誤り訂正符号の話

— 情報理論 60 周年 —

知念 宏司 (近畿大学 理工学部)

2008.3.22

日本数学会 · 日本物理学会 合同市民講演会

1　はじめに

今回は日本数学会, 日本物理学会が 31 年ぶりに同時開催となり, それを記念いたしまして, 市民講演会も両学会合同で開催されることとなりました. 私は数学会側の講演者として, 誤り訂正符号のお話をさせて頂きます. 副題が少し風変わりですが, これにつきましてはお話の中ほどでご説明いたします.

まず誤り訂正符号とは何か, 簡単に言うなら, 「ディジタル方式で情報を伝えるとき, できるだけ正確に伝えるための仕組み」ということになると思います. 特に私は数学屋ですから, その数学的理論の部分を研究しています. 他に最近よく聞く言葉に「暗号」がありますが, これは「情報を第三者に知られないよう伝える仕組み」ということになります. どちらも現代の情報技術になくてはならないもので, よく似た言葉ですが, その目的が違うわけです.

さて, 「ディジタル方式で」と言いましたが, そもそも情報をディジタル方式で記録するというのは, どういうことなのでしょうか. まずはその説明から入りたいと思います.

2　情報のディジタル記録

一言で「情報」と言いましてもさまざまなものがあります. まずは「文章.」これは文字情報ということになりますが, 普段の生活でも, われわれは電子メールで文字をやりとりしています. 文字は情報を伝える重要な手段です. 次に「画像.」これも最近は携帯電話で写真が撮れたりして, それを誰かに送ることがあります. この場合も「ディジタル記録」されたものを送っているわけです. そして「音声.」音楽 CD に記録されている音, 携帯電話での通話などがその例です.

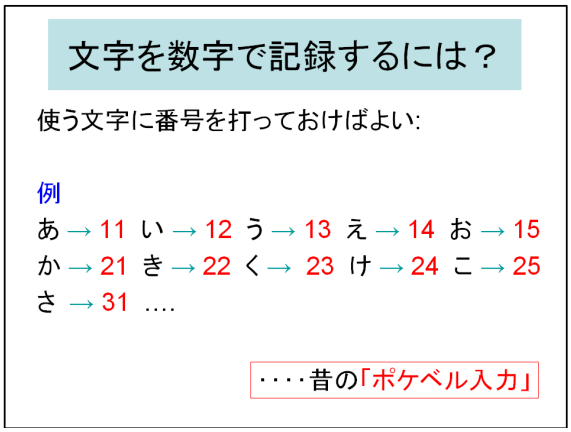
このように, 情報にはいろいろな形態がありますが, これらを

すべて数字に置き換えて記録する

というのがディジタル方式の情報記録です. 本当にそんなことができるのでしょうか.

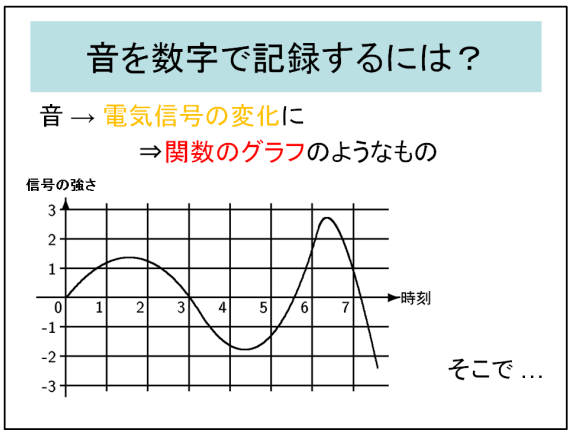
まず文字を数字で記録するにはどうすればよいか考えてみましょう. これは比較的簡単で, 使おうとする文字全部にあらかじめ番号を打っておけばよいでしょう. たとえば, ひらがな全部に次のように番号を打てば, ひらがなをそのまま記録する代わりに対応する数字を記録すればいいわけで, それを送ることで文章のやりとりが可能です*.

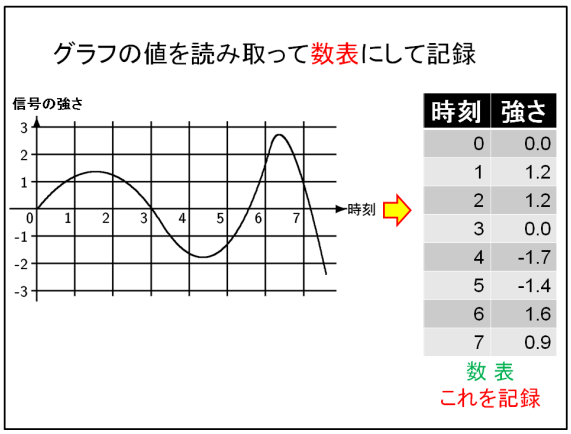
* このあとの図はすべて, 当日お見せしたスライドの一部です.



