Universidade Estadual de Campinas

Instituto de Matemática, Estatística
e Computação Científica

Michel Faleiros Martins

O Décimo Problema de Hilbert Revisitado

Campinas
2018

Michel Faleiros Martins

O Décimo Problema de Hilbert Revisitado

Dissertação apresentada ao Instituto de Matemática, Estatística e Computação Científica da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em Matemática.

Orientador: Dr. Francesco Matucci

Coorientador: Dr. Fernando Eduardo Torres Orihuela

Este exemplar corresponde à versão final da dissertação defendida pelo aluno Michel Faleiros Martins e orientada pelo Prof. Dr. Francesco Matucci

Campinas
2018

Agência(s) de fomento e nº(s) de processo(s): CAPES

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

M366d

Martins, Michel Faleiros, 1987-
O décimo problema de Hilbert revisitado / Michel Faleiros Martins. – Campinas, SP : [s.n.], 2018.

Orientador: Francesco Matucci.
Coorientador: Fernando Eduardo Torres Orihuela.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Computação Científica.

1. Turing, Máquinas de. 2. Algoritmos. 3. Equações diofantinas. 4. Teoria dos números. 5. Funções recursivas. I. Matucci, Francesco, 1977-. II. Torres Orihuela, Fernando Eduardo, 1961-. III. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Revisiting Hilbert's tenth problem
Palavras-chave em inglês:
Turing machines
Algorithms
Diophantine equations
Number theory
Recursive functions
Área de concentração: Matemática
Titulação: Mestre em Matemática
Banca examinadora:
Fernando Eduardo Torres Orihuela [Coorientador]
Saeed Tafazolian
Nazar Arakelian
Data de defesa: 01-03-2018
Programa de Pós-Graduação: Matemática

Dissertação de Mestrado defendida em 01 de março de 2018 e aprovada pela banca examinadora composta pelos Profs. Drs.

Prof(a). Dr(a). FERNANDO EDUARDO TORRES ORIHUELA

Prof(a). Dr(a). SAEED TAFAZOLIAN

Prof(a). Dr(a). NAZAR ARAKELIAN

As respectivas assinaturas dos membros encontram-se na Ata de defesa

Agradecimentos

Agradeço ao meu orientador Francesco Matucci pela compreensão da escassez de tempo que possuia, pois estive trabalhando grande parte do tempo enquanto fazia o mestrado. Também agradeço pela forma organizada com que me guiou.

Agradeço ao meu coorientador Fernando Torres por ter me sugerido inúmeras possibilidades interessantes de temas relacionados à Álgebra e Teoria dos Números.

Agradeço aos professores Plamen Koshlukov e Eduardo Garibaldi por terem me dado muitas dicas de como fazer o mestrado antes que eu o começasse e terem me motivado a fazê-lo.

Agradeço à professora Elizabeth Gasparim que durante a Iniciação Científica, nos meus primeiros dias de IMECC, me mostrou como é a rotina de um pesquisador em Matemática dando detalhes das suas prórpias pesquisas.

Agradeço ao professor Vítor Araújo que me motivou bastante com suas explicações extremamente claras enquanto fazia uma disciplina na UFBA.

Agradeço a todos os demais professores do IMECC que tive contato ao longo do mestrado pelas excelentes aulas, dadas com rigor e de forma séria, aos funcionários da biblioteca que sempre foram muito atenciosos e aos amigos que fiz e que passamos horas de estudos juntos: Rodrigo Labiak, Jean Françõis e Julieth Paola.

Agradeço à CAPES pelo período de bolsa concedido durante meus primeiros 12 meses.

Agradeço especialmente à minha esposa Camila que ao longo do mestrado sempre me deu muito apoio.

Resumo

Nesta dissertação revisitamos a solução do Décimo Problema de Hilbert publicada por Martin Davis.

A ideia é apresentar de forma compreensível os pontos principais abordados nesta demonstração acrescentando alguns que foram omitidos da original e simplificam a leitura.

O texto está intimamente relacionado à diversos temas da Teoria dos Números (equações diofantinas, equação de Pell, congruências, etc) e da Teoria da Computabilidade (funções recursivas, funções computáveis, algoritmos, máquinas de Turing, etc).

Palavras-chave: Máquina de Turing, Algoritmo, Hilbert, Equação de Pell, Diofantino, Função Recursiva, Problema de Decisão, URM-Computabilidade.

Abstract

In this thesis we revisit the Martin Davis' version of the solution of Hilbert's 10th problem.

The idea is to carefully present the main points of the proof by adding a few details left out in the original text and which simplify the reading.

This work is closely related to several themes in Number Theory (diophantine equations, Pell's equation, congruences, etc) and in Computability Theory (recursive functions, computable functions, algorithms, Turing machines, etc).

Keywords: Turing Machine, Algorithm, Hilbert, Pell Equation, Diophantine, Recursive Function, Decision Problem, URM-Computability.

Lista de Símbolos

$\mathbb{N}_0$**:** É o conjunto dos inteiros não-negativos ou naturais, $\{0, 1, 2, ...\}$.

$\mathbb{N}_1$**:** É o conjunto dos inteiros positivos, $\{1, 2, 3, ...\}$.

$\mathbb{N}_k$**:** É o conjunto dos inteiros maiores ou iguais a k, $\{k, k+1, k+2, ...\}$, onde $k \in \mathbb{N}_0$.

$\mathbb{Z}/n\mathbb{Z}$**:** É o conjunto dos resídulos módulo n, $\{\overline{0}, \overline{1}, ..., \overline{p-1}\}$, onde $n \in \mathbb{N}_2$.

$r(\text{mod } n)$**:** É o elemento de $\mathbb{Z}/n\mathbb{Z}$ congruente a r módulo n.

Sumário

Capítulo 1

Familiarização com o Décimo Problema de Hilbert

1.1 Resumo Histórico

Em 1900 o Matemático alemão David Hilbert propôs uma lista de 23 problemas na Conferência Internacional de Matemática em Paris. Estes 23 problemas foram escolhidos de forma que a solução de cada um fosse capaz de trazer grande avanço para a Matemática ao longo do século XX. Alguns dos problemas ainda permanecem abertos, outros foram resolvidos parcialmente, um foi considerado vago demais e outro foi tido como não-matemático. A maioria foi resolvido e esta dissertação detalha a solução do Matemático estado-unidense Martin Davis do décimo problema desta lista cuja primeira solução completa foi atribuída ao Matemático russo Yuri Matiyasevich em 1970.

O Décimo Problema de Hilbert questiona sobre a existência de um algoritmo geral capaz de decidir a solubilidade de quaisquer equações Diofantinas, ou seja, capaz de afirmar se uma equação Diofantina qualquer (de grau e número de incógnitas arbitrários) possui ou não soluções inteiras.

Em 1972 o Matemático alemão Carl Siegel mostrou que tal algoritmo existe para a classe de equações Diofantinas cujo grau é 2 (e número arbitrário de incógnitas) e mostrou posteriormente que não existe para a classe de equações Diofantinas cujo grau é 4. Ainda é um problema aberto a existência de um algoritmo para a classe de grau 3.

O enunciado do Décimo Problema de Hilbert mencionava o termo "algoritmo"(essencial no problema), porém naquele ano ainda não havia uma definição precisa do que seria um "algoritmo". O desenrolar da solução, 70 anos depois, se deveu em grande parte ao desenvolvimento da noção precisa de algoritmo, principalmente aos Matemáticos inglês Alan Turing e estado-unidense Alonzo Church e à contribuição dos Matemáticos estado-unidenses Martins Davis, Julia Robinson e Hilary Putnam.

Vamos revisitar a solução de Martin Davis publicada em 1973 na *The American Mathematical Monthly* que se encontra em [2] e complementá-la com tópicos extras.

1.2 Máquinas de Turing

Vamos adotar as explicações dadas por Yuri Matiyasevich em [3] sobre máquinas de Turing, as definições são aprofundadas e formalizadas mais amplamente em [2].

Quando Hilbert enunciou o Décimo Problema, o significado do termo *algoritmo* não era matematicamente preciso e portanto requeria formalização e alguma convenção que fosse aceita pela comunidade. Após uma série de estudos por Lógicos e Matemáticos, pôde-se compreender melhor e deixar mais preciso o que se entendia por *algoritmo*. Observou-se que a ideia fundamental que está por trás de todas as maneiras de se definir esta noção é sempre a mesma. Podemos nos convencer que possuimos um método geral para resolver problemas de um determinado tipo quando ao usarmos este método somos capazes de solucionar qualquer problema particular daquele mesmo tipo sem empregar nossa criatividade, ou seja, apenas mecanicamente. Isso pode ser formalizado considerando que podemos construir uma máquina ou computador que resolverá problemas de determinada classe sem nossa intervenção, ou seja, automaticamente.

As máquinas de Turing são computadores abstratos. Embora possam ser descritas em terminologia puramente teórica pela Matemática, vamos adotar a convenção de descrever as máquinas de Turing como se fossem dispositivos físicos com certas propriedades (que variam conforme a abordagem do autor, mas que se mostram equivalentes quando confrontadas).

Uma máquina de Turing como iremos definir tem memória na forma de uma *fita* dividida em *células*. A fita é unidirecional, possui apenas uma extremidade que vamos assumir ser a esquerda. À direita, a fita continua infinitamente. Diferentemente de computadores físicos reais, isto quer dizer que um cálculo usando uma Máquina de Turing nunca terminará com uma menssagem da forma *memória insuficiente*. Por outro lado, para qualquer cálculo particular irá requerir somente um número finito de células.

Cada célula, ou estará vazia ou conterá um único *símbolo* de um conjunto finito de símbolos $A = \{\alpha_1, \alpha_2, ..., \alpha_\omega\}$ chamado alfabeto. Diferentes máquinas podem ter diferentes alfabetos. Um dos símbolos tem o papel importante de marcar a célula final mais a esquerda da fita e não aparece em mais nenhum outro lugar. Nós usaremos o símbolo "$\star$" para este marcador. Além disso, precisamos denotar uma célula vazia e para isto será usado o símbolo Λ.

Os símbolos sobre a fita são lidos, apagados ou escritos por uma *cabeça* a qual em cada momento de tempo discreto escanea uma das células. A cabeça pode mover-se ao longo da fita para a esquerda e para a direita.

Em cada momento, a máquina está em um dos finitos estados que denotaremos por $q_1, q_2, ..., q_v$. Um dos estados é declarado ser o *inicial* que sempre será assumido ser q_1. Além disso, um ou mais estados são declarados serem os estados *finais*.

A próxima ação de uma máquina é totalmente determinada pelo estado atual e o símbolo que esta sendo escaneado pela cabeça. Em um único passo, a máquina pode mudar o símbolo na célula, mover a cabeça uma célula para a direita ou para a esquerda e passar para o próximo estado. As ações são definidas por um conjunto de *instruções* na forma

$$q_i\alpha_j \Rightarrow \alpha_{A(i,j)}D(i,j)q_{Q(i,j)},$$

onde

1. q_i, o estado atual, não é permitido ser um estado final;
2. α_j é o símbolo escaneado pela cabeça, $\alpha_j \in A$ a menos que a célula atual esteja vazia, caso que ocorre se $\alpha_j = \Lambda$;
3. $\alpha_{A(i,j)}$ é o símbolo a ser escrito, $\alpha_{A(i,j)} \in A$ (O caso $\alpha_{A(i,j)} = \alpha_j$ não está excluído);
4. $D(i,j)$ representa o movimento da cabeça, são três opções:
   (a) L, a cabeça move-se uma célula para a esquerda;
   (b) R, a cabeça move-se uma célula para a direita;
   (c) S, a cabeça fica parada;
5. $q_{Q(i,j)}$ é o novo estado (o caso $q_{Q(i,j)} = q_i$ não está excluído).

Nos referimos à parte de uma instrução precedendo "⇒" como LHS (*left hand side* ou lado esquerdo) e à parte após "⇒" como RHS (*right hand side* ou lado direito). Seguindo nossa convenção, se $\alpha_j = \star$, então $\alpha_{A(i,j)} = \star$ e $D(i,j) \neq L$, por outro lado, se $\alpha_j \neq \star$, então $\alpha_{A(i,j)} \neq \star$. Para cada par consistindo de estado que não seja final e um elemento de $A \cup \{\Lambda\}$, deve existir exatamente uma instrução com o correspondente LHS.

Para iniciar, algum segmento da fita está ocupado com símbolos de A sem lacunas e o restante da fita, potencialemnte infinito, está vazio, a cabeça escanea uma das células e a máquina está no estado inicial q_1. Então, o trabalho da máquina terminará após uma instrução na qual $q_{Q(i,j)}$ executado é um estado final. É possível que a máquina nunca alcance o estado final e que o trabalho continue indefinidamente.

A informação de entrada é determinada pelo conteúdo inicial da fita e a posição da cabeça sobre a fita. O resultado do cálculo, a saída, é determinada pelo conteúdo da fita quando o trabalho pára, a posição da cabeça sobre a fita e o estado final. A ação de interpretar as informações de entrada e a saída são eventos externos à máquina de Turing. Por exemplo, números naturais podem ser escritos sobre a fita em notação binária usando os símbolos $\alpha = 0$ e $\beta = 1$, ou vice-versa e nestes dois casos a mesma máquina, em geral, iria calcular duas funções diferentes.

1.2.1 Composição de Máquinas

Gostaríamos de stabelecer a existência de máquinas com propriedades particulares, contudo sem a necessidade de escrever um sistema completo de instruções para cada uma, pois isto demandaria um trabalho imenso e talvez fosse impraticável. Em vez disso, é dado um conjunto de

instruções explícitas somente para algumas máquinas fundamentais mais simples e utiliza-se a composição para produzir novas máquinas mais complexas a partir destas. Isto permitirá obter as máquinas, ao menos demonstrar a existência de outras máquinas cujas instruções seriam demasiadamente complexas.

Todas as máquinas fundamentais que estamos interessados possuem o mesmo alfabeto:

$$\{\star, 0, 1, 2, 3, \lambda\}.$$

Haverá exatamente dois estados finais q_2 e q_3 cuja interpretação será SIM e NÃO, respectivamente sempre que a máquina alcançar algum desses dois estados. As células contendo o símbolo λ terá o papel indicar células vazias no seguinte sentido: somente células vazias e células contendo o símbolo λ podem ser situadas à direita de uma célula contendo o símbolo λ e para qualquer estado q_i as instruções com LHS igual a $q_i\lambda$ e $q_i\lambda$ terão RHS idênticos.

O primeiro método para construir uma máquina M compondo-se duas máquinas M_1 e M_2 está detalhado como segue:

1. Em todas as intruções da máquina M_1, o estado final q_2 é substituído por q_{v+1} onde v é o número de estados da máquina M_1 (devemos relembrar que os estados finais podem ocorrer somente nas instruções do RHS).

2. Em todas as intruções da máquina M_2, todo estado não final q_i, é substituído por q_{v+i} (em particular, q_1 é substituído por q_{v+1}).

3. O conjunto de intruções da nova máquina M consiste de intruções de ambas as máquinas dadas, modificadas como descrito acima.

A ação da máquina M consiste da execução consecutiva das ações das máquinas M_1 e M_2 como *originalmente constituídas*, levando-se em conta que M_1 parou no estado q_2. Para denotar a máquina M usamos a notação

$$M_1; M_2.$$

Vemos que a composição de Máquinas de Turing descrita acima é uma operação associativa, de modo que a notação $M_1; M_2; M_3$ está bem definida.

O segundo método para construir uma nova Máquina de Turing a partir de duas máquinas M_1 e M_2 dadas é descrito como segue:

1. Em todas as intruções da máquina M_1, o estado final q_2 é substituído por q_{v+1} onde v é o número de estados da máquina M_1 e o estado final q_3 é substituído por q_2.

2. Em todas as intruções da máquina M_2, todo estado não final q_i, é substituído por q_{v+i} e o estado final q_2 é substituído por q_1.

3. O conjunto de intruções da nova máquina M consiste de intruções de ambas as máquinas dadas, modificadas como descrito acima.

A máquina de Turing construída deste modo será denotada por

while M_1 **do** M_2 **od**

A ação desta máquina consiste em realizar as ações das máquinas M_1 e M_2 *como originalmente constituidas* até que uma delas entre no estado final q_3.

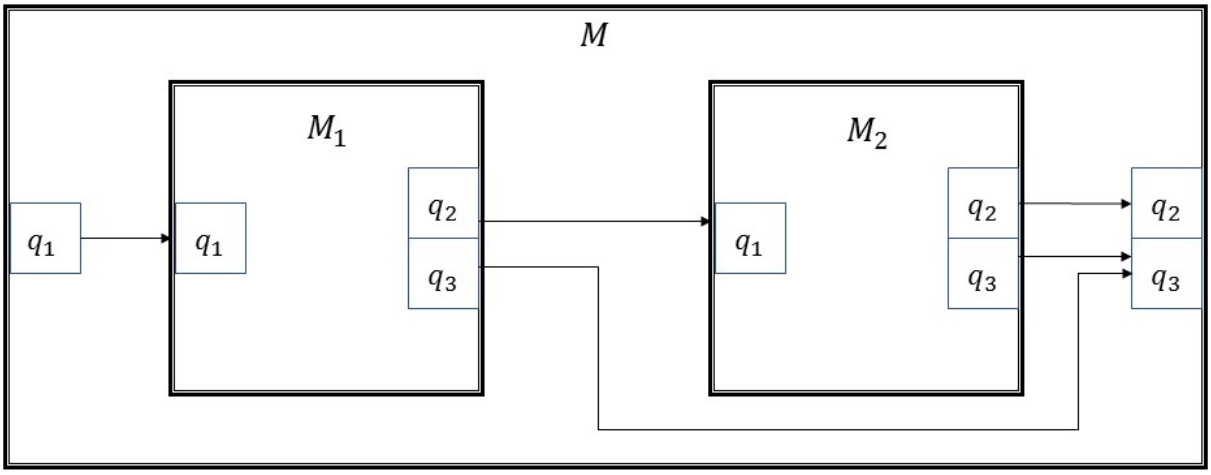

Figura 1.1: Primeiro método de composição de duas máquinas de Turing

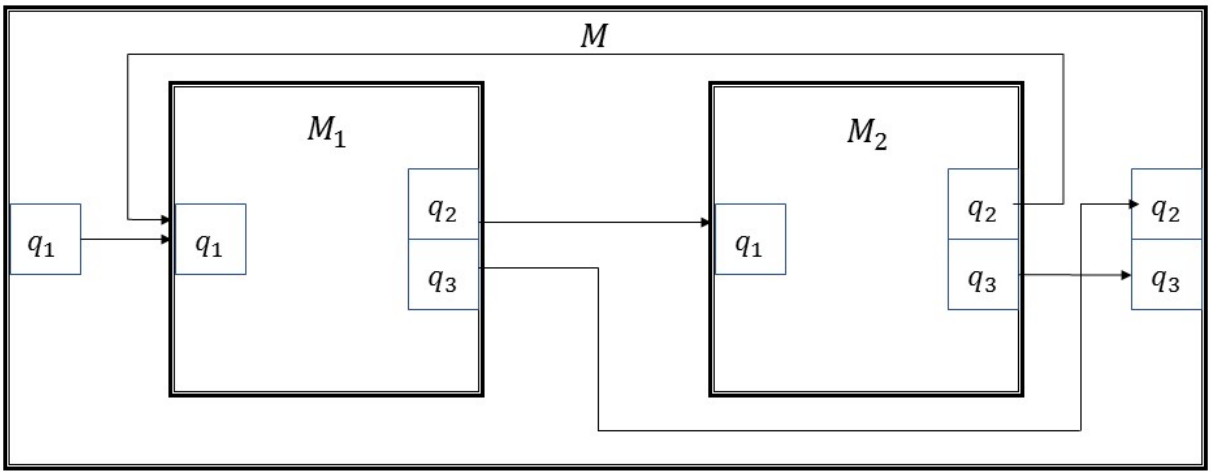

Figura 1.2: Segundo método de composição de duas máquinas de Turing

A notação introduzida acima assemelha-se a uma liguagem primitiva de programação. De fato, todo programa como este denota uma máquina de Turing particular.

