

2013 年度秋期

数の世界

(担当：角皆)

情報技術と数理の利用

コンピュータの発展・情報化社会の進展に伴い、

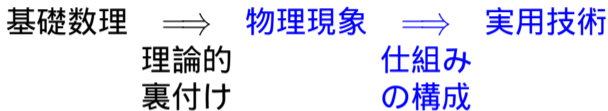
数理の解明とその利用が

ますます社会と密接になってきた

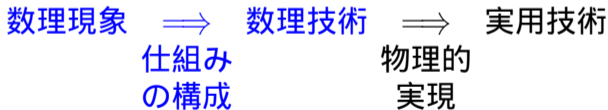
数理技術

情報技術と数理の利用

物理技術（17世紀以来）：



情報技術（20世紀以来）：



数理の解明が直接に技術発展に繋がる

計算機で扱えるもの

計算機では本質的に

有限・離散

のものしか扱えない

- 無限・連続のものに近い
- 有限・離散であることの積極的活用

有限の算術（剰余系）

m : 1 以上の整数を一つ取って固定

m で割った余りのみに注目して計算する

a と b とが m を法として合同
(congruent modulo m)

$$a \equiv b \pmod{m}$$

\Leftrightarrow a と b とを m で割った余りが等しい

$\Leftrightarrow m \mid (a - b)$ ($a - b$ が m で割切れる)

有限の算術（剰余系）

剰余のみに着目して

（**well-defined** に）足し算・掛け算が出来る

足し算表は
ほぼ当たり前
右は $m = 5$

$$3 + 4 = 7 \equiv 2 \pmod{5}$$

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

実習：剰余系の掛け算

$m = 3, 4, 5, 6, 7$ について、

演習プリントの掛け算表を埋めてみよう

掛け算表は

当たり前？

右は $m = 5$

$$2 \times 3 = 6 \equiv 1$$

$$(\text{mod } 5)$$

\times	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

(演習プリントの表には 0 の行・列はない)

実習：剰余系の掛け算

$m = 3, 4, 5, 6, 7$ について、

演習プリントの掛け算表を埋めてみよう

m の値による様子の違いは？

剰余系の演算

mod m で考えたとき、 a, b に対して、

$a + x \equiv b \pmod{m}$ となる x は
必ず丁度 1 つ見付かる

→ mod m で引き算が出来る
mod m の世界で “ $x = b - a$ ”

しかし、

$ax \equiv b \pmod{m}$ となる x は
($a \not\equiv 0 \pmod{m}$) でも) 見付かるとは限らない

→ mod m で割り算が出来るとは限らない

素数を法とする剰余系の演算

一般の m では

$\text{mod } m$ で割り算が出来るとは限らないが、

法 m が素数であるときは、

$a \not\equiv 0 \pmod{m}$ であれば、

$ax \equiv b \pmod{m}$ となる x は
必ず丁度 1 つ見付かる

→ $\text{mod } m$ で (0 以外での) 割り算も出来る
 $\text{mod } m$ の世界で “ $x = b/a$ ”

有限体

体 (field) : 四則演算 (加減乗除) が出来る集合

例 : 有理数体 \mathbb{Q} ・ 実数体 \mathbb{R} ・ 複素数体 \mathbb{C}

素数 p に対して、

p で割った余りの集合 $\xleftrightarrow{1:1} \{0, 1, \dots, p-1\}$

この中で (0 で割る以外の) 四則演算が出来る

… **有限体 (finite field)** ・ p 元体 $\mathbb{F}_p, \mathbb{Z}/p\mathbb{Z}$

有限体

体 (field) : 四則演算 (加減乗除) が出来る集合

→ 四則演算しか使わない計算なら、
有限体 \mathbb{F}_p 上でも
実数や複素数と同様に行なえる

例 : 連立 1 次方程式を解く

大小関係・極限・収束などはない

物理技術の利用においては、

微分積分（解析学）がその基礎となったが、

有限・離散な世界である計算機上の
情報・数理技術の利用においては、

抽象代数学がその基礎となっている

- 基礎理学はすぐには役に立たない
- けれども不思議といつか役に立つ
- それがいつかは判らない

→ “良いもの” を追い求めるのが大切

数理技術としての応用例 1

有限体の算術を利用した、
ちょっと不思議な応用例を紹介しよう

秘密分散

(秘密情報の安全な管理の一方法)

秘密分散

盗賊の親分が隠し財宝の在処を子分達に伝える
(会社の社長が超重要機密を重役達に伝える)

伝える相手は3人

それぞれに異なる手掛かりを教える

但し、

- どの1人も自分だけでは何も判らない
- どの2人でも教え合えば判る

ようにするにはどうしたら良いか？

秘密分散

アナログ技術で実現するのは中々難しそうだ

→ デジタル技術・数理技術の利用

→ 秘密情報を数値化・符号化して処理
(有限・離散の世界の積極的活用)