実は昔, 携帯電話が普及する前は「ポケットベル」というものがあり, みんなこの対応表に従って数字で文章を打ち込み, やりとりしていたのです.

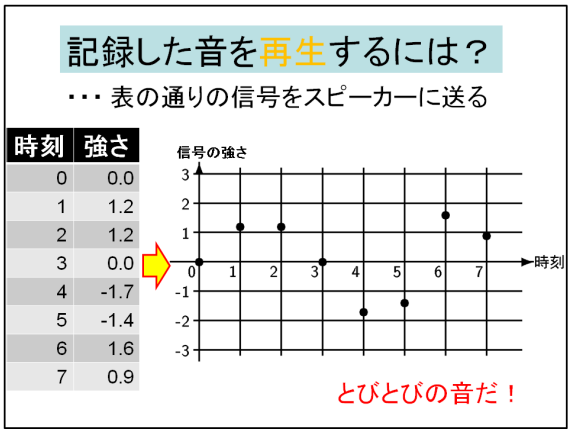
では次に, 音を数字で記録することを考えましょう. これはかなり難しそうなのですが, 音をマイクで拾って電気信号の変化に置き換える, ということを考えてみましょう. すると, ちょうど関数のグラフのようなものが得られます (当日はここでフルートを少し鳴らして, その波形を実際に観察しました). つまり, 音が鳴っているというのは, このような波が時間の経過とともに, ずーっと流れていくことなのです (もちろんこれは模式図で, 実際のフルートの波形ではありません).



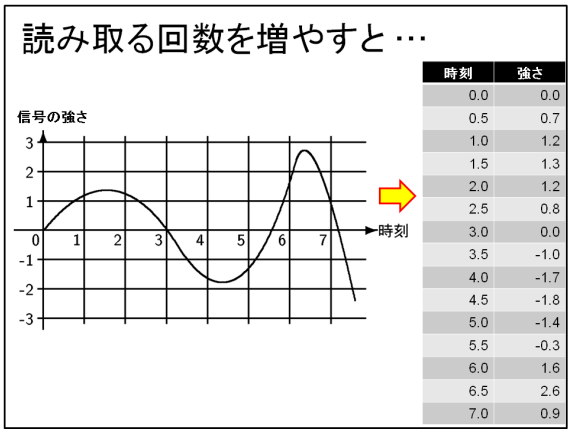


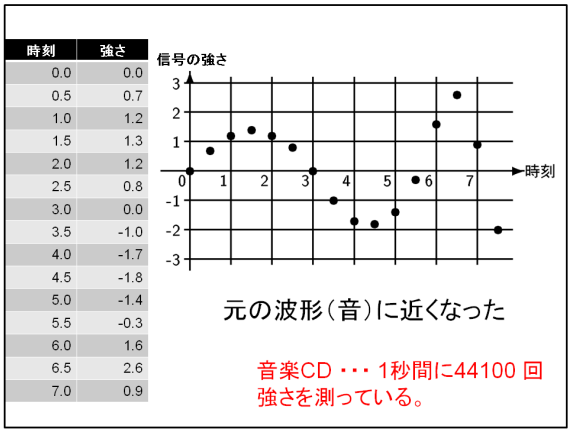
ここにヒントがあるわけです. 小学校の理科の時間などによくやったと思いますが, 波形のグラフの値を読み取ってそれを表にし, その数表を記録することにすればいいのです. 右上図では時間 1 ごとに強さを測って表にしています. このように, 波形をそのまま図形として記録するのではなく, 数表の形にして記録する, これがディジタル記録の本質といえるのです.

記録した音を再生するには, 数表の通りの信号を発生させる機械を作り, その信号をスピーカーに送ってやるのです. こんな感じです.



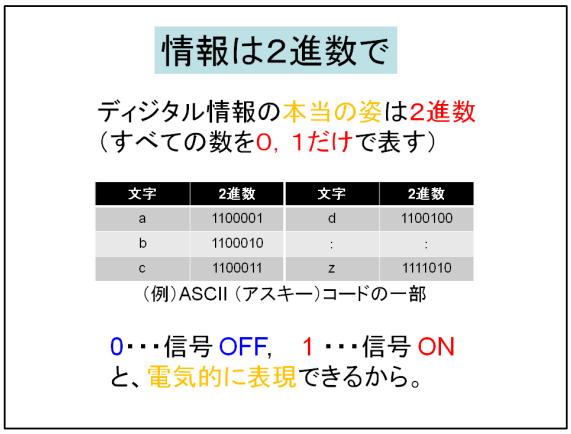
おやっ、と思いますね．とびとびの音です．これでは元の音が再現できそうな感じはしません．しかし, 値を読み取る頻度をもっと増やせばどうでしょうか．





今度はかなり細かくなって, 元の波形に似てきました．この調子で頻度を上げていくと, 元の曲線にどんどん近づいていきそうな気がします．実際の音楽 CD では, 1 秒間に 44100 回, 音を読み取っています (CD プレーヤーの説明書などに「標本化周波数 44.1 kHz」とあるのがこのことを表しています).

