A álgebra dos sistemas de identificação: da aritmética modular aos grupos diedrais

Jorge Picado

1. Introdução

> *"O Coelho Branco pôs os óculos.*
> *— Por onde devo começar, Vossa Majestade? — perguntou ele.*
> *— Começa pelo princípio, — disse o Rei, muito sério — e continua até chegares ao fim. Depois pára."*
>
> Lewis Carroll, *As Aventuras de Alice no País das Maravilhas*

De há alguns anos para cá os Bilhetes de Identidade passaram a ter um algarismo suplementar, justaposto ao seu número (Figura 1), sendo habitual ouvirmos na rua as mais diversas interpretações para o significado de tal algarismo. Destas, a mais comum é a de que "indica o número de pessoas em Portugal com o mesmo nome que o portador do BI". (No meu BI esse algarismo é o zero o que me deixa mais descansado!)

Figura 1: Exemplo de Bilhete de Identidade.

É claro que qualquer pessoa com um pouco de bom senso dirá que esta explicação é um perfeito disparate; porque é que esse número só pode variar entre 0 e 9? por outro lado, basta indagar do algarismo constante no BI de algumas pessoas para encontrarmos rapidamente algarismos iguais a 8 e a 9 em pessoas cujo nome é, além de comprido, pouco usual.

Com este artigo pretendemos explicar o papel de tal algarismo. Veremos como todos os dias lidamos com sistemas de identificação deste tipo, cuja matemática além de ser facilmente compreensível, pode ser melhorada de modo a tornar esses mesmos sistemas muito mais eficientes. Isto permite-nos ilustrar algumas das características fundamentais da matemática moderna, nomeadamente a sua "orientação abstracta, formal ou axiomática" (Sebastião e Silva [18]). Observaremos como algumas estruturas algébricas abstractas criadas pelos matemáticos têm aplicação nos sistemas de identificação com que lidamos na nossa vida quotidiana e, além disso, como a utilização dessas estruturas se torna indispensável se quisermos que esses sistemas sejam realmente eficientes.

2. Segurança na transmissão de dados

> "WRONG NUMBER - Demolition crew wrecks house at 415, not 451
>
> *It was a case of mistaken identity. A transposed address that resulted in a bulldozer blunder.*
> *City orders had called for demolition on Tuesday of the boarded--up house at 451 Fuller Ave. SE. (...)*
> *But when the dust settled, 451 Fuller stood untouched. Down the street at 415 Fuller Ave. SE, only a basement remained."*
>
> Doug Guthrie, *The Grand Rapids Press* (5/12/90)

Em qualquer texto, um erro de ortografia numa palavra pode ser facilmente detectável: ou a palavra não faz parte da língua ou é simplesmente ilegível como, por exemplo, "Mtaemática". Tal transposição de símbolos adjacentes é um erro comum em textos escritos numa máquina (as teclas correspondentes às letras trocadas foram simplesmente premidas pela ordem errada). Se, no entanto, encontrarmos a palavra "alta" algures num texto, não teremos maneira de dizer se lá deveria estar "lata", a não ser pelo contexto, lendo a frase que contém a palavra duvidosa. Pode ser mesmo que a palavra correcta seja "acta", "anta" ou "apta", mas acidentalmente premiu-se a tecla errada, um outro erro muito comum. Mais uma vez, o contexto dar-nos-á a chave para encontrar a palavra certa. Contudo, se um empregado bancário comete um tal erro aquando da escrita de um número de conta numa dada transacção, não existe à primeira vista maneira de detectar tal erro e imaginar-se-á os problemas causados.

É precisa então alguma protecção contra estes erros. Um "contexto" para o número de conta poderia ser providenciado pelo nome do titular da conta. Escrevendo sempre o nome do cliente juntamente com o número da conta, o nome poderia ser comparado com o número escrito e, em caso de inconsistência, uma mensagem de alerta poderia ser emitida. Mas se o nome for tão comum como "Silva", por exemplo, o erro poderá persistir. Além disso, este método tem o inconveniente de exigir que, numa transferência de um banco para outro, o primeiro banco tenha acesso à lista dos nomes dos titulares de contas no segundo banco, o que não é possível. O erro só seria portanto detectado quando o aviso de transferência chegasse ao segundo banco, porque só aí o teste de comparação do número com o nome poderia ser feito.

Por estas razões, o que se faz é apensar um grupo de algarismos redundantes, chamados *algarismos de teste*, ao número de conta original. O novo número obtido passa a ser o número de conta real. Nalguns casos, como veremos, em vez de algarismos utilizam-se caracteres não numéricos (por isso, também se costumam designar os algarismos de teste por *caracteres de teste*).

Vejamos um exemplo diferente. Em diversos institutos de investigação médica existem ficheiros com os dados dos pacientes em estudo. Essas fichas são, por razões óbvias,

mantidas anónimas, o que significa que a correspondência entre uma ficha e os dados pessoais do respectivo paciente só é recuperável por meio de um número de identificação, o "número do paciente". Para manter a privacidade dos dados, os cientistas não têm acesso aos dados pessoais de cada paciente, pelo que só se podem referir a uma dada ficha pelo seu número de paciente. Um erro de dactilografia na escrita do número do paciente aquando da actualização de uma dada ficha poderia diminuir ou mesmo anular a qualidade da investigação. Neste caso é claro que não é possível introduzir, como teste, o nome ou outro dado pessoal do paciente (data de nascimento, etc.). Como poderá então ser detectado e, portanto, prevenido um erro de dactilografia? Escrever cada número duas vezes é, além de moroso, pouco útil. Se uma pessoa lê, inadvertidamente, "1947" em vez de "1974" terá certamente a propensão para fazer o mesmo novamente. Tal erro de transposição é tão usual que uma outra pessoa teria alguma probabilidade de fazer o mesmo. Este tipo de problemas aumenta em algumas línguas — por exemplo, no alemão, onde números como "47" são ditos como "sete-e-quarenta", numa tradução literal. Também neste caso uma solução reside em apensar um algarismo redundante ao número de identificação original. Se se abre uma ficha para o doente 1947, esta não será identificada pelo número 1947 mas sim, digamos, pelo número 19476. Este novo número que se obtém do número "não protegido" 1947 por justaposição do algarismo de teste 6 diz-se então um "número protegido". O algarismo de teste é determinado a partir do número não protegido por um certo algoritmo (chamado *algoritmo de teste*). Daqui em diante só o número protegido será usado para identificar a ficha do paciente. Sempre que é dactilografado, o computador aplica o algoritmo de teste para verificar se o último algarismo escrito é de facto o mesmo algarismo que o algoritmo fornece quando aplicado ao número não protegido. É claro que a eficiência deste método reside na concepção do algoritmo de teste: terá que permitir a detecção dos erros de dactilografia mais comuns. Por exemplo, todos os números que possam surgir de 19476 por um desses erros, incluindo 91476, 10476, 14976, 19477 etc. deverão ser identificados como incorrectos. É então necessário que os algarismos de teste calculados para os números não protegidos 9147, 1047 e 1497 sejam todos diferentes de 6; o número 19477 é já reconhecido como errado uma vez que o algarismo de teste para o 1947 é 6 e não 7.

Por exemplo, um sistema não muito eficiente é o seguinte sistema "módulo 10". Se o número de identificação é, digamos, 1234567895 então 5 é o algarismo de teste, que é calculado por

$$1+2+3+4+5+6+7+8+9 \equiv 5 \pmod{10}.$$

Se escrevermos erradamente 12$\underline{4}$4567895 teremos

$$1+2+4+4+5+6+7+8+9 \equiv 1 \pmod{10}$$

e o erro é imediatamente detectado pois $1 \neq 5$. Este método não pode, no entanto, ser recomendado se quisermos detectar o erro 1$\underline{32}$4567895. Note-se que a ocorrência destes erros pode ser por exemplo fatal no caso dos sistemas automáticos de navegação aérea, onde estes métodos são hoje em dia muito utilizados.

3. Sistemas de identificação modulares: dos livros aos códigos de barras

> *"So abundant are the sorts ..., that to describe them all would tire the patience of Socrates himself."*
>
> Nicholas Culpeper, *Complete Herbal and English Physician* (1826)

> *"There are, however, few things which are worth knowing that can be known without patient attention."*
>
> Anne Pratt, *The Flowering Plants Grasses, Sedges and Fernes of Great Britain* (1899-1905)

O desenvolvimento nos últimos anos de sistemas automáticos, relativamente baratos, rápidos e fiáveis, de leitura de números, tornou prática usual a justaposição de um algarismo aos números de identificação de uma dada colecção de objectos, com o objectivo de detectar os erros de leitura e escrita mais comuns.

Figura 2: Vários exemplos de sistemas de identificação com algarismo de teste: sistema ISBN, bilhete de avião, identificador de via verde e sistema ZIP de correspondência postal.

Nestes sistemas com algarismos de teste — habitualmente apelidados de *sistemas (ou códigos) de identificação detectores de erros* — não se pretende que o erro seja automaticamente corrigido mas tão só que o sistema alerte o operador da ocorrência de um erro e, consequentemente, da necessidade de reescrever o número.

É o que acontece, por exemplo, nos cartões de crédito, contas bancárias, cheques, livros (sistema ISBN), publicações periódicas (sistema ISSN), publicações de pautas musicais (sistema ISMN), bilhetes de avião, códigos de barras (sistemas UPC, EAN, etc.), passaportes, produtos químicos, cartas de condução, identificadores via verde (das

auto-estradas), carros de aluguer, vales postais, correio expresso, correspondência postal (sistema ZIP), bancos de sangue, produtos farmacêuticos (sistema UPN), encomendas postais, etc. (veja [5], [6], [7], [20], [21]).

