un punto exterior escape de la zona de seguridad es $s = \mathcal{O}(n)$. $\square$

Habiendo demostrado que el tiempo de escape es lineal, podemos conectar la dinámica con la Teoría de la Complejidad algorítmica.

Para demostrar de forma rigurosa el resultado principal de esta memoria, la resolución se estructura de manera incremental a través de tres algoritmos interconectados. La presentación de esta secuencia responde a una estrategia metodológica clara: primero se establece un marco conceptual base mediante un enfoque de fuerza bruta local (Algoritmo

1) que, aunque ineficiente debido a su coste exponencial, garantiza la correctitud geométrica en zonas seguras. Posteriormente, se introduce un mecanismo de control (Algoritmo 2) que rastrea la evolución de la órbita de un punto y expande controladamente su escala de distancia hasta que es seguro invocar al algoritmo base. Finalmente, este procedimiento se integra en un recubrimiento por cuadrícula (Algoritmo 3) que elimina los bloques exponenciales y alcanza la complejidad estrictamente polinómica definitiva.

4.2.1. Algoritmo 1 [Bra05, Sección 4.4.3]

Este algoritmo tiene como objetivo clasificar si un punto del plano pertenece al conjunto de Julia o es exterior al mismo mediante la iteración directa de su órbita por fuerza bruta.

Entrada:

1. La información no uniforme del polinomio P precalculada y descrita en la sección anterior (concretamente las constantes: $r_{ij}, R, R', \tilde{d}_l, q, d, D, c$).
2. El punto $x \in \mathbb{C}$ del plano complejo que queremos evaluar, y los coeficientes $c_0, c_1, \dots, c_m$ del polinomio P , proporcionados en una cinta oráculo.
3. m el grado del polinomio y n la precisión requerida.

Salida:

1. 0 si $d(J(P), x) > 2^{-n}$.
2. 1 si $x \in J(P)$.
3. 0 o 1 en otro caso.

Proceso:

1. Calcular la constante c (Proposición 4.2.4).
2. Calcular un número natural $N = \mathcal{O}(n)$ tal que

$$N \geq 2 + \log_c(M_W \cdot R' \cdot 2^{n-1}) + q.$$

3. Calcular $P^N(x)$ con un error máximo de $\frac{\tilde{d}_l}{4}$.
 - 3.1. Si $P^N(x) \notin \tilde{U}$, devolver 0.
 - 3.2. Si $P^N(x) \in \tilde{V}$, devolver 1.
 - 3.3. En cualquier otro caso, devolver 0 o 1.

Gracias a la información no uniforme conocemos las órbitas atractoras con suficiente precisión como para decidir si se cumple una de las posibilidades 3.1 o 3.2.

Proposición 4.2.8. *Si $d(J(P), x) > 2^{-n}$, el Algoritmo 1 devuelve 0, y si $x \in J(P)$, devuelve 1.*

Demostración. ■ Si nuestro punto empieza con $d(J(P), x) > 2^{-n}$, entonces $P^N(x) \notin \tilde{U}$ ya que si recordamos la Proposición 4.2.7 estamos haciendo que N sea mayor que los pasos necesarios para salir de V y finalmente le sumamos $q+2$ pasos para asegurarnos de que sale de $\tilde{U}$, por lo que el Algoritmo 1 devuelve 0 en este caso.

- Si, por otro lado, $x \in J(P)$, entonces por la invarianza $P^n(x) \in J(P) \subset \tilde{V}$, y el Algoritmo 1 devolverá 1. $\square$

Proposición 4.2.9 (Complejidad temporal del Algoritmo 1). *La complejidad temporal del Algoritmo 1 está acotada por $S_1(n) = 2^{\mathcal{O}(n)}$. Por consiguiente, el conjunto de Julia hiperbólico es KO computable en tiempo exponencial.*

Demostración. Como $N = \mathcal{O}(n)$, realizamos un número lineal de operaciones sobre x , y por tanto requerimos un número lineal de bits de x para alcanzar el nivel de precisión (fijo) requerido al final del cálculo. Por tanto, gracias al Teorema 3.3.5 sabemos que $J(P)$ es computable en tiempo $n^{\mathcal{O}(1)} \cdot 2^{\mathcal{O}(n)} = 2^{\mathcal{O}(n)}$. $\square$

4.2.2. Algoritmo 2 [Bra05, Sección 4.5.2]

Antes de presentar el siguiente algoritmo, necesitamos un resultado que nos permita estimar la distancia desde un punto a $J(P)$ tanto por arriba como por abajo.

Proposición 4.2.10. [Bra05, Lema 4.5.1] *Existen constantes positivas $\alpha, \beta > 0$ calculables a partir de los datos iniciales tales que para cualquier $z \in B(0, R')$ y $0 < \epsilon < \alpha$ que satisfagan $d(J(P), z) \leq \epsilon$, tenemos:*

$$(|P'(z)| - \beta\epsilon)d(J(P), z) \leq d(J(P), p(z)) \leq (|P'(z)| + \beta\epsilon)d(J(P), z).$$

Esta Proposición nos dice que el cambio en la distancia de z a $J(P)$ está controlado localmente por el factor de expansión $|P'(z)|$, salvo por un pequeño error relativo.

La idea del algoritmo es iterar el punto controlando el error relativo hasta que la distancia crezca a una escala mayor ϵ , momento en el cual podremos pedirle al Algoritmo 1 que nos de una respuesta.

Entrada:

1. La información no uniforme descrita anteriormente.
2. Los coeficientes $c_0, c_1, \dots, c_m$ dada en una cinta oráculo.
3. m el grado del polinomio y n la precisión requerida.
4. El punto a estudiar dado como un número binario x de longitud $\mathcal{O}(n)$.

Salida:

1. 0 si $d(J(P), x) > 2^{-n}D$.
2. 1 si $d(J(P), x) < 2^{-n}/32$.
3. 0 o 1 en otro caso.

Proceso:

1. Calcular las constantes c (Proposición 4.2.4), α y β (Proposición 4.2.10).
2. Calcular un número natural $N = \mathcal{O}(n)$ tal que

$$N \geq 2 + \log_c(M_W \cdot R' \cdot 2^{n-1}) + q.$$

3. Calcular $P^N(x)$ con un error máximo de $\frac{\tilde{d}_l}{4}$.
 - 3.1. Si $P^N(x) \in \tilde{V}$, devolver 1.
 - 3.2. En otro caso, ir al paso 4.
4. Elegir un valor $\epsilon \in 1/\mathcal{O}(n)$ que sea una potencia de 2 tal que $\epsilon < \alpha$, $(1 + \frac{\beta\epsilon}{d})^{N+1} < 2$, y $(1 - \frac{\beta\epsilon}{d})^{N+1} > 1/2$.
 - 4.1. Si $\epsilon \leq 2^{-n}$, pedirle la respuesta al Algoritmo 1.
5. Calcular el menor paso $M \leq N + 1$ tal que

$$\frac{2^n \epsilon}{D} < \prod_{i=0}^{M-1} |P'(P^i(x))| < 2^n \epsilon.$$

Si no existe tal M , devolver 0.

6. Aplicar el Algoritmo 1 para comprobar si $d(J(P), P^i(x)) < \epsilon/16$ o $d(J(P), P^i(x)) > \epsilon/2$ para cada $0 \leq i \leq M$; para hacer esto es suficiente calcular $P^i(x)$ con precisión $\epsilon/16$.
7. Devolver 0 si el Algoritmo 1 devuelve $d(J(P), P^i(x)) > \epsilon/2$ en alguno de los casos, y 1 en caso contrario.