さて, 何でも数字で記録するのがディジタル記録, ということはおわかり頂けたと思いますが, 実際の記録には通常われわれが馴染んでいる 10 進法 (0 から 9 までの組み合わせですべての数を表す) ではなく, 2 進法 (0 と 1 だけの組み合わせですべての数を表す) を用います．

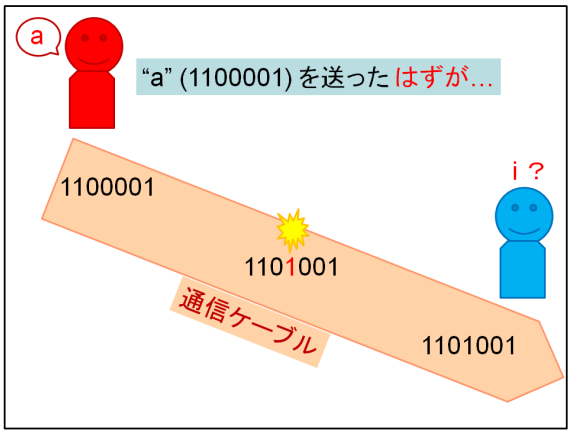


例えば英文で使われる文字を数字に置き換える規則である「ASCII (アスキー) コード」では, 小文字の a を 1100001, b を 1100010, ··· という具合に 7 桁の 2 進数で表しています. この 1100001 は 10 進法 では 97 という数です. われわれ人間はこの 7 桁の 2 進数を見て,「97 !」と, とても即座には読めません. しかし機械にとっては 2 進数は非常に都合がいいのです. それは, 「0 → 信号 OFF」,「1 → 信号 ON」という風に, 数を簡単に電気的に表せるからです. したがって, ディジタル情報の「本当の姿」は 2 進数の列, ということができます.

3　誤り訂正はなぜ必要か

正確なディジタル機器に誤りとは少々意外な感じがするかも知れませんが, 実はディジタル通信に誤りはつきものなのです. よくラジオを聴いていて部屋の蛍光灯のスイッチを入れると「バチバチ」と雑音が入ることを経験します. これは蛍光灯からの電波でラジオの電波が妨害されて雑音になるわけです. ところで, 携帯電話は電波で情報をやりとりしています. 周囲からのこうした余計な電波で妨害されると, もとの情報は大きく姿を変えてしまうでしょう. このようにして情報伝達途中には必ず何らかの誤りが起きると考えるべきなのです. あと, 音楽 CD の再生を例にとると, ディスクにゴミや傷がついた状態でプレーヤーにセットすると, その部分の情報が読み出せないことも起きるかも知れません.

実際のディジタル通信では, すべての信号は 0 または 1 のいずれかと解釈されますから, 誤り発生は数学的には, **0 と 1 が入れ替わること**と定式化できます. そして重要なことは, ディジタル通信では 0, 1 のわずかな違いが情報としては大きな違いになって現れる, ということです. 例えばある人が小文字の a (ASCII コードで 1100001) を別の人に送ったとします. 途中で「バチッ」と雑音が入って, 例えば 4 桁目の 0 が 1 に置きかわり, 1101001 になったとしましょう. これは ASCII コードの i です. このように, たった 1 桁に誤りが発生しただけで小文字の a が小文字の i という, 全く違う情報になって伝わってしまうのです (このスライド, 本当は動くのです. 印刷では止まったままで残念ですが).

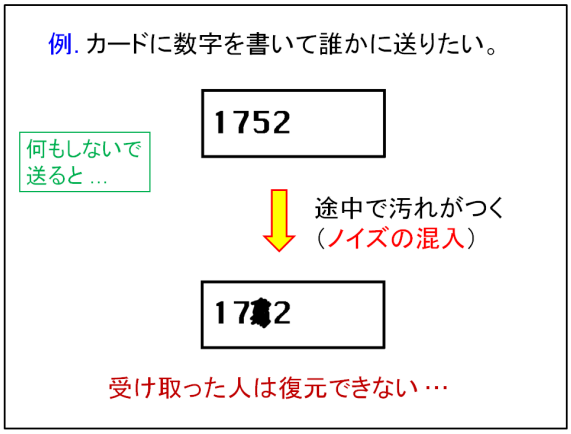


これでは困りますから, こうして発生した誤りはぜひとも本来の情報に直さなければなりません. 誤り訂正の機構を導入する必要性はここにあるのです.