O sistema ISBN

Um dos exemplos mais antigos é o sistema *International Standard Book Number* (ISBN) de catalogação de livros (Figura 3). A necessidade que as editoras e livrarias têm de catalogar os seus livros e informatizar o sistema de encomendas está na base da concepção deste sistema. Uma lista de números é transmitida e guardada com mais eficiência do que uma lista de nomes. Além disso, as listas de números transpõem a barreira da língua e dos vários alfabetos da comunidade internacional (pense-se na facilidade com que podemos encomendar um livro a um editor japonês sem ter que especificar o título do livro em japonês!). O sistema ISBN, criado em 1969 pela *International Standard Organization*, é um sistema de identificação numérica de livros, CD-Roms e publicações em braille. Associando um único número a cada título publicado o sistema providencia esse título com a sua "identidade" não duplicável, reconhecida internacionalmente. Desde então as editoras identificam os seus livros com um número $a_1\ a_2\ a_3\ a_4\ a_5\ a_6\ a_7\ a_8\ a_9\ a_{10}$ de 10 algarismos, servindo os 9 algarismos $a_1, a_2, \ldots, a_9$ para identificar o livro, divididos por 3 secções de comprimento variável, separadas por um hífen, e sendo a_{10} o algarismo de teste. A primeira secção (mais à esquerda) identifica um grupo nacional ou geográfico de editores, a segunda secção identifica um editor particular desse grupo e a terceira identifica um título particular ou determinada edição de um título. Por exemplo, se olharmos para as costas do livro "Aventuras matemáticas" de M. de Guzman observamos o seu número ISBN:

ISBN-972-662-165-8

Figura 3: Livro com ISBN 972-662-165 e algarismo de teste 8.

"972" é a identificação utilizada para os livros editados em Portugal e "662" a utilizada pela editora Gradiva. Esta etiquetou o livro com o número "165". O algarismo de teste "8" é calculado do seguinte modo:

$$10\times\underline{9}+9\times\underline{7}+8\times\underline{2}+7\times\underline{6}+6\times\underline{6}+5\times\underline{2}+4\times\underline{1}+3\times\underline{6}+2\times\underline{5}+1\times\underline{8} = 27\times 11 \equiv 0 \pmod{11}.$$

Portanto o algarismo de teste a_{10} é escolhido por forma a que a *soma de teste*

$$\begin{aligned} S &= (a_1, a_2, \cdots, a_{10}) \cdot (10, 9, 8, 7, 6, 5, 4, 3, 2, 1) \\ &= 10a_1 + 9a_2 + 8a_3 + 7a_4 + 6a_5 + 5a_6 + 4a_7 + 3a_8 + 2a_9 + a_{10} \end{aligned}$$

seja divisível por 11; ou seja, $a_{10} \equiv -\sum_{i=1}^{9}(11-i)\times a_i \pmod{11}$. No caso do ISBN ser $972-662-178-a_{10}$ obtemos, para o algarismo de teste a_{10},

$$-(10\times\underline{9}+9\times\underline{7}+8\times\underline{2}+7\times\underline{6}+6\times\underline{6}+5\times\underline{2}+4\times\underline{1}+3\times\underline{7}+2\times\underline{8}) = -298 \equiv 10 \pmod{11},$$

isto é $a_{10} = 10$. Como se pretende que os números de identificação ISBN sejam formados por 10 símbolos alfanuméricos, convencionou-se representar o algarismo de teste "10" pelo respectivo símbolo na numeração romana, "X". Tendo isto em conta, obtemos o número ISBN 972-662-178-X que identifica o livro "Ah, descobri!" de Martin Gardner.

Um outro exemplo: o livro "Hilbert" de Constance Reid possui na sua versão em alemão, publicada pela editora Springer de Berlim, o número 3-540-04999-1 (o "3" inicial indica um livro publicado em alemão); mas na sua versão em inglês, publicada pela Springer de Nova Iorque, o número é 0-387-04999-1.

É claro que a variação do comprimento das secções permite que os idiomas mais utilizados e que as grandes editoras tenham um número de identificação menor para que a terceira secção tenha mais algarismos permitindo assim catalogar um maior número de livros. Por exemplo, a língua inglesa é identificada somente pelo algarismo "0" e a editora McGraw-Hill tem um código de 2 algarismos, tendo assim à sua disposição 6 algarismos para identificar os seus livros, havendo pois espaço para 1 milhão de títulos. A Sociedade Americana de Matemática possui um número de identificação com 4 algarismos e pode portanto catalogar 10000 títulos.

Os editores e as livrarias usam o sistema ISBN de modo a que as encomendas, a catalogação, os inventários etc. sejam feitos de modo mais expedito e eficiente. Contudo, como observámos na secção anterior, estes códigos de identificação criam novos problemas: as pessoas ao manusear listas de números cometem, com frequência, alguns erros na sua leitura ou escrita; por exemplo ao dactilografá-los num computador, ao comunicá-los por telefone, etc., o que pode levar um cliente a encomendar um determinado livro e a receber outro. Para evitar estes e outros riscos que implicam gastos elevados o sistema precisa de ter capacidade para detectar tais erros. Aí reside a importância do algarismo de teste. Como veremos, com este sistema módulo 11 é possível detectar todos os erros que afectam um só algarismo e todas as trocas de algarismos adjacentes.

Para mais informações sobre o sistema ISBN consulte, por exemplo, o endereço `www.copyritepress.com/isbn.html`. O sítio `www.issn.org` contém informação pormenorizada sobre o sistema análogo ISSN de catalogação de revistas e em `www.nlc-bnc.ca/ismn` encontra ainda informação sobre o sistema ISMN de catalogação de publicações musicais.

O sistema UPC (EAN)

Sempre que vamos ao supermercado encontramos outro exemplo: o código de barras utilizado nos produtos comerciais (Figura 4). O sistema *Universal Product Code* (UPC) foi o primeiro código de barras profusamente adoptado. Foi criado nos E.U.A., em 1973, para ajudar os grandes supermercados a tornar mais eficiente o pagamento nas caixas e, simultaneamente, controlar melhor o inventário dos produtos em armazém. O sucesso do sistema conduziu rapidamente à sua implementação em todos os produtos vendidos a retalho e à sua difusão pelo mundo. Em 1976 adoptou-se na Europa o sistema análogo *European Article Number* (EAN). Este, por usar mais um algarismo, permite que um

número maior de produtos seja identificado, prevenindo assim uma maior longevidade do sistema. Aliás, a organização que gere o sistema UPC — o *Uniform Code Council* — anunciou já que prevê que o UPC se esgote por volta de 2005, altura em que os E.U.A. adoptarão o sistema EAN, como muitos outros países não europeus já fizeram, entretanto.

Olhando para o exemplo na Figura 4

Figura 4: Número de identificação UPC 01234567890 e algarismo de teste 5.

podemos ver que cada código de barras é constituído por duas partes:

- O número UPC ("*the human readable number*") com 12 algarismos.
- A codificação desse número por barras verticais, de modo a que o número seja legível por um leitor óptico. Qualquer pessoa com um pouco de paciência pode também lê-lo directamente (veja como em `howstuffworks.com/upc.htm` ou `www.uc-council.org`).

O primeiro algarismo do número UPC identifica o tipo de produto (por exemplo, "0" para mercearia e "3" para medicamentos), os cinco algarismos seguintes — 12345 — constituem o número de identificação do produtor e os seguintes cinco algarismos — 67890 — identificam o produto. Mais uma vez o último algarismo é o algarismo de teste. Este algarismo, como veremos, permite ao leitor óptico confirmar se leu o número correctamente e é calculado do seguinte modo:

$$3\times\underline{0}+1\times\underline{1}+3\times\underline{2}+1\times\underline{3}+3\times\underline{4}+1\times\underline{5}+3\times\underline{6}+1\times\underline{7}+3\times\underline{8}+1\times\underline{9}+3\times\underline{0}+1\times\underline{5}\equiv 0 \pmod{10},$$

isto é, o algarismo de teste a_{12} é o único inteiro no intervalo $[0,9]$ para o qual a soma de teste

$$\begin{aligned} S &= (a_1,a_2,a_3,a_4,a_5,a_6,a_7,a_8,a_9,a_{10},a_{11},a_{12})\cdot(3,1,3,1,3,1,3,1,3,1,3,1) \\ &= 3a_1+a_2+3a_3+a_4+\cdots+3a_{11}+a_{12} \end{aligned}$$

é divisível por 10. Portanto $a_{12}\equiv -(3a_1+a_2+3a_3+a_4+\cdots+3a_{11}) \pmod{10}$. (Em `www.uc-council.org/checkdig.htm` pode encontrar uma calculadora de algarismos de teste.)

No sistema EAN (`www.ean.be`) os números de identificação têm comprimento 13. Neste caso os primeiros 7 algarismos identificam o produtor, onde os primeiros 2 ou 3 identificam o país de origem; por exemplo "560" é o código atribuído a Portugal. O algarismo de teste a_{13} é dado por $a_{13} \equiv -(a_1+3a_2+a_3+3a_4+\cdots+a_{11}+3a_{12}) \pmod{10}$.

Em `www.uc-council.org`, `www.adams1.com/pub/russadam`, `www.cedar.buffalo.edu/Adserv/postcode.html` e `www.barcodeman.com/barspec.html`, poderá consultar mais informação sobre os códigos de barras em utilização, nomeadamente sobre diversas variantes dos sistemas UPC e EAN (como, por exemplo, os sistemas com supressão de zeros, que encurtando o tamanho dos números de identificação permitem a sua utilização em embalagens de dimensões reduzidas).

Classificação e estatística dos erros

Na base da concepção dos sistemas de identificação com algarismo de teste estão duas questões:

(1) Que erros poderão ocorrer na transmissão de mensagens com números?

(2) Com que frequência relativa ocorrem?

Como se trata de erros cometidos pelo homem, teremos que nos basear em estudos empíricos. Esses estudos foram feitos nos finais da década de 60. A Tabela 1 apresenta a classificação dos diversos tipos de erros identificados em [22] e respectivas frequências relativas. Os erros aleatórios são todos os erros não abrangidos pelas outras categorias. Erro singular significa um (e um só) algarismo errado. Uma transposição de algarismos adjacentes significa uma troca entre dois algarismos adjacentes: $\cdots ab \cdots \to \cdots ba \cdots$. Numa transposição intercalada acontece a troca entre dois algarismos que têm exactamente um algarismo a intercalá-los. Por exemplo, no número 123-5453 será natural trocar "54" com "53". Os erros fonéticos dependem evidentemente do idioma utilizado. O estudo em [22] restringe-se ao Inglês, Holandês e Alemão. São erros, por exemplo, do tipo $\cdots a0 \cdots \to \cdots 1a \cdots$ (para $a = 2, 3, \ldots, 9$): quando, pelo telefone, ao fazermos uma encomenda de Inglaterra informamos do nosso número de cartão de crédito o número "50" (*fifty*), por exemplo, pode ser entendido como "15" (*fifteen*), ou vice-versa.