Observamos que la salida que nos proporciona el Algoritmo 2 no satisface de manera directa la definición estricta de localmente computable (Definición 3.2.3). Dicha definición exige que el algoritmo decida su respuesta estableciendo una proporción de exactamente 2 entre el radio de rechazo y el de aceptación. Sin embargo, la proporción entre los radios de control del Algoritmo 2 es de:

$$\frac{\text{Radio de Rechazo}}{\text{Radio de Aceptación}} = \frac{2^{-n}D}{\frac{2^{-n}}{32}} = 32D.$$

Esto significa que el Algoritmo 2 presenta una “zona gris” de incertidumbre que es un factor de $32D$ veces mayor que la permitida por la definición estándar.

Para arreglarlo, en lugar de evaluar el Algoritmo 2 únicamente en el punto de entrada x , consideramos la bola completa de incertidumbre $B(x, 2^{-n})$ y la cubrimos por completo mediante una unión finita de bolas más pequeñas de radio $\frac{2^{-n}}{32D}$. Los centros de estas pequeñas bolas forman una cuadrícula de puntos distribuidos de manera uniforme sobre la región de búsqueda original.

El procedimiento consistirá en ejecutar el Algoritmo 2 de forma sucesiva utilizando como entrada cada uno de los puntos que forman nuestra cuadrícula. Si $J(P)$ interseca $B(x, 2^{-n})$, deberá intersectar forzosamente al menos una de las bolas pequeñas. En consecuencia, con que una sola de las ejecuciones del Algoritmo 2 devuelva 1, concluiremos con certeza que el punto se encuentra en el conjunto.

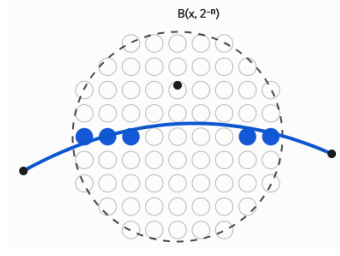


Figura 4.1: Algoritmo 2

El número total de puntos necesarios para cubrir la bola original está acotado por un factor proporcional a $(32D)^2$. Esta cota es independiente de la precisión n solicitada, por lo tanto, multiplicar el coste de un proceso por una constante preserva intacta su clase de complejidad.

Proposición 4.2.11. [Bra05, Lema 4.5.2] Si $d(J(P), x) > 2^{-n}D$, entonces el Algoritmo 2 no termina en la línea 3.1.

Demostración. Como $D \geq 1$, la demostración de la Proposición 4.2.8 nos sirve. $\square$

Proposición 4.2.12. [Bra05, Lema 4.5.3] El ϵ descrito en la línea 4 existe.

Proposición 4.2.13. [Bra05, Lema 4.5.4] Si $d(J(P), x) < 2^{-n}/32$ y el algoritmo alcanza la línea 5, entonces existe un M como el descrito en la línea 5 (y el algoritmo lo encuentra).

Proposición 4.2.14. [Bra05, Lema 4.5.5] Si $d(J(P), x) < 2^{-n}/32$ y el algoritmo alcanza la línea 6, entonces devuelve 1.

Demostración. Para que el Algoritmo 2 devuelva 1, el Algoritmo 1 debe devolver 1 cuando analice $P^i(x)$ para todo $0 \leq i \leq M$. Lo demostraremos por inducción:

- $i = 0$: Por hipótesis, sabemos que la distancia inicial es $d(J(P), x) < 2^{-n}/32$. Por la condición de la línea 4.1, sabemos que $2^{-n} < \epsilon$. Entonces:

$$d(J(P), x) < \epsilon/32 < \epsilon/16.$$

Es decir, el Algoritmo 1 devolverá 1.

- Supongamos que la cota se cumple para todos los pasos anteriores hasta $i - 1 < M$. Queremos demostrar que se mantiene para el paso i .

Por la elección de M en la línea 5, sabemos que $\prod_{j=0}^{i-1} |P'(P^j(x))| < 2^n \epsilon$.

Por la Proposición 4.2.10 obtenemos:

$$d(J(P), P^i(x)) \leq \left(1 + \frac{\beta\epsilon}{d}\right)^i \left(\prod_{j=0}^{i-1} |P'(P^j(x))|\right) d(J(P), x).$$

Sustituyendo las cotas que conocemos:

$$d(J(P), P^i(x)) < 2 \cdot (2^n \epsilon) \cdot \left(\frac{2^{-n}}{32}\right) = \frac{2 \cdot \epsilon}{32} = \frac{\epsilon}{16}.$$

Por lo tanto, nuestro Algoritmo 1 devolverá 1 como queríamos ver. $\square$

Proposición 4.2.15. [Bra05, Lema 4.5.6] Si $d(J(P), x) > 2^{-n}D$ y el algoritmo alcanza la línea 6, entonces devuelve 0.

Demostración. Supongamos, por el contrario, que el algoritmo devuelve 1. Esto significa que $d(J(P), P^i(x)) \leq \epsilon/2 < \epsilon$ para todo $i = 0, 1, \dots, M$. Por lo tanto, usando la Proposición 4.2.10 tendríamos que:

$$d(J(P), P^M(x)) \geq (1 - \frac{\beta\epsilon}{d})^M (\prod_{j=0}^{M-1} |P'(P^j(x))|) d(J(P), x) > \frac{1}{2} (2^n \epsilon / D) 2^{-n} D = \epsilon/2.$$

Lo cual es una contradicción. $\square$

Definición 4.2.16 (Complejidad de la multiplicación). Denotaremos por $M(n)$ a la complejidad temporal requerida por una máquina de Turing para multiplicar dos números enteros de n bits. Utilizando algoritmos de multiplicación rápida, como el algoritmo de Schönhage-Strassen, asumiremos que esta cota se sitúa en $M(n) = \mathcal{O}(n \log n \log \log n)$. Para propósitos de este análisis, es suficiente saber que $M(n)$ es una función estrictamente polinómica.

Proposición 4.2.17. [Bra05, Lema 4.5.7] La complejidad temporal del Algoritmo 2 es $\mathcal{O}(nM(n) + nS_1(\log n + \mathcal{O}(1)))$.

Demostración. Analizamos la complejidad del algoritmo línea por línea.

- Línea 1: Requiere tiempo constante $\mathcal{O}(1)$.
- Línea 2: Queremos encontrar el menor entero N que verifique:

$$c^{N-2-q} \geq M_W \cdot R' \cdot 2^{n-1}.$$

El término $M_W \cdot R'$ es una constante precalculada que ocupa un espacio fijo en memoria. El término 2^{n-1} tiene exactamente n bits. Al multiplicarlos en binario (que es un simple desplazamiento de bits), obtenemos un número diádico cuya longitud total crece de forma estrictamente lineal respecto a n , es decir, ocupa un espacio de $\mathcal{O}(n)$ bits en la cinta.

Para calcular N de forma eficiente, el algoritmo emplea el método de exponenciación por cuadratura, duplicando el exponente en cada paso iterativo (es decir, computando la secuencia $c^2, c^4, c^8, \dots$). Dado que $N = \mathcal{O}(n)$, solo se requieran, a lo sumo, $\mathcal{O}(\log n)$ iteraciones para alcanzar o superar la cota deseada. En cada iteración, la máquina calculará una multiplicación. Puesto que la cota superior del número a calcular tendrá una longitud máxima acotada por $\mathcal{O}(n)$ bits. Por consiguiente, el tiempo total invertido para calcular estas potencias y determinar el parámetro N es el producto del número de iteraciones por el coste de la multiplicación, $\mathcal{O}(M(n) \log n)$.