As máquinas de Turing podem lidar com sequências de caracteres, mas estamos interessados naquelas que lidam números. Para representar um inteiro $m \in \mathbb{N}_0$ utilizamos $m + 1$ células consecutivas sendo a mais a esquerda com o símbolo 0 e as demais com o símbolo 1.

Observamos que à direita da última célula desta representação não poderá haver o símbolo 1. A representação da n-upla:

$$(a_1, a_2, ..., a_n)$$

consistirá de representações dos números $a_1, a_2, ..., a_n \in \mathbb{N}_0$ dispostos um após o outro, sem lacunas, começando pela segunda célula da fita. A primeira célula conterá, conforme discutimos anteriormente, o marcador $\star$. As demais células podem ser vazias ou, de acordo com nossa convenção, conter o símbolo λ.

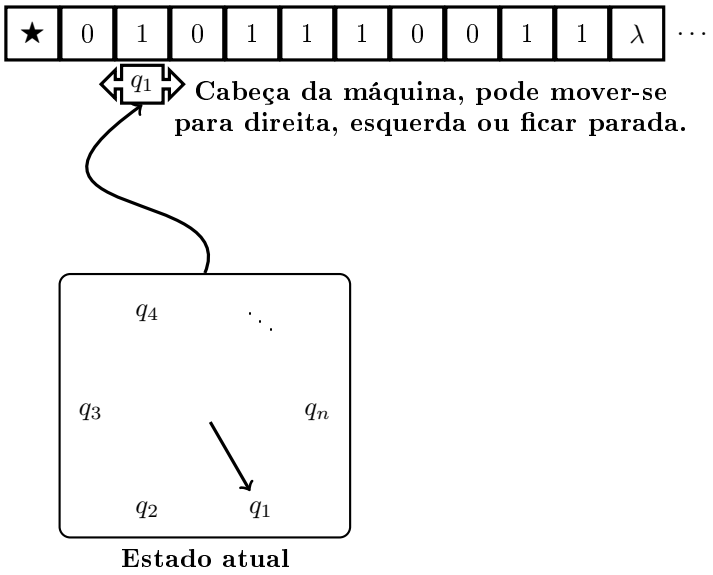

Figura 1.3: Máquina de Turing e a representação da quádrupla $(1, 3, 0, 2)$ sobre a fita.

Com estas máquinas podemos realizar diversas operações com tais $n-uplas$ e transformá-las em outras, como por exemplo:

$$(a_1, a_2, ..., a_{n-1}, 0);$$
$$(a_1, a_2, ..., a_{n-1}, a_n + a_k);$$
$$(a_1, a_2, ..., a_{n-1}, a_k + a_l);$$
$$(a_1, a_2, ..., a_{n-1}, a_k \cdot a_l);$$
$$(a_1, a_2, ..., a_{n-1}, a_k^2)$$

onde $1 \leq k, l \leq n$. Estas máquinas são construídas em detalhes em [3]. Precisamos de uma definição antes de continuarmos.

Definição 1.2.1. *Uma equação Diofantina é uma equação da tipo* $P(a_1, a_2, ..., a_n) = 0$ *onde* P *é um polinômio com coeficientes em* $\mathbb{Z}$ *nas* n *variáveis* $a_1, a_2, ..., a_n \in \mathbb{N}_1$.

Teorema 1.2.2. *Dada qualquer equação Diofantina paramétrica* $(a_1, a_2, ..., a_n$ *são os parâmetros que são fixados ao se resolver a equação Diofantina e pertencem a algum conjunto de valores pré-estabelecido*)

$$P(a_1, a_2, ..., a_n, x_1, x_2, ..., x_{m+1}) = 0 \tag{i}$$

podemos contruir uma máquina de Turing M *que pára em algum passo, começando com a representação da n-upla*

$$(a_1, a_2, ..., a_n),$$

se, e somente se, a equação (i) *é solúvel nas incógnitas* $x_1, x_2, ..., x_{m+1}$.

A demonstração passo a passo do Teorema (1.2.2) pode ser encontrada em [3].

O que mostramos com o Teorema (1.2.2) é que as equações Diofantinas são "semi-decidíveis", ou seja, se uma equação tem uma solução, então este fato pode ser revelado puramente de forma mecânica. Esta semi-decibilidade é uma propriedade intuitiva e a prova formal não contribui muito para o conhecimeto sobre equações Diofantinas. Não obstante, este fato nos mostra que as máquinas de Turing são suficientemente poderosas, apesar de seu conjunto primitivo de instruções.

Dizemos que um conjunto $\mathfrak{M}$ de n-uplas de números naturais ($\mathbb{N}_0$) é Turing semi-decidível se existe uma máquina de Turing M que, começando no estado q_1, com uma fita contendo a representação canônica (com nenhuma célula contendo λ da n-upla $(a_1, a_2, ..., a_n)$ e com sua cabeça escaneando a célula mais a esquerda da fita, em algum passo durante seu funcionamento irá parar se, e somente se, $(a_1, a_2, ..., a_n) \in \mathfrak{M}$. Neste caso dizemos que M semi-decide $\mathfrak{M}$.

Assim, o Teorema (1.2.2) mostra que todo conjunto Diofantino (veja a Definição (2.3.1)) é Turing semi-decidível. Temos também a recíproca:

Teorema 1.2.3. *Toda conjunto Turing semi-decidível é Diofantino.*

A demonstração passo a passo do Teorema (1.2.3) pode ser encontrada em [3].

1.3 A Tese de Church

Um conjunto $\mathfrak{M}$ de n-uplas com entradas em $\mathbb{N}_0$ é chamado de *Turing decidível* se existe uma máquina de Turing M que, começando no estado q_1, com a cabeça lendo a célula mais a esquerda da fita contendo a representação canônica da n-upla $(a_1, a_2, ..., a_n)$ pára em algum instante no estado q_2 se $(a_1, a_2, ..., a_n) \in \mathfrak{M}$ e no estado q_3 caso contrário.

Todo conjunto Turing decidível é também Turing semi-decidível. De fato, se M é uma máquina de Turing que revela a decibilidade de M, então a máquina que executa M

while M **do** PARA **od**; NUNCAPARA

semi-decide o conjunto M, onde as máquinas PARA e NUNCAPARA são dadas pelas sequências de instruções:

Máquina PARA:

$$\begin{aligned} q_1\star &\Rightarrow \star S q_3 \\ q_1 0 &\Rightarrow 0 S q_3 \\ q_1 1 &\Rightarrow 1 S q_3 \\ q_1 2 &\Rightarrow 2 S q_3 \\ q_1 3 &\Rightarrow 3 S q_3 \\ q_1 \lambda &\Rightarrow \lambda S q_3 \\ q_1 \Lambda &\Rightarrow \lambda S q_3 \end{aligned}$$

Máquina NUNCAPARA:

$$\begin{aligned} q_1\star &\Rightarrow \star S q_1 \\ q_1 0 &\Rightarrow 0 S q_1 \\ q_1 1 &\Rightarrow 1 S q_1 \\ q_1 2 &\Rightarrow 2 S q_1 \\ q_1 3 &\Rightarrow 3 S q_1 \\ q_1 \lambda &\Rightarrow \lambda S q_1 \\ q_1 \Lambda &\Rightarrow \lambda S q_1 . \end{aligned}$$

Se um conjunto é semi-decidível então seu complemento também é semi-decidível. De fato, é suficiente considerar a máquina

while M **do** NUNCAPARA **od**

Para uma explicação ilustrativa destes fatos veja [5]

Entretanto, não é imediato concluir a recíproca: se ambos $\mathfrak{M}$ e seu complemento são Turing semi-decidíveis, então $\mathfrak{M}$ é Turing decidível. Este fato ainda é verdadeiro e pode ser mostrado tendo em vista que a classe dos conjuntos Diofantinos é equivalente à classe dos conjuntos Turing semi-decidíveis conforme é feito em [3]

A equivalência entre a classe de conjuntos Diofantinos e a classe de conjuntos semi-decidíveis bem como o fato que será provado do Décimo problema de Hilbert ser Turing indecidível trazem novos questionamentos. A definição de conjunto Diofantino é de certa forma bastante natural ao passo que de conjunto Turing semi-decidível é repleta de detalhes técnicos que parecem artificiais. Por exemplo, poderíamos representar os números de forma binária em vez de unária. A fita poderia ser infinita em ambas direções em vez de somente em uma direção. No lugar de uma única cabeça poderia haver várias, cada uma executando seu próprio conjunto de instruções enquanto compartilhando informações sobre as células sendo escaneadas. Além disso,

poderia haver mais de uma fita. De fato, a memória não precisa ser linear, poderia ter a forma de um plano dividido em células quadradas. Para toda modificação da noção de uma máquina de Turing podemos introduzir um conceito correspondente de semi-decibilidade e enunciar a questão de como tal conceito se relaciona aos conjuntos Diofantinos.

A própria maneira de codificar equações Diofantinas poderia ser repensada, por exemplo, o natural seria escrever sobre a fita o número de incógnitas, o grau e os coeficientes de uma equação na forma unária ou em alguma notação posicional. Hilbert não impôs quaisquer restrições sobre o método desejado para resolver o Décimo Problema. Desta forma, se para alguma notação apropriada para polinômios, alguém tivesse êxito em construir uma máquina de decisão então o Décimo Problema de Hilbert teria uma solução positiva. Assim, o que garante que a Turing indecibiliadade do Décimo Problema conforme definida possa ser considerada uma solução negativa?

Para toda modificação da noção de máquina de Turing (ou qualquer outro dispositivo de cálculo abstrato) e para todo método de representar dados iniciais sobre a fita, alguém poderia tentar obter as afirmações de que a classe de conjuntos Diofantinos é idêntica à classe de conjuntos semi-decidíveis e que o Décimo Problema de Hilbert é indecidível. Isto poderia ser feito diretamente ou indiretamente. Para o primeiro, precisaríamos introduzir uma maneira adequada de codificação e provar que as relações correspondentes são Diofantinas. Para o segundo, podíamos esquecer as equações Diofantinas por enquanto e provar que a classe de conjuntos semi-decidíveis permanece a mesma não obstante às modificações na definição. Tais estudos de relações entre (semi-)decibilidade sobre diferentes dispositivos abstratos de cálculo foram de fato levados a exaustão durante o trabalho do Décimo Problema e muito antes sua indecibilidade foi estabelecida em todos os sentidos. Estes estudos mostraram que as noções de conjuntos semi-decidíveis e conjuntos decidíveis, que foram introduzidas usando máquinas de Turing, são de fato insensíveis à escolha particular do dispositivo de cálculo e do método de representação dos dados iniciais. As classes idênticas surgiram para qualquer escolha razoável. Por "razoável" entendemos que passos elementares de cálculo devem mesmo ser elementares, ou seja, facilmente realizáveis (por exemplo, não se pode reconhecer em um único passo se uma equação Diofantina particular possui solução).

Muito antes do surgimento das primeiras noções matematicamente rigorosas de dispositivos abstratos de cálculo tais como máquinas de Turing, havia a noção intuitiva de um algoritmo como um método certo para se chegar a uma solução mecânica de problemas de um tipo específico. Como um exemplo clássico, podemos mencionar o algoritmo de Euclides para encontrar o máximo divisor comum de dois inteiros positivos ou o método de extração de raízes quadradas conhecido na Grécia antiga. Quando um algoritmo era proposto, era claro que de fato havia um método universal para solucionar problemas do tipo específico. Entretanto, tais noções não são suficientes para uma prova de que não existe algoritmo para problemas do tipo sendo considerado.

Os dados iniciais para um algoritmo (no sentido intuitivo) são selecionados de algum conjunto enumerável e, sem perda de generalidade, vamos considerar somente situações nas quais os dados iniciais consistem de números naturais ou de n-uplas de números naturais de algum comprimento fixo.

O resultado da execução de um algoritmo é também um objeto de tipo específico. Poderíamos ter escolhido considerar somente algoritmos cujas saídas são números naturais, mas está mais no espírito deste tema considerar em vez disso algoritmos com as duas saídas "SIM" ou "NÃO". Correspondentemente, junto com o conceito de algoritmo, dois outros conceitos relacionados aparecem, os conceitos intuitivos de conjunto decidível e semi-decidível. Um conjunto $\mathfrak{M}$ de n-uplas é decidível (no sentido intuitivo) se existe um algoritmo (também no sentido intuitivo) que pára sobre toda n-upla de números naturais e retorna com a mensagem "SIM" ou "NÃO" dependendo se a n-upla pertence ou não ao conjunto. Analogamente, para $\mathfrak{M}$ ser semi-decidível precisaríamos de um algoritmo que retornasse "SIM" para toda n-upla em $\mathfrak{M}$ e que ou retornaria "NÃO" ou falhasse em parar se a n-upla não pertencesse à $\mathfrak{M}$. A Teoria da Computabilidade em sua totalidade poderia ser explicada em termos de conjuntos decidíveis e semi-decidíveis, da mesma forma ao se eliminar a noção de funções da Matemática lidar somente com seus gráficos.

Entre decibilidade intuitiva e semi-decibilidade existe a mesma relação que entre Turing-decibilidade e semi-decibilidade: um conjunto é dicidível se, e somente se, ambos o conjunto e seu complemento são semi-decidíveis.

Quão formal Turing (semi-)decibilidade está relacionada a (semi-)decibilidade no sentido intuitivo? Uma relação é evidente: conjuntos Turing (semi-)decibilidíveis são reconhecidos pela maioria dos matemáticos como (semi-)decidível no sentido intuitivo. O oposto é conhecido como:

Tese de Church: Todo conjunto de n-uplas que é (semi-)decidível no sentido intuitivo é também Turing (semi-)decidível.

Neste ponto se encontra algo raramente encontrado em Matemática, uma tese. O que seria uma tese? Não se trata de um teorema pois não se tem prova. Também não é uma conjectura pois não se pode obter uma prova. Nem um axioma pois não temos a liberdade para aceitar ou rejeitar. Tudo isto é devido ao fato que a tese de Church não é uma afirmação matemática precisa, porque relaciona uma noção rigorosa de Turing (semi-)decibilidade com uma não-rigorosa de (semi)decibilidade no sentido intuitivo.

Quais são os argumentos a favor da Tese de Church? Podem ser muito diferentes em aparência mas todos são essencialmente o mesmo: até que ninguém tenha encontrado um exemplo de um conjunto que pudesse ser reconhecido pelos matemáticos como (semi-)decidível no sentido intuitivo e para o qual não se pode construir uma máquina de Turing. Evidentemente esta afirmação não deve ser compreendida ao pé da letra, ou seja, pensada tal que para todo conjunto reconhecidamente (semi-)decidível alguém ter realmente construído uma máquina.

O que a Tese de Church busca? Por um lado serve como uma estrela guia: tão breve como temos estabelecido (semi-)decibilidade no sentido intuitivo, nossa chance de encontrar a máquina de Turing correspondente deve ser considerada muito alta. De fato, matemáticos profissionais normalmente se contentam por estabelecerem intuitivo (semi-)decibilidade e não condescendem à provas formais.

Por outro lado, a Tese de Church tem um papel importante em Matemática semelhante ao desempenhado na Física pela Lei da Consevação da Energia. Da mesma maneira, como até o momento nenhuma exceção à lei tem sido encontrada, não é razoável tentar construir um moto-perpétuo. Analogamente, uma vez que que a Turing indecibilidade de um conjunto tenha sido provada, não se preicsa gastar tempo procurando um método universal para reconhecer os elementos daquele conjunto. Em particular, de acordo com a Tese de Church, o resultado negativo do Décimo Problema de Hilbert nos dá um dever moral de cessar a busca por um "processo" do tipo indagado por Hilbert para o seu Décimo Problema.

Capítulo 2

Noções Preliminares

Vamos inicialmente reduzir o Décimo Problema de Hilbert a um problema de decisão equivalente, contudo mais fácil de resolver. Queremos trabalhar somente com inteiros positivos ou no máximo inteiros não-negativos, mas o Décimo Problema de Hilbert compreende equações Diofantinas cujas variáveis são inteiras e portanto podem ser negativas. Para a redução do Décimo Problema de Hilbert vamos utilizar o Teorema dos Quatro quadrados.

Observação 2.0.1. *Vamos denotar por* $\mathbb{N}_1 = \{1, 2, 3, ...\}$ *o conjunto dos inteiros positivos e* $\mathbb{N}_0 = \{0, 1, 2, ...\}$ *o conjunto dos inteiros não negativos.*

2.1 Teorema dos Quatro Quadrados

Vamos demonstrar o Teorema dos Quatro Quadrados. Como referência e aprofundamento do tema sugerimos consultar [1] ou [3].

