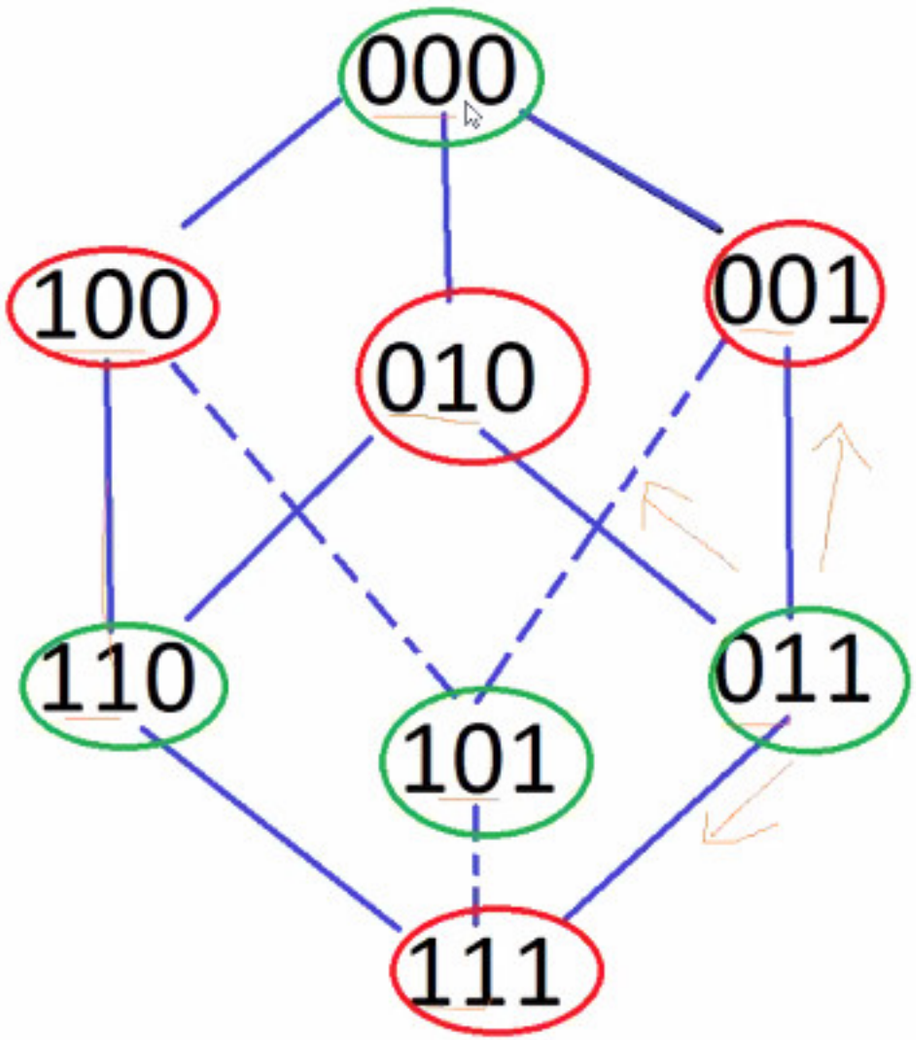


Quanto è potente il parity bit "m"?

- Facile vedere che, **qualunque sia l'm**:
- Presi due messaggi diversi, ***i loro messaggi codificati sono a minimo a distanza 2***
- → anche se c'è un errore (1 bit), un messaggio codificato non può mai diventare un altro messaggio!
- → il codice ha **potenza 1** (identifica tutti gli errori che toccano un bit)

Il fattore "m"...

- Possiamo quindi avere un errore (un bit "flippato") ogni m+1 bits → possiamo identificare un **error rate** di **1/(m+1)**
- Il codice migliore di controllo si ha con **m=1**: in quel caso, possiamo identificare un error rate di **½** (cioè, del **50%!**)

☺ ☺ ☺ ☺ ☺

Quanta ridondanza?