- Línea 3: Requiere aplicar el polinomio N veces consecutiva, y sabemos que $N = \mathcal{O}(n)$. En cada una de estas iteraciones, la máquina debe realizar los cálculos con precisión de $\mathcal{O}(n)$ bits; por lo tanto, la complejidad total de esta línea es $\mathcal{O}(n \cdot M(n))$.
- Línea 4: En este paso, la máquina debe elegir un valor ϵ (potencia de 2) que satisfaga unas cotas. Sabemos por cálculo que $(1+x)^N \approx e^{xN}$ para valores de x suficientemente pequeños. Para cumplir con las cotas, el exponente debe mantenerse acotado. Dado que en la Línea 2 ya hemos establecido que $N = \mathcal{O}(n)$, la única forma de que el producto $N \cdot \epsilon$ se mantenga como una constante es que ϵ sea inversamente proporcional a n . Es decir, $\epsilon = 1/\mathcal{O}(n)$.

ϵ tiene que ser de la forma 2^{-k} . Puesto que $\epsilon \approx 1/n$, el número de bits decimales necesarios es $k \approx \log_2(n)$. La máquina de Turing determina el valor de k evaluando la longitud binaria del número N (el cual ya está escrito en la cinta). Anotar esta potencia de 2 requiere escribir $\mathcal{O}(\log n)$ ceros seguidos de un uno en la cinta. El desplazamiento del cabezal para leer N y escribir este nuevo valor está acotado por un tiempo $\mathcal{O}(n)$.

Línea 4.1: Para garantizar un error máximo de ϵ , la precisión binaria requerida es $P = \log_2(1/\epsilon)$. Como se ha establecido en el paso anterior, $\epsilon = \frac{1}{C \cdot n}$ para cierta constante C derivada de los parámetros del polinomio. Al sustituir esta expresión, obtenemos: $P = \log_2(C \cdot n) = \log_2(n) + \log_2(C)$

En consecuencia, el tiempo de ejecución del Algoritmo 1 queda acotado por $\mathcal{O}(S_1(\log n + \mathcal{O}(1)))$.

- Línea 5: Como buscamos un entero M tal que $M \leq N+1$ y $N = \mathcal{O}(n)$, el bucle para encontrarlo se ejecutará como máximo $\mathcal{O}(n)$ veces. En cada iteración se calculará la derivada y se multiplicarán los resultados, lo que toma un tiempo $M(n)$. Esta línea requiere tiempo $\mathcal{O}(n \cdot M(n))$.
- Línea 6: Primero, calculará la órbita $P^i(x)$ para cada paso $i \leq M$. Como $M = \mathcal{O}(n)$ y cada iteración necesita tiempo $M(n)$, necesitamos un tiempo de $\mathcal{O}(n \cdot M(n))$. Luego ejecutamos $\mathcal{O}(n)$ veces el Algoritmo 1 con parámetro $\log(1/\mathcal{O}(\epsilon))$, por lo que se necesita un tiempo de $\mathcal{O}(n \cdot S_1(\log n + \mathcal{O}(1)))$. La línea 6 requiere un tiempo de $\mathcal{O}(n \cdot M(n) + n \cdot S_1(\log n + \mathcal{O}(1)))$.
- Línea 7: Tenemos las salidas del Algoritmo 1, saber si alguna ha devuelto 0 requiere tiempo $\mathcal{O}(n)$.

Al sumar todas las complejidades, vemos que la complejidad temporal del algoritmo está dominada por la complejidad temporal de la línea 6, la cual es $\mathcal{O}(n \cdot M(n) + n \cdot S_1(\log n + \mathcal{O}(1)))$. A partir de la discusión que sigue al Algoritmo 1, sabemos que $S_1(n) = 2^{\mathcal{O}(n)}$, por consiguiente $S_2(n) = \mathcal{O}(n \cdot M(n) + n \cdot 2^{\mathcal{O}(\log n)}) = \text{poly}(n)$ $\square$

4.2.3. Algoritmo 3 [Bra05, Sección 4.5.4]

Proceso: Se ejecuta exactamente igual que el Algoritmo 2, excepto que utiliza el Algoritmo 2 en lugar del Algoritmo 1 en las líneas 4.1, 6 y 7.

Ahora, será necesario justificar formalmente por qué podemos sustituir el Algoritmo 1 por el Algoritmo 2 en el diseño del Algoritmo 3, sin invalidar las demostraciones geométricas previas.

En el Algoritmo 2, las líneas 4.1, 6 y 7 requieren evaluar la distancia de un punto a $J(P)$. La respuesta de estas líneas no depende de la estructura interna del Algoritmo 1, sino de que las respuestas que proporcione satisfagan las cotas de error de la Ko-computabilidad (Definición 3.2.5) para una precisión exigida k .

El Algoritmo 2 en su forma base no es un sustituto válido, ya que su margen de incertidumbre es de un factor $32D$.

Sin embargo, el Algoritmo 3 invoca al Algoritmo 2 en su forma completa estructurado mediante el recubrimiento por cuadrícula. Como demostramos previamente, esta aplicación afina la proporción de los radios, garantizando la definición de conjunto localmente computable (Definición 3.2.3).

Llegados a este punto, recordando el Teorema 3.2.9, el cual establece la equivalencia entre la computabilidad local y la Ko-computabilidad, podemos afirmar que nuestro bloque “Algoritmo 2 + Cuadrícula” Ko computa el conjunto.

En conclusión, al sustituir la subrutina exponencial por el bloque polinómico de la cuadrícula, el Algoritmo 3 respeta las inecuaciones originales y hereda la correctitud del diseño.

Proposición 4.2.18. *La complejidad temporal del Algoritmo 3 es $\mathcal{O}(n \cdot M(n))$.*

Demostración. Al haber sustituido las llamadas exponenciales del Algoritmo 1 por el Algoritmo 2, la complejidad temporal del Algoritmo 3, pasa a ser:

$$\mathcal{O}(n \cdot M(n) + n \cdot S_2(\log n + \mathcal{O}(1)))$$

donde $S_2(k)$ representa la complejidad temporal de ejecutar el Algoritmo 2 para una precisión k .

Como demostramos en la Proposición 4.2.17, la complejidad del Algoritmo 2 es estrictamente polinómica respecto a su entrada, es decir, $S_2(k) = \text{poly}(k)$. Si sustituimos tenemos que:

$$S_3(n) = \mathcal{O}(n \cdot M(n) + n \cdot \text{poly}(\log n)).$$

Dado que $M(n)$, crece asintóticamente más rápido que cualquier polinomio en $\log n$, este último término queda absorbido. Por consiguiente, el tiempo de ejecución del Algoritmo 3 se reduce a:

$$S_3(n) = \mathcal{O}(n \cdot M(n)). \quad \square$$

Ya hemos visto que $M(n) \leq \mathcal{O}(n \log n \log \log n)$. Con esto, conseguimos acotar en tiempo polinómico todo nuestro sistema y demostrar el teorema fundamental de este trabajo de fin de grado:

Teorema 4.2.19 (Computabilidad de los Conjuntos de Julia Hiperbólicos, [Bra05, Teorema 4.5.8]). *Para cada polinomio hiperbólico fijo $P(z)$, su conjunto de Julia asociado $J(P)$ es computable en tiempo polinómico, con una complejidad temporal acotada por:*

$$\mathcal{O}(n \cdot M(n)) \leq \mathcal{O}(n^2 \log n \log \log n).$$

Este teorema corrobora de manera rigurosa que computar la estructura geométrica generada por un sistema dinámico holomorfo de comportamiento caótico, como es el caso de estos conjuntos de Julia hiperbólicos, no requiere un tiempo de ejecución exponencial, sino que puede resolverse mediante algoritmos eficientes en tiempo polinómico.