Lema 2.1.1. ***(Identidade de Euler)***. *Sejam* $a, b, c, d, A, B, C, D \in \mathbb{C}$*, então vale a identidade*

$$\begin{aligned}(a^2+b^2+c^2+d^2)(A^2+B^2+C^2+D^2) = &(aA+bB+cC+dD)^2\\ &+(aB-bA-cD+dC)^2\\ &+(aC+bD-cA-dB)^2\\ &+(aD-bC+cB-dA)^2.\end{aligned}$$

Demonstração. Sejam $\alpha, \beta, \gamma, \delta \in \mathbb{C}$. Facilmente verificamos a identidade matricial:

$$\begin{bmatrix}\alpha & \beta\\ -\overline{\beta} & \overline{\alpha}\end{bmatrix}\begin{bmatrix}\gamma & \delta\\ -\overline{\delta} & \overline{\gamma}\end{bmatrix} = \begin{bmatrix}\alpha\gamma-\beta\overline{\delta} & \alpha\delta+\beta\overline{\gamma}\\ -(\alpha\delta+\beta\overline{\gamma}) & (\alpha\gamma-\beta\overline{\delta})\end{bmatrix} \tag{I}$$

Aplicando determinantes de ambos os lados de (I) obtemos

$$(|\alpha|^2+|\beta|^2)(|\gamma|^2+|\delta|^2) = |\alpha\gamma-\beta\overline{\delta}|^2+|\alpha\delta+\beta\overline{\gamma}|^2. \tag{II}$$

Agora basta substituir em (II) $\alpha = a - bi$, $\beta = -c - di$, $\gamma = A + Bi$, $\delta = C + Di$.

□

Lema 2.1.2. *Seja* $m \in \mathbb{N}_0$ *e suponha que* $2m$ *seja soma de dois quadrados, então* m *também é soma de dois quadrados.*

Demonstração. Se $2m = r^2 + s^2$, então r e s têm a mesma paridade. Portanto, $\left(\frac{r\pm s}{2}\right)^2 \in \mathbb{N}_0$ e então podemos escrever $m = \left(\frac{r+s}{2}\right)^2 + \left(\frac{r-s}{2}\right)^2$.

□

Lema 2.1.3. *Se p é primo ímpar, então existem $u, v, k \in \mathbb{N}_0$ tais que $u^2 + v^2 + 1 = kp$.*

Demonstração. Considere os conjuntos:

$$U = \left\{ u^2 \in \mathbb{Z}/p\mathbb{Z} \,|\, 0 \leq u \leq \frac{p-1}{2} \right\},$$
$$V = \left\{ -v^2 - 1 \in \mathbb{Z}/p\mathbb{Z} \,|\, 0 \leq v \leq \frac{p-1}{2} \right\}.$$

Temos que $\#U + \#V = \left(\frac{p-1}{2} + 1\right) + \left(\frac{p-1}{2} + 1\right) = p + 1 > p$, logo $U \cap V \neq \emptyset$, ou seja, existem $u, v \in \mathbb{N}_0$ tais que $u^2 \equiv -v^2 - 1 (\text{mod } p)$.

□

Teorema 2.1.4. ***(Teorema dos Quatro Quadrados)*** *Todo $n \in \mathbb{N}_0$ pode ser escrito como a soma 4 quadrados.*

Demonstração. Podemos escrever $n = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$ com $p_1, p_2, ..., p_k$ são primos distintos e os expoentes são inteiros positivos. Se provarmos o Teorema para primos, basta aplicarmos sucessivamente o Lema (2.1.1) quando n for composto. Como $2 = 1^2 + 1^2$ precisamos provar somente para os primos ímpares. Pelo Lema (2.1.3) existem $a, b, c, d \in \mathbb{N}_0$ e $m \in \mathbb{N}_1$ tais que $mp = a^2 + b^2 + c^2 + d^2$ com $c = 1$ e $d = 0$. Agora vamos provar que se $m > 1$ podemos encontrar um $0 < n < m$ tal que np pode ser escrito como a soma de 4 quadrados. Com efeito, se m é par, então nenhum, dois ou quatro dos números a, b, c, d são pares e podemos usar o Lema 2.1.2 adequadamente e por fim escolher $n = \frac{m}{2}$. Portanto, podemos supor $m > 1$ ímpar. Sejam $A, B, C, D \in \mathbb{Z}$ satisfazendo as congruências

$$\begin{cases} A \equiv a (\text{mod } m) \\ B \equiv b (\text{mod } m) \\ C \equiv c (\text{mod } m) \\ D \equiv d (\text{mod } m) \end{cases}$$

onde $A, B, C, D \in \{-\frac{m-1}{2}, -\frac{m-1}{2} + 1, ..., -1, 0, 1, ..., \frac{m-1}{2}\}$. Sendo assim, temos que

$$A^2 + B^2 + C^2 + D^2 < 4 \cdot \frac{m^2}{4} \quad \text{e} \quad A^2 + B^2 + C^2 + D^2 \equiv 0 (\text{mod } m)$$

Portanto, $A^2 + B^2 + C^2 + D^2 = nm$ com $0 < n < m$. Pela escolha de A, B, C, D temos que

$$aB - bA - cD + dC \equiv 0 (\text{mod } m),$$
$$aC + bD - cA - dB \equiv 0 (\text{mod } m),$$
$$aD - bC + cB - dA \equiv 0 (\text{mod } m),$$

e

$$aA + bB + cC + dD \equiv a^2 + b^2 + c^2 + d^2 \equiv 0 (\text{mod } m).$$

Logo, pelo Lema(2.1.1) temos que

$$\begin{aligned} np = \frac{1}{m^2} \cdot mp \cdot nm &= \frac{1}{m^2}(a^2+b^2+c^2+d^2)(A^2+B^2+C^2+D^2) \\ &= \left(\frac{aA+bB+cC+dD}{m}\right)^2 + \left(\frac{aB-bA-cD+dC}{m}\right)^2 \\ &\quad + \left(\frac{aC+bD-cA-dB}{m}\right)^2 + \left(\frac{aD-bC+cB-dA}{m}\right)^2. \end{aligned}$$

Assim sucessivamente até encontrarmos $n = 1$ de modo que p pode ser escrito como a soma de 4 quadrados.

□

Teorema 2.1.5. ***(Teorema Chinês dos Restos)****: Sejam* $r_1, r_2, ..., r_k \in \mathbb{N}_0$ *e* $m_1, m_2, ..., m_k \in \mathbb{N}_1$ *com* $\text{mdc}(m_1, m_2, ..., m_k) = 1$*. Então, o sistema de congruências*

$$\begin{cases} x \equiv r_1 (\text{mod } m_1) \\ x \equiv r_2 (\text{mod } m_2) \\ \quad \vdots \\ x \equiv r_k (\text{mod } m_k) \end{cases}$$

admite solução $x \in \mathbb{N}_1$ *única módulo* $m = m_1 m_2 \cdots m_k$*.*

Demonstração. Considere a aplicação

$$\begin{aligned} &f : \mathbb{Z}/(m) \to \mathbb{Z}/(m_1) \times \mathbb{Z}/(m_2) \times \cdots \times \mathbb{Z}/(m_k) \\ &r(\text{mod } m) \mapsto (r(\text{mod } m_1), r(\text{mod } m_2), ..., r(\text{mod } m_k)). \end{aligned}$$

Esta aplicação está bem definida, ou seja, o valor de $f(r(\text{mod } m))$ independe da escolha do representante da classe de $r(\text{mod } m)$, pois quaisquer dois representantes diferem de um múltiplo de m, que tem imagem $(0(\text{mod } r_1), 0(\text{mod } r_2), ..., 0(\text{mod } r_k)$ no produto $\mathbb{Z}/(r_1) \times \mathbb{Z}/(r_2) \times \cdots \times \mathbb{Z}/(r_k)$. O Teorema é equivalente a mostrar que f é bijetiva, pois sendo sobrejetiva nos garante que o sistema de congruências possua solução e sendo injetiva nos garante que a solução seja única módulo m. Como o domínio e o contradomínio têm o mesmo número de elementos (m elementos), basta mostrarmos que f é injetiva. Suponha que $f(b_1(\text{mod } m)) = f(b_2(\text{mod } m))$ então $b_1 \equiv b_2(\text{mod } m_i)$ para todo $1 \leq i \leq k$ e daí $m_i | b_1 - b_2$, mas os $m_i's$ são primos dois a dois e então $m | b_1 - b_2$, ou seja, $b_1 \equiv b_2(\text{mod } m)$. □

2.2 Redução do Décimo Problema de Hilbert

Um sistema de equações Diofantinas (lembre-se da Definição (1.2.1)

$$\begin{cases} P_1(a_1, a_2, ..., a_n) = 0 \\ P_2(a_1, a_2, ..., a_n) = 0 \\ \quad \vdots \\ P_k(a_1, a_2, ..., a_n) = 0 \end{cases}$$

tem uma solução em $a_1, a_2, .., a_n$ se, e somente se, a equação Diofantina

$$P_1^2(a_1, a_2, ..., a_n) + P_2^2(a_1, a_2, ..., a_n) + \cdots P_k^2(a_1, a_2, ..., a_n) = 0$$

tem uma.

Suponha que estejamos procurando soluções em $\mathbb{N}_1$ para a equação Diofantina

$$P(a_1, a_2, ..., a_n) = 0. \tag{III}$$

Considere o seguinte sistema:

$$\begin{cases} P(b_1, b_2, ..., b_n) = 0 \\ b_1 = c_{1,1}^2 + c_{1,2}^2 + c_{1,3}^2 + c_{1,4}^2 + 1 \\ b_2 = c_{2,1}^2 + c_{2,2}^2 + c_{2,3}^2 + c_{2,4}^2 + 1 \\ \qquad\qquad \vdots \\ b_n = c_{n,1}^2 + c_{n,2}^2 + c_{n,3}^2 + c_{n,4}^2 + 1 \end{cases} \tag{IV}$$

onde P é o mesmo polinômio só que agora nas variáveis $b_1, b_2, ..., b_n \in \mathbb{Z}$ e $c_{1,1}, ..., c_{n,4} \in \mathbb{Z}$.

Assim, qualquer solução do sistema (IV) inclui um solução da equação(III) em $\in \mathbb{N}_1$. A recíproca também é verdadeira, pois para qualquer solução de (III) em $\mathbb{N}_1$, pelo Teorema (2.1.4), existem $c_{1,1}, ..., c_{1,4}, c_{2,1}, ..., c_{m,4} \in \mathbb{Z}$ que solucionam o sistema (IV). Como vimos, podemos condensar o sistema (IV) em uma única equação polinomial de coeficientes em $\mathbb{Z}$, a qual terá soluções em $\mathbb{Z}$ se, e somente se, a equação (III) tem solução em $\mathbb{N}_1$.

Suponha que estejamos procurando soluções em $\mathbb{Z}$ para a equação Diofantina

$$P(a_1, a_2, ..., a_n) = 0. \tag{V}$$

Considere a equação:

$$P(b_1 - c_1, b_2 - c_2, ..., b_n - c_n) = 0 \tag{VI}$$

onde $b_1, b_2, ..., b_n, c_1, c_2, ..., c_n \in \mathbb{N}_1$

Assim, toda solução da equação (VI) gera uma solução

$$\begin{cases} a_1 = b_1 - c_1 \\ a_2 = b_2 - c_2 \\ \quad \vdots \\ a_n = b_n - c_n \end{cases}$$

da equação (V) $a_1, a_2, ..., a_n \in \mathbb{Z}$. Reciprocamente, dada uma solução $a_1, a_2, ..., a_n \in \mathbb{Z}$ da equação (V) podemos encontrar $b_1, b_2, ..., b_n, c_1, c_2, ..., c_n \in \mathbb{N}_1$ que satisfaçam a equação (VI). Sendo assim, de fato basta assumirmos como na definição (1.2.1) que estamos interessados nos polinômios cujas variáveis são inteiras positivas em vez de inteiras.

Assim, temos mostrado que o Décimo Problema de Hilbert como um problema de decisão é equivalente se considerarmos somente as soluções inteiras positivas.

2.3 Conjuntos e Funções Diofantinos

Definição 2.3.1. *Um conjunto D de n-uplas ordenadas de inteiros positivos é chamado Diofantino se existe um polinômio em $n+m$ variávies $P(x_1, x_2, ..., x_n, y_1, y_2, ..., y_m)$, com $m \geq 0$ e coeficientes inteiros, tal que uma dada n-upla $(a_1, a_2, ..., a_n) \in D$ se, e somente se, existem $b_1, b_2, ..., b_n \in \mathbb{N}_1$ tais que $P(a_1, a_2, ..., a_n, b_1, b_2..., b_m) = 0$.*

Vamos escrever a relação entre o conjunto D e o polinômio P simplesmente como

$$(a_1, a_2, ..., a_n) \in D \Leftrightarrow (\exists\, b_1, b_2, ..., b_m)[P(a_1, a_2, ..., a_n, b_1, b_2, ..., b_m) = 0]$$

ou de forma equivalente

$$D = \{(a_1, a_2, ..., a_n) | (\exists\, b_1, b_2, ..., b_m)[P(a_1, a_2, ..., a_n, b_1, b_2, ..., b_m) = 0]\}.$$

No que segue, vamos desenvolver algumas técnicas para facilitar a demonstração que alguns conjuntos são Diofantinos.

Primeiro, alguns exemplos de conjuntos Diofantinos.

i. Os números que não são potência de 2:

$$x \in D \Leftrightarrow (\exists\, y, z)[x = y(2z + 1)];$$

ii. Os números compostos:

$$x \in D \Leftrightarrow (\exists\, y, z)[x = (y + 1)(z + 1)];$$

iii. A relação de ordenação dos inteiros positivos, i.e., os conjuntos $\{(x, y)|x < y\}$,$\{(x, y)|x \leq y\}$:

$$x < y \Leftrightarrow (\exists\, z)[x + z = y],$$
$$x \leq y \Leftrightarrow (\exists\, z)[x + z - 1 = y];$$

iv. A relação de divisibilidade, i.e., $\{(x, y)|x \,\text{divide}\, y\}$:

$$x|y \Leftrightarrow (\exists\, z)[xz = y].$$

v. O conjunto W de ternos (x, y, z) tais que $x|y$ e $x < z$:

$$x|y \Leftrightarrow (\exists\, r)[y = xr] \,\text{e}\, \Leftrightarrow (\exists\, s)[z = x + s]$$

Então,

$$(x, y, z) \in W \Leftrightarrow (\exists\, r, s)[(y - xr)^2 + (z - x - s)^2 = 0].$$

O último exemplo mostra uma técnica que pode ser aplicada de modo mais geral. Se um conjunto Diofantino D é equivalente à solução de um sistema de equações polinomiais $P_1 = 0,\ P_2 = 0, ..., P_k = 0$ podemos substituir este sistema por uma única equação polinomial:

$$P_1^2 + P_2^2 + \cdots + P_k^2 = 0$$

conforme vimos em (2.2).

Observação 2.3.2. *Quando usarmos o termo "função" estaremos nos referindo a uma função que assume valores inteiros positivos definida sobre uma ou mais variáveis inteiras positivas.*

Definição 2.3.3. *Uma função $f : \mathbb{N}_1^n \to \mathbb{N}_1$ é chamada Diofantina se o conjunto*

$$\{(x_1, x_2, ..., x_n, y)|y = f(x_1, x_2, ..., x_n)\}$$

é Diofantino, i.e., f é Diofantina se seu gráfico é Diofantino.

2.4 Uma Importante Função Diofantina

Iremos agora mostrar algumas propriedades relacionadas a uma importante função Diofantina que serão úteis mais tarde. Considere os números triagulares:

$$T(n) = 1 + 2 + \cdots + n = \frac{n(n+1)}{2}.$$

Como $T(n)$ é uma função estritamente crescente, para cada inteiro positivo z, existe um único $n \geq 0$ tal que

$$T(n) < z \leq T(n+1) = T(n) + n + 1.$$

Então cada z é unicamente representável como

$$z = T(n) + y; \quad y \leq n + 1,$$

ou de forma equivalente, unicamente representável como:

$$z = T(x + y - 2) + y.$$

Neste caso, podemos escrever $x = L(z)$ e $y = R(z)$; e fazer

$$P(x, y) = T(x + y - 2) + y.$$

Podemos ver que estas três funções $L(z)$, $R(z)$ e $P(x, y)$ são Diofantinas, pois

$$z = P(x, y) \Leftrightarrow 2z = (x + y - 2)(x + y - 1) + 2y$$

$$x = L(z) \Leftrightarrow (\exists\, y)[2z = (x + y - 2)(x + y - 1) + 2y]$$

$$y = R(z) \Leftrightarrow (\exists\, x)[2z = (x + y - 2)(x + y - 1) + 2y].$$

Vemos assim que a função $P(x, y)$ associa bijetivamente cada par de inteiros positivos (x, y) a um inteiro positivo z. Vemos também que $x + y = n + 2 \Rightarrow x \leq n + 1$ e então, $z > T(n) \geq n \geq x - 1 \Rightarrow z \geq x = L(z)$ Analogamente, $z \geq R(z)$.

Resumindo, temos construido três funções Diofantinas que satisfazem as propriedades:

1. Para todo x, y, temos $L(P(x, y)) = x$ e $R(P(x, y)) = y$;
2. Para todo z, temos $P(L(z), R(z)) = z$, $L(z) \leq z$ e $R(z) \leq z$.

Considere a função $F(a, b)$ definida como

$$F(a, b) = r$$

onde r é o único inteiro positivo tal que

$$r \equiv L(b) \pmod{(1 + aR(b))}$$

$$r \leq 1 + aR(b)$$

A função $F(a, b) = r$, assim definida, é Diofantina, pois se considerarmos os polinômios:

$$P_1(b, x, y) = 2b - [(x + y - 2)(x + y - 1) + 2y]$$

$$P_2(a,r,x,y,z) = x - [r + (1+ay)(z-1)]$$
$$P_3(a,r,y,w) = 1 + ay - (r + w - 1)$$

$P_1 = 0$ assegura que $x = L(b)$, $P_2 = 0$ garante que $y = R(b)$ e $P_3 = 0$ força que r seja o menor resíduo possível. Podemos escrever $Q(a,b,r,x,y,z,w) = P_1^2 + P_2^2 + P_3^2$ de modo que $Q = 0$ se, e somente se, $F(a,b) = r$, portanto F é Diofantina.

