Check Digit Calculation for Contract-IDs

Introduction

The Contract Identifier (short: CID; also known as eMA-ID or EVCO-ID) as described by the eMI³ Group and standardized in ISO/IEC-15118 Annex H allows specifying an optional but highly recommended check digit. The purpose of the check digit is the detection of typing errors in human-machine interaction. The syntax of a Contract-ID is:

<CID> = <Country Code> <S> <Provider ID> <S> <ID Type> <ID Instance> <S> **<Check Digit>**

This syntax is based on DIN SPEC 91286 (2011), from where ISO/IEC-15118 adapts and extends it for further international use. The in there specified former check digit is not empowered to detect all common typing errors. Therefore, the here described new algorithm was introduced with the novel of the Contract-ID since it performs better than existing systems such as ISO/IEC 7064, MOD 37, 36 and ISO/IEC 7064 1271-36.

The check digit system described within this document can detect the five most frequent error types made by human operators transmitting a character sequence:

1) single error:	·····a·····	→	·····b·····
2) adjacent transposition:	·····ab····	→	·····ba····
3) twin error:	·····aa····	→	·····bb····
4) jump transposition:	·····abc···	→	·····cba···
5) jump twin error:	·····aca···	→	·····bcb···

The mathematical theory behind is explained by *Chen et al* in the article:

Chen, Y., Niemenmaa, M., & Vinck, A. (2013). A check digit system over a group of arbitrary order. *2013 8th International Conference on Communications and Networking in China (CHINACOM)* (pp. 897-902). IEEE. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6694722

Mathematical Algorithm

Over the 36 alpha-numeric characters, results in the group $(Z_2 \times Z_2) \times (Z_3 \times Z_3)$ are to be applied. Z_2 is the modulo group 'mod 2' and contains only 0 and 1 as values, i.e. all even values correspond to 0 and all odd values to 1. Correspondingly, Z_3 is the modulo group 'mod 3' and contains only 0, 1 and 2 as values, i.e. 3 corresponds to 0, 4 to 1, 5 to 2, 6 to 0, etc. The steps to calculate the check digit in the theory are:

1. For each a from $0, 1, \ldots 35$ there exist unique q and r such that $a = q \cdot 9 + r$, where q can be considered as an element in $(Z_2 \times Z_2)$; and r as an element in $(Z_3 \times Z_3)$.

   Thus, for a string with n characters, $a_1, \ldots, a_n$, we easily have $(q_1, \ldots, q_n)$ and $(r_1, \ldots, r_n)$, where q_i and r_i are the quotient and remainder, respectively when dividing a_i by 9. (In the check equation, q_i and r_i are considered as elements of $(Z_2 \times Z_2)$ and $(Z_3 \times Z_3)$, respectively.)
2. To calculate the check digit a_{n+1}, two check equations are used. In particular two matrices are used: the binary matrix P_1 and the other ternary matrix P_2, where P_1 is a matrix which has $x^2 + x + 1$ as its characteristic polynomial; and P_2 is a matrix whose characteristic matrix is $x^2 + x + 2$.
3. Then we can calculate q_{n+1} and r_{n+1} from the following two check equations, respectively.

   $q_1 P_1 + q_2 {P_1}^2 + \ldots + q_n {P_1}^n + q_{n+1} {P_1}^{n+1} = 0$ (the calculation is in Z_2)

   $r_1 P_2 + r_2 {P_2}^2 + \ldots + r_n {P_2}^n + r_{n+1} {P_2}^{n+1} = 0$ (the calculation is in Z_3)

   The check symbol $a_{n+1} = q_{n+1} * 9 + r_{n+1}$.

Initiation for the Contract-ID

P_1 is initial set to the 2×2 binary matrix $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$ and P_2 to the 2×2 ternary matrix $\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}$.

The calculations in the two check equations are over Z_2 and Z_3, respectively., i.e., in the check equation which employs P_1, the calculation is over Z_2; and in the check equation which employs P_2, the calculation is over Z_3.

For the Contract-ID, the number of digits is $n = 14$, and $n + 1 = 15$ is the check digit.

Setup for the Contract-ID

The binary and ternary matrixes to be used for the Contract-ID check digit calculation:

$$P_1 = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \text{ and}$$

$$P_2 = \begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}$$

The exponents of P_1 over Z_2:

$$P_1 = {P_1}^4 = {P_1}^7 = {P_1}^{10} = {P_1}^{13} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

$$P_1^2 = {P_1}^5 = {P_1}^8 = {P_1}^{11} = {P_1}^{14} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

$$P_1^3 = {P_1}^6 = {P_1}^9 = {P_1}^{12} = {P_1}^{15} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