- Quanto stiamo "sprecando" per controllo dell'errore?
- Presto detto: **1/(m+1)**
  (ogni m+1 bits, 1 è per il controllo)
- Il data-rate effettivo sarà dunque
  $1 - 1/(m+1) =$ **m / (m+1)**
- Ad esempio con m=3 → ¼ del data-rate sprecato → ¾ del datarate

Quindi...

- Nel caso migliore (m=1) è vero che possiamo identificare un error rate del 50%, ma...

Ad ogni modo…

- …il codice "parity bit m" ha potenza 1: significa che non appena c'è un errore di "potenza 2" (distanza di Hamming 2, cioè che può invertire due bits) il codice fallisce
- Come facciamo a rilevare errori di potenza più grande…?

I repetition codes

- I i re re pe pe ti ti tion tion co co des des so so no no sem sem pli pli ci ci ! !
- Nel **repetition code R_N** : ogni bit si ripete per **N** volte
- Quindi ad esempio,

  R_3 : 010 → 000111000

  110 → 111111000

La potenza dei repetitions

- E' facile vedere che per ogni due messaggi diversi, i loro messaggi codificati con R_N sono distanti almeno N
- Quindi, con R_N possiamo fare ***error detection fino a potenza N-1***

Quindi…

- Possiamo ***alzare la potenza*** quanto vogliamo!!!

Error correction

- Supponiamo ora di non accontentarci di "sapere" se c'è stato un errore, ma anche di ***volerlo correggere***. Si può fare?

Con i Repetition Codes...

- ...possiamo alzare la potenza di detection fin che vogliamo...
- Forse possiamo usare questi codici anche per ***error correction***...???

Vediamo…

- Error correction significa, dato un valore sbagliato, farlo tornare al valore corretto
- Se avessimo un valore sbagliato (trovato tramite l'error detection con R_N), dovremmo decidere a che valore corretto riportarlo: come fare?
- Potremmo ad esempio portarlo al valore corretto ***"più vicino"***, secondo la distanza di Hamming

Funziona?

- Diciamo che abbiamo un codice i cui messaggi corretti sono distanti **minimo** N (come ad esempio R_N)
- Intervengono **K** errori, che alterano il messaggio, portandoci quindi a distanza **K**
- Riusciamo a tornare indietro al messaggio corretto solo quando?
- Quando questo nuovo messaggio "sbagliato" è ancora più vicino al messaggio iniziale piuttosto che ad altri

Quindi?

- Se un codice che genera messaggi legali distanti **minimo N**
- Ci basta che **K sia meno della metà di N**!
- E quindi il K massimo è semplicemente:

K = N/2 –1 se N pari

= (N-1)/2 se N dispari

Allora:

- Ad esempio, abbiamo che con R_3 , il K massimo è (3-1)/2 = 1
- → R_3, oltre a fare error detection a potenza **2**, è un codice di ***error-correction*** di potenza **1** (può ***correggere*** il messaggio ***senza bisogno di ritrasmissione*** quando un bit arbitrario viene "flippato" !)

Piccolo problema anche qui:

- Si paga un bel prezzo: con R_N , il data-rate diventa **1/N** ☹

Abbiamo visto

- Un esempio molto semplice di error detection (**parity bit m**)
- Un esempio molto semplice di error correction ($\mathbf{R_N}$)
- Nella pratica?