Seja $r_1, r_2, ..., r_k \in \mathbb{N}_0$ uma sequência arbitrária. Vamos encontrar um b_0 tal que

$$F(a, b_0) \leq b_0$$

$$F(a, b_0) = r_a \quad \text{para todo} \quad 1 \leq a \leq k.$$

Da definição de F e das propriedades de L obtemos que $F(a,b_0) \leq L(b_0) \leq b_0$. Seja $l_0 > r_a$ para todo $1 \leq a \leq k$ e $l_0 l$ divisível por $k!$. Então $1 + l_0, 1 + 2l_0, ..., 1 + kl_0$ é uma sequência de termos dois a dois primos entre si. De fato, se $d|(1 + il_0)$ e $d|(1 + jl_0)$ com $i < j$ então $d|(j(1 + il_0) - i(1 + jl_0))$,i.e., $d|j - i < k$ mas daí $d|l_0$, logo $d|1$, portanto $d = 1$. Pelo Teorema Chinês dos Restos, existe um m_0 inteiro positivo tal que

$$\begin{aligned} m_0 &\equiv r_1 \pmod{1 + l_0} \\ m_0 &\equiv r_2 \pmod{1 + 2l_0} \\ &\vdots \\ m_0 &\equiv r_k \pmod{1 + kl_0} \end{aligned}$$

Seja $b_0 = P(m_0, l_0)$ de modo que $m_0 = L(b_0)$ e $l_0 = R(b_0)$. Assim, para $a = 1, 2, ..., k$ temos

$$r_a \equiv L(b_0) \pmod{1 + aR(b_0)}$$

e $r_a < l_0 = R(b_0) < 1 + aR(b_0)$. Logo, pela definição de F, $F(a,b_0) = r_a$ para todo $1 \leq a \leq k$.

Esta função $F(a,b)$ será útil quando formos demonstrar o Teorema (3.2.1).

Teorema 2.4.1. *Um conjunto $A \subset \mathbb{N}$ é Diofantino se, e somente se, existe um polinômio P tal que $A = Im(P) \cap \mathbb{N}$.*

Demonstração. ($\Leftarrow$) Se $A = Im(P) \cap \mathbb{N}$ então

$$x \in A \Leftrightarrow (\exists\, x_1, x_2, ..., x_m \in \mathbb{N})[x = P(x_1, x_2, ..., x_m)]$$

Portanto, A é Diofantino por definição.
($\Rightarrow$) Seja A um conjunto Diofantino. Então,

$$x \in A \Leftrightarrow (\exists\, x_1, x_2, ..., x_m \in \mathbb{N}_1)[Q(x, x_1, x_2, ..., x_m)] = 0$$

para algum polinômio Q. Vamos fazer $P(x, x_1, x_2, ..., x_m) = x[1 - Q^2(x_1, x_2, ..., x_m)]$. Então, se $a \in A$, escolhemos $a_1, a_2, ..., a_m$ tais que $Q(a_1, a_2, ..., a_m) = 0$. Então $P(a, a_1, a_2, ..., a_m) = a$ e assim $a \in Im(P)$. Por outro lado, se $P(t, t_1, t_2, ..., t_m) = u > 0$ então, $Q(t, t_1, t_2, ..., t_m) = 0$ (caso contrário, $1 - Q^2 \leq 0 \Rightarrow u \leq 0$) $\Rightarrow t = u$ e $t \in A$. □

Capítulo 3

A Função Exponencial

3.1 Lemas Auxiliares

Neste capítulo vamos demonstrar alguns lemas relacionados a um caso particular da equação de Pell. Iindicamos as referências (1) e (4) para este assunto. Mais tarde esses lemas serão úteis na demonstração de teoremas que facilitarão concluirmos que certas funções são Diofantinas

Vamos considerar a seguinte equação de Pell:

$$x^2 - dy^2 = 1, \quad x, y \in \mathbb{Z} \tag{1}$$

onde

$$d = a^2 - 1, \quad a \in \mathbb{N}_2.$$

Vamos denotar uma solução (raiz) de (1) pelo par (x, y).

As equações de Pell na forma geral são do tipo (1) com d não quadrado perfeito.

Lema 3.1.1. *Não existem $x, y \in \mathbb{Z}$ que satisfaçam simultaneamente a equação (1) e a condição $1 < x + y\sqrt{d} < a + \sqrt{d}$.*

Demonstração. Por contradição, suponha que existam tais inteiros. Então, da equação (1) temos que

$$x^2 - dy^2 = (x + y\sqrt{d})(x - y\sqrt{d}) = 1 = (a + \sqrt{d})(a - \sqrt{d}) = a^2 - d.$$

Logo, usando a condição $1 < x + y\sqrt{d} < a + \sqrt{d}$, temos que

$$x - y\sqrt{d} < 1, \quad a - \sqrt{d} < 1, \quad \text{e} \quad x - y\sqrt{d} > a - \sqrt{d}.$$

Donde concluimos que

$$-1 < -x + y\sqrt{d} < -a + \sqrt{d}.$$

Logo

$$0 < 2y\sqrt{d} < 2\sqrt{d} \Rightarrow 0 < y < 1.$$

O que nos leva a um absurdo. □

Lema 3.1.2. *Sejam $r, s, t, u \in \mathbb{Z}$ tais que os pares (r, s) e (t, u) satisfaçam a equação ((1)). Sejam $p, q \in \mathbb{Z}$ tais que*

$$p + q\sqrt{d} = (r + s\sqrt{d})(t + u\sqrt{d}). \tag{2}$$

Então, (p, q) também é raiz de (1).

Demonstração. Conjugando a igualdade (2) temos que

$$p - q\sqrt{d} = (r - s\sqrt{d})(t - u\sqrt{d}). \tag{3}$$

Multiplicando ambas as igualdades (2) e (3) obtemos:

$$p^2 - dq^2 = (r^2 - ds^2)(t^2 - du^2) = 1.$$

□

Definição 3.1.3. *Definimos a sequência de pares $\{(x_n(a), y_n(a))\}_{n\geq 0}$, com $a > 1$, fazendo*

$$x_n(a) + y_n(a)\sqrt{d} = (a + \sqrt{d})^n.$$

Vamos evitar escrever explicitamente a dependência do parâmetro a, a menos que pelo contexto se faça necessário. Sendo assim, denotamos o par $(x_n(a), y_n(a))$ simplesmente como (x_n, y_n).

Lema 3.1.4. *O par (x_n, y_n) satisfaz a equação (1) para todo $n \geq 0$.*

Demonstração. Temos que $(x_0, y_0) = (1, 0)$ e $(x_1, y_1) = (a, 1)$ satisfazem (1). Suponha que (x_n, y_n) satisfaça (1) para algum $n \geq 0$, então, pelo Lema (3.1.2) temos que (x_{n+1}, y_{n+1}) também satisfaz (1). Logo, por indução, (x_n, y_n) satifaz (1) para todo $n \geq 0$. □

Lema 3.1.5. *Seja (x, y) uma solução de (1) com $x, y \in \mathbb{N}_0$. Então $(x, y) = (x_n, y_n)$ para algum $n \geq 0$.*

Demonstração. Primeiro notamos que se $x + y\sqrt{d} = 0$, então $x = y = 0$, mas o par $(0, 0)$ não satisfaz (1), logo $x + y\sqrt{d} \geq 1$. Como a sequência $\{(a + \sqrt{d})^n\}_{n\geq 0}$ é estritamente crescente, então para algum $n \geq 0$ temos que

$$(a + \sqrt{d})^n \leq x + y\sqrt{d} < (a + \sqrt{d})^{n+1}. \tag{4}$$

Se ocorrer a igualdade em (4) nada há o que provar, caso contrário temos que

$$x_n + y_n\sqrt{d} < x + y\sqrt{d} < (x_n + y_n\sqrt{d})(a + \sqrt{d}). \tag{5}$$

Mas, como $(x_n + y_n\sqrt{d})(x_n - y_n\sqrt{d}) = 1$ e $x_n + y_n\sqrt{d} > 0$, então $x_n - y_n\sqrt{d} > 0$. Assim, multiplicando (5) por $x_n - y_n\sqrt{d}$, obtemos

$$1 < (x + y\sqrt{d})(x_n - y_n\sqrt{d}) < a + \sqrt{d}.$$

Mas, isto contradiz o Lema (3.1.1) após aplicarmos o Lema (3.1.2). □

Lema 3.1.6. *Temos que $x_{m\pm n} = x_m x_n \pm d y_n y_m$ e $y_{m\pm n} = x_n y_m \pm x_m y_n$, para todos m, n cujos índices sejam não-negativos.*

Demonstração. Temos que

$$\begin{aligned} x_{m+n} + y_{m+n}\sqrt{d} &= (a + \sqrt{d})^{m+n} \\ &= (x_m + y_m\sqrt{d})(x_n + y_n\sqrt{d}) \\ &= (x_m x_n + d y_n y_m) + (x_n y_m + x_m y_n)\sqrt{d}. \end{aligned}$$

Portanto,

$$x_{m+n} = x_m x_n + d y_n y_m \quad \text{e} \quad y_{m+n} = x_n y_m + x_m y_n.$$

Também temos que

$$(x_{m-n} + y_{m-n}\sqrt{d})(x_n + y_n\sqrt{d}) = x_m + y_m\sqrt{d}$$

donde obtemos, multiplicando ambos os lados por $(x_n - y_n\sqrt{d})$, que

$$\begin{aligned} x_{m-n} + y_{m-n}\sqrt{d} &= (x_m + y_m\sqrt{d})(x_n - y_n\sqrt{d}) \\ &= (x_m x_n - d y_n y_m) + (x_n y_m - x_m y_n)\sqrt{d}. \end{aligned}$$

Portanto,

$$x_{m-n} = x_m x_n - d y_n y_m \quad \text{e} \quad y_{m-n} = x_n y_m - x_m y_n.$$

□

Lema 3.1.7. *Temos que* $y_{m\pm1} = a y_m \pm x_m$ *e* $x_{m\pm1} = a x_m \pm d y_m$.

Demonstração. Basta fazer $n = 1$ no Lema (3.1.6). □

Lema 3.1.8. *Temos que* $\text{mdc}(x_n, y_n) = 1$ *para todo* $n \geq 0$.

Demonstração. Seja $d = \text{mdc}(x_n, y_n)$, então $d|x_n$ e $d|y_n$ o que implica que $d|(x_n^2 - d y_n^2)$, ou seja, $d|1$ e daí $d = 1$. □

Lema 3.1.9. *Temos que* $y_n|y_{nk}$ *para todo* $n, k \geq 1$.

Demonstração. Para $k = 1$ a afirmação é imediata. Fixamos n. Por indução, suponha que $y_n|y_{nm}$ para algum $m \geq 1$. Usando a fórmula do Lema (3.1.6, temos que

$$y_{n(m+1)} = x_n y_{nm} + x_{nm} y_n.$$

Portanto, $y_n|y_{n(m+1)}$ e então $y_n|y_{nk}$ para todo $k \geq 1$. □

Lema 3.1.10. *Temos que* $y_n|y_m$ *se, e somente se,* $n|m$.

Demonstração. ($\Leftarrow$) Basta aplicar o Lema (3.1.9). ($\Rightarrow$) Suponha que $y_n|y_m$, mas $n \nmid m$. Então podemos escrever $m = nq + r$, $0 < r < n$. Logo, pelo Lema (3.1.6), obtemos

$$y_m = y_{nq+r} = x_r y_{nq} + x_{nq} y_r. \tag{6}$$

Pelo Lema (3.1.9) temos que $y_n|y_{nq}$ o que implica por (6) que $y_n|x_{nq} y_r$. Mas $\text{mdc}(y_n, x_{nq}) = 1$. De fato, se $d|y_n$ e $d|x_{nq}$, então pelo Lema (3.1.9) $d|y_{nq}$ o que pelo Lema (3.1.8) implica em $d = 1$. Assim, $y_n|y_r$. Mas, como $r < n$, pelo Lema (3.1.7), obtemos que $y_r < y_n$. Isto é uma contradição. Então $n|m$. □

Lema 3.1.11. *Temos a congruência* $y_{nk} \equiv kx_n^{k-1}y_n (\text{mod } y_n^3)$ *para todo,* $n, k \geq 1$.

Demonstração.

$$\begin{aligned} x_{nk} + y_{nk}\sqrt{d} &= (a + \sqrt{d})^{nk} \\ &= (x_n + y_n\sqrt{d})^k \\ &= \sum_{j=0}^{k} \binom{k}{j} x_n^{k-j} y_n^j d^{\frac{j}{2}}. \end{aligned}$$

Assim,

$$y_{nk} = \sum_{\substack{j=1 \\ j \text{ ímpar}}}^{k} \binom{k}{j} x_n^{k-j} y_n^j d^{\frac{j-1}{2}}.$$

Mas todos os termos desta expansão em que $j > 1$ são divisíveis por (y_n^3). □

Lema 3.1.12. *Temos que* $y_n^2 | y_{ny_n}$ *para todo* $n \geq 1$.

Demonstração. Basta fazer $k = y_n$ no Lema (3.1.11). □

Lema 3.1.13. *Se* $y_n^2 | y_m$, *então* $y_n | m$.

Demonstração. Pelo Lema (3.1.10), $n|m$. Seja $m = nk$. Usando o Lema (3.1.11), temos que $y_n^2 r = y_m = kx_n^{k-1}y_n + sy_n^3$, para certos $r, s \in \mathbb{Z}$, o que implica em $y_n(r - sy_n) = kx_n^{k-1}$ (já que $y_n \neq 0$) e obtemos que $y_n | kx_n^{k-1}$. Mas, pelo Lema (3.1.8), $\text{mdc}(x_n, y_n) = 1$. Então, $y_n | k$ e daí $y_n | m$. □

Lema 3.1.14. *Temos as duas equações de recorrência*

$$x_{n+1} = 2ax_n - x_{n-1} \quad \text{e} \quad y_{n+1} = 2ay_n - y_{n-1}$$

para todo $n \geq 1$.

Demonstração. Pelo Lema (3.1.7),

$$\begin{aligned} x_{n+1} &= ax_n + dy_n, \quad y_{n+1} = ay_n + x_n, \\ x_{n-1} &= ax_n - dy_n, \quad y_{n-1} = ay_n - x_n. \end{aligned}$$

Assim, obtemos $x_{n+1} + x_{n-1} = 2ax_n$ e $y_{n+1} + y_{n-1} = 2ay_n$. □

Estas duas equações de recorrência lineares e homogêneas de segunda ordem do Lema (3.1.14) podem ser utilizadas com os valores iniciais de cada uma $x_0 = 1$, $x_1 = a$ e $y_0 = 0$, $y_1 = 1$, respectivamente, para se determinar as soluções x_n e y_n explicitamente, bastando para isso resolver as equações características correspondentes e determinar coeficientes usando os valores iniciais.

Também podemos usá-las para estabelecer propriedades via indução, provando o caso base para $n = 0$ e $n = 1$ e em seguida inferindo que o resultado ocorre para $n + 1$ supondo-o verdadeiro para n e $n - 1$.

A seguir veremos alguns exemplos destas propriedades que podem ser deduzidas dessas equações de recorrência.

Lema 3.1.15. *Temos a congruência* $y_n \equiv n(\text{mod } a-1)$.

Demonstração. Para $n = 0, 1$ a afirmação é imediata. Por indução, como $a \equiv 1(\text{mod } a-1)$, do Lema (3.1.14) vem:

$$\begin{aligned} y_{n+1} &= 2ay_n - y_{n-1} \\ &\equiv 2n - (n-1) \\ &\equiv n+1(\text{mod } a-1). \end{aligned}$$

□

Lema 3.1.16. *Se* $a \equiv b(\text{mod } c)$, *então para todo* $n \geq 0$,

$$x_n(a) \equiv x_n(b)(\text{mod } c) \quad \text{e} \quad y_n(a) \equiv y_n(b)(\text{mod } c).$$

Demonstração. Para $n = 0, 1$ ambas as congruências são imediatas. Por indução, do Lema (3.1.14)

$$\begin{aligned} y_{n+1}(a) &= 2ay_n(a) - y_{n-1}(a) \\ &\equiv 2by_n(b) - y_{n-1}(b) \\ &\equiv y_{n+1}(b)(\text{mod } c). \end{aligned}$$

Analogamente prova-se a congruência para x_n no lugar de y_n. □

Lema 3.1.17. *Temos que* n *e* y_n *têm a mesma paridade.*

Demonstração. Do Lema (3.1.14), $y_{n+1} = 2ay_n - y_{n-1} \equiv y_{n-1}(\text{mod } 2)$. Assim, quando n é par, $y_n \equiv y_0 = 0(\text{mod } 2)$ e quando n é ímpar, $y_n \equiv y_1 = 1(\text{mod } 2)$. □

Lema 3.1.18. *Temos a congruência* $x_n - y_n(a-k) \equiv k^n(\text{mod } 2ak - k^2 - 1)$ *onde* $x_n = x_n(a)$, $y_n = y_n(a)$ *e* $k \in \mathbb{N}_1$.

Demonstração. Para $n = 0$, $x_0 - y_0(a-k) = 1$ e para $n = 1$, $x_1 - y_1(a-k) = k$ de sorte que para $n = 0, 1$ temos verificado que o Lema (3.1.18) ocorre. Usando o Lema (3.1.14), por indução vem

$$\begin{aligned} x_{n+1} - y_{n+1}(a-k) &= 2a(x_n - y_n(a-k)) - (x_{n-1} - y_{n-1}(a-k)) \\ &\equiv 2ak^n - k^{n-1} \\ &\equiv k^{n-1}(2ak-1) \\ &\equiv k^{n-1}k^2 \\ &\equiv k^{n+1}(\text{mod } 2ak - k^2 - 1). \end{aligned}$$

□

Lema 3.1.19. *Para todo* $n \geq 0$, $y_{n+1} > y_n \geq n$.

Demonstração. Pelo Lema (3.1.7), $y_{n+1} > y_n$. Como $y_0 = 0$, segue por indução que $y_n \geq n$ para todo $n \geq 0$. □

Lema 3.1.20. *Para todo* $n \geq 0$, $x_{n+1} > x_n \geq a^n$ *e* $x_n \leq (2a)^n$.

Demonstração. Pelos Lemas (3.1.7) e (3.1.14), $ax_n \leq x_{n+1} \leq 2ax_n$. Mas, $ax_n > x_n$ e $x_0 = 1$ de maneira que o resultado segue por indução.

□

A seguir algumas propriedades de periodicidade da sequência $\{x_n\}_{n\geq 0}$ serão obtidas.