4　誤り訂正の考え方

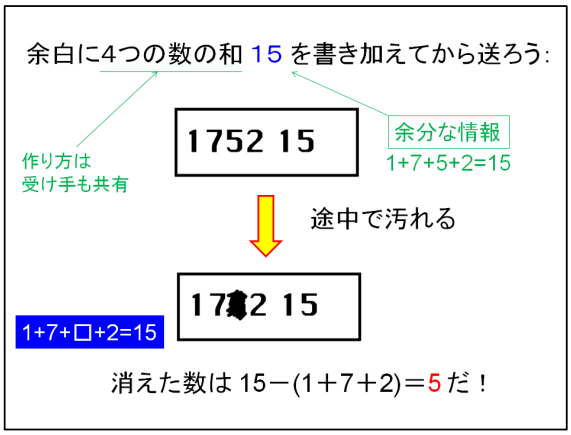
元の情報に何も手を加えずに送って, 発生した誤りを直す方法はさすがにありません. あらかじめ**余分な情報を少し付け加えておき, それを頼りに誤りを直す**という方法を取ります. ヒントは「虫食い算」です.

2 進数からはちょっと離れて次のような例を考えてみます. いま, カードに数字を書いて誰かに送りたいとします. 途中で汚れがついて 1 文字読めなくなるとします (誤りの発生です). 何もしないで送ると, 受け取った人は読めなくなった数を復元することはできません.

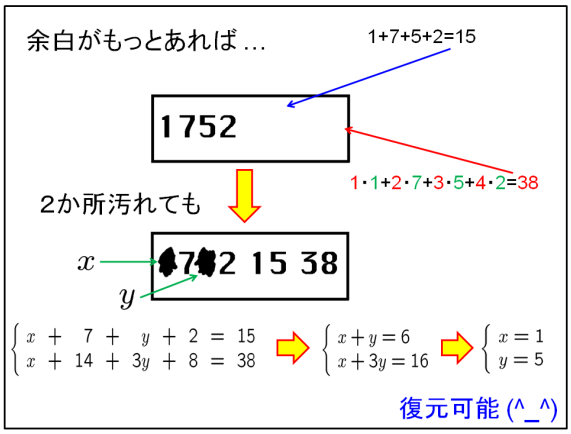


そこでカードに余白があるのを利用して, ここに少し余分な情報を付け加えておきましょう. 付け加えるのは 15 という数, これは何かというと, 送りたい 1752 の各桁を足したものです ($1+7+5+2=15$). この状態にして送ります. 同じように途中で汚れがついて 1 文字読めなくなる状況を考えますが, 今度は余分に付け加えた 15 という数を頼りに, もとの情報を復元することができます. もちろん, 「付け加えたのは各桁の和である」

という規則だけは情報の受け手もあらかじめ知っているとするのです. 簡単にわかるように, 消えた数は $1+7+\bullet+2=15$ の $\bullet$ の部分で, $15-(1+7+2)=5$ と求めることができます. 最も簡単な虫食い算ですね.



もっと余白があればどうでしょうか. もう 1 つ余分な情報を書き加えておきましょう. 今度は 38 を付け加えました. これは $1\cdot1+2\cdot7+3\cdot5+4\cdot2=38$ と計算したもので, 一番左の桁はそのまま, 次の桁は 2 倍, その次は 3 倍, 最後は 4 倍して足したものです. 「重み付きの和」ですね. すると途中で 2 か所読めなくなってももとの情報を完全に復元できます.



つまり, 付け加え方の規則から, 連立 1 次方程式

$$\begin{cases} x+7+y+2=15 \\ x+14+3y+8=38 \end{cases}$$

を解けばよいことになります. 答は $x=1$, $y=5$ です. 連立 1 次方程式が登場しましたが, おもしろいことに本物の符号でも, 連立 1 次方程式のより高度な理論が本質的なところで使われているのです[†].

[†]符号理論の基礎は係数体が有限体の線型代数学と思って差し支えありません.

上のことから, 付け加える余分な情報が多いほどたくさんの誤りを訂正できることがわかります. これはよいことなのですが, 一方, 余分な情報があまりに多いと情報伝達の効率が落ちることになります. したがって, 状況に応じてちょうどよい付け加え方を考える必要があり, ここに符号理論研究の 1 つの意義があるわけです.

5 「情報理論 60 周年」— クロード · シャノンのこと

さて, ここで少し話題を変えて, 歴史的なお話をしたいと思います. 現在われわれが恩恵を受けている情報技術ですが, その基礎を作った人は誰かといいますと, クロード · シャノン (Claude Elwood Shannon, 1916 – 2001) という人です. 1948 年に「通信の数学的理論」という論文を発表し[‡], 情報というものを数学的に取り扱うことができるようにしました. 今年 2008 年は, この画期的論文が発表されてちょうど 60 年です. それで「情報理論 60 周年」という風変わりな副題をつけたわけです. また, 情報量の単位「ビット (bit)」を, この人を記念して「シャノン (Sh)」という単位に言い換えようということも提唱されています (国際規格 ISO/IEC 2382-16:1996). まだそれほど普及はしていないようですが, 情報理論の最近の教科書の中には, 取り入れているものがあります.