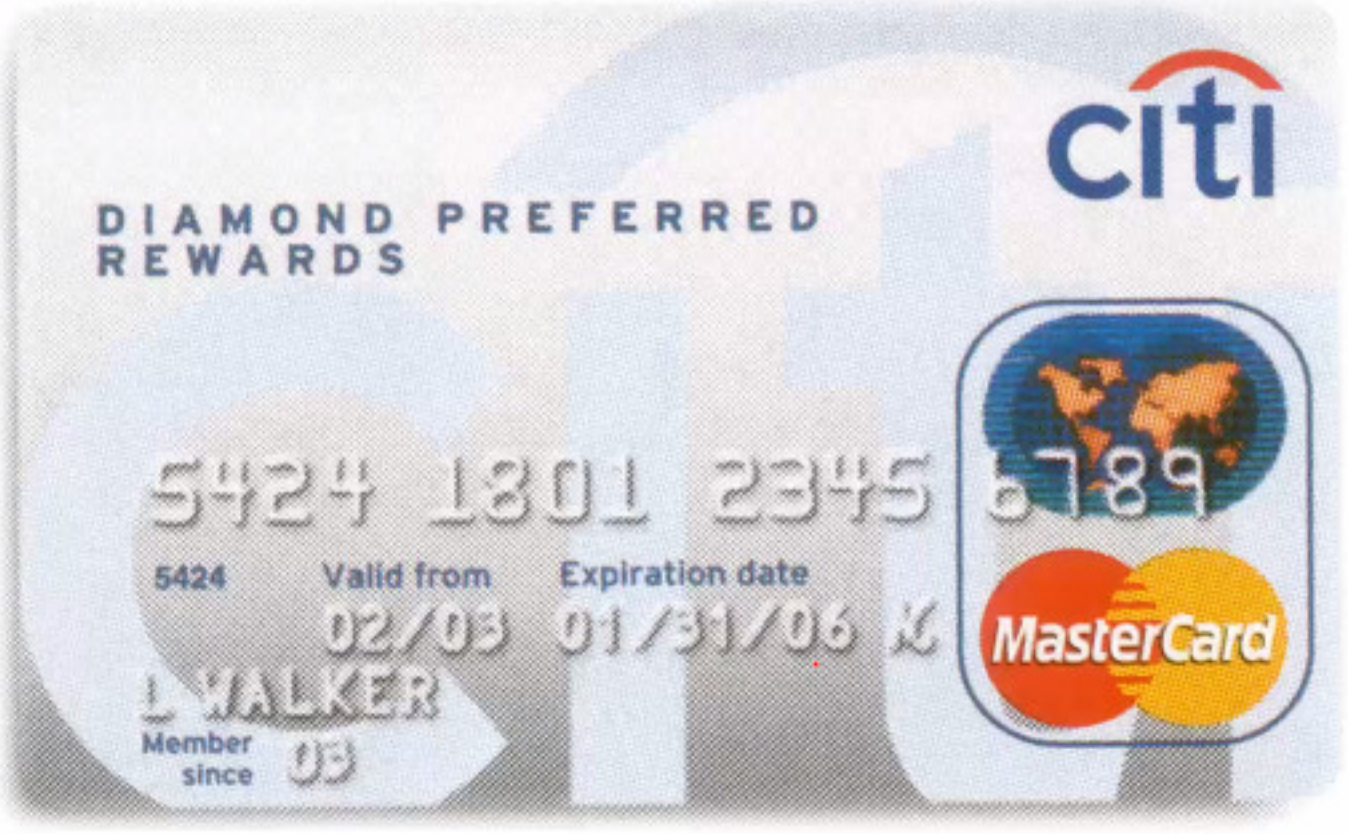
In pratica…

- Codici di questo tipo si usano dappertutto, non solo per le trasmissioni di rete, ma anche dove c'e' informazione (che poi in ogni caso può essere scambiata)
- Vediamo cosa succede con i codici di ***error detection***

Built-in error detection

- Ad esempio, in molti casi l'informazione extra per l'error detection viene aggiunta all'inizio, e fa parte dei dati stessi
- In tal modo, c'è error detection *indipendentemente dal metodo di trasmissione*

Esempio: le carte di credito



Il Codice di Luhn

- Funziona analogamente al parity bit code
- Funziona ***in base 10***, quindi è comodo da usare anche per umani
- Come si fa: si ***raddoppiano le cifre dispari*** (contando da destra a sinistra)
- Si fa poi la ***somma delle singole cifre***
- La cifra di Luhn, che si aggiunge, è quanto manca per arrivare a un ***multiplo di 10***