Tipo de erro		**Frequência relativa (em %)**
Erros singulares	$a \to b$	79.1
Transposições de algar. adjacentes	$ab \to ba$	10.2
Transposições intercaladas	$acb \to bca$	0.8
Erros gémeos	$aa \to bb$	0.5
Erros fonéticos	$a0 \to 1a$ $(a = 2, 3, \ldots, 9)$	0.5
Erros gémeos intercalados	$aca \to bcb$	0.3
Erros aleatórios		8.6

Tabela 1: Tipos de erros mais comuns.

Estes estudos estatísticos também nos dizem que, a ocorrerem erros num número, a ocorrência de mais do que um é muito pouco provável.

Em certas circunstâncias um erro pouco habitual pode tornar-se muito comum. Por exemplo, na Suécia os números de identificação nos BI consistem em 6 algarismos para a data de nascimento (ano/mês/dia), seguidos de três algarismos para evitar duplicações. Muitas pessoas no entanto permutam os algarismos do ano com os do dia criando um erro do tipo $a_1a_2a_3a_4a_5a_6\cdots \to a_5a_6a_3a_4a_1a_2\cdots$. São especificidades locais que devem ser tidas em conta aquando da concepção de um sistema de identificação.

Como veremos, os erros singulares são totalmente detectados pela maioria dos sistemas em uso. Será pois importante, para uma comparação qualitativa dos diversos sistemas, conhecer, no conjunto dos erros duplos não aleatórios, a frequência relativa de cada um dos diversos tipos de erros duplos:

Tipo de erro	**Frequência relativa (em %)**
Transposições de algarismos adjacentes	83.0
Transposições intercaladas	6.7
Erros gémeos	4.4
Erros fonéticos	3.7
Erros gémeos intercalados	2.2

Tabela 2: Frequência dos erros duplos.

Análise do sistem ISBN

> *"— E se subtraíres um a trezentos e sessenta e cinco, quantos ficam?*
> *— Trezentos e sessenta e quatro, claro.*
> *Humpty Dumpty desconfiou da resposta.*
> *— Preferia ver isso num papel — disse ele."*
>
> Lewis Carroll, *Alice do Outro Lado do Espelho*

Os dados estatísticos acima apresentados dizem-nos que os erros singulares e as transposições de algarismos adjacentes constituem os erros mais frequentes. Normalmente acontecem separadamente, e uma só vez no máximo, em cada mensagem. O sistema ISBN foi concebido de modo a detectar estes erros (detecta mesmo qualquer transposição de algarismos não adjacentes):

Proposição 3.1. *Suponhamos que, na leitura de um dado número ISBN, ocorre um (e um só) dos dois seguintes erros: um erro singular ou uma transposição. Então a soma de teste não é um múltiplo de 11.*

Demonstração. A demonstração deste facto é simples:

- **Caso 1: "ocorre um erro singular"**

  Seja $a_1 \cdots a_i \cdots a_{10}$ um número ISBN e suponhamos que $a_1 \cdots a'_i \cdots a_{10}$ é o resultado da ocorrência de um erro singular na i-ésima posição ($i \in \{1, 2, \ldots, 10\}$). Denotemos por S e S', respectivamente, as somas de teste do número inicial (o número correcto) e do número errado. É claro que $S \equiv 0 (\mathrm{mod}\ 11)$ e $S' - S = (11 - i)(a'_i - a_i)$.

  Se suposermos que S' é múltiplo de 11 então, como 11 é primo, concluímos que 11 divide $11 - i$ ou divide $a'_i - a_i$, o que é um absurdo pois $11 - i$ e $a'_i - a_i$ são números inteiros não nulos entre -10 e 10. Logo S' não é múltiplo de 11.

- **Caso 2: "ocorre uma transposição"**

  Consideremos um número ISBN, $a_1 \cdots a_i \cdots a_j \cdots a_{10}$, e o resultado da ocorrência de uma transposição dos algarismos a_i e a_j nas posições i e j ($i \neq j$), $a_1 \cdots a_j \cdots a_i \cdots a_{10}$. Neste caso a diferença $S' - S$ entre as respectivas somas de teste é igual a

  $$(11 - i)a_j + (11 - j)a_i - (11 - i)a_i - (11 - j)a_j = (j - i)(a_j - a_i).$$

  A suposição de S' ser múltiplo de 11 é então absurda pois conduz-nos à conclusão de que um dos números $j - i$ ou $a_j - a_i$, que são números inteiros não nulos entre -10 e 10, é múltiplo de 11. Em conclusão, S' não é múltiplo de 11. ∎

Corolário 3.2. *Admitindo que na leitura dos números ISBN só ocorrem erros singulares ou transposições, não mais do que uma vez em cada número, então não ocorrem erros na leitura de um número ISBN se e só se a sua soma de teste (depois de lido o número) for múltipla de 11.* ∎

Portanto, na recepção de uma dada mensagem, contendo um número ISBN cuja soma de teste é múltipla de 11, a probabilidade desse número ser o correcto é muito elevada, atendendo aos dados estatísticos acima apresentados. Se, pelo contrário, a soma de teste não for um múltiplo de 11, o número que temos não é o número correcto.

Acabámos de ver como simples manipulações algébricas permitem provar que é possível a detecção dos erros referidos. Estas técnicas podem também ser utilizadas para averiguar que tipos de erros estes sistemas não detectam e desafiar-nos assim a conceber outros sistemas de identificação ainda mais eficientes.

Análise do sistema UPC (EAN)

Existe uma diferença de concepção entre o sistema UPC e o sistema ISBN. Este foi desenhado para detectar os erros mais frequentes cometidos por operadores humanos aquando da comunicação de números longos. Aquele foi desenhado de forma a ser lido e comunicado por máquinas (leitores ópticos). Testes de qualidade realizados garantem

que estas são muito precisas e, quando muito, cometem erros singulares. O sistema UPC foi pois concebido para detectar somente erros singulares.

Usando a técnica de subtrair à soma de teste de um número UPC a soma de teste de um número UPC com um erro singular, podemos facilmente concluir que este sistema detecta este tipo de erros. Na prova deveremos distinguir dois casos: quando o erro ocorre num algarismo com índice par ou quando o erro ocorre num algarismo de índice ímpar, pois a diferença $S' - S$ é igual a $3(a'_i - a_i)$, caso i seja par, e a $a'_i - a_i$, caso i seja ímpar. Se S' fosse múltiplo de 10, como S o é, teríamos $10|(S' - S)$. Mas quer $mdc(1, 10)$ quer $mdc(3, 10)$ são iguais a 1 logo, em ambos os casos, 10 dividiria $a'_i - a_i$, o que é absurdo, pois $a'_i - a_i$ é um número inteiro não nulo entre -9 e 9.

Este sistema não detecta todas as transposições de algarismos adjacentes, uma vez que $mdc(2, 10)$ não é igual a um. Por exemplo, se $a_9 = 6$ e $a_8 = 1$ forem trocados então $S' - S = 2(a_9 - a_8) = 2(6 - 1)$, que é um múltiplo de 10, enquanto que $a_9 - a_8$ o não é. Mais exactamente, as trocas de algarismos adjacentes $\cdots a_{i+1}a_i \cdots \quad \rightarrow \quad \cdots a_i a_{i+1} \cdots$ que este sistema não detecta são aquelas em que $|a_{i+1} - a_i| = 5$:

Com efeito, a diferença $S' - S$ é igual a $2(a_{i+1} - a_i)$, caso i seja par, e a $2(-a_{i+1} + a_i)$, caso i seja ímpar, pelo que

$$10|S' \Leftrightarrow 10|(S' - S) \Leftrightarrow 10|2(a_{i+1} - a_i) \Leftrightarrow |a_{i+1} - a_i| = 5.$$

Este sistema tem assim uma eficiência de $88,9\%$ na detecção deste tipo de erros, uma vez que detecta 80 das 90 possíveis transposições de algarismos adjacentes (não detecta os casos "05", "50", "16", "61", "27", "72", "38", "83", "49" e "94"). É, no entanto, um problema sem relevância prática, atendendo à precisão dos leitores ópticos assegurada pelos seus construtores.

Mais um exemplo: os cheques bancários

Cada cheque bancário tem três números legíveis por computador impressos na sua parte inferior. Um destes números é o número do cheque e um outro é o número da conta. O terceiro número é, no caso de alguns bancos, um número de nove algarismos $a_1\, a_2\, a_3\, a_4\, a_5\, a_6\, a_7\, a_8\, a_9$ identificando o banco emissor. Neste caso, o algarismo de teste a_9 (entre 0 e 9) é tal que

$$(7, 3, 9, 7, 3, 9, 7, 3, -1).(a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9) \equiv 0 \pmod{10}.$$

A prova de que este sistema também detecta todos os erros singulares é análoga às anteriores. Contudo, por causa dos quatro coeficientes diferentes 7,3,9 e -1, deveremos estudar quatro casos separadamente. O facto crucial na prova será o de que $mdc(7, 10) = mdc(3, 10) = mdc(9, 10) = mdc(-1, 10) = 1$. Este sistema não detecta todas as transposições de algarismos adjacentes uma vez que $mdc(4, 10)$, $mdc(6, 10)$ e $mdc(2, 10)$ não são iguais a 1. Também neste caso, as transposições de algarismos adjacentes $\cdots a_{i+1}a_i \cdots \rightarrow \cdots a_i a_{i+1} \cdots$ que este sistema não detecta são exactamente aquelas em que a diferença $a_{i+1} - a_i$ é, em módulo, igual a 5.

Para a análise de mais exemplos de sistemas de identificação detectores de erros consulte, por exemplo, [5], [6] e [20]. Em `www.adams1.com/pub/russadam/info.html` pode encontrar calculadoras dos algarismos de teste de alguns desses sistemas.

Análise geral dos sistemas modulares

"Generalizemos para simplificar, ou para compreender melhor!"