秘密分散

駄目な例 1 : 秘密情報を 3 桁の数字列として
各人に 1 桁ずつ教える

- 2 人がつるんでも判らない
- 各人は何も知らないよりも情報がある

駄目な例 2 : 秘密情報を 3 桁の数字列として
各人に 2 桁ずつ教える

- 2 人がつるめば判るが、
- 各人は何も知らないよりも情報がある

秘密分散

- 素数 p を固定（これは公開）
- 秘密情報は有限体 \mathbb{F}_p の元 b とする
- ランダムに \mathbb{F}_p の元 a を選ぶ（これも秘密）
- \mathbb{F}_p 上の“直線” $y = ax + b$ を考える
- 各子分に対して
 - ★ 異なる \mathbb{F}_p の元 $x_i (\neq 0)$ を選び
 $y_i = ax_i + b$ を計算する
 - ★ “直線上の点” (x_i, y_i) を教える

秘密分散

2人つるむと判る理由：

2点を通る“直線” $y = ax + b$ は唯一に定まる

2点 $(x_i, y_i), (x_j, y_j)$ を通る直線は

$$a = \frac{y_i - y_j}{x_i - x_j}, \quad b = y_i - \frac{y_i - y_j}{x_i - x_j} x_i$$

→ 秘密情報 b が判明した

秘密分散

1 人では判らない理由 :

2 点を通る “直線” $y = ax + b$ は必ず存在する

2 点 $(x_i, y_i), (0, b)$ を通る直線の傾きは

$$a = \frac{y_i - b}{x_i}$$

どの値 b も同様に可能性がある

→ 何も知らないのと同じ

実習：秘密分散

みなさんに配った“秘密情報の一部”（鍵）

$$(x, y)$$

1 次式 $y \equiv ax + b \pmod{11}$ で

秘密情報 b を分散して伝えたもの

近くの人の鍵を見せてもらって

秘密情報 b を復元しよう

（鍵が同じ値だったら他の近くの人に）

実習：秘密分散

- 法 $p = 11$ に関する掛け算表を埋めよう
(なくても出来れば、埋めなくても良い)
- 引き算 $b - a$ はどうするの？
 - $a + x = b$ となる x が $x = b - a$
 - $b - a < 0$ なら p を足せば良い
- 割り算 $\frac{b}{a}$ はどうするの？
 - $ax = b$ となる x が $x = \frac{b}{a}$
 - 拡張互除法で計算できるが、
今は表から探そう

「割り算」って何さ？ — 定義に立ち戻ること

有限体での $\frac{b}{a}$ とは何か？

「割り算」とは何だったか？

$\frac{b}{a}$ とは、 $ax = b$ となる（ただ一つの） x のこと

定義に立ち戻る（定義から出発する）ことで、
何であるかがはっきりする

秘密分散

今は 1 次式（“直線” $y = ax + b$ ）を使ったので、
2 人が見せ合えば判ったが、

3 人の協力で判るようにするには、
2 次式（“放物線” $y = ax^2 + bx + c$ ）を使えば良い
一般に、

k 人の協力で判るようにするには、
(k - 1) 次式を使って仕組みを設計すればよい
(“n 人中 k 人”で出来る)

秘密分散

- 秘密を分散する人数は余り関係ない
→ 人数より大きな素数 p を用いれば良い
- あてずっぽうでも確率 $\frac{1}{p}$ で当たってしまう
→ 実際には大きな素数 p を使う
(100桁とか200桁とか)
- 割り算の計算を効率良く行なう必要がある
→ **Euclidの拡張互除法**を用いる

暗号 (cryptography)

- 秘密通信
- 電子認証・電子署名
- 鍵共有

暗号の利用

- 古典的：戦争・謀略など

- 現代：情報通信一般

→ 個人の独立を守るための重要な数理技術

既に身近な暗号の利用

メディアセンターのコンピュータを使う際の
パスワードによる本人認証にも
暗号（暗号化）が使われている

入力したパスワードを
保管してあるデータと照合しているのだが、

実は、
パスワードそのものを保管しているのではない

定まった方式（暗号化関数）で
パスワードを変換して保管している

パスワードによる本人認証

- 暗号化関数でパスワードを変換して保管
- 入力したパスワードを暗号化関数で変換して、
保管してある文字列と照合

暗号化関数に要請される性質は？

- 間違った文字列では変換結果が一致しない
→ 異なる入力には異なる値を返す（単射）
- 保管してある文字列が露見しても
元のパスワードが判明しない
→ 一方向性関数 (one-way function)