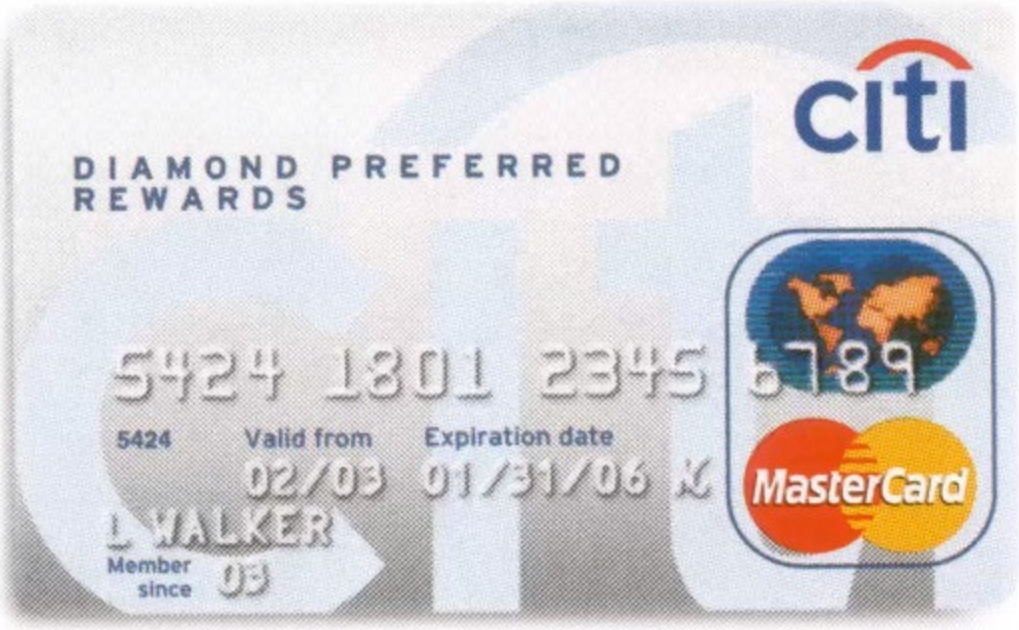
Esempio

- Numero: **537**
- Raddoppiamo le cifre dispari:
- **10 3 14**
- Facciamo la somma delle singole cifre:
- **1 + 0 + 3 + 1 + 4 = 9**
- Cifra di Luhn: quanto manca per un multiplo di 10 → **1**
- → Il numero codificato è **5371**

Error detection

- ***L'error detection*** si fa nel modo ovvio, ribaltando la procedura:
- Si raddoppiano le cifre ***pari***
- Si fa la somma delle singole cifre
- Se il risultato è ***multiplo di dieci***, allora ok, altrimenti c'è un errore

Proviamo:



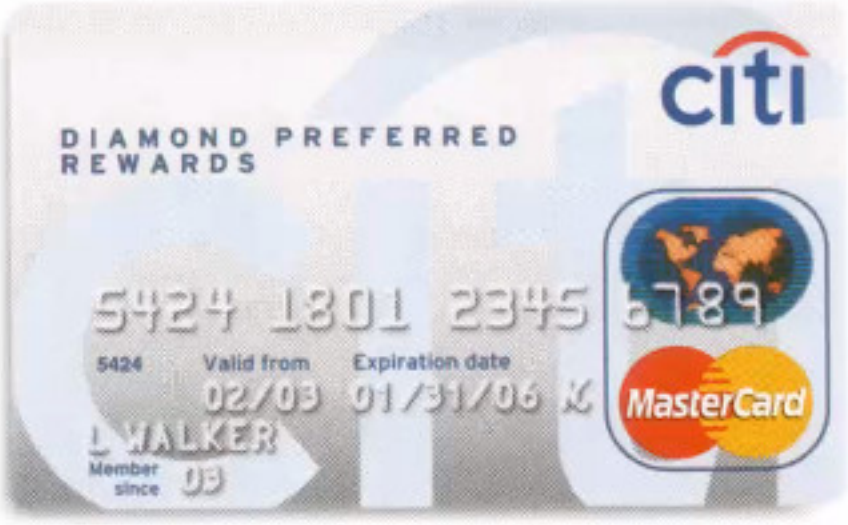
- **10 4 4 4 2 8 0 1 4 3 8 5 12 7 16 9**