Jacques Hadamard

Todos estes sistemas de identificação possuem características comuns que nos permitem abordar o problema de uma forma geral e sistemática:

Fixemos um conjunto A de cardinal k, uma bijecção $\iota : A \to \mathbb{Z}_k$ e um n-uplo $(p_1, p_2, \ldots, p_n)$ de inteiros não nulos. Os números de identificação que constituem o sistema são palavras $a_1 a_2 \ldots a_n$ de comprimento n, definidas no alfabeto A, tais que

$$(p_1, p_2, \ldots, p_n) \cdot (\iota(a_1), \iota(a_2), \ldots, \iota(a_n)) \equiv 0 \pmod{k}$$

ou seja

$$p_1\iota(a_1) + p_2\iota(a_2) + \ldots + p_n\iota(a_n) \equiv 0 \pmod{k}$$

(não existe nenhuma razão significativa para usar o zero nesta condição; qualquer outro elemento de $\mathbb{Z}_k$ serve de modo equivalente.)

Aos sistemas deste tipo chamaremos *sistemas de identificação módulo k*. No sistema ISBN,

$$k = 11, A = \{0, 1, \ldots, 9, X\}, \iota(0) = 0, \iota(1) = 1, \ldots, \iota(9) = 9, \iota(10) = X$$

e

$$(p_1, p_2, \ldots, p_n) = (10, 9, 8, 7, 6, 5, 4, 3, 2, 1);$$

no sistema UPC,

$$k = 10, A = \{0, 1, \ldots, 9\}, \iota(a) = a \text{ e } (p_1, p_2, \ldots, p_n) = (3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1)$$

e, no sistema EAN,

$$k = 10, A = \{0, 1, \ldots, 9\}, \iota(a) = a \text{ e } (p_1, p_2, \ldots, p_n) = (1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1);$$

no caso dos cheques,

$$k = 10, A = \{0, 1, \ldots, 9\}, \iota(a) = a \text{ e } (p_1, p_2, \ldots, p_n) = (7, 3, 9, 7, 3, 9, 7, 3, -1).$$

Ao n-uplo $(p_1, p_2, \ldots, p_n)$ chama-se *vector de verificação de algarismos* do sistema. Para não sobrecarregarmos muito a notação cometeremos o abuso, daqui em diante, de identificar $\iota(a)$ com a.

Podemos imediatamente determinar quais os erros singulares e as transposições que são detectáveis:

Proposição 3.3. *Consideremos um número $a_1a_2\ldots a_n$ de um sistema de identificação módulo k.*

(a) *Um erro singular $a_i \to a_i'$ na i-ésima posição é detectável se e só se*

$$p_i(a_i' - a_i) \not\equiv 0 \pmod{k}.$$

(b) *Uma transposição dos algarismos a_i e a_j nas posições i e j é detectável se e só se*

$$(p_i - p_j)(a_j - a_i) \not\equiv 0 \pmod{k}.$$

Demonstração. (a) É óbvio pois a soma de teste do número incorrecto e a soma de teste do número correcto diferem de $p_i(a_i' - a_i)$. Portanto o erro será detectável se e só se $p_i(a_i' - a_i) \not\equiv 0 (\text{mod } k)$.

(b) Neste caso a diferença entre a soma de teste do número errado e a soma de teste do número correcto é igual a $(p_ia_j + p_ja_i) - (p_ia_i + p_ja_j) = (p_i - p_j)(a_j - a_i)$. Portanto, o erro é detectável se e só se $(p_i - p_j)(a_j - a_i) \not\equiv 0 \pmod{k}$. ∎

Podemos agora determinar facilmente as condições nos pesos p_i que assegurem a detecção de todos os erros de um determinado tipo.

Corolário 3.4. *Um sistema de identificação módulo k com vector de verificação de algarismos $(p_1, p_2, \ldots, p_n)$ detecta:*

(a) *Os erros singulares na posição i se e só se $mdc(p_i, k) = 1$;*

(b) *As transposições de algarismos nas posições i e j se e só se $mdc(p_i - p_j, k) = 1$.*

Demonstração. (a) Pela proposição anterior o sistema detecta todos os erros singulares na posição i se e só se para quaisquer $a_i, a_i' \in \{0, 1, \ldots, k-1\}$ com $a_i \neq a_i'$, $p_i(a_i' - a_i) \not\equiv 0 \pmod{k}$. Esta condição é claramente equivalente a $mdc(p_i, k) = 1$. De facto, sob aquela condição, se $mdc(p_i, k)$ fosse igual a $d > 1$ teríamos $p_i = dd_1$ e $k = dd_2$ com $d_2 \in \{1, 2, \ldots, k-1\}$. Fazendo $a_i' = d_2$ e $a_i = 0$ obteríamos a condição absurda $p_i(a_i' - a_i) \equiv 0 \pmod{k}$. Reciprocamente, sendo $mdc(p_i, k) = 1$, se existissem diferentes a_i e a_i' em $\{0, 1, \ldots, k-1\}$ tais que $k | p_i(a_i' - a_i)$ teríamos $k | (a_i' - a_i)$ o que é também absurdo pois $a_i' - a_i \in \{1, 2, \ldots, k-1\}$.

(b) Pela proposição anterior o sistema detecta todas as transposições dos algarismos nas posições i e j se e só se para quaisquer $a_i, a_j \in \{0, 1, \ldots, k-1\}$ com $a_i \neq a_j$, se tem $(p_i - p_j)(a_j - a_i) \not\equiv 0 \pmod{k}$. A prova da alínea anterior diz-nos que esta condição é equivalente a $mdc(p_i - p_j, k) = 1$. ∎

Analogamente podemos fazer o mesmo relativamente aos outros tipos de erros, obtendo a Tabela 3.

Tipo de erro	**Condições**
Erro singular: $a_i \to a_i'$	$mdc(p_i, k) = 1$
Transposição: $\ldots a_i \ldots a_j \ldots \to \ldots a_j \ldots a_i \ldots$	$mdc(p_i - p_j, k) = 1$
Erro gémeo: $aa \to bb$ (posições $i, i+1$)	$mdc(p_i + p_{i+1}, k) = 1$
Erro fonético: $a0 \to 1a$ (posições $i, i+1$), $a \in \{2, \ldots, k-1\}$	$ap_{i+1} \not\equiv (a-1)p_i (\mathrm{mod}\ k)$
Erro gémeo intercalado: $aca \to bcb$ (posições $i, i+2$)	$mdc(p_i + p_{i+2}, k) = 1$
Erro gémeo generalizado: $a \ldots a \to b \ldots b$ (posições i, j)	$mdc(p_i + p_j, k) = 1$

Tabela 3: Condições de detecção dos erros não aleatórios da Tabela 1.

Com esta tabela torna-se agora muito simples a qualquer pessoa desenhar sistemas de identificação modulares que detectem determinados tipos de erros. É também óbvio porque é que os sistemas módulo 11 são muito comuns. Esta tabela revela ainda porque é que os sistemas que usem $k < 10$ são pouco utilizados: é impossível que todos os erros singulares e todas as transposições sejam detectados se não obrigarmos todos os algarismos a variarem entre 0 e $k-1$, obrigação essa que torna o sistema pouco útil devido ao seu reduzido tamanho. Por exemplo, o sistema módulo 7 usado nos bilhetes de avião e pela *UPS* (veja [7]), como utiliza todos os algarismos entre 0 e 9 não distingue entre a_i e a_i' quando $|a_i - a_i'| = 7$.

No caso $k = 10$ as condições da Tabela 3 são incompatíveis: é impossível satisfazer a segunda se quisermos satisfazer a primeira pois, nesse caso, p_{i+1} e p_i são ímpares; além disso, a segunda condição também contradiz a quarta — de $mdc(p_{i+1} - p_i, 10) = 1$ pode deduzir-se a existência de $a \in \{2, 3, \ldots, 9\}$ tal que $p_i \equiv a(p_i - p_{i+1}) (\mathrm{mod}\ 10)$. É por isso que o sistema UPC detectando todos os erros singulares, só tem 88.9% de eficiência na detecção das transposições de algarismos adjacentes. Melhor eficácia num sistema destes é impossível. De facto, sendo p_{i+1} e p_i necessariamente ímpares, a diferença $p_{i+1} - p_i$ é um número par, digamos $2t$ ($t \in \mathbb{Z}$). Então, como

$$S' - S = (p_{i+1} - p_i)(a_i - a_{i+1}),$$

o sistema não detectará a transposição dos algarismos a_i e a_{i+1} se e só se $10 | 2t(a_i - a_{i+1})$, ou seja, não detectará nenhuma caso t seja múltiplo de 5 e, caso t não seja múltiplo de 5, não detectará aquelas em que $|a_i - a_{i+1}| = 5$. Portanto, qualquer sistema deste tipo que tenha 100% de eficiência na detecção dos erros singulares, detectará somente 88.9% das transposições adjacentes no caso em que a diferença entre os pesos p_{i+1} e p_i não seja múltipla de 10 ou, caso contrário, não detectará nenhuma.

4. Os Bilhetes de Identidade

> *"If you want a description of scientific method in three syllables, I propose: GUESS AND TEST."*
>
> George Pólya, *Mathematical Discovery* (Vol. II)

É agora relativamente simples adivinharmos qual o algoritmo utilizado nos BI. Trata-se de facto de um sistema de identificação modular; com uma boa amostra para testar, um simples programa de computador permitiu-nos resolver rapidamente essse puzzle: utiliza o mesmo algoritmo do ISBN,

$$k = 11 \text{ e } (p_1, p_2, \ldots, p_{10}) = (10, 9, 8, 7, 6, 5, 4, 3, 2, 1),$$

com uma simples (mas importante) diferença: o alfabeto utilizado é o conjunto $\{0, 1, \ldots, 9\}$; no caso, como acontece no meu BI (Figura 1), em que o algarismo de teste deveria ser igual a 10, em vez de usar um símbolo não numérico para identificar esse número, usa o algarismo 0 (!) o que evidentemente retira muita eficiência ao algoritmo.

É surpreendente que, apesar de existirem algoritmos muito eficazes como veremos já de seguida, o nosso Ministério da Justiça utilize um método tão pobremente concebido! Não acredite pois na frase "este é um algarismo de controle que permite detectar eventuais erros no seu número de identificação" que aparece algures no impresso de renovação dos BI.