The exponents of P_2 over Z_3:

$$\mathrm{P}_2 = {\mathrm{P}_2}^9 = \begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}$$

$${\mathrm{P}_2}^2 = {\mathrm{P}_2}^{10} = \begin{pmatrix} 1 & 2 \\ 2 & 2 \end{pmatrix}$$

$${\mathrm{P}_2}^3 = {\mathrm{P}_2}^{11} = \begin{pmatrix} 2 & 2 \\ 2 & 0 \end{pmatrix}$$

$${\mathrm{P}_2}^4 = {\mathrm{P}_2}^{12} = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$$

$${\mathrm{P}_2}^5 = {\mathrm{P}_2}^{13} = \begin{pmatrix} 0 & 2 \\ 2 & 1 \end{pmatrix}$$

$${\mathrm{P}_2}^6 = {\mathrm{P}_2}^{14} = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$$

$${\mathrm{P}_2}^7 = {\mathrm{P}_2}^{15} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

$${\mathrm{P}_2}^8 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Lookup Tables

The correlation between the used alphabet in the Contract-ID and the values of q_n and r_n can be implemented by using the following lookup tables:

Alpha to q_1		Alpha to q_2		Alpha to r_1		Alpha to r_2		Reverse lookup	
char	**q1**	**char**	**q2**	**char**	**r1**	**char**	**r2**	**result**	**Check digit**
0	0	0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	1	16	1
2	0	2	0	2	0	2	2	32	2
3	0	3	0	3	1	3	0	4	3
4	0	4	0	4	1	4	1	20	4
5	0	5	0	5	1	5	2	36	5
6	0	6	0	6	2	6	0	8	6
7	0	7	0	7	2	7	1	24	7
8	0	8	0	8	2	8	2	40	8
9	0	9	1	9	0	9	0	2	9
A	0	A	1	A	0	A	1	18	A
B	0	B	1	B	0	B	2	34	B
C	0	C	1	C	1	C	0	6	C
D	0	D	1	D	1	D	1	22	D
E	0	E	1	E	1	E	2	38	E
F	0	F	1	F	2	F	0	10	F
G	0	G	1	G	2	G	1	26	G
H	0	H	1	H	2	H	2	42	H
I	1	I	0	I	0	I	0	1	I
J	1	J	0	J	0	J	1	17	J
K	1	K	0	K	0	K	2	33	K
L	1	L	0	L	1	L	0	5	L
M	1	M	0	M	1	M	1	21	M
N	1	N	0	N	1	N	2	37	N
O	1	O	0	O	2	O	0	9	O
P	1	P	0	P	2	P	1	25	P
Q	1	Q	0	Q	2	Q	2	41	Q
R	1	R	1	R	0	R	0	3	R
S	1	S	1	S	0	S	1	19	S
T	1	T	1	T	0	T	2	35	T
U	1	U	1	U	1	U	0	7	U
V	1	V	1	V	1	V	1	23	V
W	1	W	1	W	1	W	2	39	W
X	1	X	1	X	2	X	0	11	X
Y	1	Y	1	Y	2	Y	1	27	Y
Z	1	Z	1	Z	2	Z	2	43	Z

Example

The calculation of the check digit for the single Contract-ID 'DE83DUIEN83QGZ' is shown in the next steps as an example. (With $n = 14$ digits.)

STEP 1:

According to the lookup table, we get the following matrices for the digits of the given Contract-ID:

$$D \rightarrow \begin{pmatrix} q_1 \\ r_1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

$$E \rightarrow \begin{pmatrix} q_2 \\ r_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}$$

$$8 \rightarrow \begin{pmatrix} q_3 \\ r_3 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 2 & 2 \end{pmatrix}$$

$$3 \rightarrow \begin{pmatrix} q_4 \\ r_4 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$$

$$D \rightarrow \begin{pmatrix} q_5 \\ r_5 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

$$U \rightarrow \begin{pmatrix} q_6 \\ r_6 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

$$I \rightarrow \begin{pmatrix} q_7 \\ r_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

$$E \rightarrow \begin{pmatrix} q_8 \\ r_8 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}$$

$$N \rightarrow \begin{pmatrix} q_9 \\ r_9 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 2 \end{pmatrix}$$

$$8 \rightarrow \begin{pmatrix} q_{10} \\ r_{10} \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 2 & 2 \end{pmatrix}$$

$$3 \rightarrow \begin{pmatrix} q_{11} \\ r_{11} \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$$

$$Q \rightarrow \begin{pmatrix} q_{12} \\ r_{12} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 2 & 2 \end{pmatrix}$$

$$G \rightarrow \begin{pmatrix} q_{13} \\ r_{13} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 2 & 1 \end{pmatrix}$$

$$Z \rightarrow \begin{pmatrix} q_{14} \\ r_{14} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 2 & 2 \end{pmatrix}$$

STEP 2.

Calculate the check digit $\begin{pmatrix} q_{15} \\ r_{15} \end{pmatrix}$ using the following check equation:

Check equation 1) calculated over Z_2:

$$q_1 P_1 + q_2 P_1^2 + \cdots + q_{15} P_1^{15} = 0$$

Check equation 2) calculated over Z_3:

$$r_1 P_2 + r_2 P_2^2 + \cdots + r_{15} P_2^{15} = 0$$

The calculation gives us $\begin{pmatrix} q_{15} \\ r_{15} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$ which corresponds to the check digit D.