シャノン氏の年表を少し詳しく見てみましょう.

クロード・シャノン 略年表

1916	アメリカ ミシガン州生まれ
1936	ミシガン大学卒業
1941	ベル研究所 研究員
1948	論文「通信の数学的理論」発表
1957	マサチューセッツ工科大学(MIT)教授
1985	第1回 京都賞(基礎科学部門)受賞
1996	情報量の単位「ビット」を「シャノン」に
1998	MITで「情報理論50周年」記念シンポジウム(IEEE)
2001	没
2008	(今日)日本数学会・日本物理学会 合同市民講演会

※ 長年、MIT で活躍した研究者だった

1916 年アメリカ生まれです. 大学卒業後, ベル研究所の研究員になっています. ベル研究所はアメリカの電話会社が設立した研究所ですが, この当時のベル研といえば, 例えばトランジスターを発明したグループがいたりして[§], 錚々たるメンバーが名を連ねていたすごい所だったのです. 1948 年に上述の論文を発表, その後マサチューセッツ工科大学 (MIT) の先生になっています. 日本に関係のある経歴としては, 1985 年, 第 1 回京都賞受賞, というのがあります. ご存知の通り, 京都賞は稲盛財団が運営する賞で, 基礎科学部門, 先端技術部門, 思想 · 芸術部門の 3 部門で毎年賞が授与されています. シャノン氏はその記念

[‡]“A mathematical theory of communication”, 初出は Bell System Technical Journal で 1948 年 7 月と 10 月の 2 度に分けて出版, 改訂版が 1949 年に Illinois 大学から出版されました.

[§]1948 年, ウォルター · ブラッテン (Walter Houser Brattain, 1902 – 1987), ジョン · バーディーン (John Bardeen, 1908 – 1991), ウィリアム · ショックレー (William Bradford Shockley, 1910 – 1989). この功績で 1956 年ノーベル物理学賞受賞.

すべき第 1 回,「基礎科学部門［授賞対象分野: 数理科学 (純粋数学を含む)］」という形での受賞です. 情報理論だから先端技術部門かな, と思ったら実は基礎科学部門, というのもなかなか興味深いですね. それから, 今年が 60 周年なら 10 年前は 50 周年という, もっときりのいい年だったわけですが, この年には MIT で大きな記念シンポジウムが開かれました. これについては後で触れます. そして最後のは ... これは普通シャノンさんの年表には書かないものですが, 今日お越し頂いた皆さんのためにも, ちょっと書かせて頂きました.

さて, シャノン氏は上述の「画期的」論文でどんなことをしたかというと, 次のような定理を証明したのです: 「通信路容量を $C(p)$ とするとき、$R < C(p)$ なら, 任意の $\varepsilon > 0$ に対し, 符号化率 R で復号誤り率 $P_{\mathrm{err}} < \varepsilon$ の 2 元符号が存在する. 」これは難しいですね. 「任意の $\varepsilon > 0$ に対し · · · が存在する」などと, およそ日常使わない言い回しが出てきます. まあ数学屋というものは, 日々こんな文章を相手にしているのですが, これはさすがに難しいですので, 易しく言い換えてみましょう. こんな感じでしょうか:「扱える情報量に少し余裕があれば、いくらでも正確に情報を伝える仕組みは存在する¶.」なかなかありがたい定理ですが, 実は大きな問題があります. それは, この定理が「存在定理」であるということです. これも数学者がよく使う言葉で, 定理によってある数学的対象 (今の場合, 性能のよい符号) の存在は保障されるが具体例は教えてくれない, というタイプの定理をこう呼びます. 具体例がわからないのに存在がわかる, というのも変な感じがするかも知れませんが, 数学の世界には実はそういう定理がたくさんあるのです. ですから,「性能のよい符号は探せば必ずありますから, 皆さんがんばって探して下さい !」と, この定理は言っているわけです. それ以来, 世界中の研究者がよい符号の研究を続け, 現在こうしてコンピュータも携帯電話も音楽プレーヤーも, 満足に使える時代がやってきた, というわけです. それでもシャノンの定理の言うような「究極の符号」には, なかなか到達できないのが現状のようです.

6 MIT とその周辺

さて, ここで少し寄り道をして, 皆さんをちょっとアメリカへご案内しましょう‖. どこへ行くかといいますと, シャノンさんが活躍していた MIT (マサチューセッツ工科大学) です. 場所はほぼボストン, 正確にはその北隣のケンブリッジという街です. チャールズ川というのが町の境界線になっていまして, その川沿いにキャンパスが広がっています. 先ほども申しましたが, 今からちょうど 10 年前, ここでシャノン氏の業績を記念した大きなシンポジウムが開かれました.

¶日常の言葉によるもっと的確な言い換えをご存知の会員の方はぜひご教示下さい.