5. As limitações dos sistemas modulares

> *"Alice desatou a rir.*
> *— Não vale a pena tentar — disse. — Ninguém pode acreditar em coisas impossíveis.*
> *— Cá a mim parece-me que não tens muita prática — opinou a Rainha. — Quando eu era da tua idade, treinava quatro horas e meia por dia, e às vezes chegava a imaginar seis coisas impossíveis antes do pequeno-almoço."*
>
> Lewis Carroll, *Alice do Outro Lado do Espelho*

Como acabámos de observar (Tabela 3), os sistemas módulo 11 são melhores que os sistemas módulo 10. É o caso do sistema ISBN que detecta a 100% todos os erros singulares, transposições e erros gémeos. No entanto, a taxa de detecção dos erros fonéticos é inferior a 100%. Por isso alguns bancos alemães como o *Dresdner Bank* usam um outro sistema módulo 11 onde o vector de verificação de algarismos é a progressão geométrica módulo 11 das potências de 2, que é preferível pois 2 é uma raiz primitiva módulo 11:

$$(2, 2^2, 2^3, 2^4, \ldots, 2^{10}) = (2, 4, 8, 5, 10, 9, 7, 3, 6, 1).$$

Este sistema detecta a 100% todos os erros da Tabela 1. No que concerna pois aos erros não aleatórios da Tabela 1, não existe melhor código que este (chamado *sistema módulo 11 geométrico*). Existem no entanto outros tipos de erros (veja Tabela 4), incluídos nos erros aleatórios da Tabela 1, onde a eficiência deste método módulo 11 não é total.

Tipo de erro	**Descrição**
Erro duplo: $a_i \dots a_j \to a'_i \dots a'_j$	Erro singular em duas posições i e j.
Transposição dupla: $a_i a_{i+1} \dots a_j a_{j+1} \to a_j a_{j+1} \dots a_i a_{i+1}$	Transposições de pares de algarismos adjacentes.
Translacção: $\dots a_i a_{i+1} a_{i+2} \dots \to \dots (a_i+q)(a_{i+1}+q)(a_{i+2}+q) \dots$	Modificação de algarismos adjacentes pela mesma quantidade q.

Tabela 4: Erros duplos generalizados.

O sistema seguinte, que utiliza contudo dois algarismos de teste, melhora o método módulo 11 geométrico relativamente à detecção destes erros [11]:

Sendo $a_1 a_2 \dots a_n$ um número de identificação do sistema, a_{n-1} e a_n são algarismos de teste; em primeiro lugar deve calcular-se a_{n-1}, que protege o número $a_1 a_2 \dots a_{n-2}$, pela fórmula

$$a_{n-1} \equiv \Big(1 - \sum_{i=1}^{n-2} (a_i \times 5^{[\frac{i}{2}]})\Big) \pmod{11},$$

e depois

$$a_n \equiv \Big(1 - \sum_{i=1}^{n-2} (a_i \times 2^{i+1}) - a_{n-1} \times 2\Big) \pmod{11},$$

que protege o número $a_1 a_2 \dots a_{n-1}$.

No caso $n \leq 11$, além de continuar a detectar os mesmos erros que o método módulo 11 geométrico, detecta ainda todos os erros duplos e translações [11]. As transposições duplas são detectadas com uma única excepção. Para mais pormenores consulte [4] e [11].

Existem também alguns sistemas em utilização onde $k > 11$. Isso torna-se uma necessidade quando, por exemplo, os números de identificação requerem um alfabeto maior do que $\{0, 1, \dots, 9\}$, nomeadamente caracteres não numéricos. É natural que os melhores sejam aqueles em que k é um número primo. Um sistema módulo 13 usado no *Hospital de Rhode Island* é descrito em [14]. Sistemas módulo 37, 43, 97 ou 103 são também utilizados (encontra mais pormenores em [1], [4] e [17]):

(1) *Sistema módulo 37*

Neste caso $A = \{0, 1, \dots, 9, a, b, \dots, z, *\}$. Com este sistema podemos usar letras nos números de identificação. O cardinal de A é 37. Felizmente 37 é um número primo e 2 é uma raiz primitiva módulo 37. Toma-se então $p_i = 2^i$ como no sistema módulo 11 geométrico. Fazendo a correspondência $a \mapsto 10, b \mapsto$

$11, \ldots, z \mapsto 35, * \mapsto 36$ trabalhamos em $\mathbb{Z}_{37}$. Por exemplo, o número de identificação $abcd1234$ terá como algarismo de teste o símbolo c pois

$$2\times\underline{10}+2^2\times\underline{11}+2^3\times\underline{12}+2^4\times\underline{13}+2^5\underline{1}+2^6\times\underline{2}+2^7\times\underline{3}+2^8\times\underline{4} = 1936 \equiv 12 \pmod{37}.$$

Todos os erros não aleatórios da Tabela 1 são detectáveis.

(2) *Código 39* (`www.barcodeman.com/c39_1.html`)

Trata-se de um código de barras (cada carácter é representado por 9 barras, 3 das quais mais largas que as outras, daí o nome do sistema), que suporta 43 caracteres:

$$A = \{0, 1, \ldots, 9, A, B, C, \ldots, X, Y, Z, -, ., *, \$, /, +, \%\}.$$

É um sistema módulo 43 onde cada p_i é igual a 1. Por exemplo o número $12345ABCDE/$ terá como algarismo de teste a letra T pois $1 + 2 + 3 + 4 + 5 + 10 + 11 + 12 + 13 + 14 + 40 = 115 \equiv 29 \pmod{43}$.

(3) *Sistema módulo 97*

Neste caso o alfabeto é o subconjunto $\{0, 1, \ldots, 9\}$ de $\mathbb{Z}_{97}$. Como o vector de verificação de algarismos é dado por $p_i = 10^{i-1}$, cada número de identificação, antes do cálculo do algarismo de teste, pode ser interpretado como um número decimal. Por exemplo, se 56643 é tal número de identificação então $56643 \equiv 92 (\text{mod } 97)$ e 92 é o número de teste com dois algarismos. Como 97 é um número primo, todos os erros não aleatórios da Tabela 1 são detectados.

(4) *Código de barras 128*

É um sistema de identificação módulo 103 que permite a utilização de todos os 128 caracteres ASCII. Para mais informações consulte `www.adams1.com/pub/russadam/128code.html`.

Uma grande empresa de correio tem também implementado um sistema com 4 algarismos de teste [23].

O problema dos sistemas modulares com $k > 10$ reside na necessidade de utilizar mais do que um algarismo de teste ou, alternativamente, de introduzir caracteres não numéricos para os algarismos de teste (como, por exemplo, no caso ISBN no qual se considera a letra X para representar o número 10) o que acarreta alguns problemas técnicos e custos mais elevados na concepção dos sistemas automáticos de leitura e escrita. O uso da letra X poderá ainda ser problemático pela seguinte razão: para o titular de um dado número de BI ou de um dado número de conta bancária, ou para um paciente num banco de dados médico, a letra X poderá sugerir uma "marca especial", deveras desagradável. A maioria dos sistemas módulo 11 em utilização evita isso: uns não utilizando os números de identificação cujo algarismo de teste é superior a 9 (é o caso, por exemplo, do *Dresdner Bank* nos seus números de conta), outros fazendo o

algarismo de teste igual a 0 no caso em que deveria ser 10 (caso dos BI e de alguns bancos).

Esta desvantagem dos sistemas módulo k, com $k > 10$, levanta o problema interessante de averiguar se é possível conceber um sistema módulo 10 que detecte todos os erros singulares e transposições (que são na prática os mais relevantes, como vimos). Já observámos no final da Secção 3 que com um sistema deste tipo tal é impossível. Só alterando (generalizando) o sistema poderemos ter esperança de encontrar tal solução.

Para isso recordemos que para detectar todos os erros singulares somos forçados a escolher pesos p_i relativamente primos com k. Isto significa que

$$0, p_i \times 1, p_i \times 2, \ldots, p_i \times (k-1)$$

constitui uma permutação dos elementos do grupo $(\mathbb{Z}_k, \oplus_k)$. Este facto motiva a seguinte definição, que generaliza este tipo de sistemas substituindo a função $x \mapsto p_i x$ por uma função arbitrária $x \mapsto \phi_i(x)$:

Definição 5.1. Sejam A um conjunto de cardinal k, $\iota : A \to \mathbb{Z}_k$ uma bijecção e $(\phi_1, \phi_2, \ldots, \phi_n)$ um n-uplo de aplicações de $\mathbb{Z}_k$ em $\mathbb{Z}_k$. Um *sistema de identificação módulo k* é constituído pelas palavras em A de comprimento n, $a_1 a_2 \ldots a_n$, satisfazendo a condição de teste

$$\phi_1(\iota(a_1)) + \phi_2(\iota(a_2)) + \cdots + \phi_n(\iota(a_n)) \equiv 0 \pmod{k}$$

ou seja

$$\phi_1(\iota(a_1)) \oplus_k \phi_2(\iota(a_2)) \oplus_k \cdots \oplus_k \phi_n(\iota(a_n)) = 0.$$

Continuaremos a cometer o abuso de identificar $\iota(a)$ com a. Note-se que o algarismo de teste a_n é então igual a

$$\phi_n^{-1}\Big(-\sum_{i=1}^{n-1} \phi_i(a_i)\Big).$$

A $(\phi_1, \phi_2, \ldots, \phi_n)$ chama-se *vector de verificação de algarismos* do sistema.

Esta generalização deve-se à IBM que concebeu há alguns anos um sistema deste tipo, hoje utilizado em cartões de crédito, bibliotecas, bancos de sangue, farmácias e bancos. Todos os sistemas modulares que discutimos anteriormente são descritos por esta definição. É muito simples reescrevermos as condições da Tabela 3 de detecção dos diversos tipos de erros. Por exemplo, no caso dos erros singulares $a_i \to a_i'$ (com $a_i \neq a_i'$) terá que ser $\phi_i(a_i) \neq \phi_i(a_i')$, ou seja, ϕ_i terá que ser uma permutação. No caso das transposições, se $a_i \neq a_j$ então $\phi_i(a_i) + \phi_j(a_j) \neq \phi_i(a_j) + \phi_j(a_i)$ ou seja $\phi_i(a_i) - \phi_j(a_i) \neq \phi_i(a_j) - \phi_j(a_j)$, isto é, a aplicação $x \mapsto \phi_i(x) - \phi_j(x)$, $x \in \mathbb{Z}_k$, é injectiva e, portanto, bijectiva.