somma =70 che è multiplo di 10 quindi ok

Che proprietà ha il codice di Luhn

- Non è così difficile (provate come esercizio!), ma si può dimostrare che è un codice di ***error detection di potenza 1***, cioè trova tutti gli errori nel caso che una cifra sia sbagliata (trascritta male etc)
- Di più: trova anche praticamente tutti gli errori nel caso due cifre contigue siano state invertite (***trasposizione***), es. "25" invece che "52"
  (tutti tranne "90" ←→ "09")

In pratica…



- I produttori di carte di credito (***Visa, Mastercard*** etc), partono col loro numero di 15 cifre, e usano il ***codice di Luhn*** per creare il numero a 16 cifre che usano direttamente nella carta
- In questo modo fanno error detection sul **100%** degli errori per cifre singole, e sul **98%** degli errori di trasposizione

Altri esempi pratici…
i nostri cellulari!!

Proviamo...

- **TIRATELO FUORI!!**
- **#USCITELO**
- Fate il numero: ***#06#**
- → esce un numero di 15 (o 17) cifre
- Questo numero usa ***error detection con Luhn***!!
- Che numero è?

Ricordate come funziona il GSM?

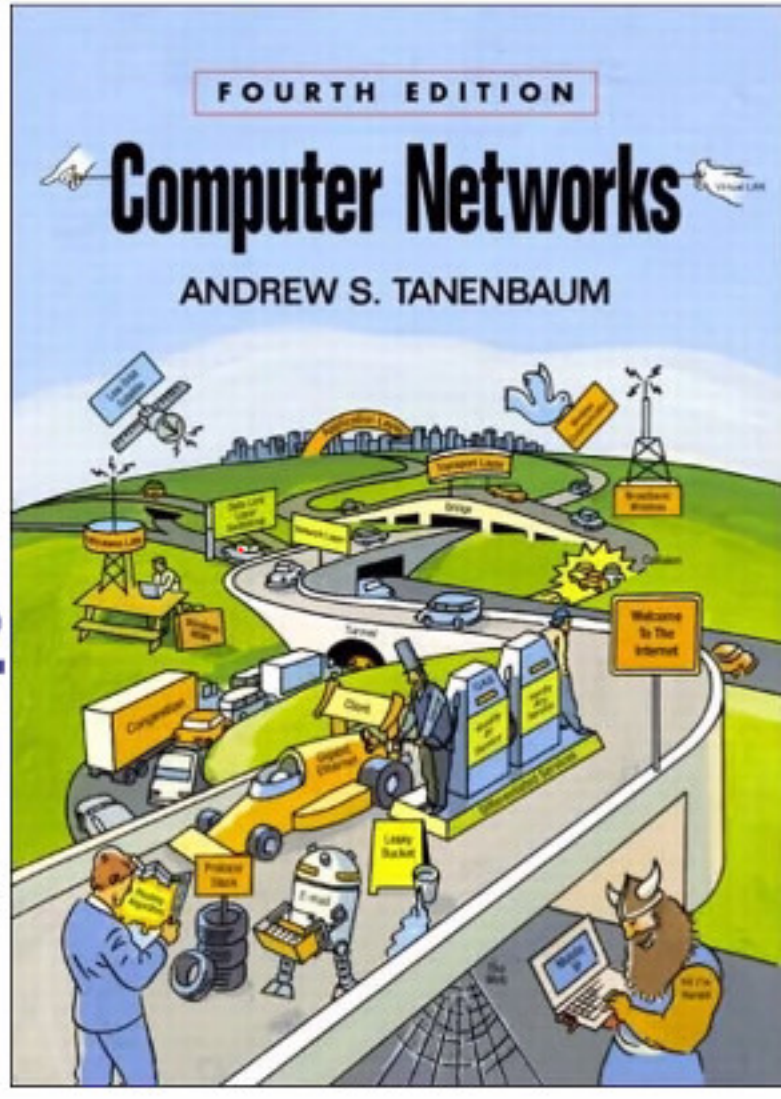
- Evoluzione di 1G: il numero identificativo del cellulare viene inviato alla stazione
- Il cosiddetto **IMEI** (*International Mobile Equipment Identity*)