Lema 3.1.21. *Temos a congruência $x_{2n\pm k} \equiv -x_k (\text{mod } x_n)$ para todos n, k cujos índices são não-negativos.*

Demonstração. Pelo Lema (3.1.6),

$$\begin{aligned} x_{2n\pm k} &= x_n x_{n\pm k} \pm d y_n y_{n\pm k} \\ &\equiv d y_n (y_n x_k \pm x_n y_k) \\ &\equiv d y_n^2 x_k (\text{mod } x_n) \end{aligned}$$

e sabendo que x_n e y_n satisfazem a equação de Pell, $dy_n^2 = x_n^2 - 1$, e então

$$\begin{aligned} x_{2n\pm k} &\equiv (x_n^2 - 1) x_k \\ &\equiv -x_k (\text{mod } x_n). \end{aligned}$$

□

Lema 3.1.22. *Temos a congruência $x_{4n\pm k} \equiv x_k (\text{mod } x_n)$ para todos n, k cujos índices são não-negativos.*

Demonstração. Pelo Lema (3.1.21),

$$x_{4n\pm k} \equiv -x_{2n\pm k} \equiv x_k (\text{mod } x_n).$$

□

Lema 3.1.23. *Seja $x_r \equiv x_s (\text{mod } x_n)$ com $0 \leq r \leq s \leq 2n$. Então, $r = s$, a menos que $a = 2$, $n = 1$, $r = 0$ e $s = 2$.*

Demonstração. Primeiro suponha que x_n seja ímpar e seja $q = \frac{x_n - 1}{2}$. Então, os números

$$-q, -q+1, -q+2, ..., -1, 0, 1, ..., q-2, q-1, q$$

formam um sistema completo de resíduos mutuamente incongruentes módulo x_n. Pelo Lema (3.1.20),

$$1 = x_0 < x_1 < \cdots < x_{n-1}.$$

Usando o Lema (3.1.7), $x_{n-1} \leq \frac{x_n}{a} \leq \frac{x_n}{2}$, assim, $x_{n-1} \leq q$. Também, pelo Lema (3.1.21), os números

$$x_{n+1}, x_{n+2}, ..., x_{2n-1}, x_{2n}$$

são congruentes módulo x_n, respectivamente, a:

$$-x_{n-1}, -x_{n-2}, \cdots, -x_1, -x_0 = -1.$$

Desta forma os números $x_0, x_1, x_2, ..., x_{2n}$ são mutuamente incongruentes módulo x_n. Isto conclui o resultado.

Agora suponha que x_n seja par e seja $q = \frac{x_n}{2}$. Neste caso, os números

$$-q+1, -q+2, \cdots, -1, 0, 1, \cdots, q-1, q \tag{7}$$

que formam um sistema completo de resíduos mutuamente incongruentes módulo x_n. Como acima, $x_{n-1} \leq q$. Assim, a sequência (7) não cobre (módulo x_n) a sequência $x_0, x_1, ..., x_{2n}$ somente quando $x_{n-1} = q = \frac{x_n}{2}$, de sorte que precisamos avaliar este caso. Se isto ocorre, $x_{n+1} \equiv -q (\text{mod } x_n)$, e daí poderíamos ter $r = n-1$, $s = n+1$ como possibilidade para contradizer o resultado. Pelo Lema (3.1.7),

$$x_n = ax_{n-1} + dy_{n-1}$$

de modo que $x_n = 2x_{n-1}$ implica que $a = 2$ e $y_{n-1} = 0$, ou seja, $n = 1$. Assim, o resultado falha se, e somente se, $a = 2$, $n = 1$, $r = 0$ e $s = 2$.

□

Lema 3.1.24. *Seja $x_r \equiv x_s (\text{mod } x_n)$, $0 < r \leq n$, $0 \leq s < 4n$, então ou $r = s$ ou $s = 4n - r$.*

Demonstração. Suponha que $s \leq 2n$. Então, pelo Lema (3.1.23), $r = s$, a menos que a exceção discutida ocorra (neste caso deveríamos ter $r = 0$, se tivéssemos $r \leq s$, que é absurdo, pois $r > 0$ por hipótese). Então, $r > s$ e daí $s = 0$ (s troca de lugar com r no Lema (3.1.23)). Mas, então

$$r = 2 > 1 = n$$

gera uma contradição. Para o outro caso, $s > 2n$, seja $t = 4n - s$ de modo que $0 < t < 2n$. Pelo Lema (3.1.22), $x_t \equiv x_s (\text{mod } x_n)$. Novamente, pelo Lema (3.1.23), $t = r$ a menos que o caso excepcional ocorra. Mas, este último está fora de questão, pois $r, s > 0$. □

Lema 3.1.25. *Se $0 < r \leq n$ e $x_r \equiv x_s (\text{mod } x_n)$, então $s \equiv \pm r (\text{mod } 4n)$.*

Demonstração. Escrevemos $s = 4nq + t$ com $0 \leq t < 4n$. Pelo Lema (3.1.22),

$$x_r \equiv x_s \equiv x_t (\text{mod } x_n).$$

Pelo Lema (3.1.24) $r = t$ ou $r = 4n - t$. Assim, $s \equiv t \equiv \pm r (\text{mod } 4n)$. □

3.2 A Função Exponencial é Diofantina

Nosso objetivo nesta seção é mostrar que a função exponencial $h : \mathbb{N}_1^2 \to \mathbb{N}_1$ definida por $h(n, k) = n^k$ é Diofantina. Para isto provaremos o Teorema (3.2.1) que nos auxilará nesta tarefa. A demonstração do Teorema (3.2.1) utilizará diversos dos Lemas de Pell vistos anteriormente.

Teorema 3.2.1. *Sejam $a, b \in \mathbb{N}_2$ e $p, q, r, s, t, u, k, \overline{A}, \overline{B}, \overline{C}, \overline{D}, \overline{E}, \overline{F} \in \mathbb{N}_1$. Considere o seguinte sistema de equações Diofantinas*

$$\begin{cases} p^2 - (a^2-1)q^2 = 1 & \text{(I)} \\ r^2 - (a^2-1)s^2 = 1 & \text{(II)} \\ t^2 - (b^2-1)u^2 = 1 & \text{(III)} \\ s = \overline{A}q^2 & \text{(IV)} \\ b = a + \overline{C}r = 1 + 4\overline{B}q & \text{(V)} \\ t = p + \overline{D}r & \text{(VI)} \\ u = k + 4(\overline{E}-1)q & \text{(VII)} \\ q = k + (\overline{F}-1) & \text{(VIII)} \end{cases}$$

Então, dados $a > 1, p, k$ o sistema (I)-(VIII) tem solução se, e somente se, $p = x_k(a)$, onde $(x_k(a), y_k(a))$ é definido em (3.1.3).

Demonstração. ($\Leftarrow$) Suponha que o sistema de (I)-(VIII) tenha uma solução. Por (V), $b > a > 1$. Então, (I), (II) e (III) implicam pelo Lema (3.1.5) que existem $l, m, n \in \mathbb{N}_1$ tais que

$$(p, q) = (x_l(a), y_l(a)), \quad (r, s) = (x_m(a), y_m(a)), \quad (t, u) = (x_n(b), y_n(b)).$$

Por (IV), $q \leq s$ de modo que $l \leq m$, pois $\{y_n\}_{n\geq 0}$ é crescente. Por (V) e (VI) temos as congruências

$$b \equiv a (\text{mod } x_m(a)); \quad x_n(b) \equiv x_l(a) (\text{mod } x_m(a))$$

e pelo Lema (3.1.16) temos que

$$x_n(b) \equiv x_n(a) (\text{mod } x_m(a)).$$

Assim,

$$x_l(a) \equiv x_n(a) (\text{mod } x_m(a)).$$

Pelo Lema (3.1.25),

$$n \equiv \pm l (\text{mod } 4m). \tag{i}$$

A equação (IV) implica que

$$y_l^2(a) | y_m(a)$$

de modo que pelo Lema (3.1.13)

$$y_l(a) | m$$

e (i) implica que

$$n \equiv \pm l (\text{mod } 4y_l(a)). \tag{ii}$$

Pela equação (V) temos que

$$b \equiv 1(\text{mod } 4y_l(a))$$

de modo que pelo Lema (3.1.15) obtemos

$$y_n(b) \equiv n(\text{mod } 4y_l(a)). \tag{iii}$$

Pela equação (VII) temos que

$$y_n(b) \equiv k(\text{mod } 4y_l(a)). \tag{iv}$$

Combinando (ii), (iii), (iv) vem

$$k \equiv \pm l(\text{mod } 4y_l(a)) \tag{v}$$

A equação (VIII) implica em

$$k \leq y_l(a)$$

e pelo Lema (3.1.19)

$$l \leq y_l(a).$$

Como os números

$$-2q+1, -2q+2, ..., -1, 0, 1, ..., 2q$$

formam um sistema completo de resíduos mutuamente incongruentes módulo $4q = 4y_l(a)$, estas desigualdades e (v) implicam que $k = l$, pois como $0 < k, l \leq y_l(a)$ a outra possibilidade seria $k \equiv -l(\text{mod } 4q)$ e daí $k + l \equiv 0(\text{mod } 4q)$ que é absurdo. Então,

$$p = x_l(a) = x_k(a).$$

($\Rightarrow$) Reciprocamente, seja $p = x_k(a)$. Fazemos $q = y_k(a)$ de modo que (I) ocorra. Seja $j = 2ky_k(a)$ e sejam $r = x_j(a)$ e $s = y_j(a)$. Assim, escolhemos uma solução para (II) . Pelos Lemas (3.1.10) e (3.1.12) $q^2|s$. Então podemos escolher $\overline{A}$ satisfazendo (IV). Além disso, pelo Lema (3.1.17), s é par de modo que r é ímpar. Pelo Lema (3.1.8), $\text{mdc}(r, s) = 1$. Então $\text{mdc}(r, 4q) = 1$. (Se $\overline{B}$ é um divisor primo de r e de $4q$, então $\overline{B}|q$ pois r é ímpar e então $\overline{B}|s$ visto que $q|s$).

Pelo Teorema Chinês dos Restos (2.1.5) podemos encontrar $\overline{M} \in \mathbb{N}_1$ tal que

$$\begin{cases} \overline{M} \equiv 1(\text{mod } 4q) \\ \overline{M} \equiv a(\text{mod } r). \end{cases}$$

Como $\overline{M} + 4\overline{N}rq$, para algum $\overline{N} \in \mathbb{N}_1$, também satisfará estas congruências, $b, \overline{B}, \overline{C}$ satisfazendo (V) podem ser encontrados. A equação (III) é satisfeita pondo-se $t = x_k(b)$, $u = y_k(b)$. Como $b > a$, $t = x_k(b) > x_k(a) = p$. Pelo Lema (3.1.16) e (V), $t \equiv p(\text{mod } r)$. Assim $\overline{D}$ pode ser escolhido de modo que (VI) seja satisfeita. Pelo Lema (3.1.19), $u \geq k$ e pelo Lema (3.1.15), $u \equiv k(\text{mod } b-1)$ e então usando (V), $u \equiv k(\text{mod } 4q)$. Assim, $\overline{E}$ pode ser escolhido de modo que a equação (VII) seja satisfeita. Pelo Lema (3.1.19) novamente, $q \geq k$, assim (VIII) pode ser satisfeita pondo-se $\overline{F} = q - k + 1$. □

Corolário 3.2.2. *A função $g : \mathbb{N}_1^2 \to \mathbb{N}_1$ definida por $g(c,k) = x_k(c+1)$ é Diofantina.*

Demonstração. Basta adicionarmos ao sistema (I)-(VIII) a equação

$$a = c + 1. \tag{IX}$$

Pelo Teorema (3.2.1), o sistema (I)-(IX) tem uma solução se, e somente se, $p = x_k(a) = g(c,k)$. Assim, é fácil escrever uma definição Diofantina para g somando-se os quadrados de 9 polinômios.

□

Utilizaremos mais dois Lemas para mostrar que a função exponencial é Diofantina.

Lema 3.2.3. *Se $a > q^k$, então $2aq - q^2 - 1 > q^k$ onde $a, q, k \in \mathbb{N}_1$.*

Demonstração. Seja $f : \mathbb{R} \to \mathbb{R}$ definida por $f(x) = 2ax - x^2 - 1$. Então, como $a \geq 2$, $f(1) = 2a - 2 = a + (a-2) \geq a$. Para $1 \leq x < a$, $f'(x) = 2a - 2x > 0$, então f é estritamente crescente para $1 \leq x < a$ de modo que $f(x) \geq f(1) \geq a$. Logo, para $a > q^k \geq q \geq 1$ temos que $2aq - q^2 - 1 \geq a > q^k$

□

Sejam $n, m, \overline{G}, \overline{H}, \overline{I}, \overline{J}, \overline{K}, \overline{L} \in \mathbb{N}_1$. Adicionamos ao sistema (I)-(VIII) as equações

$$\begin{cases} (p - q(a-n) - m)^2 = (\overline{G} - 1)^2(2an - n^2 - 1)^2 & \text{(X)} \\ m + \overline{H} = 2an - n^2 - 1 & \text{(XI)} \\ \overline{I} = n + \overline{K} = k + \overline{L} & \text{(XII)} \\ a^2 - (\overline{I}^2 - 1)(\overline{I} - 1)^2\overline{J}^2 = 1 & \text{(XIII)} \end{cases}$$

Lema 3.2.4. *Temos que $m = n^k$ se, e somente se, as equações (I)-(VIII) e (X)-(XIII) têm uma solução nas varáveis restantes.*

Demonstração. ($\Leftarrow$) Suponha que o sistema (I)-(VIII) e (X)-(XIII) tenha solução. Por (XII), $\overline{I} > 1$. Então $(\overline{I} - 1)\overline{J} > 0$ e assim por (XIII) $a > 1$. Assim, pelo Teorema (3.2.1) segue que $p = x_k(a)$ e $q = y_k(a)$. Por (X) e pelo Lema (3.1.18),

$$m \equiv n^k (\text{mod } 2an - n^2 - 1).$$

A equação (XII) implica que

$$k, n < \overline{I}.$$

Por (XIII), usando o Lema (3.1.5), para algum $j \in \mathbb{N}_0$ temos que $a = x_j(\overline{I})$ e $(\overline{I}-1)\overline{J} = y_j(\overline{I})$. Pelo Lema (3.1.15),

$$j \equiv 0 (\text{mod } \overline{I} - 1)$$

de modo que $j \geq \overline{I} - 1$. Assim, pelo Lema (3.1.20),

$$a \geq \overline{I}^{\overline{I}-1} > n^k.$$

Agora por (XI), $m < 2an - n^2 - 1$, e pelo Lema (3.2.3)

$$n^k < 2an - n^2 - 1.$$

Como m e n^k são congruentes e ambos menores que o módulo $2an - n^2 - 1$, a única possibilidade é serem iguais.

($\Rightarrow$) Reciprocamente, suponha que $m = n^k$. Devemos encontrar soluções para o sistema (I)-(VIII) e (X)-(XIII). Escolhemos qualquer número $\overline{I}$ tal que $\overline{I} > n$ e $\overline{I} > k$. Pomos $a = x_{\overline{I}-1}(\overline{I})$ de modo que $a > 1$. Pelo Lema (3.1.15),

$$y_{\overline{I}-1}(\overline{I}) \equiv 0 (\text{mod } \overline{I} - 1).$$

De modo que podemos escrever

$$y_{\overline{I}-1}(\overline{I}) = \overline{J}(\overline{I} - 1)$$

assim,(XIII) é satisfeita. A equação (XII) pode ser satisfeita pondo-se

$$\overline{K} = \overline{I} - n, \quad \overline{L} = \overline{I} - k.$$

Como antes, $a > n^k$ de modo que novamente pelo Lema (3.2.3),

$$m = n^k < 2an - n^2 - 1$$

e (XI) pode ser satisfeita. Pondo-se $p = x_k(a)$ e $q = y_k(a)$ o Lema (3.1.18) permite-nos definir $\overline{G}$ tal que

$$p - q(a - n) - m = \pm(\overline{G} - 1)(2an - n^2 - 1)$$

de modo que (X) é satisfeito. Finalmente, o sistema (I)-(VIII) pode ser satisfeito pelo Teorema (3.2.1). □

Teorema 3.2.5. *A função exponencial $h(n, k) = n^k$ é Diofantina.*

Demonstração. Segue imediatamente do Lema (3.2.4) e do fato das equações de (I)-(VIII) e (X)-(XIII) serem polinomiais. □

Capítulo 4

Técnicas Principais

4.1 Predicados Diofantinos

Tendo visto que a função exponencial é Diofantina estamos guarnecidos e prontos para provar que novas funções são Diofantinas usando como recurso este fato e a liguagem dos predicados Diofantinos.

Como exemplo considere a função

$$\begin{aligned} H : \mathbb{N}_1^3 &\to \mathbb{N}_1 \\ (a, b, c) &\mapsto a^{b^c}. \end{aligned}$$

Podemos concluir que H é Diofantina, pois

$$r = a^{b^c} \Leftrightarrow (\exists\, s \in \mathbb{N}_1)(r = a^s \ \wedge \ s = b^c)$$

onde $\wedge$ é o símbolo usado no lugar do conectivo lógico "e". Pelo teorema (3.2.5), existe um polinômio P tal que

$$\begin{aligned} r = a^s &\Leftrightarrow (\exists\, t_1, t_2..., t_n \in \mathbb{N}_1)[P(r, a, s, t_1, t_2..., t_n) = 0], \\ s = b^c &\Leftrightarrow (\exists\, u_1, u_2..., u_n \in \mathbb{N}_1)[P(s, b, c, u_1, u_2..., u_n) = 0]. \end{aligned}$$

Então,

$$r = a^{b^c} \Leftrightarrow (\exists\, s, t_1, t_2..., t_n, u_1, u_2..., u_n \in \mathbb{N}_1)[P^2(r, a, s, t_1, t_2..., t_n) + P^2(s, b, c, u_1, u_2..., u_n) = 0]$$

Notamos que este procedimento não se aplica somente a este caso particular, mas pode ser estendido de modo que se já sabemos que certas expressões são diofantas podemos combiná-las usando operações lógicas "$\wedge$" e "$\exists$" livremente que a expressão resultante irá definir um novo conjunto Diofantino. (Tais expressões são chamadas de predicados Diofantinos). Nesta linguagem é também permitido o uso do operador lógico "$\vee$" para "ou", pois

$$\begin{gathered} (\exists\, t_1, t_2, ..., t_n \in \mathbb{N}_1)[P_1 = 0] \ \vee \ (\exists \in \mathbb{N}_1\, u_1, u_2, ..., u_n)[P_2 = 0] \\ \Leftrightarrow \\ (\exists \in \mathbb{N}_1\, t_1, t_2, ..., t_n, u_1, u_2, ..., u_n)[P_1 \cdot P_2 = 0]. \end{gathered}$$

Lema 4.1.1. *Para $0 < k \leq n, n \in \mathbb{N}_1$ e $m > 2^n$, temos que*

$$\left\lfloor \frac{(m+1)^n}{m^k} \right\rfloor = \sum_{i=k}^{n} \binom{n}{i} m^{i-k}$$

onde $\lfloor x \rfloor$ é a função piso, ou seja, $\lfloor x \rfloor$ é o único inteiro tal que $\lfloor x \rfloor \leq x < \lfloor x \rfloor + 1$.