パスワードによる本人認証

良い暗号化関数で変換して保管していても

通信経路の途中で盗聴されてしまっても

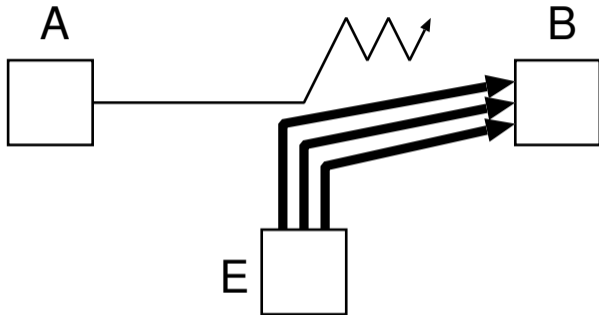
パスワードが露見してしまう

→ **暗号**による秘密通信

安全な情報伝達を阻害するもの

- 妨害（DoS 攻撃など）
- 盗聴
- 改竄
- なり済まし など

DoS (Denial of Service) 攻撃

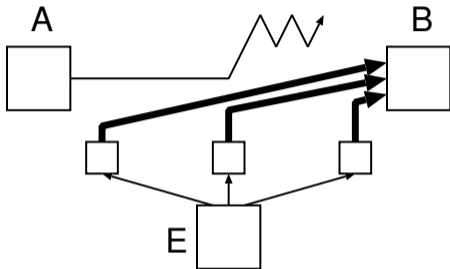


B を機能停止に追い込むには

E に相当のマシンパワーが必要

そこで実際には …

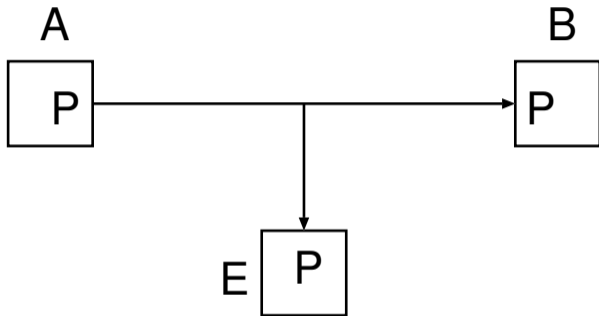
DoS (Denial of Service) 攻撃



実際には、

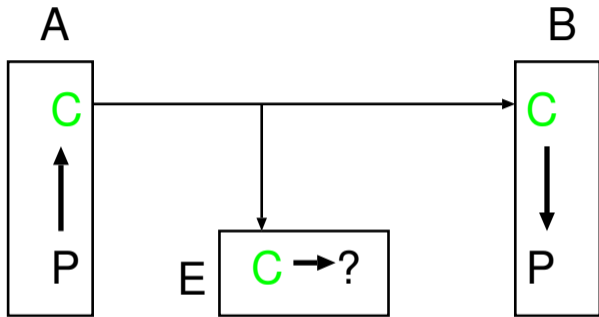
コンピュータウイルス・乗っ取りなどで
制御下に置いた多数の機械から一斉に攻撃
(Distributed DoS, DDoS)

盗聴



現在の計算機ネットワークの仕組みでは、
事実上、通信経路は誰にでも見られる

暗号通信で盗聴対策

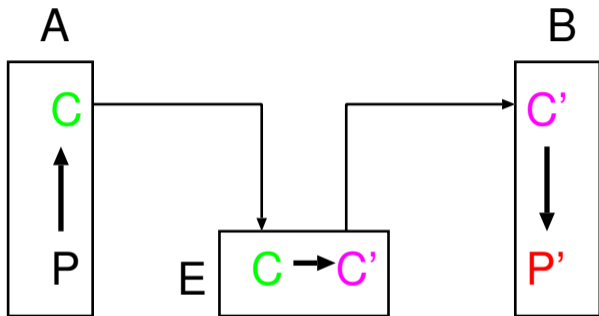


P : 平文 (plain text), C : 暗号文 (ciphertext)

P → C : 暗号化 (encryption)

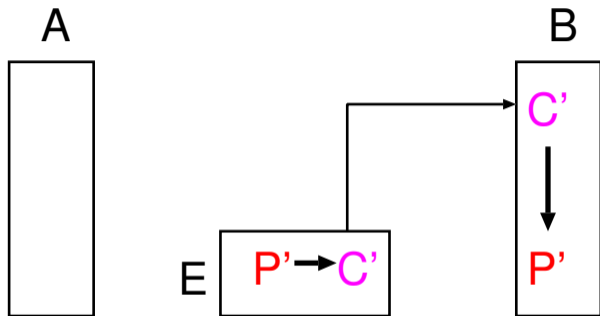
C → P : 復号 (decryption) ・ 解読

改竄



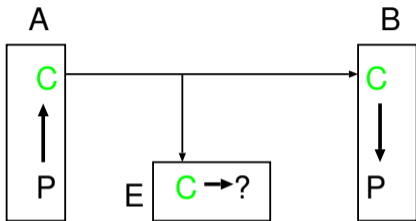
A が送信した情報であることを
確かめられるような仕組みが必要
(電子認証・電子署名)

なり済まし



A が送信した情報であることを
確かめられるような仕組みが必要
(電子認証・電子署名)

暗号 (cryptography)



- 送信者 **A** が平文 **P** を暗号化、暗号文 **C** を送信
- 受信者 **B** が暗号文 **C** を受信、平文 **P** に復号
- 盗聴者 **E** は暗号文 **C** を知っても
平文 **P** を復元できない

→ **B** だけが復号鍵を持っていることが必要

次回に続く