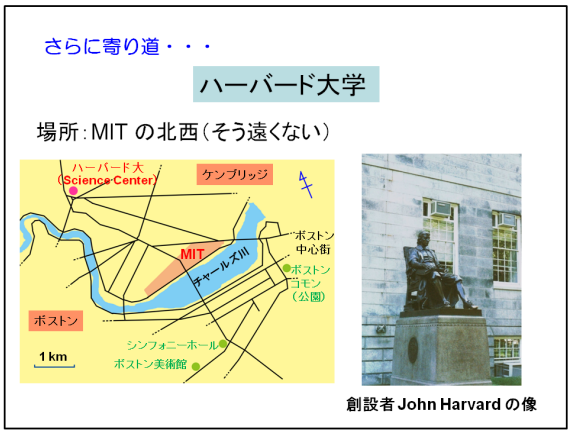
‖以下, 写真はすべて筆者.

主催は IEEE (アイ・トリプル・イー) という団体です. 略さずに言うと “The Institute of Electrical and Electronic Engineers” (米国電気電子学会) となります. 工学系の団体ですが, 数学もこうした分野に大きく関連しますので, 数学者も数多くここから論文を出版しています. 会場で撮った写真に垂れ幕が写っていますが, “FIFTY YEARS OF INFORMATION THEORY” (情報理論の 50 年) と書かれているのにお気づきでしょうか. このシンポジウムでは計約 470 件の研究発表が行われた他, 前述のシャノン氏の論文「通信の数学的理論」がリプリントされて参加者に配られました.

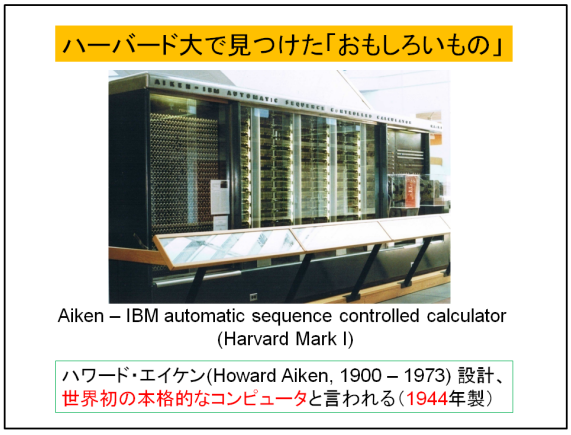
さて, この MIT を訪れたとき, ちょっとおもしろいものを見つけました. 写真を見ますと何か作品が展示してあります. 実はこれ, 学生用アートギャラリーです. MIT といいますと, 理科系専門の大学というイメージが強いのですが, 今は芸術系の学部もあるそうで**, そのためもあるのでしょう. しかし理科系中心の日本の大学にはなかなかこうしたスペースはないでしょうから, やはり興味深いですね.

ついでにもう少し寄り道しましょう. 今度はハーバード大学へ行ってみましょう. 場所は同じケンブリッジで, MIT の北西にあたります. ハーバード大の敷地は非常に広いのですが, 数学教室のある Science Center は大体地図の赤丸の位置です. 学内にはハーバードさんの銅像があり, 訪れた人は大抵ここで記念写真を撮るという, ちょっとした観光名所にもなっています.

**ボストン在住のシンポジウム参加者から聞きました.



ここでもまたおもしろいものを見つけました. Science Center の中に展示してあったのですが, これは大きな機械ですね. 展示のためガラスケースに入っています. 上の文字を読んでみると “Aiken - IBM automatic sequence controlled calculator” とあります. 通称 “Harvard Mark I” と呼ばれるこの機械は, ハワード·エイケン (Howard Aiken, 1900 – 1973) という人が作った, 世界初の本格的なコンピュータと言われる機械です. 「本格的な」ということの中身は, 計算を始めてから結果が出るまで, 途中で人間が操作をする必要がなく, すべて自動でやってくれる, ということで, そうした計算機はこれが初めて, と言われているのです.



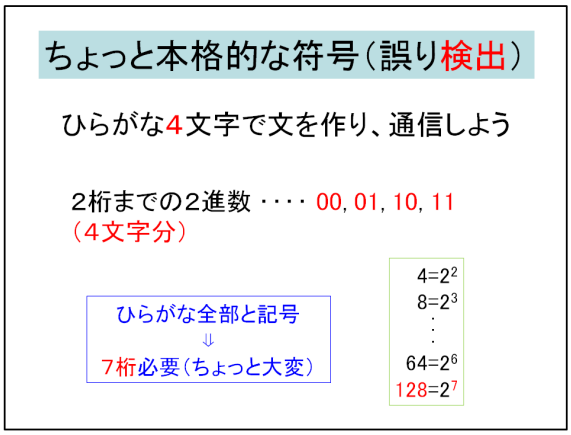
こうして見てみますと, MIT にはシャノンのような人がいるし, ハーバードにはこんな機械を作る人がいるし, ということで, やはりこのあたりは当時, 計算機科学発展の中心の一つであったことが覗えます.