Anexo A

Superficies de Riemann y métrica hiperbólica

En este anexo introduciremos los conceptos fundamentales sobre superficies hiperbólicas, la métrica hiperbólica y las isometrías asociadas. El desarrollo y los resultados presentados a continuación están basados principalmente en el trabajo de Milnor [Mil06] y el de Beardon y Minda [BM06].

A.1. Superficies de Riemann y aplicaciones conformes

Definición A.1.1. *Una superficie de Riemann U es una variedad compleja de dimensión uno, la cual supondremos siempre conexa.*

Definición A.1.2. *Sean $U, V \subseteq \mathbb{C}$ dos dominios abiertos y conexos. Un isomorfismo conforme entre U y V es una función holomorfa y biyectiva $f : U \rightarrow V$, con inversa también holomorfa.*

Definición A.1.3. *Un automorfismo conforme de un dominio U será aquel isomorfismo conforme que vaya de U a sí mismo.*

A.1.1. Aplicaciones propias y mapas recubridores

Definición A.1.4. [Mil06, Apéndice E] *Una aplicación $p : S' \rightarrow S$ entre superficies de Riemann es una aplicación propia si la imagen inversa $p^{-1}(K)$ de cualquier subconjunto compacto $K \subset S$ es un subconjunto compacto de S' .*

Observación A.1.5. [Mil06, Apéndice E] Toda aplicación propia entre superficies de Riemann es “finita a uno” (la preimagen de cada punto es un conjunto finito de puntos) y posee un grado finito bien definido $d \geq 1$, contando multiplicidades. En este caso, nos referimos a ella como un recubrimiento ramificado de d hojas.

Definición A.1.6. [Mil06, Apéndice E] *Una aplicación holomorfa $p : S' \rightarrow S$ entre superficies de Riemann se denomina mapa recubridor (o recubrimiento) si cada punto de S posee un entorno conexo U que está uniformemente recubierto, en el sentido de que cada componente conexa de $p^{-1}(U) \subset S'$ se aplica sobre U mediante un isomorfismo conforme.*

Lema A.1.7. Sea $p : S' \rightarrow S$ una aplicación propia entre superficies de Riemann. Si p no tiene puntos críticos (es decir, es un isomorfismo local en todo su dominio), entonces p es un mapa recubridor.

A.2. Métricas conformes e isometrías

Definición A.2.1. Sea $U \subseteq \mathbb{C}$ un dominio abierto. Una métrica conforme sobre U viene dada por una aplicación $\lambda : U \rightarrow \mathbb{R}^+$, llamada densidad, continua y estrictamente positiva, que asigna a cada vector del espacio tangente una norma dada por:

$$\|v\|_\lambda = \lambda(z)|v|.$$

Es decir, su elemento diferencial de longitud viene dado por:

$$ds = \lambda(z)|dz|.$$

A partir de ahora, denotaremos por $\|\cdot\|$ la norma inducida por la métrica hiperbólica. Para cualquier superficie de Riemann S y un punto $p \in S$, denotaremos por $T_p S$ a su espacio tangente en p , el cual es un espacio vectorial complejo de dimensión 1. Tal y como establecimos en nuestra definición inicial, esta norma nos indica geoméricamente cuánto mide un vector infinitesimal en dicho espacio tangente.

Lema A.2.2. Sea S una superficie de Riemann, $p \in S$ y f una función holomorfa. Dados dos vectores no nulos cualesquiera $v, w \in T_p S$, se cumple que:

$$\frac{\|Df_p(v)\|}{\|v\|} = \frac{\|Df_p(w)\|}{\|w\|}.$$

Es decir, el valor de este cociente depende únicamente del punto p y de la función f , pero no del vector tangente evaluado.

Demostración. Sabemos que el espacio tangente $T_p S$ es un espacio vectorial complejo de dimensión 1. Por lo tanto, si tenemos dos vectores no nulos $v, w \in T_p S$, uno es múltiplo del otro, es decir, $w = \alpha v$ para algún escalar $\alpha \in \mathbb{C} \setminus \{0\}$. Por la linealidad de la derivada (diferencial) Df_p , tenemos que $Df_p(w) = Df_p(\alpha v) = \alpha Df_p(v)$. Entonces:

$$\frac{\|Df_p(w)\|}{\|w\|} = \frac{\|\alpha Df_p(v)\|}{|\alpha| \cdot \|v\|} = \frac{|\alpha| \cdot \|Df_p(v)\|}{|\alpha| \cdot \|v\|} = \frac{\|Df_p(v)\|}{\|v\|}.$$

Con esto, podemos referirnos a la norma de la derivada sin importar el vector tangente escogido. $\square$

Definición A.2.3. Sean (U, d_U) y (V, d_V) dos superficies con sus respectivas distancias. Una aplicación $f : U \rightarrow V$ es una isometría (global) si para todo par de puntos $z, w \in U$ se cumple que:

$$d_V(f(z), f(w)) = d_U(z, w).$$

En el contexto de las métricas conformes definidas por densidades, la aplicación f será una isometría local si preserva el elemento diferencial de longitud:

$$\lambda_V(f(z))|f'(z)| = \lambda_U(z).$$

A.3. La métrica hiperbólica en el disco unidad

Trabajaremos sobre el disco unidad abierto $\mathbb{D} = \{z \in \mathbb{C} \mid |z| < 1\}$. Definiremos el disco hiperbólico como el disco unidad $\mathbb{D}$ dotado de una métrica específica, la métrica hiperbólica o métrica de Poincaré. En la geometría euclídea, el elemento diferencial de longitud es constante, representado como $ds = |dz|$. Intuitivamente significa que la “regla” con la que medimos distancias tendrá el mismo tamaño estés en el punto en el que estés del disco. Sin embargo, para poder estudiar el comportamiento de los dominios del plano complejo nos interesa tener una forma de medir con la que consigamos que el borde del dominio actúe como una frontera inalcanzable.

Para lograrlo, aplicaremos el concepto de métrica conforme visto en la sección anterior al caso específico del disco unidad. Dotaremos a $\mathbb{D}$ de una función de densidad particular que cambie la escala a medida que nos acercamos al borde.

Definición A.3.1. *Definimos la métrica hiperbólica (o métrica de Poincaré) en el disco unidad $\mathbb{D}$ mediante la densidad $\lambda_{\mathbb{D}}(z)$, cuya expresión diferencial es:*

$$\lambda_{\mathbb{D}}(z)|dz| = \frac{2|dz|}{1 - |z|^2}.$$

Intuitivamente, a medida que te vas acercando al borde del disco unidad $\mathbb{D}$ ($|z| \rightarrow 1$), la regla se va encogiéndose lo que provoca que la frontera esté infinitamente lejos.

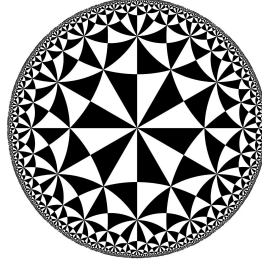


Figura A.1: Representación del disco de Poincaré (obra Límite Circular IV de M.C. Escher). [Esc24].