A tabela seguinte resume essas condições:

Tipo de erro	**Condições**
Erro singular	ϕ_i é uma permutação
Transposição	$x \to \phi_i(x) - \phi_j(x)$ define uma permutação
Erro gémeo	$x \to \phi_i(x) + \phi_{i+1}(x)$ define uma permutação
Erro gémeo intercalado	$x \to \phi_i(x) + \phi_{i+2}(x)$ define uma permutação

Tabela 5: Condições de detecção de alguns erros não aleatórios da Tabela 1.

Como a aplicação

$$\begin{array}{ccc} \mathbb{Z}_k & \to & \mathbb{Z}_k \\ x & \mapsto & p_i x \end{array}$$

é uma permutação se e só se $mdc(p_i, k) = 1$, imediatamente se observa como estas condições generalizam as anteriores.

Listemos alguns exemplos de sistemas de identificação módulo 10, retirados de [4] (as respectivas taxas de detecção de erros são apresentadas na Tabela 6):

(1) *Sistema IBM*

Desenvolvido pela IBM, é definido por $k = 10$ e pelo vector de verificação de algarismos $(\ldots, \phi, id, \phi, id)$, sendo ϕ a permutação

$$\phi(x) = \begin{cases} 2x & \text{se } 2x < 10 \\ 2x - 9 & \text{se } 2x \geq 10 \end{cases}$$

ou seja, $\phi = (124875)(36)$.

Figura 5: Número de identificação 110010002112 e algarismo de teste 7 satisfazendo $1 + \phi(1) + 0 + \phi(0) + 1 + \phi(0) + 0 + \phi(0) + 2 + \phi(1) + 1 + \phi(2) + 7 \equiv 0 \pmod{10}$.

Detecta todas as transposições de algarismos adjacentes, com excepção dos casos "09" e "90", ou seja, detecta 88 casos em 90. Tem pois uma taxa de detecção de transposições adjacentes de 97.8%, melhor do que os 88.9% que tínhamos no sistema UPC/EAN. Por outro lado não detecta nenhuma transposição intercalada.

(2) *Sistema IBM generalizado*

É um melhoramento do sistema (1), uma vez que já permite detectar muitas transposições intercaladas. Usa no vector de verificação de algarismos potências de ϕ:

$$(\phi^{n-1}, \phi^{n-2}, \ldots, \phi, id).$$

(3) *Sistema IBM generalizado modificado*

A permutação ϕ é modificada do seguinte modo:

$$\delta(x) = \phi(x) + 6 \pmod{10}, \quad x = 0, 1, \ldots, 9,$$

isto é, $\delta = (069571832)$.

A equação de teste mantém-se, substituindo ϕ por δ.

(4) *Sistema Colenbrander*

A permutação ϕ é novamente modificada:

$$\lambda(x) = \phi(x + 4 \pmod{10}) + 1 \pmod{10}, \quad x = 0, 1, \ldots, 9,$$

ou seja, $\lambda = (0973612485)$.

O vector de verificação será $(\lambda^{n-1}, \lambda^{n-2}, \ldots, \lambda, id)$. Este sistema já detecta 100% dos erros fonéticos, o que não acontece com o sistema IBM generalizado modificado.

(5) *Método PTT (banco alemão)*

Trata-se de um dos métodos mais exóticos em utilização. Utiliza três permutações: $\phi_1 = (0123456789), \phi_2 = (026387514)$ e $\phi_3 = (0316)(29857)$.

Estas permutações são o resultado da sucessiva aplicação de funções módulo 11 e módulo 10:

$$\phi_i(x) = (i(x + 1) \pmod{11}) \pmod{10}.$$

O vector de verificação de algarismos é igual a $(\phi_1, \phi_2, \phi_3, \phi_1, \phi_2, \phi_3, \phi_1, \phi_2, -id)$ sendo $k = 10$.

Para mais exemplos consulte [4] e [6].

Sistema	$ab \to ba$	$acb \to bca$	$aa \to bb$	$a0 \to 1a$	$aca \to bcb$
UPC, EAN	88.9	0	88.9	100	88.9
vector $(1, 3, 7, 1, 3, 7, \ldots)$	88.9	88.9	59.3	100	59.3
vector $(1, 3, 9, 7, 1, 3, 9, 7, \ldots)$	88.9	88.9	88.9	100	0
IBM	97.8	0	93.3	87.5	88.9
IBM generalizado	97.8	82.8	93.3	89.6	95.6
PTT	96.3	96.3	95.6	95.8	95.6
IBM generalizado modif.	97.8	95.6	93.3	90.3	95.6
Colenbrander	97.8	95.6	93.3	100	95.6

Tabela 6: Taxas de detecção de erros de diversos sistemas módulo 10 (em %).

Portanto o sistema IBM, bem como o IBM generalizado, o IBM generalizado modificado e o Colenbrander, tem respectivamente taxas de 100% e 97, 8% na detecção dos erros singulares e transposições adjacentes. Melhor é impossível num sistema módulo 10. De facto, a condição $\phi_i(a) - \phi_j(a) \neq \phi_i(b) - \phi_j(b)$, sendo simétrica, diz-nos que se não detecta a troca "*ab*" também não detecta a troca "*ba*"; portanto melhor que os 88 casos que o sistema IBM detecta só a totalidade dos 90 casos, o que é impossível pelo seguinte teorema de Gumm [9]:

Teorema 5.2. *Seja k um inteiro par. Qualquer sistema de identificação módulo k que detecte todos os erros singulares não detecta todas as transposições; mais concretamente, para quaisquer i e j existe uma transposição envolvendo os algarismos das posições i e j que não é detectável.*

Demonstração. Suponhamos $k = 2t$. Uma vez que o sistema detecta todos os erros singulares, todas as funções ϕ_i são permutações. Além disso, se detectasse todas as transposições envolvendo as posições i e j, a função dada por $\phi(x) = \phi_i(x) - \phi_j(x)$ seria uma permutação de $\mathbb{Z}_{2t}$. Mas então obteríamos uma contradição. De facto, adicionando (módulo $2t$) os elementos de $\mathbb{Z}_{2t}$, teríamos

$$\sum_{x \in \mathbb{Z}_{2t}} x = (0 + t) + (1 + 2t - 1) + (2 + 2t - 2) + \cdots + (t - 1 + t + 1) = t$$

e, por outro lado, sendo ϕ uma permutação de $\mathbb{Z}_{2t}$,

$$\sum_{x \in \mathbb{Z}_{2t}} x = \sum_{x \in \mathbb{Z}_{2t}} \phi(x) = \sum_{x \in \mathbb{Z}_{2t}} (\phi_i(x) - \phi_j(x)) = \sum_{x \in \mathbb{Z}_{2t}} \phi_i(x) - \sum_{x \in \mathbb{Z}_{2t}} \phi_j(x) = t - t = 0.$$

■

Em conclusão, só alterando (ou generalizando) estes algoritmos poderemos eventualmente obter um método que, no caso $k = 10$, detecte a 100% os erros singulares e as transposições de algarismos adjacentes.

6. A solução: sistemas de identificação definidos sobre um grupo

> *"— Quem é que te recitou todas essas coisas tão difíceis?*
> *— Li eu num livro — respondeu Alice."*
>
> Lewis Carroll, *Alice do Outro Lado do Espelho*

Olhando para a Definição 5.1 observamos que o essencial é a estrutura do grupo $(\mathbb{Z}_k, \oplus_k)$. Podemos então generalizar a definição mais uma vez, substituindo esta estrutura — que, como acabámos de concluir, não se revela suficientemente rica para permitir detectar a 100% todos os erros singulares e todas as transposições adjacentes

— por um grupo arbitrário $(G, \cdot)$. O problema resumir-se-á então em determinar, caso existam, grupos com uma estrutura mais rica para este fim. Como veremos nesta secção, existem de facto grupos com essa estrutura, que permitem atingir taxas de 100%. Esta ideia deve-se originalmente a Verhoeff [22], que provou em 1969 que o sistema que se obtém utilizando o grupo não-abeliano de ordem 10 detecta todos os erros singulares e transposições adjacentes, sem a necessidade de introduzir um símbolo não numérico para algarismo de teste.

Definição 6.1. Sejam $(G, \cdot)$ um grupo de ordem k, $(\phi_1, \phi_2, \ldots, \phi_n)$ um n-uplo de aplicações $\phi_i : G \to G$ e $c \in G$. Um *sistema de identificação de base G* é constituído pelas palavras em G de comprimento n, $a_1 a_2 \ldots a_n$, tais que

$$\phi_1(a_1) \cdot \phi_2(a_2) \cdot \cdots \cdot \phi_n(a_n) = c.$$

Proposição 6.2. *Seja $a_1 a_2 \ldots a_n$ um número de identificação de um sistema de identificação de base G.*

(a) *Um erro singular $a_i \to a_i'$ na i-ésima posição é detectável se e só se $\phi_i(a_i) \neq \phi_i(a_i')$.*

(b) *Uma transposição dos elementos adjacentes a_i e a_{i+1} é detectável se e só se*

$$\phi_i(a_i)\phi_{i+1}(a_{i+1}) \neq \phi_i(a_{i+1})\phi_{i+1}(a_{i+1}).$$

Demonstração. (a) Sendo S a soma de teste do número correcto e S' a soma de teste do número errado, o erro será detectável se e só se $S' \neq c$ ou seja $S'S^{-1} \neq e$ (onde e denota o elemento neutro de G). Mas

$$S'S^{-1} = \phi_1(a_1) \cdots \phi_{i-1}(a_{i-1})\phi_i(a_i')\phi_i(a_i)^{-1}\phi_{i-1}(a_{i-1})^{-1} \cdots \phi_1(a_1)^{-1},$$

pelo que $S'S^{-1} \neq e$ se e só se $\phi_i(a_i) \neq \phi_i(a_i')$, pois $\phi_1(a_1) \cdots \phi_{i-1}(a_{i-1})$ e

$$\phi_{i-1}(a_{i-1})^{-1} \cdots \phi_1(a_1)^{-1}$$

são inversos um do outro.

(b) Neste caso $S'S^{-1}$ é igual a

$$\phi_1(a_1) \cdots \phi_{i-1}(a_{i-1})\phi_i(a_{i+1})\phi_{i+1}(a_i)\phi_{i+1}(a_{i+1})^{-1}\phi_i(a_i)^{-1} \cdots \phi_1(a_1)^{-1}$$

pelo que o erro é detectável se e só se $\phi_i(a_{i+1})\phi_{i+1}(a_i)\phi_{i+1}(a_{i+1})^{-1}\phi_i(a_i)^{-1} \neq e$.