7　誤り検出符号の実例

では, 本題に戻りまして, ちょっと本格的な符号を作ってみましょう. ただし, お話を簡単にするため, 誤りの検出（誤りが起きたかどうかだけを判定し, 訂正まではしない）のみを行う符号にします. 訂正ができなければ全く実用にならないかというと, そうでもあ

りません. 誤りが発生したことがわかれば同じ情報を再度送ってもらう, という使い方は可能です. このあと数学は ... あんまり要らないですね.

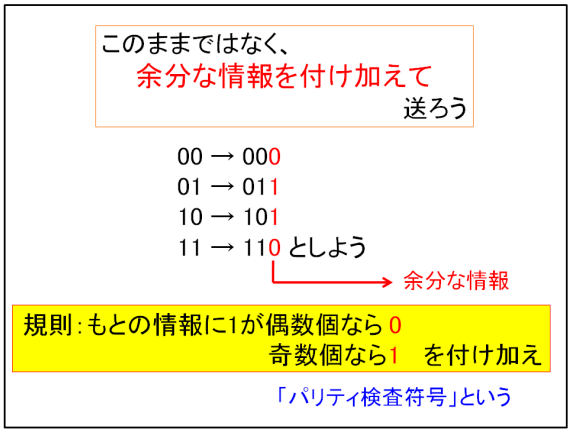
今度考えるのは, ひらがな 4 文字で文章を作って, それを 2 進数にして送ろう, というものです. 4 文字とは少ないですが, 4 文字ですと 2 桁の 2 進数で済みます. もしひらがな全部 (それに句読点などの記号類) を使おうとすると, 7 桁ぐらいの 2 進数が必要ですので, こういう場でご説明するのはとても大変です.



4 つの文字を 00, 01, 10, 11 という 4 つの数で置き換えて通信をしますが, 使う文字をあらかじめ決めておかなければなりません.「け, た, ぶ, や」の 4 つにしましょう (変な文字を選んだものですが). これらをそれぞれ 00, 01, 10, 11 に対応させます. 例えば

$$01\ 00\ 11\ 10\ 11\ 00\ 01$$

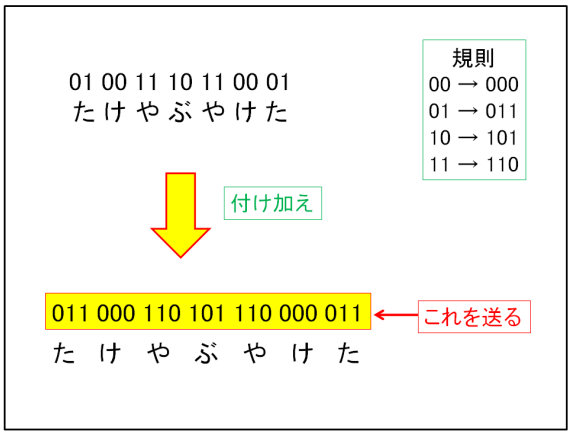
は「たけやぶやけた (竹薮焼けた).」そう, よくご存知の古典的回文です. まあほとんどこの文しか表せなくてやはり不便です (皆さんは文字数＝桁数を適宜増やしてお楽しみ下さい).



これを送るのですが, このまま送るのではなく, 余分な情報を少し付け加えてから送りましょう. 図のように, 2 桁ずつ区切ったもとの情報に 1 が偶数個なら 0 を, 奇数個なら 1 を付け加える, という規則です (これには「パリティ検査符号」という名前がついています). したがって「たけやぶやけた」は

011 000 110 101 110 000 011

となります. これを送るわけです.

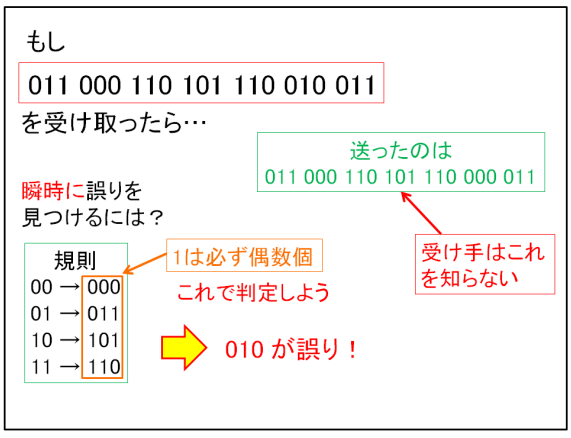


もし受け手がこれをそのまま受け取れば, 各 3 桁組の左 2 桁を取り出して 01 00 11 10 11 00 01 と戻します. すると, 最初に決めた 2 進数とひらがなとの対応規則から,「たけやぶやけた」に戻せるわけです.