A partir de la métrica construida, podemos definir la longitud hiperbólica de una curva suave γ en $\mathbb{D}$ calculando la integral de su norma:

$$L_{\mathbb{D}}(\gamma) := \int \|\gamma'(t)\| dt = \int \lambda_{\mathbb{D}}(\gamma(t)) |\gamma'(t)| dt.$$

Y la distancia hiperbólica entre dos puntos $z, w \in \mathbb{D}$ como el ínfimo de las longitudes de todas las curvas suaves que los unen:

$$d_{\mathbb{D}}(z, w) = \inf_{\gamma} L_{\mathbb{D}}(\gamma).$$

Proposición A.3.2 ([BM06, Teorema 2.1]). *Sea $\mathbb{D}$ el disco unidad dotado de la métrica hiperbólica. El grupo de isometrías de $(\mathbb{D}, d_{\mathbb{D}})$ que preservan la orientación coincide exactamente con el grupo de automorfismos conformes del disco, $\text{Aut}(\mathbb{D})$.*

A.4. Uniformización y superficies hiperbólicas

Hasta ahora hemos definido la métrica hiperbólica en el disco unidad $\mathbb{D}$, pero las componentes conexas de los dominios que estudiaremos no tienen por qué ser topológicamente el disco. Para poder usar la métrica hiperbólica en nuestro estudio, debemos definir algunos conceptos previos.

Teorema A.4.1 (Teorema de Uniformización, [Mil06, Teorema 1.1]). *Cualquier superficie de Riemann simplemente conexa es isomorfa a una de las tres superficies canónicas:*

1. $\widehat{\mathbb{C}} = \mathbb{C} \cup \{\infty\}$, la esfera de Riemann.
2. $\mathbb{C}$, el plano complejo.
3. $\mathbb{D}$, el disco unidad abierto.

Teorema A.4.2 (Teorema de la aplicación de Riemann). *Todo dominio abierto $U \subset \mathbb{C}$ simplemente conexo, $U \neq \mathbb{C}$, es conformemente equivalente al disco unidad abierto $\mathbb{D}$.*

Teorema A.4.3 (Uniformización de cualquier superficie de Riemann, [Mil06, Teorema 2.1]). *Para toda superficie de Riemann S , existe una superficie simplemente conexa $\tilde{S}$ y una aplicación exhaustiva $\pi : \tilde{S} \rightarrow S$ que es un isomorfismo local. A esta superficie $\tilde{S}$ le llamaremos recubridor universal de S , que solo puede ser isomorfo a $\widehat{\mathbb{C}}$, $\mathbb{C}$ o $\mathbb{D}$.*

Definición A.4.4. *Una superficie hiperbólica será aquella superficie de Riemann tal que su recubridor universal es isomorfo al disco unidad $\mathbb{D}$.*

Teorema A.4.5. [Mil06, Cap 2] *Cualquier subconjunto abierto de $\widehat{\mathbb{C}}$ al que le falten tres o más puntos será una superficie hiperbólica.*

Definición A.4.6. *Sea $\pi : \tilde{S} \rightarrow S$ un recubrimiento entre dos superficies de Riemann. Un automorfismo del recubrimiento es una función $\gamma : \tilde{S} \rightarrow \tilde{S}$ holomorfa y biyectiva que cumple $\pi \circ \gamma = \pi$.*

Proposición A.4.7. *Sea $\pi : \tilde{S} \rightarrow S$ un recubrimiento universal. $\text{Aut}(\pi)$ actúa de forma transitiva.*

Con esta definición podemos trasladar nuestra métrica hiperbólica a la superficie hiperbólica S . Sea $\pi : \mathbb{D} \rightarrow S$ su recubridor universal, queremos definir una métrica λ_S tal que la proyección π sea una isometría local:

$$\lambda_S(p) = \frac{\lambda_{\mathbb{D}}(z)}{|\pi'(z)|}.$$

Para demostrar que la métrica λ_S está bien definida, debemos comprobar que el valor obtenido es independiente de la elección de la preimagen z .

Escogemos otro $w \in \mathbb{D}$ tal que $\pi(w) = p$. Como γ también es un automorfismo del disco sabemos que por la Proposición A.4.7 existe un automorfismo del recubrimiento γ tal que $\gamma(z) = w$. Sabemos por la Proposición A.3.2 que este automorfismo de $\mathbb{D}$ es una isometría de la métrica hiperbólica, es decir, $\lambda_{\mathbb{D}}(\gamma(z))|\gamma'(z)| = \lambda_{\mathbb{D}}(z)$.

Tenemos que $\pi(\gamma(z)) = \pi(z)$, derivando obtenemos $\pi'(\gamma(z))\gamma'(z) = \pi'(z)$. Ahora, usando lo que hemos visto y que $\gamma(z) = w$ tenemos que:

$$\frac{\lambda_{\mathbb{D}}(w)}{|\pi'(w)|} = \frac{\lambda_{\mathbb{D}}(\gamma(z))}{|\pi'(\gamma(z))|} = \frac{\lambda_{\mathbb{D}}(z)/|\gamma'(z)|}{|\pi'(z)|/|\gamma'(z)|} = \frac{\lambda_{\mathbb{D}}(z)}{|\pi'(z)|} = \lambda_S(p).$$

Con lo que acabamos de ver que $\lambda_S(p)$ no depende de la elección de la preimagen, por lo tanto la densidad $\lambda_S(p)$ es independiente de la preimagen escogida y la métrica está bien definida en toda la superficie.

A.5. Lema de Schwarz y Teorema de Schwarz-Pick

Proposición A.5.1 (Lema de Schwarz). *Sea $f : \mathbb{D} \rightarrow \mathbb{D}$ una función holomorfa tal que $f(0) = 0$. Entonces:*

1. $|f(z)| \leq |z|$ para todo $z \in \mathbb{D}$.
2. $|f'(0)| \leq 1$.

Además, si $|f(z)| = |z|$ para algún $z \neq 0$ o $|f'(0)| = 1$, entonces la función es una rotación.

Teorema A.5.2 (Teorema de Schwarz-Pick). *Sean X e Y superficies de Riemann hiperbólicas dotadas de sus respectivas métricas hiperbólicas (λ_X, λ_Y) y distancias inducidas (d_X, d_Y) . Toda función holomorfa $f : X \rightarrow Y$ es una contracción débil, es decir:*

1. $\lambda_Y(f(z))|f'(z)| \leq \lambda_X(z) \quad \forall z \in X$.
2. $d_Y(f(z), f(w)) \leq d_X(z, w) \quad \forall z, w \in X$.

Además, se da la igualdad en (1) para algún punto z (o en (2) para un par de puntos distintos) si y solo si f es un isomorfismo local. En caso de que f no sea un isomorfismo local, la función es una contracción estricta.

Corolario A.5.3. [Bra05, Lema 4.4.3] *Sean U y V dos superficies de Riemann hiperbólicas tales que $V \subset U$ es un subconjunto abierto y $V \neq U$. Entonces, para todo punto $z \in V$, las densidades de sus respectivas métricas hiperbólicas cumplen la desigualdad estricta:*

$$\lambda_U(z) < \lambda_V(z).$$

Demostración. Consideremos la aplicación inclusión $i : V \hookrightarrow U$, definida por $i(z) = z$ para todo $z \in V$. Al ser V un subconjunto abierto de U , la aplicación i es holomorfa e inyectiva. Además, dado que $V \neq U$, i no es exhaustiva, lo que impide que sea un isomorfismo conforme global (isometría) entre ambas superficies. Aplicando el Teorema A.5.2, sabemos que esta debe actuar como una contracción estricta. Por lo tanto, tenemos que:

$$\lambda_U(i(z)) \cdot |i'(z)| < \lambda_V(z).$$

Como la derivada de la inclusión es idénticamente la unidad ($i'(z) = 1$) y la imagen es el propio punto ($i(z) = z$), al sustituir estos valores en la desigualdad obtenemos directamente:

$$\lambda_U(z) < \lambda_V(z).$$

para todo $z \in V$, lo que completa la demostración. □

Anexo B

Relación entre la Ko-computabilidad y las clases P y NP

Este anexo contiene el desarrollo formal y las demostraciones omitidas en el Capítulo 3 relativas a las clases de complejidad y su equivalencia con la Ko-computabilidad.