Demonstração. Pelo Teorema Binomial temos que

$$\frac{(m+1)^n}{m^k} = \sum_{i=0}^{n} \binom{n}{i} m^{i-k} = A + B$$

onde

$$A = \sum_{i=0}^{k-1} \binom{n}{i} m^{i-k} \quad \text{e} \quad B = \sum_{i=k}^{n} \binom{n}{i} m^{i-k}.$$

Então, B é inteiro e

$$0 < A \leq \frac{1}{m} \sum_{i=0}^{k-1} \binom{n}{i} < \frac{1}{m} \sum_{i=0}^{n} \binom{n}{i} = \frac{1}{m}(1+1)^n < 1.$$

Assim,

$$B \leq \frac{(m+1)^n}{m^k} < B + 1$$

e temos o resultado.

□

Lema 4.1.2. *Para $0 < k \leq n, n \in \mathbb{N}_1$ e $m > 2^n$,*

$$\left\lfloor \frac{(m+1)^n}{m^k} \right\rfloor \equiv \binom{n}{k} (\text{mod } m)$$

Demonstração. No Lema (4.1.1) todos os termos do somatório para os quais $i > k$ são divisíveis por m.

□

Teorema 4.1.3. *A função $f(n,k) = \binom{n}{k}$ com $n, k \in \mathbb{N}_0$ é Diofantina.*

Demonstração. Como

$$\binom{n}{k} \leq \sum_{i=0}^{n} \binom{n}{i} = 2^n < m,$$

se $m > 2^n$, então o Lema (4.1.2) determina $\binom{n}{k}$ como o único inteiro positivo congruente a $\left\lfloor \frac{(m+1)^n}{m^k} \right\rfloor$ módulo m e menor do que m. Portanto,

$$s = \binom{n}{k}$$

$$\Leftrightarrow$$

$$(\exists\, m, a, b \in \mathbb{N}_0)(a = 2^n \wedge m > a \wedge b = \left\lfloor \frac{(m+1)^n}{m^k} \right\rfloor \wedge s \equiv b (\text{mod } m) \wedge s < m)$$

Para concluir que $\binom{n}{k}$ é Diofantina, é suficiente notar que as expressões acima separadas por $\wedge$ são predicados Diofantinos, $a = 2^n$ é Diofantina pelo Teorema (3.2.1). A desiguladade $m > a$ é também Diofantina pois $m > a \Leftrightarrow (\exists\, r \in \mathbb{N}_1)(m = a + r)$. Também,

$$s \equiv b(\text{mod } a) \wedge s < m \quad \Leftrightarrow \quad (\exists\, p, q \in \mathbb{N}_1)(b = s + (p-1)m \wedge m = s + q)$$

Finalmente,

$$b = \left\lfloor \frac{(m+1)^n}{m^k} \right\rfloor$$
$$\Leftrightarrow$$
$$(\exists\, t, u, v \in \mathbb{N}_1)(v = m + 1 \wedge t = v^n \wedge u = m^k \wedge b \leq \frac{t}{u} < b + 1)$$
$$\Leftrightarrow$$
$$(\exists\, t, u, v \in \mathbb{N}_1)(v = m + 1 \wedge t = v^n \wedge u = m^k \wedge bu \leq t < (m+1)u).$$

□

Lema 4.1.4. *Se $m > (2n)^{n+1}$ com $m, n \in \mathbb{N}_0$ então*

$$n! = \left\lfloor \frac{m^n}{\binom{m}{n}} \right\rfloor$$

Demonstração. Seja $m > (2n)^{n+1}$. Então,

$$\begin{aligned} \frac{m^n}{\binom{m}{n}} &= \frac{m^n n!}{m(m-1)\cdots(m-n+1)} \\ &= \frac{n!}{\left(1 - \frac{1}{m}\right)\cdots\left(1 - \frac{n-1}{m}\right)} \\ &= \frac{n!}{\left(1 - \frac{n}{m}\right)^n} \end{aligned}$$

Mas,

$$\begin{aligned} \frac{1}{1 - \frac{n}{m}} &= 1 + \frac{n}{m} + \left(\frac{n}{m}\right)^2 + \cdots \\ &= 1 + \frac{n}{m}\left(1 + \frac{n}{m} + \left(\frac{n}{m}\right)^2 + \cdots\right) \\ &< 1 + \frac{n}{m}\left(1 + \frac{1}{2} + \frac{1}{4} + \cdots\right) \\ &= 1 + \frac{2n}{m} \end{aligned}$$

e

$$\begin{aligned}\left(1+\frac{2n}{m}\right)^n &= \sum_{j=0}^{n}\binom{n}{j}\left(\frac{2n}{m}\right)^j \\ &< 1+\frac{2n}{m}\sum_{j=1}^{n}\binom{n}{j} \\ &< 1+\frac{2n}{m}\cdot 2^n.\end{aligned}$$

Assim,

$$\begin{aligned}\frac{m^n}{\binom{m}{n}} &< n!+\frac{2n}{m}n!\cdot 2^n \\ &\leq n!+\frac{2^{n+1}n^{n+1}}{m} \\ &< n!+1.\end{aligned}$$

□

Teorema 4.1.5. *A função $f(n)=n!$ com $n\in\mathbb{N}_1$ é Diofantina.*

Demonstração.

$$m=n!$$
$$\Leftrightarrow$$
$$(\exists\, r,s,t,u,v\in\mathbb{N}_0)[s=2x+1\ \wedge\ t=x+1\ \wedge\ r=s^t\ \wedge\ u=r^n\ \wedge\ v=\binom{r}{n}\ \wedge\ mv\leq u<(m+1)v]$$

□

Lema 4.1.6. *Seja $bq\equiv a(\mathrm{mod}\ m)$ com $a,b,q,m\in\mathbb{N}_1$. Então,*

$$\prod_{k=1}^{n}(a+bk)\equiv b^n n!\binom{q+n}{n}(\mathrm{mod}\ m).$$

Demonstração.

$$\begin{aligned}b^n n!\binom{q+n}{n} &= b^n(q+n)(q+n-1)\cdots(q+1) \\ &= (bq+nb)(bq+(n-1)b)\cdots(bq+b) \\ &\equiv (a+nb)(a+(n-1)b)\cdots(a+b)(\mathrm{mod}\ m)\end{aligned}$$

□

Teorema 4.1.7. *A função $h(a,b,n)=\prod_{k=1}^{n}(a+bk)$ com $a,b,n\in\mathbb{N}_1$ é Diofantina.*

Demonstração. No Lema (4.1.6) escolhemos $m = b(a+bn)^n + 1$. Então, $\text{mdc}(m, b) = 1$ e $m > \prod_{k=1}^{n}(a+bk)$. Daí a congruência $bq \equiv a(\text{mod } m)$ tem solução para q e então $\prod_{k=1}^{n}(a+bk)$ é determinado como o único número o qual é congruente módulo m a $b^n n!\binom{q+n}{n}$ e é também menor que m. Ou seja,

$$\begin{aligned} h &= \prod_{k=1}^{n}(a+bk) \\ &\Leftrightarrow \\ (\exists\, p,q,r,s,t,u,v,c,d \in \mathbb{N}_0)[&r = a+bn \wedge s = r^n \wedge m = bs+1 \\ &\wedge\ bq = a+mt \wedge u = b^n \wedge v = n! \wedge v = n! \\ &\wedge\ z < m \wedge c = q+n \wedge d = \binom{c}{n} \wedge h + mp = uvd]. \end{aligned}$$

Usando o Teorema (3.2.5) para a função exponencial e os Teoremas (4.1.5) para $v = n!$ e (4.1.3) para $x = \binom{c}{n}$, obtemos o resultado.

□

4.1.1 Quantificadores Limitados

A linguagem dos predicados Diofantinos permite o uso dos operadores lógicos $\wedge$, $\vee$ e $\exists$. Outras operações usadas por lógicos são:

$\neg$	para "não"
$(\forall n)$	para "para todo n"
$\rightarrow$	para "se, então $\cdots$"

Entretanto, como ficará claro mais tarde, o uso de qualquer destas operações podem levar a expressões que definem conjuntos que não são Diofantinos. Existem também os quantificadores existenciais limitados:

"$(\exists\, k)_{\leq n} \cdots$" que significa "$(\exists\, k)(k \leq n \wedge \cdots)$"

e os quantificadores universais limitados:

"$(\forall\, k)_{\leq n} \cdots$" que significa "$(\forall\, k)(k > n \vee \cdots)$".

Consequentemente, estas operações podem ser vinculadas à linguagem dos predicados Diofantinos, ou seja, os conjuntos definidos pelas expressões desta linguagem ampliada ainda serão Diofantinos.

Teorema 4.1.8. *Se P é um polinômio,*

$$A = \{(b, a_1, a_2, ..., a_n) | (\exists\, c)_{\leq b}(\exists\, b_1, b_2, ..., b_m)[P(b, c, a_1, a_2, ..., a_n, b_1, b_2, ..., b_m) = 0]\}$$

e

$$B = \{(b, a_1, a_2, ..., a_n) | (\forall\, c)_{\leq b} (\exists\, b_1, b_2, ..., b_m)[P(b, c, a_1, a_2, ..., a_n, b_1, b_2, ..., b_m) = 0]\}$$

então A e B são Diofantinos.

Vamos demonstrar separadamente para os conjuntos A e B. Primeiro provamos para A:

Demonstração. O fato de A ser Diofantino é imediato. Com efeito,

$$(b, a_1, a_2, ..., a_n) \in A \Leftrightarrow (\exists\, c, b_1, b_2, ..., b_m)(c \leq b \wedge P = 0).$$

□

A prova para o conjunto B é mais intrincada. Precisamos de um Lema antes.

Lema 4.1.9. *Temos a equivalência*

$$(\forall\, k)_{\leq b} (\exists\, b_1, b_2, ..., b_m)[P(b,k, a_1, a_2, ..., a_n, b_1, b_2, ..., b_m) = 0]$$
$$\Leftrightarrow$$
$$(\exists\, u)(\forall\, k)_{\leq b} (\exists\, b_1, b_2, ..., b_m)_{\leq u} [P(b, k, a_1, a_2, ..., a_n, b_1, b_2, ..., b_m) = 0].$$

Demonstração. ($\Leftarrow$) É imediato. ($\Rightarrow$) Para a recíproca, suponha que o lado esquerdo seja verdadeiro dados $b, a_1, a_2, ..., a_n$. Então, para cada $k = 1, 2, ..., b$ existem números definidos $b_1^{(k)}, b_2^{(k)}, ..., b_m^{(k)}$ para os quais:

$$P(b, k, a_1, a_2, ..., a_n, b_1^{(k)}, b_2^{(k)}, ..., b_m^{(k)}) = 0.$$

Fazendo u ser o máximo dos mb números

$$\{b_j^{(k)} | j = 1, ..., m; k = 1, 2, ..., b\},$$

segue que o lado direito da equivalência também é verdadeiro.

□

Lema 4.1.10. *Seja $Q(b, u, a_1, a_2, ..., a_n)$ um polinômio com as propriedades*

$$Q(b, u, a_1, a_2, ..., a_n) > u, \quad \text{(i)}$$
$$Q(b, u, a_1, a_2, ..., a_n) > b, \quad \text{(ii)}$$

$k \leq b$ e $b_1, b_2, ..., b_m \leq u$ implicam que

$$|P(b, k, a_1, a_2, ..., a_n, b_1, b_2, ..., b_m)| \leq Q(b, u, a_1, a_2, ..., a_n). \quad \text{(iii)}$$

Então,

$$(\forall\, k)_{\leq b}(\exists\, b_1, b_2, ..., b_m)_{\leq u}[P(b,k,a_1,a_2,...,a_n,b_1,b_2,...,b_m) = 0]$$
$$\Leftrightarrow$$
$$(\exists\, c, t, c_1, c_2, ..., c_m)[1+ct = \prod_{k=1}^{b}(1+kt)$$
$$\wedge\ t = Q(b,u,a_1,a_2,...,a_n)! \wedge \prod_{j=1}^{u}(c_1 - j) \equiv 0(\text{mod } 1+ct)$$
$$\wedge\ \cdots \wedge \prod_{j=1}^{u}(c_m - j) \equiv 0(\text{mod } 1+ct)$$
$$\wedge\ P(b,c,a_1,a_2,...,a_n,c_1,c_2,...,c_m) \equiv 0(\text{mod } 1+ct)].$$

O ponto deste Lema é que enquanto o lado direito da equivalência parece o mais complicado dos dois, na verdade é livre de quantificadores universais limitados.

Demonstração. ($\Leftarrow$) Para cada $k = 1, 2, ..., b$, seja p_k um fator primo de $1+kt$. Seja $b_i^{(k)}$ o resto quando a_i é dividido por p_k $(k = 1, 2, ..., b; i = 1, 2, ..., m)$. Segue que para cada k, i temos

$$1 \leq b_i^{(k)} \leq u \tag{a}$$
$$P(b,k,a_1,a_2,...,a_n,b_1^{(k)},b_2^{(k)},...,b_m^{(k)}) = 0 \tag{b}$$

Para demonstrar (a), note que $p_k|1+kt$, $1+kt|1+ct$ e $1+ct|\prod_{j=1}^{u}(c_i-j)$ Ou seja, $p_k|\prod_{j=1}^{u}(c_i-j)$. Como p_k é primo, $p_k|c_i - j$ para algum $j = 1, 2, ..., u$. Isto é

$$j \equiv c_i \equiv b_i^{(k)}(\text{mod } p_k).$$

Como $t = Q(b,u,a_1,a_2,...,a_n)!$, (ii) implica que todo divisor de $1+kt$ deve ser maior que $Q(b,u,a_1,a_2,...,a_n)$. Assim, $p_k > Q(b,u,a_1,a_2,...,a_n)$ e por (i) $p_k > u$. Então, $j \leq u < p_k$. Como $b_i^{(k)}$ é o resto quando a_i é dividido por p_k temos também que $b_i^{(k)} < p_k$. Assim,

$$b_i^{(k)} = j$$

Para demonstrar (b), primeiro notamos que

$$1 + ct \equiv 1 + kt \equiv 0(\text{mod } p_k).$$

Então

$$k + kct \equiv c + kct(\text{mod } p_k),$$

isto é, $k \equiv c(\text{mod } p_k)$. Já temos obtido

$$b_i^{(k)} \equiv c_i(\text{mod } p_k).$$

Desta forma,

$$\begin{aligned} P(b,k,a_1,a_2,...,a_n,b_1^{(k)},b_2^{(k)},...,b_m^{(k)}) &\equiv P(b,c,a_1,a_2,...,a_n,c_1,c_2,...,c_m). \\ &\equiv 0(\text{mod } p_k) \end{aligned}$$

Finalmente

$$|P(b,k,a_1,a_2,...,a_n,b_1^{(k)},b_2^{(k)},...,b_m^{(k)})|\leq Q(b,u,a_1,a_2,...,a_n) < p_k.$$

Isto prova (b) e completa a prova.
($\Rightarrow$) Seja

$$P(b,k,a_1,a_2,...,a_n,b_1^{(k)},b_2^{(k)},...,b_m^{(k)}) = 0,$$

para cada $k = 1,2,...,t$ onde cada $b_j^{(k)} < u$. Fazemos $t = Q(b,u,a_1,a_2,...,a_n)!$ e como $\prod_{k=1}^{b}(1+kt) \equiv 1(\text{mod } t)$ podemos encontrar c tal que

$$1 + ct = \prod_{k=1}^{b}(1+kt).$$

Agora, afirmamos que para $1 \leq k < l \leq b$,

$$(1+kt, 1+lt) = 1.$$

Com efeito, seja p tal que $p|1+kt$ e $p|1+lt$. Então, $p|l-k$ de modo que $p < b$. Mas, como $Q(b,u,a_1,a_2,...,a_n) > b$ temos que $p|t$ o que é absurdo. Desta forma os números $1+kt$ formam um sequência admissível de módulos e pelo Teorema Chinês dos Restos (2.1.5) podemos obter para cada $i, 1 \leq i \leq m$, um número c_i tal que

$$c_i \equiv b_i^{(k)}(\text{mod } 1+kt), \quad k = 1,2,...,b.$$

Como acima, $k \equiv c(\text{mod } 1+kt)$. Assim,

$$P(b,c,a_1,a_2,...,a_n,c_1,c_2,...,c_n) \equiv P(b,k,a_1,a_2,...,a_n,b_1^{(k)},b_2^{(k)},...,b_m^{(k)}) \equiv 0(\text{mod } 1+kt),$$

Como os números $1+kt$'s são dois a dois primos entre si e cada um divide

$$P(b,c,a_1,a_2,...,a_n,c_1,c_2,...,c_m)$$

temos que o mesmo ocorre com o seu produto. Ou seja,

$$P(b,c,a_1,a_2,...,a_n,c_1,c_2,...,c_m) \equiv 0(\text{mod } 1+ct).$$

Finalmente,

$$c_i \equiv b_i^{(k)}(\text{mod } 1+kt),$$

ou seja,

$$1+kt|c_i - b_i^{(k)}.$$

Como $1 \leq b_i^{(k)} \leq u$,

$$1+kt|\prod_{j=1}^{u}(c_i - j).$$

E novamente como os $1+kt$'s são dois a dois primos entre si,

$$1+ct|\prod_{j=1}^{u}(c_i - j).$$

□

Agora é fácil completar a prova do Teorema (4.1.8) para o conjunto B usando os Lemas (4.1.9) e (4.1.10):

Demonstração. Primeiro encontramos um polinômio Q satisfazendo (i), (ii) e (iii). Podemos escrever

$$P(b,k,a_1,a_2,...,a_n,b_1,b_2,...,b_m)=\sum_{r=1}^{N}t_r$$

onde cada t_r tem a forma

$$t_r=cb^ak^da_1^{q_1}a_2^{q_2}\cdots a_n^{q_n}b_1^{s_1}b_2^{s_2}\cdots b_m^{s_m}$$

para $c\in\mathbb{Z}$. Seja, $u_r=|c|b^{a+d}a_1^{q_1}a_2^{q_2}\cdots a_n^{q_n}u^{s_1+s_2+\cdots+s_m}$ e seja

$$Q(b,u,a_1,a_2,...,a_n)=u+b+\sum_{r=1}^{N}u_r.$$

Então (i), (ii) e (iii) ocorrem. Desta forma:

$$\begin{aligned}
&(\forall\, k)_{\leq r}(\exists\, r_1,r_2,...,r_m)[P(r,k,a_1,a_2,...,a_n,r_1,r_2,...,r_m)=0])\\
&\qquad\qquad\Leftrightarrow\\
&(\exists\, u,c,t,c_1,c_2,...,c_m)[1+ct=\prod_{k=1}^{r}(1+kt)\\
&\qquad\qquad\wedge\ t=Q(r,u,a_1,a_2,...,a_n)!\wedge 1+ct|\prod_{j=1}^{u}(c_i-j)\\
&\qquad\qquad\wedge\ \cdots\wedge\ 1+ct|\prod_{j=1}^{u}(c_m-j)\\
&\qquad\qquad\wedge\ P(r,c,a_1,a_2,...a_n,c_1,c_2,...,c_m)\equiv 0(\mathrm{mod}\ 1+ct)]\\
&\qquad\qquad\Leftrightarrow\\
(\exists\, u,c,t,c_1,c_2,...,c_n,e,f,g_1,g_2,...g_m,h_1,&h_2,...,h_n,l)[e=1+ct\ \wedge\ e=\prod_{k=1}^{r}(1+kt)\\
&\qquad\qquad\wedge\ f=Q(r,u,c_1,c_2,...,c_n)\ \wedge t=f!\\
&\qquad\qquad\wedge\ g_1=c_1-u-1\ \wedge\ g_2=c_2-u-1\ \wedge\ \cdots\wedge\ g_m=c_m-u-1\\
&\qquad\qquad\wedge\ h_1=\prod_{k=1}^{u}(g_1+k)\ \wedge\ h_2=\prod_{k=1}^{u}(g_2+k)\\
&\qquad\qquad\wedge\ \cdots\wedge\ h_m=\prod_{k=1}^{u}(g_m+k)\ \wedge\ e|h_1\ \wedge\ e|h_2\ \wedge\ e|h_m\\
&\qquad\qquad\wedge\ l=P(b,c,a_1,a_2,...,a_n,c_1,c_2,...,c_n)\ \wedge\ e|l]
\end{aligned}$$

e esta é Diofantina pelos Teoremas (4.1.3), (4.1.5) e (4.1.7). □

4.2 Funções Recursivas

Vamos observar dois exemplos primeiro.