STEP 3.

To validate a given check digit the steps 1 and 2 are to be applied to the ID without the appended check digit. The calculated result is to be compared to the given check digit in place.

Implementation

Due to the fact that only basic mathematical operations (+, *, lookup) on small numbers ($\{0,1,2\}$) are required, the check digit calculation and evaluation can be implemented very efficiently. Even bitwise implementation is possible, if wished. Moreover, depending on the hardware and software used, the speed for $mod\ 2$ and $mod\ 3$ can be much faster than for $mod\ 36$ or $mod\ 1271$. Finally, the same operations are applied for the calculation and evaluation which can lead to lower implementation efforts. The following considerations refer to the computational efforts per calculation/evaluation step:

STEP 1:

The lookup tables can be 'hardcoded' per alphanumeric character of the ID-String so that for each character only 1 lookup is necessary where each lookup results in 4 numbers (⇒ 14 lookups · 4 values = 56 values)

STEP 2:

Only the 56 values of step 1 are used as input for the solving of the two check equations in step 2. In order to derive the solution vectors q_{15} and r_{15}, the first 14 terms ($x_1 P_m^1 \ldots x_{14} P_m^{14}$, with $x = \{q, r\}$ and $m = \{1,2\}$) are to be folded at first. Due to the chosen initiation of P_1 and P_2, there are only three distinct matrices for the first check equation and eight distinct matrices for the second check equation across the 14 exponents. Due to the distributive property of matric multiplication, the first 14 terms of each check equation can be aggregated to the vector terms t_1 and t_2 of three and eight blocks respectively. This allows for a more efficient implementation than in the accompanying reference implementation where the 14 terms are considered independently (.xls-file). For example, the aggregation term for the first check equation is:

$$T_1 = (q_1 + q_4 + q_7 + q_{10} + q_{13})P_1 + (q_2 + q_5 + q_8 + q_{11} + q_{14})P_1^2 + (q_3 + q_6 + q_9 + q_{12})P_1^3$$

Folding this term within each block requires across all blocks 11 vector summations (⇒ 22 summations). Multiplying a vector with a 2x2-matrice requires 4 multiplications and 2 summations. Since P_1^3 is the identity matrice, this multiplication has to be done only for the first two blocks (⇒ 4 summations and 8 multiplications). Considering the summing up of the three blocks (⇒ 6 summations) eventually leads to a total effort of 32 summations and 8 multiplications for t_1. Correspondingly, t_2 consists of eight blocks resulting in seven vector summations (⇒ 14 summations). With P_2^8 being the identity matrice, only the first seven blocks have to be multiplied with the corresponding 2x2-matrices (⇒ 14 summations and 28 multiplications). Considering the summing up of the seven blocks (⇒ 14 summations) eventually leads to a total effort of 42 summations and 28 multiplications for t_2.

If the used programming language does not allow for native calculations in Z_2 and Z_3, the terms t_1 and t_2 can be calculated in Z or R (due to the distributive property of matrice operations). In this case the modulo-calculation needs to be applied after the folding on each value of both vector terms (⇒ 4 modulo operations).

Finally, solving of the resulting two aggregated check equations ($t_m + x_{15} * P_m^{15} = 0$, with $x = \{q, r\}$ and $m = \{1,2\}$) requires two comparisons for both entries of vector q_{15} as well as three comparisons for both entries of vector r_{15} (⇒ 10 comparisons in total; alternatively, a conventional resolving could also be done very efficiently: Due to P_1^{15} and P_2^{15} being constant, vector q could be caculated with two summations and one 'mod 2'-operation. Vector r could be calculated with two summations and two 'mod 3'-operations; for the transformation of the linear equations cf. also the comments in the accompanying .xls-file).

Step 3:

Cf. Step 1 and 2.

Conclusion

All in all, each check digit can be either calculated or evaluated with the following basic mathematical operations on very small numbers (for Z_2 in $\{0,1\}$ and for Z_3 in $\{0,1,2\}$):

- 14 table lookups
- 36 multiplications
- 74 summations
- 4 modulo operations
- 10 comparisons

Although this algorithm may require a few more basic mathematical operations than modulo check digits, all hard- and software is easily able to conduct these calculations very fast – and you catch all the most frequent error types.

Acknowledgment
The above descriptions originate from email-exchange with Yanling Chen (Department of Telematics at NTNU, Trondheim, Norway) between 12/2012 – 01/2013.

The algorithm is based on:

Chen, Y., Niemenmaa, M., & Vinck, A. (2013). A check digit system over a group of arbitrary order. *2013 8th International Conference on Communications and Networking in China (CHINACOM)* (pp. 897-902). IEEE. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6694722

Edited by Jonas Fluhr (FIR at RWTH Aachen) and Simon Schilling (smartlab Innovationsgesellschaft mbH).