Para estudiar si $P = NP$, es necesario clasificar los elementos más difíciles de la clase NP.

Definición B.0.1 (Problemas NP-duros y NP-completos). *Un problema de decisión X se dice que es NP-duro (NP-hard) si cualquier problema perteneciente a la clase NP se puede reducir a X mediante una reducción en tiempo polinómico. Si, además, el propio problema X pertenece a la clase NP, decimos que X es un problema NP-completo.*

De la definición anterior se desprende que los problemas NP-completos son los problemas más difíciles de NP, en el sentido de que si se encontrase un algoritmo en tiempo polinómico para resolver uno solo de ellos, se podrían resolver todos los problemas de NP eficientemente, lo que implicaría que $P = NP$.

El primer problema que fue identificado como NP-completo por el Teorema de Cook-Levin [Coo71] fue el problema SAT:

Definición B.0.2 (Problema SAT). *El problema de satisfacibilidad booleana, denotado como SAT, consiste en determinar si, dada una fórmula lógica booleana compuesta por variables y operadores lógicos, existe alguna asignación de valores de verdad para dichas variables que haga que la fórmula entera se evalúe como cierta.*

Lema B.0.3. [Bra05, Lema 3.2.3] *Sea $M^\phi(n)$ una máquina de Turing con oráculo que se ejecuta en un tiempo máximo acotado por $p(n)$. Si $x \in \mathcal{D}_{p(n)}$ es un número diádico y $y \in \mathbb{R}$ es un número real tal que $|y - x| \leq 2^{-p(n)-1}$, entonces existe un oráculo ψ que representa a y y un oráculo ϕ' que representa a x tales que la máquina devuelve exactamente el mismo resultado para ambos: $M^{\phi'}(n) = M^\psi(n)$.*

Demostración. Como sabemos que la máquina $M^\phi(n)$ se ejecuta en un tiempo máximo $p(n)$, la capacidad de la máquina para interactuar con el oráculo se ve fuertemente limitada. Tal y como vimos en la Observación 3.2.13 (del documento principal), la máquina solo podrá leer y procesar, en el peor de los casos, $p(n)$ bits de la cinta. Por lo tanto, la máxima precisión que la máquina puede solicitar al oráculo sin exceder su tiempo de ejecución es

$p(n)$, lo que equivale a recibir un número diádico con un error estrictamente menor que $2^{-p(n)}$.

Vamos a construir dos oráculos válidos, ψ (para y) y ϕ' (para x), que devuelvan exactamente la misma secuencia en sus primeros $p(n) + 1$ dígitos diádicos.

- Para representar y : al pedir una precisión r , exigimos que devuelva un valor tal que $|y - \psi(r)| < 2^{-r-1}$, es decir, nos ponemos más exigentes. Además, vamos a forzar que $\psi(r)$ esté en el intervalo abierto $(x - 2^{-p(n)-1}, x)$ si $y < x$, o en el intervalo cerrado-abierto $[x, x + 2^{-p(n)-1})$ si $y \geq x$. Sabemos que $\psi(r)$ caerá en alguno de esos intervalos porque, por hipótesis, el propio número y ya está en esa zona.
- Para representar x : Exigiremos que $|\phi'(r) - x| < 2^{-r}$ y forzaremos que $\phi'(r)$ esté en el intervalo $[x, x + 2^{-p(n)-1})$ si $\psi(r)$ está ahí, o en $(x - 2^{-p(n)-1}, x)$ si $\psi(r)$ está ahí.

Los dos son oráculos válidos ya que su error es estrictamente menor al exigido por la definición original. Además, bajo estas condiciones, $\psi(r)$ y $\phi'(r)$ coincidirán obligatoriamente en sus primeros $p(n) + 1$ dígitos diádicos. Esto se debe a que: dado que $x \in \mathcal{D}_{p(n)}$, su representación binaria exacta termina en el dígito $p(n)$. La distancia que delimita los intervalos construidos es exactamente $2^{-p(n)-1}$, lo que equivale a modificar el bit situado en la posición $p(n) + 1$. Al obligar a que ambas aproximaciones caigan estrictamente en el mismo lado de x (es decir, ambas en el intervalo por la derecha $[x, x + 2^{-p(n)-1})$ o ambas en el intervalo por la izquierda $(x - 2^{-p(n)-1}, x)$), forzamos a que compartan el mismo prefijo binario.

Como la máquina solo podrá leer los primeros $p(n)$ bits, la ejecución de cada oráculo será idéntica. En consecuencia, $M^{\phi'}(n) = M^{\psi}(n)$. $\square$

Teorema B.0.4. [Bra05, Teorema 3.2.2] Sea $C \subset \mathbb{R}^2$ un conjunto compacto. Si todo conjunto Ko P-computable es también computable en tiempo polinómico (localmente), entonces $P = NP$.

Demostración. Para esta demostración asumiremos que trabajamos en dimensión $k = 1$, ya que su generalización a la dimensión 2 se desarrolla de forma análoga. Nuestro objetivo será construir un conjunto C que sea Ko P-computable, tal que si existe un algoritmo que hace que C sea computable en tiempo polinómico, entonces dicho algoritmo podría usarse para resolver el problema *SAT* en tiempo polinómico (lo que implicaría $P = NP$).

Paso 1: Construcción del conjunto C

Vamos a coger el intervalo $[0, 1]$ y lo dividiremos en infinitos intervalos cerrados disjuntos K_i , definidos de la siguiente manera:

$$K_i = \left[\frac{1}{i+1}, \frac{1}{i} \right] \quad \text{para } i \geq 1.$$

Observemos que la longitud de cada intervalo es $\frac{1}{i} - \frac{1}{i+1} = \frac{1}{i(i+1)}$, lo cual es de orden $\mathcal{O}\left(\frac{1}{i^2}\right)$.

A continuación, imagina que disponemos de un diccionario infinito o codificación que asigna a cada número natural i , un texto generado. Bajo esta codificación, un número i puede representar:

- Una cadena de texto que corresponda a una fórmula lógica proposicional válida, como por ejemplo podría ser " $A \wedge B$ ".

- Una cadena de texto sin sentido sintáctico, es decir, “texto basura” (por ejemplo, “jhdjj”).

De esta manera, nuestro diccionario contendrá codificados absolutamente todos los posibles problemas de la clase *SAT*.

Hace falta destacar que, si un número i codifica una fórmula proposicional ϕ que contiene r variables, necesariamente se cumple que $r < \log_2 i$. Intuitivamente, esto se debe a que la longitud del texto de la fórmula (la cantidad de caracteres) siempre será igual o mayor que la cantidad de variables que contiene, y para traducir dicha fórmula a un número i , su representación binaria requerirá una gran cantidad de dígitos para poder albergar toda la información.

Ahora seguimos con la construcción de nuestro conjunto C . Para cada intervalo K_i , codificamos i .

