The Crypto-Word Game For Learning Number Systems

Ronald I. Frank
CS & IS Departments
Pace University
Graduate Center 1 Martine
White Plains, NY 10606
rfrank@pace.edu

Abstract

This paper introduces a game which provides a fun way to practice number system manipulations. The game gives practice in using number systems other than base 10, which helps motivate learning the conversion algorithms that appear in many introductory texts [Knuth Vol. 2, Frank] and which are given below for reference. The competitive format of a "game" makes the practice more interesting so that the students will spend more time on task and more time thinking about and using other number systems. The game also gives an example of an encryption / decryption algorithm pair.

Keywords: Number systems, games, encryption, decryption, cryptography,

1. THE BASIS OF THE GAME

If we choose a number system with a large enough base where the digits 0-9 are extended with letters, this extended set will contain all the letters needed to extract a meaningful English word (or name). We could always choose base 36. The digits would be 0, 1, …, to 9, A, B, … to Z.

The problem in the game is to choose the minimal base, b, necessary to well-form a Latin 1 (ASCII) alphabet word. The base 10 number equivalent of that base b word or name is an encryption of the word into a number.

Given the decimal number and the base b, we can decrypt the word and recover the original text. The reason for choosing the minimal base b, is that always choosing base b = 36 yields less of a problem decrypting the decimal number. Given the base, there is a unique decryption. However, since large decimal numbers can be decrypted into many words, one for each base, not knowing the base creates a more difficult but more interesting decryption problem.

To prevent "misunderstandings", we suggest the "word" and the base b, always be stored in a neutral safe place for confirmation of student answers at the end.

2. THE GAME

Player (Team) 1:

A. Picks a name or word w.
B. Picks a base b, just large enough to convert w into a base 10 number N_{10}.
C. Stores the answer and b, in a neutral safe place.
D. Presents N_{10} to the opponent(s) (Team 2).
E. Presents hints as to the correct answer to help recognize a successful decryption.

Player (Team) 2:

A. Picks a base b (singly or for a team, many).
B. Decrypts N_{10} (singly or in parallel).
C. Decides if it (they) matches the hints.
D. Presents the single final answer to Team 1 for verification using the answer word and base stored in a neutral safe place.

The teams could each be a Team 1 and a Team 2 at the same time to create a race condition. I.e., they each create a problem for the other team at the same time and then race to a solution.

The game can be played as a homework problem, with the answers presented in the session following the presentation of the problem. The instructor can be Team 1.

3. THE TOOLS OF THE GAME

The required base conversion algorithm should be covered in class or assigned as reading. The computations will have to be done by hand, so teams are useful. Worst case is 26 decryptions, with 13 as an average.
A calculator won't give the answer directly! At best, they convert base 2, 8, 10, and 16. In general, this game re-

quires higher bases, however the algorithm requires substantial calculation, so calculators are really useful.

I have interactive codes (.exe files) that convert integer-integer (10 to b and b to 10) [also fraction - fraction (10 to b and b to 10)] for arbitrary positive integer b. These allow very rapid decryptions, but the class does not learn the algorithm. However, they do develop a sense of number systems. I suggest only calculator assisted manual computations. I also suggest the game be done using positive integers only, to simplify the problem even though it could be done with fractions.

I would not have written this note without using the codes because the examples given below otherwise would represent hours of tedious work. From this observation, I suggest giving no more than a few assignments using this game.

4. THE ALGORITHM(S)

The "Galactic Principle" is that WE do all computations in base 10

A. Given an integer number in base 10, convert to base b

1) Divide (C, C++, JAVA / and %) the decimal number by the base b (rendered in decimal by the Galactic Principle)
2) Record the quotient and the remainder
3) If the quotient is < base we are done and the base b numerals are the final quotient and the remainders in reverse order [from the radix point out]
4) Otherwise [q>b] q is the new decimal number. Go to 1) above

B. Given an integer number in base b, convert to base 10

0) Notice that the digits are base b digits
1) Multiply the left most digit by b (converting digits to decimal equivalents and using decimal arithmetic) or exit if 1 digit
2) Add the next digit to the right, or exit if none left
3) Multiply the result by b
4) Go to 2) above

C. Given a fraction in base 10, convert to a fraction in base b

1) Multiply (C, C++, JAVA *) the decimal number by the base b
(rendered in decimal by the Galactic Principle)
2) Record the integral part and the fraction
3) If the fraction is 0 or repeats a previous one, we are done and the base b numerals, from the radix point out, are the recorded integral parts in the order generated
4) Otherwise, drop the recorded integral part and go to 1) above

D. Given a fraction in base b, convert to a fraction in base 10

0) Notice that the digits are base b digits
1) Multiply the RIGHT most digit by (1/b) (converting digits to decimal equivalents and using decimal arithmetic)
2) Add the next digit to the LEFT, or exit if none left
3) Multiply the result by (1/b)
4) Go to 2) above

5. EXAMPLE OF THE GAME

Player (Team) 1:

A. Picks a short name or word w = rfrank .
B. Picks a base b, just large enough to convert rfrank into a base 10 number N_{10} using algorithm 4B above. The largest letter is R, the 18^{th} letter of the alphabet. Therefore, b = 28, and number N_{10} = 474500984. [Saved away]. The conversion is done using the basic definition of place-based number systems
C. Presents N_{10} = 474500984 to the opponent(s).
D. Presents HINT: 6 answer characters denoting a person .