(i) O conjunto P dos números primos:

$$x \in P \Leftrightarrow x > 1 \wedge (\forall y, z)_{\leq x}[yz < x \vee yz > x \vee y = 1 \vee z = 1].$$

Uma outra definição Diofantina dos primos é:

$$x \in P \Leftrightarrow x > 1 \wedge ((x-1)!\,, x) = 1$$
$$\Leftrightarrow$$
$$x > 1 \wedge (\exists\, y, z, u, v)[y = x{-}1 \wedge z = y! \wedge (uz - vx)^2 = 1].$$

(ii) A função $g(y) = \prod_{k=1}^{y}(1 + k^2)$. Aqui nós usamos a função $F(a, b)$ definida aqui (2.4) para codificar a sequência $g(1), g(2), ..., g(r)$ em um único número b_0, ou seja, de modo que

$$F(a, b_0) = g(a), \quad a = 1, 2, ..., r$$

Desta forma, $z = g(a)$

$\Leftrightarrow (\exists\, b_0)\{F(1, b_0) = 2 \wedge (\forall\, a)_{\leq r}[k = 1 \vee F(a, b_0) = (1 + a^2)F(a - 1, b_0)] \wedge z = F(r, b_0)\}$
$\Leftrightarrow (\exists\, b_0)\{F(1, b_0) = 2 \wedge (\forall\, a)_{\leq r}[k = 1 \vee (\exists\, t, u, v)(t = a - 1$
$\wedge\ u = F(t, b_0) \wedge v = F(a, b_0) \wedge v = (1 + a^2)u) \wedge z = F(r, b_0)]\}.$

Agora podemos construir as *funções recursivas* usando certas funções iniciais denominadas *funções recursivas primitivas.*

As funções recursivas são todas aquelas possíveis de serem obtidas a partir das funções recursivas primitivas dadas por:

$$c(x) = 1, \quad s(x) = x + 1, \quad U_i^n(x_1, x_2, ..., x_n) = x_i, \quad 1 \leq i \leq n$$

iterativamente aplicando-se as três operações: *Composição, Recursão Primitiva e Minimalização* definidas abaixo.

Composição

Quando fazemos a Composição a partir das funções dadas $g_1, g_2, ..., g_m$ e $f(t_1, t_2, ..., t_m)$ obtemos a função

$$h(x_1, x_2, ..., x_n) = f(g_1(x_1, x_2, ..., x_n), \cdots, g_m(x_1, x_2, ..., x_n)).$$

Recursão Primitiva

Quando fazemos a Recursão Primitiva a partir das funções dadas f e g obtemos a função $h(x_1, x_2, ..., x_n, z)$ que satisfaz as equações:

$$h(x_1, x_2, ..., x_n, 1) = f(x_1, x_2, ..., x_n)$$
$$h(x_1, x_2, ..., x_n, t+1) = g(t, h(x_1, x_2, ..., x_n, t), x_1, x_2, ..., x_n).$$

Quando $n = 0$, f torna-se uma constante de modo que h é obtida diretamente de g.

Minimalização

Quando fazemos a Minimalização a partir das funções dadas f e g obtemos a função

$$h(x_1, x_2, ..., x_n) = \min_y[f(x_1, x_2, ..., x_n, y) = g(x_1, x_2, ..., x_n, y)]$$

assumindo que f e g são tais que para cada $x_1, x_2, ..., x_n$ exista ao menos um y satisfazendo a equação $f(x_1, x_2, ..., x_n, y) = g(x_1, x_2, ..., x_n, y)$ (ou seja, h está definida em todos os pontos).

Teorema 4.2.1. *Uma função é Diofantina se, e somente se, é recursiva.*

Vamos observar algumas funções recursivas antes de demonstrar este teorema.

(1) A função $f : \mathbb{N}_1^2 \to \mathbb{N}_1$ definida por $f(x, y) = x + y$ é recursiva:

$$x + 1 = s(x),$$
$$x + (t + 1) = s(x + t) = g(t, x + t, x),$$

onde $g(u, v, w) = s(U_2^3(u, v, w))$.

(2) A função $f : \mathbb{N}_1^2 \to \mathbb{N}_1$ definida por $f(x, y) = x \cdot y$ é recursiva:

$$x \cdot 1 = U_1^1(x)$$
$$x \cdot (t + 1) = x \cdot t + x = g(t, x \cdot t, x),$$

onde $g(u, v, w) = U_2^3(u, v, w) + U_3^3(u, v, w)$.

(3) Para cada k fixo, a função constante $c_k(x) = k$ é recursiva, pois $c_1(x)$ é uma das funções iniciais e $c_{k+1}(x) = c_k(x) + c(x)$.

(4) Qualquer polinômio $P(x_1, x_2, ..., x_n)$ com coeficientes inteiros positivos é recursiva, pois qualquer função deste tipo pode ser escrita como uma iteração finita de adições e multiplicações de variáveis e $c(x)$. Por exemplo:

$$2x^2y + 3xz^3 + 5 = c_2(x) \cdot x \cdot x \cdot y + c_3(x) \cdot x \cdot z \cdot z \cdot z + c_5(x).$$

Assim, **(1)**, **(2)** e **(3)** e a **Composição** dão o resultado.

Agora podemos provar o Teorema (4.2.1).

Demonstração. ($\Rightarrow$) Seja f uma função Diofantina, assim podemos escrever

$$y = f(x_1, x_2, ..., x_n)$$
$$\Leftrightarrow$$
$$(\exists\, t_1, t_2, ..., t_m \in \mathbb{N}_1)[P(x_1, x_2, ..., x_n, y, t_1, t_2, ..., t_m)] = Q(x_1, x_2, ..., x_n, y, t_1, t_2, ..., t_m)],$$

onde P e Q são polinômios com coeficientes inteiros positivos. Então, pela função $F(a,b)$ definida em (2.4) temos que

$$\begin{aligned} f(x_1,x_2,...,x_n) &= F(1,\min_b[P(x_1,x_2,...,x_n,F(1,b),F(2,b),...,F(m+1,b))] \\ &= Q(x_1,x_2,...,x_n,F(1,b),F(2,b),...,F(m+1,b))]. \end{aligned}$$

Como P, Q e $F(a,b)$ são recursivas, o mesmo ocorre com f (usando Composição e Minimalização).

($\Rightarrow$) Como $F(a,b)$ é Diofantina e as demais funções iniciais também o são, é suficiente provar que as funções Diofantinas são fechadas sob composição, recursão primitiva e minimalização.

Composição: Se $h(x_1,x_2,...,x_n) = f(g_1(x_1,x_2,...,x_n),...,g_m(x_1,x_2,...,x_n))$, onde $f,g_1,g_2,...,g_m$ são Diofantinas, de modo que h também é:

$$\begin{gathered} y = h(x_1,x_2,...,x_n) \\ \Leftrightarrow \\ (\exists\, t_1,t_2,...,t_m \in \mathbb{N}_1)[t_1 = g_1(x_1,x_2,...,x_n) \wedge \cdots \wedge t_m = g_m(x_1,x_2,...,x_n) \wedge y = f(t_1,t_2,..,t_m)]. \end{gathered}$$

Recursão Primitiva: Se

$$\begin{aligned} &h(x_1,x_2,...,x_n,1) = f(x_1,x_2,...,x_n) \\ &h(x_1,x_2,...,x_n,t+1) = g(t,h(x_1,x_2,...,x_n,t),x_1,x_2,...,x_n), \end{aligned}$$

e f,g são Diofantinas, então (usando a função $F(a,b)$ definida em (2.4) para codificar os números $h(x_1,x_2,...,x_n,1), h(x_1,x_2,...,x_n,2),...,h(x_1,x_2,...,x_n,z)$:

$$\begin{gathered} y = h(x_1,x_2,...,x_n,z) \\ \Leftrightarrow \\ \begin{aligned} (\exists\, b)\{(\exists\, c)[c = F(1,b) \wedge c = f(x_1,x_2,...,x_n)] \wedge (\forall\, t)_{\leq z})[(t=z) \vee (\exists\, c)(c = F(t+1,b))] \\ \wedge c = g(t,F(t,b)),x_1,x_2,...,x_n))] \wedge y = F(z,b)\} \end{aligned} \end{gathered}$$

de modo que, pelo Teorema (4.1.8) h é Diofantina. □

Minimalização: Se

$$h(x_1,x_2,...,x_n) = \min_y[f(x_1,x_2,...,x_n,y) = g(x_1,x_2,...,x_n)],$$

onde f,g são Diofantinas, então o mesmo vale para h, com efeito,

$$\begin{gathered} y = h(x_1,x_2,...,x_n) \\ \Leftrightarrow \\ (\exists\, z)[z = f(x_1,x_2,...,x_n,y)\, s \wedge z = g(x_1,x_2,...,x_n,y)] \\ \wedge (\forall\, t)_{\leq y}[(t = y \vee (\exists\, u,v)(u = f(x_1,x_2,...,x_n,t) \wedge v = g(x_1,x_2,...,x_n,t) \wedge (u < v \vee v < u)]. \end{gathered}$$

4.3 URM-Computabilidade

Nesta seção vamos considerar as máquinas registradoras ilimitadas (abreviadamente URM na sigla em inglês para *Unlimited Register Machines*) que da mesma forma que as máquinas de Turing definidas na seção (1.2), capturam exatamente a noção geral de computabilidade. As URM's são mais fáceis de compreender do que as máquinas de Turing, pois operam em alto nível de abstração e se assemelham com determinadas linguagens de programação computacional como Pascal. Desta forma são mais simples para tratamento teórico do que as máquinas de Turing e tais máquinas são investigadas em [6]. As URM's são equivalentes às máquinas de Turing no sentido do Teorema (4.3.3) e devido a maior facilidade para tratamento teórico que as máquinas de Turing, vamos adotá-las para provar a equivalência com as funções recursivas no sentido do Teorema (4.3.3).

Definição 4.3.1. *Uma máquina registradora ilimitada, denotada por URM, é uma abstração de um dispositivo de cálculo com as seguintes características:*

1. ***Registradores:*** *Uma URM possui posições de alocação chamadas registradores os quais podem armazenar números em $\mathbb{N}_0$. Qualquer URM-programa pode utilizar um número finito de registradores. Os registradores normalmente são referidos com letras maiúsculas substcritas $R_1, R_2, R_3, ...$ Os subscritos, que pertencem à $\mathbb{N}_1$, são chamados de índices do respectivo registrador. O número armazenado em qualquer instante por um registrador normalmente é referido com letras minúsculas subscritas $r_1, r_2, r_3, ...$ Os registradores são ilimitados nos seguintes sentidos:*

   (a) *Embora um URM-programa possa utilizar somente uma quantidade finita de registradores não há uma cota superior que quantos de fato um determinado programa realmente utilizará.*

   (b) *Não há cota superior para o tamanho dos números naturais que podem ser armazenados em qualquer dos registradores.*

2. ***Programa:*** *Os números armazenados nos registradores de uma URM são manipulados de acordo com o programa. Um URM-programa é uma lista finita de instruções básicas. As instruções são escritas em uma ordem fixa e numeradas como $1, 2, 3, ...$ Por razões históricas, o número de instruções é chamado de número da linha. Podemos nos referir ou à linha do programa ou à linha na URM. Vamos utilizar o símbolo $\mathbb{U}$ para denotar o conjunto de todos os URM-programas.*

   (a) ***Comprimento do Programa:*** *seja $P \in \mathbb{U}$ um URM-programa. Por definição, P é uma lista finita de instruções básicas. Definimos $\lambda : \mathbb{U} \to \mathbb{N}_1$ como:*

   $$\forall P \in \mathbb{U} : \lambda(P) = \textit{número de instruções básicas em } P.$$

   (b) ***Número de Rigistradores Usados****: seja $P \in \mathbb{U}$ um URM-programa. Por definição, P usa um número finito de registradores. Definimos a função $\rho : \mathbb{U} \to \mathbb{N}_1$ como:*

   $$\forall P \in \mathbb{U} : \rho(P) = \textit{maior número armazenado em um registrador usado por } P.$$

   Assim, em qualquer URM-programa P, nenhuma instrução se refere a qualquer registrador com índice maior do que R_u, onde $u = \rho(P)$.

3. ***Intruções Básicas:***

Nome	***Notação***	***Efeito***	***Descrição***
ANULAR	$Z(n)$	$0 \to R_n$	*Substitui o número em R_n por 0.*
SUCEDER	$S(n)$	$r_n + 1 \to R_n$	*Adiciona 1 ao número em R_n.*
COPIAR	$C(m, n)$	$r_m \to R_n$	*Substitui o número em R_n pelo número em R_m (não modifica o número de R_m).*
PULAR	$J(m, n, q)$	$r_m = r_n? \Rightarrow q$	*Se os números em R_m e R_n são iguais, vai para a instrução número q, caso contrário vai para a próxima instrução.*

4. ***Operação:***

   (a) *Quando uma URM executa um programa, sempre começa executando a primeira instrução do programa.*

   (b) *Quando uma instrução tem sido executada, move-se para a próxima instrução e a executa a menos que tenha sido executado a instrução PULAR.*

   i. ***Apontador da Instrução:*** *O número da linha que está sendo executada em determinado intante é denominado de apontador da instrução. Pode ser pensado como um registrador com propósito especial na URM que guarda o número da linha.*

   ii. ***Estágio de Cálculo:*** *O estágio de cálculo (ou apenas estágio) de um URM-programa é o número de instruçoes básicas que têm sido executadas até certo ponto. Desta forma cada estágio corresponde ao processamento de uma instrução.*

   iii. ***Estado:*** *O estágio de um URM-programa em um instante particular é definido como:*

      i. *o valor do apontador da instrução;*

      ii. *o valor, naquele instante, de cada um dos registradores que são usados pelo programa.*

5. ***Terminação****: Um URM-programa pára ou termina quando não existem mais instruções a serem executadas. Isto pode acontecer de duas maneiras:*

   (a) *Se o programa executa a última instrução e esta não é um PULAR para uma anterior, o programa pára.*

   (b) *Se o programa executa um PULAR para uma instrução não existente, o programa parará. Tal instrução PULAR é conhecida como PULAR PARA FORA.*

   A linha na qual um URM-programa executa uma instrução particular e pára é chamada de linha de saída. Se o programa, quando executando, nunca chega a um estado como estes, então é chamado de um laço infinito. Podemos observar que se um programa termina ou não, pode depender da entrada em particular. Pode entrar num laço infinito para certa entrada, mas pode terminar perfeitamente bem para outra.

6. ***Entrada:*** *A entrada de um URM-programa é*

(a) *ou uma k-upla* $(n_1, n_2, ...n_k) \in \mathbb{N}_0^k$,

(b) *ou um número* $n \in \mathbb{N}_0$.

No último caso é conveniente considerar um único número natural como uma 1*-upla ordenada* $(n_1) \in \mathbb{N}_0^1 = \mathbb{N}_0$. *Assim podemos discutir entradas em URM-programas apenas usando t-uplas e evitar o trabalho desnecessário de repetir os casos onde* $k = 1$. *A convenção normalmente usada considera que um URM-programa P começa o cálculo com*

I. *a entrada* $(n_1, n_2, ..., n_k)$ *nos registradores* $R_1, R_2, .., R_k$,

II. *0 em todos os outros registradores usados por P.*

Ou seja, o estado inicial da URM é

I. $\forall i \in [1...k] : r_i = n_i$,

II. $\forall i > k : r_i = 0$.

É usual para a entrada (toda ela ou parte dela) ser sobrescrita durante o curso das operações do programa. Isto é, no término do programa, $R_1, R_2, ..., R_k$ *não são garantidos ainda conterem* $n_1, n_2, ..., n_k$ *a menos que o programa tenha sido explicitamente escrito de modo a garantir que isto ocorra.*

7. ***Saída:*** *No final da execução do URM-programa, a saída estará armazenada no registrador* R_1.

8. ***Programa Nulo:*** *Um programa nulo ou vazio é um URM-programa que não contém instruções.*

Definição 4.3.2. *Uma função* $f : \mathbb{N}_0^k \to \mathbb{N}_0$ *é dita ser URM-computável se existe um URM-programa que a calcula.*

Conforme podemos perceber, a execução de uma única instrução para uma URM, por exemplo $\mathsf{PULAR}(m, n, q)$, pode ser bastante complexa do ponto de vista computacional, pois os conteúdos r_n e r_m dos registradores R_n e R_m, respectivamente, precisam ser lidos, sendo que r_n e r_m podem ser arbitrariamente grandes e em seguida precisam ser comparados. Esta tarefa pode tomar tempo e espaço demasiadamente longos.