- Si el número i corresponde a texto basura, incluimos el quinto central del intervalo K_i en C .
- Si el número i corresponde a una fórmula proposicional ϕ con r variables, subdividimos el quinto central de K_i en 2^r subintervalos de igual longitud. Cada subintervalo corresponde a una asignación de verdad para las variables de ϕ (combinación de las r variables). Incluimos un subintervalo en C si y solo si la asignación de verdad correspondiente evalúa la fórmula como verdadera ($\phi = \text{TRUE}$).

Finalmente, incluimos el punto 0 en el conjunto C para garantizar que sea cerrado y compacto, ya que es el único punto de acumulación de los intervalos K_i . Como los extremos de los intervalos $K_i = [\frac{1}{i+1}, \frac{1}{i}]$ tienden a 0 cuando $i \rightarrow \infty$, cualquier sucesión infinita de puntos tomada a lo largo de estos intervalos convergerá inevitablemente a 0. Dado que, por definición topológica, un conjunto debe contener a todos sus puntos de acumulación para ser cerrado, la inclusión del 0 es estrictamente necesaria.

Paso 2: C es Ko P-computable

Dada una entrada n y un oráculo ϕ que representa un número real x :

- Si la máquina detecta que $x < 2^{-n}$, devuelve 1. Esto es correcto ya que $0 \in C$ y no se necesita una precisión absoluta en la frontera.
- Si $x \geq 2^{-n-1}$, entonces x pertenece a algún intervalo K_i con $i < 2^{3n}$. Esto es debido a la construcción de los intervalos, $K_i = [\frac{1}{i+1}, \frac{1}{i}]$, si sabemos que $x \in K_i$ entonces su valor máximo posible es $\frac{1}{i}$, por lo tanto:

$$2^{-(n+1)} \leq x \leq \frac{1}{i}$$

$$i \leq 2^{n+1} < 2^{3n}.$$

Para poder saber en que subintervalo cae x , necesitamos pedirle al oráculo suficiente precisión (decimales en binario). Si el subintervalo más pequeño posible mide 2^{-k} , necesitaremos leer k bits del oráculo para poder ver en que subintervalo cae.

Sabiendo que $i < 2^{3n}$ y que la longitud de un intervalo K_i es $\frac{1}{i} - \frac{1}{i+1} = \frac{1}{i(i+1)} \approx \frac{1}{i^2}$, si tomamos el peor caso (cuando i se aproxima a 2^{3n}) tenemos que:

$$\text{Longitud } K_i \approx \frac{1}{(2^{3n})^2} = \frac{1}{2^{6n}} = 2^{-6n}.$$

La longitud del quinto central de ese intervalo será, como mínimo $\frac{1}{5}2^{-6n}$. Este, lo hemos dividido en 2^r subintervalos y sabemos que $r < \log_2 i$. Entonces:

$$r < \log_2(2^{3n}) = 3n.$$

Por lo tanto, vamos a dividir el quinto central en, a lo sumo, 2^{3n} subintervalos. El tamaño del subintervalo final será:

$$\text{Tamaño subintervalo} \approx \frac{\frac{1}{5}2^{-6n}}{2^{3n}} = \frac{1}{5}2^{-6n} \cdot 2^{-3n} = \frac{1}{5}2^{-9n}.$$

Con esto hemos visto que, en el peor de los casos, el tamaño del subintervalo lógico más pequeño que nos vamos a encontrar es del orden de 2^{-9n} . Para poder saber si x cae dentro o fuera, necesitamos pedirle al oráculo una precisión de aproximadamente $9n$ bits. Por lo tanto, consultando $\mathcal{O}(n)$ bits del oráculo, podemos determinar con suficiente precisión los dos subintervalos adyacentes de K_i donde podría estar x .

Como consultar una cantidad lineal de bits requiere un tiempo lineal, el proceso completo de ubicar el punto se mantiene dentro del tiempo polinómico.

- A partir de aquí, la máquina decodifica el índice i , obtiene la fórmula ϕ y sustituye los valores de verdad asociados al subintervalos correspondiente para poder saber si x está en C o no. Todo este proceso de decodificación y evaluación booleana requiere únicamente tiempo polinómico respecto a n .

Como todo se hace en tiempo polinómico, si vemos que lo que hace es Ko computar tendremos que C es Ko P-computable.

Para verificar que el conjunto C es Ko-computable, debemos comprobar que la máquina de Turing con oráculo $M^\phi(n)$ descrita anteriormente satisface las condiciones de la definición de Ko computabilidad:

- Si $x \in C$: Por construcción, x debe ser o bien el punto 0 o bien pertenecer a un subintervalo de algún K_i donde la fórmula ϕ_i se evalúa como verdadera. En el primer caso ($x = 0$), la máquina detecta $x < 2^{-n}$ y devuelve 1. En el segundo, la máquina identifica el índice i , evalúa la fórmula para la asignación correspondiente a la posición de x y devuelve 1. En ambos casos, $M^\phi(n) = 1$.
- Si $B(x, 2^{-n}) \cap C = \emptyset$: Esto implica que x está a una distancia mayor a 2^{-n} de cualquier punto de C . Por tanto, x no puede estar cerca de 0 (ya que $0 \in C$) ni puede estar dentro de un subintervalo “verdadero” de ningún K_i . Al ejecutar el algoritmo, la máquina determinará que la región que contiene a x no pertenece a C y devolverá 0.

Con esto hemos visto que C es Ko P-computable.

Paso 3: Si C es computable en tiempo polinómico, entonces $P = NP$

Asumimos que C es computable en tiempo polinómico, es decir, que existe una función local $f(d, l)$ que se ejecuta en tiempo polinómico.

Ahora cojamos cualquier fórmula proposicional ϕ del problema *SAT* de longitud $n = |\phi|$. Consultamos en nuestro diccionario para conseguir la codificación i de esta fórmula. La longitud de i es de $\mathcal{O}(n)$ bits, lo que significa que el valor numérico de i es a lo sumo $2^{\mathcal{O}(n)}$ (si un número tiene x bits, su mayor valor numérico es $2^x - 1$, osea $i \leq 2^x - 1 < 2^x$).

La longitud del intervalo correspondiente K_i es proporcional a $\frac{1}{i^2} \sim 2^{-\mathcal{O}(n)}$.

Escojamos un punto d exactamente en el centro del quinto central de K_i y una precisión $l = \mathcal{O}(n)$, tal que se cumpla que:

- $B(d, 2^{-l})$ cubra exactamente el quinto central de K_i .
- $B(d, 2 \cdot 2^{-l})$ esté contenida estrictamente dentro de K_i (sin invadir otros intervalos).

Ahora, si consultamos $f(d, l)$, la función devolverá 1 si y solo si la fórmula ϕ es satisfacible. Por como hemos construido nuestro conjunto C , $B(d, 2^{-l})$ intersecará C si existe alguna combinación de variables que satisface la fórmula proposicional.

Como por hipótesis, nuestra función $f(d, l)$ se ejecuta en tiempo polinómico sobre l , es decir, se ejecuta en tiempo polinómico respecto a la longitud original de la fórmula n . Por lo tanto, tendríamos un algoritmo capaz de resolver cualquier instancia *SAT* en tiempo polinómico.

En consecuencia, deducimos que $\text{SAT} \in P$. En este punto usaremos un resultado fundamental, el *Teorema de Cook-Levin* [Coo71] el cual establece rigurosamente que el problema *SAT* es NP-completo. Al ser *SAT* un problema NP-completo, se garantiza que es NP-duro y que cualquier otro problema en NP puede reducirse polinómicamente a él. Por consiguiente, el hecho de haber hallado un algoritmo eficiente para *SAT* implica que toda la clase NP puede resolverse en tiempo polinómico, lo que nos permite concluir formalmente que $P = NP$. $\square$

Teorema B.0.5. [Bra05, Teorema 3.2.2] Si $P = NP$, entonces todo conjunto Ko P-computable es localmente computable en tiempo polinómico.