■

Então, imediatamente:

Corolário 6.3. (a) *O sistema detecta todos os erros singulares na posição i se e só se ϕ_i é uma permutação de G.*

(b) *O sistema detecta todas as transposições de elementos adjacentes nas posições i e $i+1$ se e só se $a \neq b \Rightarrow \phi_i(a)\phi_{i+1}(b) \neq \phi_i(b)\phi_{i+1}(a)$.* ■

Assim para assegurar que todos os erros singulares são detectados basta assumir que cada ϕ_i é uma permutação. Neste caso, substituindo $\phi_i(a)$ por a e $\phi_i(b)$ por b em (b) podemos concluir que detecta todas as transposições de elementos nas posições adjacentes i e $i+1$ se e só se

$$a \neq b \Rightarrow a\,\phi_{i+1}\phi_i^{-1}(b) \neq b\,\phi_{i+1}\phi_i^{-1}(a). \tag{6.3.1}$$

Quando G é abeliano esta condição pode ser simplificada:

Corolário 6.4. *Suponhamos que G é abeliano e que cada ϕ_i é uma permutação. Então as seguintes condições são equivalentes:*

(a) *O sistema detecta todas as transposições de elementos nas posições adjacentes i e $i+1$.*

(b) *$x \mapsto \phi_i(x)\phi_{i+1}(x)^{-1}$ define uma permutação de G.*

(c) *$x \mapsto x(\phi_{i+1}\phi_i^{-1}(x))^{-1}$ define uma permutação de G.* ∎

De modo análogo, podemos obter as condições para a detecção dos outros tipos de erros, que listamos nas Tabelas 7 (caso geral) e 8 (caso abeliano).

Tipo de erro	**Condições**
Transposição	$a \neq b \Rightarrow a\,\phi_{i+1}\phi_i^{-1}(b) \neq b\,\phi_{i+1}\phi_i^{-1}(a)$
Transposição intercalada	$a \neq c \Rightarrow a\,b\,\phi_{i+2}\phi_i^{-1}(c) \neq c\,b\,\phi_{i+2}\phi_i^{-1}(a)$
Erro gémeo	$a \neq b \Rightarrow a\,\phi_{i+1}\phi_i^{-1}(a) \neq b\,\phi_{i+1}\phi_i^{-1}(b)$
Erro gémeo intercalado	$a \neq c \Rightarrow a\,b\,\phi_{i+2}\phi_i^{-1}(a) \neq c\,b\,\phi_{i+1}\phi_i^{-1}(c)$

Tabela 7: Condições de detecção de erros (num grupo arbitrário).

Tipo de erro	**Condições**
Transposição	$x \mapsto x(\phi_{i+1}\phi_i^{-1}(x))^{-1}$ define uma permutação
Transposição intercalada	$x \mapsto x(\phi_{i+2}\phi_i^{-1}(x))^{-1}$ define uma permutação
Erro gémeo	$x \mapsto x\phi_{i+1}\phi_i^{-1}(x)$ define uma permutação
Erro gémeo intercalado	$x \mapsto x\phi_{i+2}\phi_i^{-1}(x)$ define uma permutação

Tabela 8: Condições de detecção de erros (num grupo abeliano).

No caso abeliano, o Teorema 5.2 de Gumm é imediatamente generalizável a este contexto:

Teorema 6.5. *Seja G um grupo abeliano finito tal que $\prod_{x\in G} x \neq e$. Qualquer sistema de identificação de base G que detecte todos os erros singulares não detecta todas as transposições adjacentes.*

Demonstração. Uma vez que o sistema detecta todos os erros singulares, todas as funções ϕ_i são permutações. Além disso, se detectasse todas as transposições envolvendo as posições i e $i+1$, a função ϕ dada por $\phi(x) = \phi_i(x)\phi_{i+1}(x)^{-1}$ seria uma permutação de G. Mas então obteríamos uma contradição. De facto, por um lado $\prod_{x\in G} x \neq e$ e, por outro lado, sendo ϕ uma permutação de G, $\prod_{x\in G} x$ coincide com $\prod_{x\in G} \phi(x)$ e, consequentemente, é igual a

$$\prod_{x\in G}\left(\phi_i(x)\phi_{i+1}(x)^{-1}\right) = (\prod_{x\in G}\phi_i(x))(\prod_{x\in G}\phi_{i+1}(x))^{-1} = (\prod_{x\in G} x)(\prod_{x\in G} x)^{-1} = e.$$

■

Voltando ao caso geral, a condição (6.3.1) diz-nos que a permutação $\theta = \phi_{i+1}\phi_i^{-1}$ satisfaz a condição

$$a \neq b \Rightarrow a\,\theta(b) \neq b\,\theta(a).$$

A qualquer permutação de G que satisfaça esta condição chama-se *permutação anti-simétrica* [6]. É evidente que a existência de um sistema de base G que detecte todos os erros simples e transposições adjacentes é equivalente à existência de uma permutação de G anti-simétrica: existindo tal permutação, bastará considerar para vector de verificação de algarismos o n-uplo $(\theta^{n-1}, \ldots, \theta, id)$. (No caso abeliano, $(\ldots, \theta, id, \theta, id)$ também serve; o primeiro será mais eficiente na detecção de outros tipos de erros duplos.)

As permutações anti-simétricas estão intimamente relacionadas com as permutações completas [12]. Uma permutação ϕ diz-se *completa* caso $a \neq b$ implique $a\,\phi(a) \neq b\,\phi(b)$. Num grupo abeliano a existência de uma permutação anti-simétrica é equivalente à existência de uma permutação completa pois θ é anti-simétrica se e só se a permutação $x \mapsto \theta(x)^{-1}$ é completa.

Portanto, um sistema de identificação de base num grupo abeliano G detecta todos os erros singulares e as transposições adjacentes se e só se existe uma permutação completa de G. Fazendo uma abordagem análoga na análise dos outros tipos de erros duplos da Tabela 7 obtem-se:

Teorema 6.6. (Ecker e Poch [4]) *Um sistema de identificação de base num grupo abeliano G detecta os erros singulares e um dos tipos de erros duplos da Tabela 7 com uma eficiência de* 100% *se e só se existe uma permutação de G completa.*

O conceito de permutação completa foi introduzido por H. B. Mann em [12]; curiosamente o problema da existência de permutações completas de um grupo abeliano finito foi totalmente resolvido por L. J. Paige há muito tempo atrás, sem qualquer motivação utilitária:

Teorema 6.7. (Paige [13]) *Seja G um grupo abeliano finito. As seguintes asserções são equivalentes:*

(i) *Existe uma permutação de G completa.*

(ii) *G não possui elementos de ordem 2 ou possui mais do que um.*

(iii) $\prod_{x\in G} x = e$.

Este teorema garante-nos que, no caso abeliano, a condição $\prod_{x\in G} x = e$ do Teorema 6.5 é, além de necessária, suficiente para que o sistema detecte todos os erros singulares e todas as transposições adjacentes.

É óbvio, pela condição (ii), que existe uma permutação completa do grupo cíclico C_n de ordem n se e só se n é ímpar. Mais geralmente:

Lema 6.8. *Seja $G = C_{\beta_1} \oplus C_{\beta_2} \oplus \cdots \oplus C_{\beta_t}$ $(t > 1)$, onde $\beta_1|\beta_2|\cdots|\beta_t$. Então existe uma permutação completa de G se e só se β_t é ímpar ou β_{t-1} é par.*

Demonstração. Suponhamos que G possui uma permutação completa e que β_t é par. Nesse caso $(e, e, \ldots, e, \frac{\beta_t}{2})$ tem ordem 2 pelo que G possui pelo menos um elemento $(a_1, a_2, \ldots, a_t) \neq (e, e, \ldots, e, \frac{\beta_t}{2})$ de ordem 2. Necessariamente existe $i \leq t-1$ tal que $a_i \neq e$ pelo que C_{β_i} possui um elemento de ordem 2. Logo β_i é par. Como $\beta_i|\beta_{t-1}$ também β_{t-1} é par.

Reciprocamente, se β_t é ímpar não existe nenhum elemento de ordem 2 em C_{β_t} pelo que também tais elementos não existem em G. Se β_{t-1} é par então $(e, \ldots, e, \frac{\beta_{t-1}}{2}, e)$ e $(e, \ldots, e, e, \frac{\beta_t}{2})$ são elementos de G de ordem 2. ∎

Uma vez que, como se pode provar facilmente, um grupo abeliano finito G não possui elementos de ordem 2 se e só se $|G|$ é ímpar e que se G possui mais do que um elemento de ordem 2 então $|G|$ é múltiplo de 4 (o recíproco é no entanto falso; basta considerar o grupo $\mathbb{Z}_4$), podemos concluir que, se existe uma permutação completa de G, $|G|$ é ímpar ou é múltiplo de 4.

Apesar de não existirem permutações completas de todo o grupo de ordem múltipla de 4, o Teorema de Estrutura dos grupos abelianos finitos ([19], Teorema 8.14), que afirma que todo o grupo abeliano finito de ordem maior do que 1 é soma directa de grupos cíclicos não triviais $C_{\beta_1}, C_{\beta_2}, \ldots, C_{\beta_t}$, que podem ser escolhidos de tal forma que β_i divide β_{i+1} para $i = 1, 2, \ldots, t-1$ (os números $\beta_1, \beta_2, \ldots, \beta_t$ dizem-se os *factores invariantes* do grupo), e o Lema 6.8 permitem-nos afirmar o seguinte:

Teorema 6.9. *Seja $k \in IN$. Existe um grupo abeliano finito de ordem k com uma permutação completa se e só se k é ímpar ou é um múltiplo de 4.*

Demonstração. Observámos já que a condição "k é ímpar ou é um múltiplo de 4" é necessária para que exista um grupo abeliano finito de ordem k com uma permutação completa. Vejamos que é também suficiente:

Suponhamos que k é ímpar. Como o produto dos factores invariantes $\beta_1, \beta_2, \ldots, \beta_t$ de qualquer grupo G de ordem k é igual a k é evidente que β_t será ímpar pelo que existirá uma permutação de G completa.