Teorema 4.3.3. *Para qualquer função* $f : \mathbb{N}_0^k \to \mathbb{N}_0$ *são equivalentes as afirmações:*

1. *f é URM-computável*

2. *f é Turing-computável*

3. *f é recursiva*

Vamos provar que 1 implica 3 em (4.3.15). A volta é deixada como leitura em alguma das referências [2],[7] e [9] ou com as mesmas notações deste texto em [11]. Vamos basear a solução do Décimo Problema de Hilbert na URM-computabilidade em conjunto com a Turing-computabilidade. Em [2] pode-se encontrar a demonstração da equivalência entre (2) e (3). O que produz o mesmo resultado que baseado em Turing-computabilidade.

Lema 4.3.4. *A toda instrução básica I em um URM-programa podemos associar um número em $\mathbb{N}_1$ de modo que cada um destes números corresponda a uma única instrução.*

Demonstração. Seja $\mathbb{I}$ o conjunto de todas as instruções básicas de uma URM. Definimos a aplicação $\beta : \mathbb{I} \to \mathbb{N}$ como

$$\begin{aligned}\beta(Z(n)) &= 6n - 3,\\ \beta(S(n)) &= 6n,\\ \beta(C(m,n)) &= 2^m 3^n + 1,\\ \beta(J(m,n,q)) &= 2^m 3^n 5^q + 2,\end{aligned}$$

de forma que obtemos as congruências

$$\begin{aligned}\beta(Z(n)) &\equiv 3(\text{mod } 6),\\ \beta(S(n)) &\equiv 0(\text{mod } 6),\\ \beta(C(m,n)) &\equiv 1(\text{mod } 3) \equiv 1 \text{ ou } 4(\text{mod } 6),\\ \beta(J(m,n,q)) &\equiv 2(\text{mod } 3) \equiv 2 \text{ ou } 5(\text{mod } 6).\end{aligned}$$

Assim, se $\beta(I_1) = \beta(I_2)$, então ambas as instruções devem ser do mesmo tipo, pois devem ter o mesmo resíduo módulo 6 e os resíduos são incongruentes módulo 6 para cada tipo, conforme mostrado pelo sistema de congruências acima. Segue, do Teorema Fundamental da Aritmética, que β é unicamente determinada para qualquer instrução básica dada de uma URM, pois é injetora. □

Definição 4.3.5. *O número $\beta(I)$ é definido como o número de codificação para a instrução I.*

Lema 4.3.6. *A qualquer URM-programa podemos associar um número em $\mathbb{N}_1$ de modo que cada um destes números corresponda a um único URM-programa.*

Demonstração. Seja $\mathbb{U}$ o conjunto de todos os URM-programas. Seja $P \in \mathbb{U}$ um programa com k instruções básicas:

Linha	**Comando**
1	I_1
2	I_2
⋮	⋮
k	I_k

Definimos a aplicação $\gamma : \mathbb{U} \to \mathbb{N}_0$ como

$$\gamma(P) = \prod_{i=1}^{k} p_i^{\beta(I_i)}$$

onde

1. p_i é o i-ésimo número primo
2. $\beta(I_i)$ é a única codificação para a instrução i.

Então, segue do Teorema Fundamental da Aritmética que γ é unicamente especificada para qualquer URM-programa, pois é injetora. □

Definição 4.3.7. *Dado $P \in \mathbb{U}$, o número $\gamma(P)$ é referido como o número de codificação de P.*

Definição 4.3.8. *O comprimento de $n \in \mathbb{N}_2$ é definido como o número de fatores primos distintos de n e é denotado por $len(n)$.*

Definição 4.3.9. *Sejam $S \subset \mathbb{N}_0^k$ e a função $f : S \to \mathbb{N}_0$. Suponha que para toda k-upla em $\mathbb{N}_0^k \backslash S$, f não esteja definida. Então f é chamada de função parcial de $\mathbb{N}_0^k$ em $\mathbb{N}_0$. Caso $S = \mathbb{N}_0^k$, então f é chamada de função total.*

Definição 4.3.10. *Sejam $g : \mathbb{N}_0^k \to \mathbb{N}_0$ e $h : \mathbb{N}_0^k \to \mathbb{N}_0$ funções parciais. Escrevemos*

$$g(n_1, n_2, ..., n_k) \approx h(n_1, n_2, ..., n_k)$$

se, e somente se, ocorre uma das duas possibilidades:

1. *ambos $g(n_1, n_2, ...n_k)$ e $h(n_1, n_2, ..., n_k)$ estão bem definidos e são iguais ou*
2. *nem $g(n_1, n_2, ...n_k)$ nem $h(n_1, n_2, ..., n_k)$ estão definidos.*

Assim, por exemplo, outra maneira de escrever que $g = h$, é:

$$\forall x \in \mathbb{N}_0^k : g(x) \approx h(x)$$

neste caso, as funções g e h são totais (caso particular de função parcial), ou seja, estão bem definidas para todas as k-uplas pertencentes a $\mathbb{N}_0^k$.

Lema 4.3.11. *A todo estado de um URM-programa podemos associar um número em $\mathbb{N}_1$ de modo que cada um destes números corresponda a um único estado.*

Demonstração. Seja P um URM-programa. Suponha que, em um dado estágio do cálculo

1. o valor do apontador da instrução é a
2. o valor do registrador R_k é r_k.

Seja $b = \rho(P)$ o número de registradores usados por P. Então podemos definir a codificação de estado s como

$$s = p_1^a p_2^{r_1} p_3^{r_2} \cdots p_{b+1}^{r_b}$$

onde p_j é o j-ésimo número primo. Então, segue do Teorema Fundamental da Aritmética que s é unicamente especificado para qualquer estado dado. □

Definição 4.3.12. *Este número de codificação s é chamado de codificação de estado.*

Definição 4.3.13. *Seja $\mathbb{S} = \prod_{i=1}^{n} S_i \times S_2 \times \cdots \times S_n$ o produto cartesiano de n conjuntos $S_1, S_2, ..., S_n$. Uma relação n-ária sobre $\mathbb{S}$ é uma $(n+1)$-upla ordenada definida como*

$$\mathfrak{R} = (S_1, S_2, ..., S_n, R)$$

onde R é um subconjunto arbitrário $R \subset \mathbb{S}$. Para indicar que $(s_1, s_2, ..., s_n) \in R$ escrevemos $\mathfrak{R}(s_1, s_2, ..., s_n)$.

Definição 4.3.14. *Seja $\mathfrak{R} \subset \mathbb{N}_0^k$ uma relação n-ária sobre $\mathbb{N}_0^k$. Então $\mathfrak{R}$ é uma relação recursiva se, e somente se, sua função característica $\chi_{\mathfrak{R}}$ é uma função recursiva primitiva.*

Teorema 4.3.15. *Toda função URM-computável é recursiva.*

Demonstração. Seja $f : \mathbb{N}_0^k \to \mathbb{N}_0$ uma função computável. Então existe um URM-programa que computa f por definição. Seja P o URM-programa com o menor número de codificação que computa f. Seja $m = \gamma(P)$ o número de codificação de P. Considere a função $g : \mathbb{N}_0^k \to \mathbb{N}_0$ dada por

$$g(n_1, n_2, ..., n_k) \approx \mu t((S_k(m, n_1, n_2, ..., n_k, t))_1 > len(m)$$

onde

1. $len(m)$ é o comprimento de m;
2. μt é a operação de minimização sobre S_k;
3. $\approx$ denota a igualdade de função parcial;
4. $S_k(m, n_1, n_2, ..., n_k, t)$ é o número de codificação de estado no estágio t do cálculo realizado por P com a entrada $(n_1, n_2, ..., n_k)$ o qual é dado pelo número s definido em (4.3.12);
5. $(S_k(m, n_1, n_2, ..., n_k, t))_1$ é o número de codificação da instrução dado pela função β a ser realizada no estágio t.

Assim, a desigualdade

$$(S_K(m, n_1, n_2, ..., n_k, t))_1 > len(m)$$

expressa o fato que no estágio t o cálculo terminou. Assim, o valor de $g(n_1, n_2, ..., n_k)$ é a codificação de estado do primeiro estágio na qual o cálculo tem parado, se existe um, e indefinido caso contrário.

Como as funções nesta desigualdade e o relação $>$ são todos primitivos recursivos, segue que a desigualdade expressa uma relação recursiva primitiva sobre $m, n_1, n_2, ..., n_k, t$.

Desta forma g é uma função recursiva por definição, visto que pode ser obtida por minimização sobre uma relação recursiva.

Considere a função $h : \mathbb{N}_0^k \to \mathbb{N}_0$ dada por

$$h(n_1, n_2, ..., n_k) \approx S_k(m, n_1, n_2, ..., n_k, g(n_1, n_2, ..., n_k)).$$

A função h é recursiva porque é obtida por substituição de funções recursivas conhecidas. A função h está definida se, e somente se, o cálculo pára e produz o valor da codificação do estado no momento em que se tem parado. A saída deste cálculo, a qual produz o valor de f, é o número no registrador R_1. Mas, o número em R_1 é o expoente de $p_2 = 3$ conforme o Lema (4.3.12) na expressão da codificação do estado $h(n_1, n_2, ..., n_k)$ na forma $p_1^a p_2^{r_1} p_3^{r_2} \cdots p_{k+1}^{r_k}$. Portanto, a função f é dada por

$$f(n_1, n_2, ...n_k) \approx (h(n_1, n_2, ..., n_k))_1.$$

Segue que f é uma função recursiva, pois é obtida por substituição de funções recursivas conhecidas. □

Capítulo 5

Conclusão

5.1 Um Conjunto Diofantino Universal

Uma enumeração explícita de todos os conjuntos Diofantinos de inteiros positivos será descrita. Qualquer polinômio com coeficientes inteiros positivos pode ser construído via recursão utilizando o inteiro 1 e indeterminadas fazendo multiplicações e adições sucessivas. Da seguinte maneira:

Primeiro fixamos o alfabeto

$$x_0, x_1, x_2, x_3, ...$$

de variáveis independentes, e então definimos recursivamente os polinômios P_i como

$$\begin{aligned} P_1 &= 1 \\ P_{3i-1} &= x_{i-1} \\ P_{3i} &= P_{L(i)} + P_{R(i)} \\ P_{3i+1} &= P_{L(i)} \cdot P_{R(i)} \end{aligned}$$

onde $i \geq 1$ e as funções $L(i)$ e $R(i)$ usadas acima são as mesmas definidas na seção (2.4).

Temos, de fato, obtido todos os polinômios com coeficientes inteiros positivos por meio desta recursão. Por exemplo, considere o polinômio $P = 2x_0^3 + 3x_1x_2 + 5$. A seguir $\{k_j\}_{\geq 0}$ denota uma sequência de índices em $\mathbb{N}_1$.

Reescrevemos P como $P = 2 \cdot x_0 \cdot x_0 \cdot x_0 + 3 \cdot x_1 \cdot x_2 + 5 = (P_{k_1} \cdot P_{k_2}) \cdot (P_{k_2} \cdot P_{k_2}) + P_{k_3} \cdot (P_{k_4} \cdot P_{k_5}) + P_{k_6} = P_{k_7} \cdot P_{k_8} + P_{k_3} \cdot P_{k_9} + P_{k_6} = (P_{k_{10}} + P_{k_{11}}) + P_{k_6} = P_{k_{12}} + P_{k_6} = P_{k_{13}}$. Isto é sempre possível, não importa qual polinômio P iniciamos, pois os pares $(L(i), R(i))_{i \geq 1}$ percorrem todos os pares de inteiros positivos; com $P_1 = 1$ conseguimos gerar todos os polinômios idênticos aos inteiros positivos e com $P_{3i-1} = x_{i-1}$ todos os polinômios da forma x_{i-1}. Na verdade, geramos todos os polinômios, mas eventualmente algum se repete com a recursão, por exemplo, $P_3 = P_{13} = 2$ e $P_7 = P_{10} = 1 + x_0$.

Escrevemos $P_i = P_i(x_0, x_1, ..., x_n)$, onde n é grande o bastante de modo que todas as variáveis ocorrendo em P_i estão incluidos. (Não necessariamente P_i dependerá de todas estas variáveis.) Finalmente, seja

$$D_n = \{x_0 | (\exists\, x_1, x_2, ..., x_n)[P_{L(n)}(x_0, x_1, ..., x_n) = P_{R(n)}(x_0, x_1, ..., x_n)]\}.$$

Aqui, $P_{L(n)}$ e $P_{R(n)}$ de fato não envolvem todas das variáveis $x_0, x_1, ..., x_n$ mas também não envolvem quaisquer outras. (Já mostramos na seção (2.4) que $L(n), R(n) \leq n$.) Pela forma como a sequência P_i foi construída, temos que a sequência de conjuntos

$$D_1, D_2, D_3, D_4, ...$$

percorre todos os conjuntos Diofantinos em $\mathbb{N}_1$.

Teorema 5.1.1. ***(Teorema da Universalidade)*** *Temos que o conjunto* $\{(n, x)|x \in D_n\}$ *é Diofantino.*

Demonstração. Novamente, usando a função $F(a, b)$ definida na seção (2.4), temos que:

$$\begin{aligned} & x \in D_n \\ & \Leftrightarrow \\ (\exists\, b)\{F(1,b) = 1 \ \wedge\ & F(2,b) = x \\ & \wedge\ (\forall i)_{\leq n}[F(3i,b) = F(L(i),b) + F(R(i),b)] \\ & \wedge\ (\forall i)_{\leq n}[F(3i+1,b) = F(L(i),b) \cdot F(R(i),b)] \\ & \wedge\ F(L(n),b) = F(R(n),b)\} \end{aligned}$$

O predicado do lado direito da equivalência é Diofantino, assim devemos verificar somente a afirmação. Seja $x \in D_n$ para x, n dados. Então, existem números $t_1, t_2, ..., t_n$ tais que

$$P_{L(n)}(x, t_1, t_2, ..., t_n) = P_{R(n)}(x, t_1, t_2, ..., t_n).$$

Escolhemos b conforme fizemos na seção (2.4) para a função $F(a, b)$ tal que

$$F(j, b) = P_j(x, t_1, t_2, ..., t_n), \quad j = 1, 2, ..., 3n + 2. \tag{1}$$

Então, em particular, $F(2, b) = x$ e $F(3i - 1, b) = t_{i-1}, i = 2, 3, ..., n + 1$. Desta forma, o lado direito da equivalência é verdadeiro. Reciprocamente, suponha que o lado direito ocorra dados n, x. Seja

$$t_1 = F(5, b), t_2 = F(8, b), ..., t_n = F(3n + 2, b).$$

Assim, (1) deve ocorrer. Como $F(L(n), b) = F(R(n), b)$ devemos ter

$$P_{L(n)}(x, t_1, t_2, ..., t_n) = P_{R(n)}(x, t_1, t_2, ..., t_n),$$

de modo que $x \in D_n$. □

Como $D_1, D_2, D_3, ...$ gera uma enumeração de todos os conjuntos Diofantinos, é fácil construir um conjunto diferente de todos estes e, portanto, não-Diofantino. Podemos fazer

$$V = \{n|n \notin D_n\}.$$

Teorema 5.1.2. *O conjunto* V *não é Diofantino.*

Demonstração. Podemos aplicar o método diagonal de Cantor. Se V fosse Diofantino, então para algum i fixo, $V = D_i$. O mesmo ocorre com i, ou seja, $i \in V$? Temos:

$$i \in V \Leftrightarrow i \in D_i \quad \text{e} \quad i \in V \Leftrightarrow i \notin D_i.$$

Chegamos a uma contradição. □

Teorema 5.1.3. *A função $g(n,x) = 1$ definida por*

$$g(n,x) = 1 \quad \text{se} \quad x \notin D_n,$$
$$g(n,x) = 2 \quad \text{se} \quad x \in D_n,$$

não é recursiva.

Demonstração. Se g fosse recursiva, então ela seria Diofantina (4.2.1), digamos:

$$y = g(n,x) \Leftrightarrow (\exists\, y_1, y_2, ..., y_m)[P(n,x,y,y_1,...,y_m) = 0].$$

Mas então, seguiria que

$$V = \{x | (\exists\, y_1, y_2, ..., y_m)[P(x,x,1,y_1,y_2,...,y_m) = 0]\}$$

o que contradiz o Teorema (5.1.2).

□

Teorema 5.1.4. *O Décimo Problema de Hilbert é insolúvel: não existe um algoritmo baseado em instruções para máquinas de Turing ou URM's tal que exista uma máquina que utilizando este algoritmo seja capaz de parar após um número finito de execuções em um estado que revele se qualquer equação Diofantina que está sendo testada pelo algoritmo possui ou não solução.*

Demonstração. Usando o Teorema (5.1.1), escrevemos:

$$x \in D_n \Leftrightarrow (\exists\, z_1, z_2, ..., z_k)[P(n,x,z_1,z_2,...,z_k) = 0].$$

onde P e algum polinômio definido (embora complicado). Suponha que houvesse um algoritmo para testar se equações Diofantinas possuem ou não solução inteira positiva, isto é, um algoritmo para o Décimo Problema de Hilbert. Então, dados n, x este algoritmo poderia ser usado para testar se a equação

$$P(n,x,z_1,z_2,...,z_k) = 0$$

tem uma solução, i.e., se $x \in D_n$ ou não. Assim, o algoritmo poderia ser usado para calcular a função $g(n,x)$. Como as funções recursivas são exatamente aquelas para as quais um algoritmo de cálculo existe, g teria que ser recursiva. Isto contradiria o Teorema (5.1.3). □

Bibliografia

[1] F. B. Martinez, C. G. Moreira, N. Saldanha e E. Tengan. *Teoria dos Números: um passeio com primos e outros números familiares pelo mundo inteiro.* Projeto Euclides, IMPA, 2013.

[2] M. Davis. *Computability and Unsolvability.* Dover, 1982.

[3] Y. V. Matiyasevich. *Hilbert's Tenth Problem.* MIT Press, 1996.

[4] C. Y. Shine. *21 Aulas de Matemática Olímpica.* SBM, 2009.

[5] M. Davis and R. Hersh. *Hilbert's 10th Problem.* Scientific American, 1973.

[6] S. Barry Cooper *Computability Theory.* CRC Press, 2004.

[7] H. Rogers Jr. *Theory of Recursive Functions and Effective Computability.* McGraw-Hill, 1967.

[8] Z. Melzak. *A Note on Hilbert's Tenth Problem.* McGrill University, 1959.

[9] N. Cutland. *Computability: an introduction to recursive function theory.* Cambridge Press, 1992.

[10] M. Davis., R. Sigal, E. Weyuker *Computability, Complexity, and Languages.* Academic Press, 1983.

[11] *https://proofwiki.org/wiki/Recursive_Function_is_URM_Computable*

[12] P. Ribenboim *Números Primos - Velhos Mistérios e Novos Recordes* IMPA, 2014.

[13] J. Jones, D. Sato, D. Wiens *Diophantine Representation of the Set of Prime Numbers* MAA, 1976.