Player (Team) 2:

- Picks bases and calculates "answers" using algorithm 4A above. These are the answers:
  - b = 8 N_8 = 3422047570
  - b = 16 N_{16} = 1C484F78
  - b = 25 N_{25} = 1NEI1E9
  - b = 26 N_{26} = 1DO92DM
  - b = 27 N_{27} = 161N3QE
  - b = 28 N_{28} = RFRANK ←
  - b = 29 N_{29} = N3PFCQ
  - b = 30 N_{30} = JFO39E
  - b = 31 N_{31} = GHOKGB
  - b = 32 N_{32} = C43MON
  - b = 33 N_{33} = E4GJRO
  - b = 34 N_{34} = AF2JXA
  - b = 35 N_{35} = 9172PY
  - b = 36 N_{36} = 7UI7AW

6. A HARDER FORM OF THE GAME

Team 1 chooses a word, finds a base b $\geq$ the minimum base, generates N_{10} for that word as before. Now however

they go one step further. Team 1 converts N_{10} into a number in a base b' yielding ($N_{b'}$). They give $N_{b'}$ and b' to team 2. Team 2 first must convert ($N_{b'}$) to N_{10} then proceed as before.

.

7. EXAMPLE OF THE HARDER FORM OF THE GAME

Player (Team) 1:

A. Picks a short name or word w = GAME [Saved away].
B. Picks a base b large enough to convert GAME into a base 10 number N_{10} using algorithm 4B above.
C. The largest letter is M, the 13th letter of the alphabet. Therefore, MINIMUM b = 23. They pick b = 27 giving the number N_{10} = 322826.
D. Converts N_{10} = 322826 to N_7 = 2513133 (using 4A)
E. Presents N_7 = 2513133 and b' = 7 to the opponent(s)
F. Presents HINT: 4 answer characters denoting a game.

Player (Team) 2:

- First they must convert N_7 = 2513133 to N_{10} = 322826 using algorithm 4B. Then Team 2 picks bases and calculates "answers" using algorithm 4-A above. These are the answers:
  - b = 8 N_8 = 1166412
  - b = 16 N_{16} = 4ED0A
  - b = 23 N_{23} = 13C5L
  - b = 25 N_{25} = KGD1
  - b = 26 N_{26} = I9EA
  - b = 27 N_{27} = GAME ←
  - b = 28 N_{28} = EJLE
  - b = 29 N_{29} = D6OR
  - b = 30 N_{30} = BSKQ
  - b = 31 N_{31} = APSN
  - b = 32 N_{32} = 9R8A
  - b = 33 N_{33} = 8WEK
  - b = 34 N_{34} = 878U
  - b = 35 N_{35} = 7 I I L
  - b = 36 N_{36} = 6X3E

8. DISCUSSION OF THE HARDER GAME

If Team 1 did not give the base b' = 7 in step E above, the game would be impossibly hard. Team 2 would have to convert N = 2513133 to N_k for k = 6, …, 36 (5 is largest digit here), then for each of these 31 numbers they would have to try all bases from 11-36 (those that use letters). This yields a maximum work effort of 31*25 = 775 number conversions. This teaches a very elementary insight into using number properties for hard to decipher (reversible) encodings.

9. DISCUSSION

I have found many IS students reluctant to learn number systems. By providing competition and a game, there is more interest in the process and therefore more learning.

Although we mostly use the 2, 8,16 family of non-decimal bases, there are uses of general bases. The Radix Sort [Knuth Vol 3] is one such example.

I have taught number systems in both introductory undergraduate computer architecture courses and in more advanced graduate computer architecture courses. I have taught these topics in both IS and CS courses. In all courses, I find some students who need extra help in understanding algebraic manipulations. All of these courses use base 2, 8, and 16 throughout. Floating point units use fractional numbers in base 2 (IEEE 754) and 16 (IBM).

Telecom courses mention encryption. The game gives a feel for the problem without using much math.

The competitive play is better for undergraduates. Graduate students also need the practice, but assigned examples are sufficient.

Above, there is an assumption of limiting ourselves to using only letters of our alphabet for digits. We could extend the alphabet to include all ASCII printable characters or even beyond to Unicode characters yielding bases b equal to hundreds, thousands, or even tens of thousands, but not for this game!

10. REFERENCES

Knuth, Donald E. The Art Of Computer Programming Vol. 2. (Seminumerical Algorithms), 3rd Ed. (1998) pp. 319-325 Addison Wesley Longman. Reading, MA. ISBN 0-201-89684-2.

The Art Of Computer Programming Vol. 3. (Sorting and Searching), 2rd Ed. (1998) pp. 169 -on. Addison Wesley Longman. Reading, MA. ISBN 0-201-89685-0.

Frank, Ronald I. "Definition Of Number, Number Representations, and Algorithms For Base Conversions [in C++]" - Technical Report 135 January 1999, Pace University White Plains, NY.
[This contains the C++ source code for the windows .exe files that compute the four algorithms.]