No caso $4|k$ é também claro que existirá um grupo $G = C_{\beta_1} \oplus C_{\beta_2} \oplus \ldots \oplus C_{\beta_t}$ $(t > 1)$ de ordem k cujos factores invariantes β_{t-1} e β_t são pares (basta considerar um grupo que tenha pelo menos um divisor elementar [19] igual a 2; por exemplo, $C_2 \oplus C_{\frac{k}{2}}$). ∎

Portanto, se quisermos responder à questão inicial que nos motiva (a construção de um método num alfabeto de cardinal 10 que detecte a 100% os erros singulares e as transposições adjacentes) teremos que efectuar a procura nos grupos não-abelianos. Felizmente, nestes grupos a resposta é afirmativa. De facto, apesar do problema da determinação dos grupos onde existem permutações anti-simétricas não estar completamente resolvido, sabe-se (cf. [8], [10]) que existem permutações anti-simétricas de, por exemplo, todos os grupos de ordem ímpar (neste caso $x \mapsto x^{-1}$ define uma permutação anti-simétrica; no caso abeliano, $x \mapsto x$ define uma permutação completa), grupos alternantes A_n $(n \geq 3)$, grupos simétricos S_n $(n \geq 3)$, grupos solúveis não-abelianos e grupos diedrais. Em [8] os autores conjecturam mesmo que todos os grupos não-abelianos possuem tais permutações. Uma vez que, para cada inteiro $m \geq 3$, existe um grupo diedral D_m de ordem $2m$ (grupo das simetrias de um polígono regular com m lados [19]), temos nestes grupos a solução que faltava para o caso em que $k \geq 3$ é par e não é múltiplo de 4. Logo:

Teorema 6.10. *Para cada inteiro $k \geq 3$ existe um grupo de ordem k com uma permutação anti-simétrica.* ∎

Existe assim, para qualquer alfabeto de cardinal superior a 2, um método 100% eficiente na detecção de todos os erros singulares e transposições adjacentes. No caso $k = 10$ basta tomar o grupo D_5 das simetrias de um pentágono regular. Este grupo é formado pelos elementos ρ_i e α_i $(i \in \{1, 2, 3, 4, 5\})$, sendo ρ_i a reflexão sobre o eixo r_i (Figura 6) e α_i a rotação de ângulo $\frac{i-1}{5} \times 2\pi$.

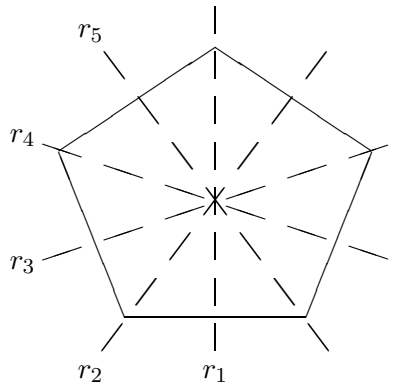

Figura 6: Reflexões de um pentágono regular.

Denotando α_i por $i - 1$ e ρ_i por $i + 4$, a operação de D_5 é definida pela Tabela 9.

$\cdot$	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	0	6	7	8	9	5
2	2	3	4	0	1	7	8	9	5	6
3	3	4	0	1	2	8	9	5	6	7
4	4	0	1	2	3	9	5	6	7	8
5	5	9	8	7	6	0	4	3	2	1
6	6	5	9	8	7	1	0	4	3	2
7	7	6	5	9	8	2	1	0	4	3
8	8	7	6	5	9	3	2	1	0	4
9	9	8	7	6	5	4	3	2	1	0

Tabela 9: Operação do grupo diedral D_5.

Tomando a permutação anti-simétrica $\theta = (14)(23)(58697)$ obtemos deste modo um sistema com 100% de eficácia na detecção dos erros singulares e das transposições de algarismos adjacentes.

7. Comentários finais

> *"Não existe nenhuma área da matemática, por mais abstracta que seja, que não possa ser um dia aplicada a fenómenos do mundo real."*
>
> N. Lobachevsky

> *"Está agora completamente confirmado o paradoxo de que as maiores abstracções são as melhores armas para controlarmos os nossos raciocínios sobre os factos concretos."*
>
> A. N. Whitehead

(1) Como vimos, a Teoria dos Grupos fornece-nos uma solução óptima para o problema; se quisermos um sistema de identificação com um alfabeto de cardinal k, que detecte, além dos erros singulares, as transposições adjacentes, bastará usar:

- No caso de k ser ímpar, o grupo abeliano $\mathbb{Z}_k$ e a permutação completa $\phi : x \mapsto x$;
- No caso de k ser múltiplo de 4, o grupo abeliano $\mathbb{Z}_2 \oplus \mathbb{Z}_{\frac{k}{2}}$. Sucessivas aplicações do Lema 3 de [13] permitem-nos construir uma permutação completa de $\mathbb{Z}_2 \oplus \mathbb{Z}_{\frac{k}{2}}$.
- No caso $k = 2m$ (m ímpar), o grupo diedral D_m. Este grupo é gerado por dois elementos a e b, sujeitos às relações $a^m = e$, $b^2 = e$, $ba = a^{m-1}b$, pelo que

$$D_m = \{e, a, a^2, \ldots, a^{m-1}, b, ab, a^2b, \ldots, a^{m-1}b\}.$$

A permutação que a a^i $(i = 0, 1, \ldots, m-1)$ faz corresponder a^{m-1-i} e que mantém os outros elementos de D_m inalterados é uma permutação anti-simétrica ([4], Teorema 4.4).

(2) Estranhamente, apesar destes métodos serem muito mais eficientes que os modulares (no caso em que k é par), não estão praticamente em utilização. Que eu conheça, só o *Deutsche Bundesbank* utiliza uma permutação anti-simétrica do grupo D_5 para justapôr um algarismo de teste aos números de série das notas bancárias que emite ([15], pp. 65-67).

(3) Bastaria no algoritmo implementado no sistema dos BI substituir a operação $\oplus_{11}$ pela operação da Tabela 9 e o vector $(10, 9, \ldots, 1)$ pelo vector $(\theta^{n-1}, \ldots, \theta, id)$, sendo $\theta = (14)(23)(58697)$, para termos um sistema realmente eficiente.

(4) É possível generalizar um pouco mais estes métodos, definindo-os sobre *quase-grupos* (isto é, quadrados latinos). Com isso obtêm-se métodos um pouco mais eficientes relativamente aos outros tipos de erros duplos ([4], [16]).

(5) Não discutimos aqui o problema da correcção automática dos erros. Em muitos casos é claro que a correcção não é permitida (por exemplo, nos bancos). Mas em muitos outros casos é importante ter métodos que possam, além de detectar os erros, fazer a sua correcção automática. É possível construir um sistema módulo 37 que detecte e corrija todos os erros singulares e transposições. Evidentemente o número de algarismos de teste terá que aumentar. Para mais informações sobre estes sistemas, chamados *códigos correctores de erros*, consulte [2], [3] e [17].

"Fez-se um longo silêncio.
— É tudo? — perguntou Alice timidamente.
— É tudo — confirmou Humpty Dumpty. — Adeus."

Lewis Carroll, *Alice do Outro Lado do Espelho*

Referências bibliográficas

[1] T. Briggs, Weights for modulus 97 systems, *Computer Bulletin* 15 (1971) 79.

[2] D.A.H. Brown, Some error correcting codes for certain transposition and transcription errors in decimal integers, *The Computer Journal* 17 (1974) 9-12.

[3] D.A.H. Brown, Biquinary decimal error detection codes with one, two and three check digits, *The Computer Journal* 17 (1974) 201-204.

[4] A. Ecker e G. Poch, Check Character Systems, *Computing* 37 (1986) 277-301.

[5] J.A. Gallian, The Zip Code Bar Code, *The UMAP Journal* 7 (1986) 191-195.

[6] J.A. Gallian, The mathematics of identification numbers, *The College Math. Journal* 22 (1991) 194-202.

[7] J.A. Gallian e S. Winters, Modular Arithmetic in the marketplace, *The Amer. Math. Monthly* 95(1988) 548-551.

[8] J.A. Gallian e M.D. Mullin, Groups with anti-symmetric mappings, *Arch. Math.* 65 (1995) 273-280.

[9] H.P. Gumm, Encoding of numbers to detect typing errors, *Int. J. Appl. Engineering Ed.* 2 (1986) 61-65.

[10] S. Heiss, Antisymmetric mappings for finite solvable groups, *Arch. Math.* 69 (1997) 445-454.

[11] H.L. Larsen, Generalized double modulus 11 check digit error detection, *BIT* 23 (1983) 303-307.

[12] H.B. Mann, The construction of orthogonal Latin squares, *Ann. Math. Statistics* 13 (1942) 418-423.

[13] L.J. Paige, A note on finite abelian groups, *Bull. Amer. Math. Soc.* 53 (1947) 590-593.

[14] J. Pezzulo, *Communications of the ACM* 32 (1989) 1131-1132.

[15] R.-H. Schulz, *Codierungstheorie. Eine Einführung*, Braunschweig-Wiesbaden, 1991.

[16] R.-H. Schulz, A note on check character systems using Latin squares, *Discrete Math.* 97 (1991) 371-375.

[17] A.S. Sethi, V. Rajaraman e P.S. Kenjale, An error-correcting coding system for alphanumeric data, *Information Processing Letters* 7 (1978) 72-77.

[18] J. Sebastião e Silva, *Para uma teoria geral dos homomorfismos*, 1944 (Obras de José Sebastião e Silva, Vol. I, INIC, 1985, pp. 135-367).

[19] M. Sobral, *Álgebra*, Universidade Aberta, Lisboa, 1996.

[20] D. Sutherland, Error-detecting identification codes for Algebra students, *School Science and Math.* 90 (1990) 283-290.

[21] P. Tuchinsky, International Standard Book Numbers, *The UMAP Journal* 6 (1985) 41-54.

[22] J. Verhoeff, *Error detecting Decimal Codes*, Mathematisch Centrum, Amsterdão, 1969.

[23] N.R. Wagner e P.S. Putter, Error detecting decimal digits, *Communications of the ACM* 32 (1989) 106-110.

Departamento de Matemática, Universidade de Coimbra, 3001-454 Coimbra

E-mail: picado@mat.uc.pt