では, もし

011 000 110 101 110 010 011

を受け取ったらどうでしょうか. 誤りが 1 つ含まれているのですが, おわかりでしょうか. 正しい情報と見比べればすぐにわかるのですが (最後から 2 番目のグループが違っています), 受け手は正しい情報を知らないのですから, 見比べるのは反則です. また, 受け手が正しい 3 桁組の一覧表を持っていて, 順にそれと見比べながら調べていくのも時間がかかり過ぎます. 受け取った情報を見て瞬時に間違いが発生したかどうかを判断できることが必要です.

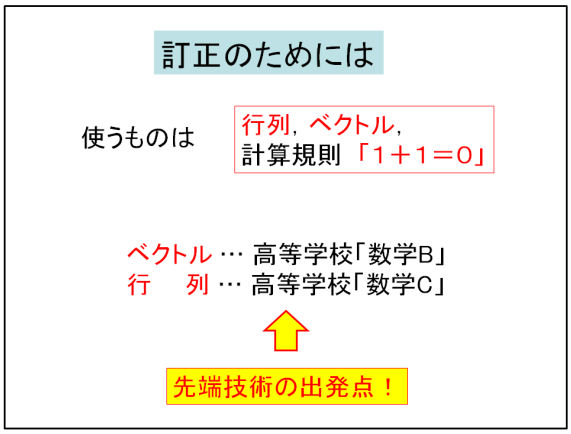


そこで, また 1 の個数を調べる, ということを考えてみましょう. 正しい 4 つの 3 桁組には, 実は共通するある特徴があります. それは, どの組にも 1 が偶数個含まれている, ということです. 000 には 1 は含まれていませんから, 0 個, そして 0 は偶数ですね. 他の

011, 101, 110 には 2 個の 1 が含まれています. これを利用しましょう. つまり, 受け手は送られてきた数の列を 3 桁ずつ読み取って 1 の個数を数えます. 1 が偶数個ならそれは正しい, 1 が奇数個ならそれは誤り, と判断していくのです[††].

訂正までできて, ある程度効率もよい符号を作るには, やはり少々難しい数学を使う必要があります[‡‡]. 主に使われるのは**ベクトル**や**行列**の高度な理論, それに「$1+1=0$」というちょっと不思議な計算規則です. 小学校以来, $1+1=2$ と習って久しいわけですから, $1+1=0$ には違和感があるでしょうか. あるいは新鮮な感じを受ける方もおありかも知れません.

ところで, このベクトルや行列は高等学校でもその初歩を学習します (ベクトル → 数学 B, 行列 → 数学 C). 皆様の中には高校数学は一体何の役に立つのかな, とお思いの方も多いでしょうが, こうした先端技術を支える学問の出発点としての役割を果たしているのです.

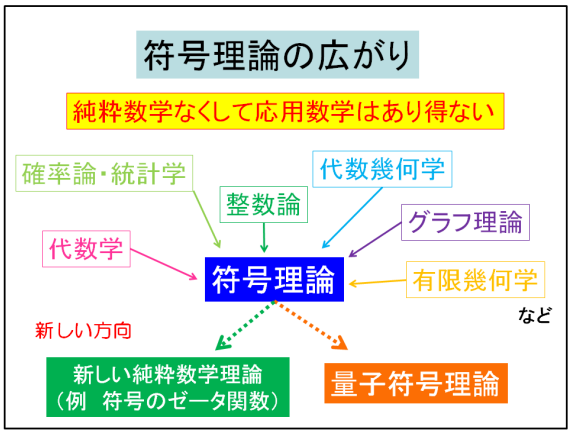


8　最後に

私の専門は数学 (整数論) で, そうした立場から符号理論というものに接してきました. すると, 符号理論はそれだけで発展してきたわけでは決してなく, 純粋数学の分野で得られてきた, さまざまな高度な成果をつぎ込むことによって実際に役立つ理論が形成されてきたという感じを強く持ちます. まさに, **純粋数学なくして応用数学はあり得ない**ということを実感できる分野, それが数学理論としての符号理論ではないかと思います.

[††]010 を 000 と直すわけにはいきません. それは, 1 か所間違って 010 になるのは他にも 011 → 010 (一番右の桁が 1 → 0), 110 → 010 (一番左の桁が 1 → 0) という場合があるからです.

[‡‡]いわゆる「繰り返し符号 (repetition code)」は訂正もできますが難しい数学はいりません. ただ効率はよくありません.



そして最近では, 符号理論の中から新しい純粋数学理論が生まれたりもしています. 私が現在興味を持っている「符号のゼータ関数」などはその例だと思います. さらに「量子コンピュータ」の研究からは「量子符号」という分野も生まれ, 物理学とのつながりも出てきました. 符号理論は今後もさらなる発展と広がりを見せてくれるであろうと思っています.