Altri esempi pratici

- I codici ISBN per i libri (ISBN10)
- Si costruisce la sequenza a scala
- `S1 S2 S3 ...`
  ...data dalle somme parziali delle cifre dell'ISBN (da sinistra a destra)
- La somma dei due ultimi S deve essere **multiplo di 11**

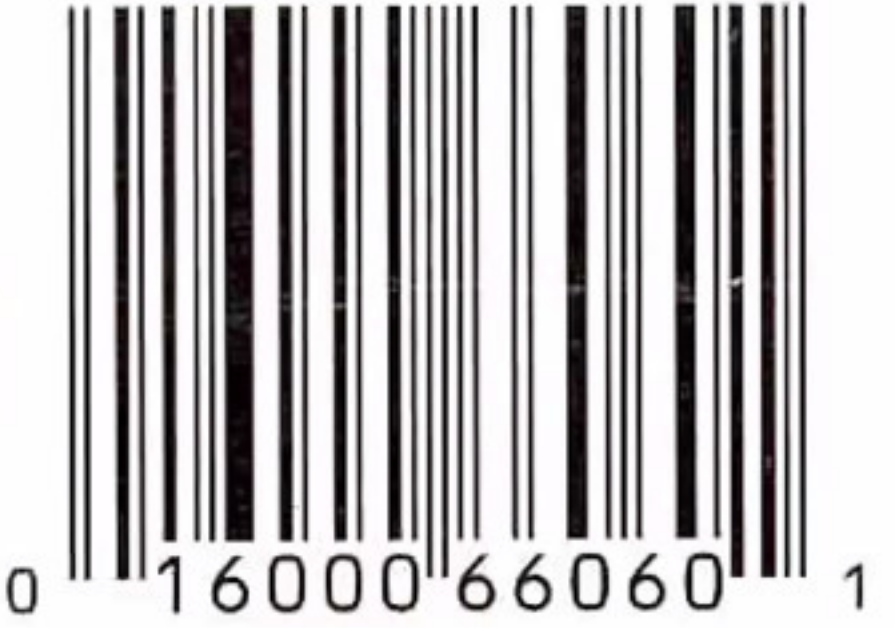
Esempio

- Il codice ISBN del libro del Tanebaum:
- ISBN: **0-13-038488-7**
- Sequenza delle S:
  0 1 4 4 7 15 19 27 35 42
- Somma delle ultime due:
  35 + 42 = **77**
- Multiplo di 11 → **OK!**



Altri esempi pratici nella nostra vita!!

- I **codici a barre** (UPC = Universal Product Code)
- (somma delle cifre dispari) x 3 + (somma delle cifre pari) == multiplo di 10



Esempio:

- Fusilli Voiello:
  076810 500407
  →26 x 3 + 12 =
  78 + 12 =
  90

Esempio:

- Yogurt Bianca Bontà:
  000965 011051
  → 16 x 3 + 12 =
  48 + 12 =
  60

ISBN13

- C'è un nuovo tipo di codici ISBN dal 2007: **ISBN13** (**13** cifre invece di **10**)
- Usa lo stesso codice di error detection usato dai codici a barre

ISBN 978-0-7334-2609-4

9780733426094