Demostración. Asumimos que $P = NP$ y que nuestro conjunto C es Ko P-computable por una máquina de Turing con oráculo $M^\phi(n)$ que se ejecuta en un tiempo polinómico $p(n)$. Queremos demostrar que C es localmente computable en tiempo polinómico, pero nosotros ya sabemos que es localmente computable. Por lo tanto, nos queda ver que podemos calcular el valor de la función local $f(d, l)$ en tiempo polinómico respecto a l y a la longitud de $n = |d|$.

Como sabemos que la máquina $M^\phi(n)$ se ejecuta en un tiempo polinómico $p(n)$, para cualquier número real y , la máquina solo tiene tiempo para leer $p(n)$ bits del oráculo. Por lo tanto, no puede leer los infinitos decimales del número real y . Por el Lema B.0.3 sabemos que, si tomamos un número diádico $x \in \mathcal{D}_{p(n)}$ tal que $|y - x| \leq 2^{-p(n)-1}$, existen oráculos para ambos que fuerzan a la máquina a devolver exactamente el mismo resultado.

Nuestro objetivo es ver que $f(d, l)$ pertenece a la clase *NP*, por lo tanto, necesitamos construir un algoritmo que no dependa de un oráculo. Para conseguirlo, en vez de consultar a un oráculo sobre un número real y desconocido, propondremos como candidatos distintos números diádicos $x \in \mathcal{D}_{p(n)}$. Como conoceremos todos los bits de x , podremos dejar de usar el oráculo y simular nosotros mismos el comportamiento de la máquina.

El problema que surge ahora es que nos gustaría que la máquina pudiese trabajar números reales $y \in C$ que se encuentren cerca de x . Como vimos en el Lema B.0.3,

podemos dividir el entorno de x en: los puntos que se acercan por la derecha y los que se acercan por la izquierda. Para asegurarnos de cubrir cualquier posible punto real y cercano a x , definimos dos simulaciones extremas para nuestra máquina original:

- $M^+(x, n)$: Simula la máquina asumiendo que el número se rellena con infinitos ceros por la derecha. Esto representa el caso en el que el punto real y cae en el intervalo $[x, x + 2^{-p(n)-1})$.
- $M^-(x, n)$: Simula la máquina asumiendo que, tras los bits de x , el número se rellena con infinitos unos. Esto representa el caso en el que el punto real y cae en el intervalo $(x - 2^{-p(n)-1}, x)$.

Como las dos simulaciones consisten en leer cadenas de texto prefijadas, se ejecutan en tiempo polinómico.

Con estas dos simulaciones, cubrimos el entorno de x . Además, observamos que:

- Si existe algún $y \in C \cap [x - 2^{-p(n)-1}, x + 2^{-p(n)-1}]$, entonces alguna de las dos simulaciones lo detectará, por lo que $M^+(x, n) = 1 \vee M^-(x, n) = 1$.
- Si $C \cap [x - 2^{-p(n)-1}, x + 2^{-p(n)-1}] = \emptyset$, tendremos que las simulaciones no detectarán nada y $M^+(x, n) = 0 \wedge M^-(x, n) = 0$.

A continuación, construimos nuestra función local, dada una entrada (d, l) , elegimos una precisión n tal que $p(n) > l + 2$ y:

$$f(d, l) = \begin{cases} 1 & \text{si } \exists x \in \mathcal{D}_{p(n)} \cap \overline{B(d, 2^{-l} + 2^{-p(n)})} \text{ tal que } (M^+(x, n) = 1) \vee (M^-(x, n) = 1) \\ 0 & \text{en caso contrario} \end{cases} \quad (\text{B.0.1})$$

Como verificar si un candidato específico x cumple la condición geométrica y hace que nuestras simulaciones devuelvan 1 requiere únicamente tiempo polinómico (ya que M^+ y M^- son polinómicas), el problema de decidir si existe tal x pertenece, por definición, a la clase NP.

Calcular la función $f(d, l)$ equivale exactamente a resolver este problema de decisión. Por nuestra hipótesis, asumimos que $P = NP$, lo que implica que todo problema en NP admite un algoritmo que lo resuelve en tiempo polinómico. Por lo tanto, podemos afirmar que existe un algoritmo capaz de calcular $f(d, l)$ en tiempo polinómico sin necesidad de explorar todos los posibles valores de x .

Ahora solo nos queda ver que $f(d, l)$ satisface las condiciones de la función local.

- Caso $\overline{B(d, 2^{-l})} \cap C \neq \emptyset$. Sea un $y \in C \cap \overline{B(d, 2^{-l})} \neq \emptyset$, entonces existirá un $x \in \mathcal{D}_{p(n)}$ tal que $|y - x| \leq 2^{-p(n)-1}$. Aplicando la desigualdad triangular tenemos que

$$|x - d| \leq |x - y| + |y - d| \leq 2^{-p(n)-1} + 2^{-l} \leq 2^{-l} + 2^{-p(n)}.$$

Por lo tanto $x \in \mathcal{D}_{p(n)} \cap \overline{B(d, 2^{-l} + 2^{-p(n)})}$. También tenemos que $y \in [x - 2^{-p(n)-1}, x + 2^{-p(n)-1}] \cap C$, y por lo tanto, se cumple que $(M^+(x, n) = 1) \vee (M^-(x, n) = 1)$, y $f(d, l) = 1$ en este caso.

- Caso $\overline{B(d, 2 \cdot 2^{-l})} \cap C \neq \emptyset$. Entonces, para cualquier $x \in \mathcal{D}_{p(n)} \cap \overline{B(d, 2^{-l} + 2^{-p(n)})}$, si $y \in [x - 2^{-n}, x + 2^{-n}]$, entonces

$$|y - d| \leq |x - d| + |x - y| \leq 2^{-l} + 2^{-p(n)} + 2^{-n} < 2^{-l} + 2^{-l-1} + 2^{-l-1} = 2 \cdot 2^{-l}.$$

Por lo tanto $y \notin C$. Hemos obtenido que $[x - 2^{-n}, x + 2^{-n}] \cap C = \emptyset$, entonces $(M^+(x, n) = 1) \vee (M^-(x, n) = 1)$ no se cumple para ningún x . Por lo tanto, $f(d, l) = 0$ en este caso. $\square$

Bibliografia

- [BM06] A. F. Beardon and D. Minda. The hyperbolic metric and geometric function theory. In *Proceedings of the International Workshop on Quasiconformal Mappings and their Applications (IWQCMA05)*, 2006.
- [Bra05] Mark Braverman. Hyperbolic julia sets are poly-time computable. *Electron. Notes Theor. Comput. Sci.*, 120(C):17–30, February 2005.
- [Coo71] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, page 151–158, New York, NY, USA, 1971. Association for Computing Machinery.
- [Esc24] Escher in Het Paleis. Circle Limit IV (Heaven and Hell). <https://escherinhetpaleis.nl/en/about-escher/escher-today/circle-limit-iv-heaven-and-hell>, 2024. Accedido: 06-05-2026.
- [Mil06] John Milnor. *Dynamics in One Complex Variable. (AM-160): Third Edition. (AM-160)*. Princeton University Press, 2006.
- [Sip12] Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, Boston, MA, 3rd edition, 2012.