

# Didactic Games For Teaching Information Theory<sup>\*</sup>

Michal Forišek<sup>1</sup> and Monika Steinová<sup>2</sup>

<sup>1</sup> Comenius University, Bratislava, Slovakia, [forisek@dcs.fmph.uniba.sk](mailto:forisek@dcs.fmph.uniba.sk)

<sup>2</sup> ETH Zurich, Switzerland, [monika.steinova@inf.ethz.ch](mailto:monika.steinova@inf.ethz.ch)

**Abstract.** We developed a set of didactic games and activities that can be used to illustrate and teach various concepts from Information Theory. For each of the games and activities we list the topics it covers, give its rules and related information, describe our practical experiences and give an overview of its scientific background. We also discuss the proper ways to integrate these games into the knowledge acquisition process.

## 1 Introduction

It is a well-established fact that games do belong in the classroom. Various aspects in which games contribute to the educational process were researched and supporting evidence for this fact has been given in many publications.

In recent years, the main focus in this area was on computer games. For the general setting, the review [4] notes that people acquire new knowledge and complex skills from game play, and that the computer games can teach higher-order thinking skills such as strategic thinking, interpretative analysis, problem solving, plan formulation and execution, and adaptation to rapid change. McFarlane et al. [16] point out that games provide a forum in which learning arises as a result of tasks stimulated by the content of the games, knowledge is developed through the content of the game, and skills are developed as a result of playing the game. Recently, Interactive Software Federation of Europe sponsored a major study [24] on the use of computer games in schools in Europe. Other relevant publications on general use of computer games in classrooms include [18, 23].

It would only seem natural that out of all school subjects, computer science would be the one where including computer games would provide the most benefits. As an example, Van Emde Boas [3] mentions how strategic computer games can be used to teach concepts related to information systems modelling and relational databases. Essentially, the player is forced to extract useful information from the game and build an abstract model in order to succeed.

Computer games are becoming well established in the teaching process. However, we are strongly convinced that pen and paper, or even physical, games and activities can have a stronger impact in the classroom – especially if they are tailored to the topic we intend to teach.

---

<sup>\*</sup> This work was partially supported by FILEP grant 351 of ETH Zurich.

The use of such games in teaching mathematics is well researched. See Hatch [6] for an overview, and Rowe [17] for a concrete example. For use of classroom games in teaching concepts from statistics and economy, see multiple publications by Holt and various coauthors, especially [11, 12].

In teaching computer science, the flagship is the book by Bell et al. [2], where the authors give twenty games and activities that can be used to illustrate and teach various concepts from computer science. Ginat [5] focuses on (combinatorial) mathematical games and shows that the development of strategies for such games exercises both rigor and heuristic reasoning – which both present key competences in real-life problem solving. Hill et al. [10] use puzzles and games to teach various concepts in operating systems, and Levitin and Papalaskari [14] show that the reasoning students use to solve pen and paper puzzles can be used to introduce and illustrate algorithm design techniques.

## 2 Role of games in knowledge acquisition

Whenever we decide to use a didactic game or activity in the classroom, we should see the game as a tool, and we should be aware of the goals this tool helps us to accomplish. We find it very helpful to think about the knowledge acquisition process in terms of the Theory of Generic Models (TGM), as developed by Hejný [8, 9]. (See also Sfard [22] for a different but similar theory.) According to Hejný's theory, the process of knowledge acquisition should follow the following six steps:

1. motivation for knowledge acquisition
2. gathering experiences, forming separate models
3. generalization
4. discovery of the generic model
5. abstraction, crystallization of the knowledge
6. abstract, automated knowledge

To briefly illustrate these steps, consider a small child that learns how to count. Separate models include the pieces of knowledge *two candies plus one candy are three candies* and *two cars plus one car are three cars*. In the generalization process, the child then forms a generic model *anything can be counted on fingers, and two fingers plus one finger are three fingers*.

This generic model covers many separate models the child previously developed. But it is still bound to concrete objects, which may hinder the child in further development of this knowledge. In such case, this knowledge needs to be abstracted – in our example towards the symbolic equation  $2 + 1 = 3$ . And only then the usage of this knowledge can become automated – so that the child can apply the equation  $2 + 1 = 3$  without needing to think about any concrete objects.

And where exactly do classroom games enter the picture?

Classroom games and activities of the type we present in our paper may play several important roles. The two most significant ones correspond to the first two steps of the knowledge acquisition process described above.

As we already mentioned in the overview of past research, games can be an excellent way to involve and activate the students. When playing the game, the students encounter problems they need to solve, and their desire to play the game well is a strong motivational factor to do so.

But, in our opinion, even more important is the contribution of the games towards the second step: forming separate models. For example, in Section 5 we present a game where one of the players controls a “robot” using just a few simple commands. On their own, the players will develop a separate model of information coding and using a transmission channel in the context of the game.

We conclude this section with a brief note on the third step: generalization, progress towards a generic model. For each of the games we provide a subsection with background information. This subsection usually contains the generic models the students will later form in their studies of Information Theory. It is possible for the teacher to reveal some or all of these after the students finish playing the game – but this is not required, and we recommend not to do it.

The most exciting part of the knowledge acquisition process is the arrival at the fourth step – the discovery of a generic model. As Hejný showed, for the best didactic effect it is important for the students to make this discovery on their own. By revealing the generic model too early, the teacher is taking the ecstasy of this discovery away from the students. Hence we usually (and with much success) apply the following approach:

- Play a suitable game.
- At some later point in time, introduce a new concept.
- Thanks to playing the game, the students now have two separate models (one from the game, one from the teacher’s introduction), and hence it is easier for them to start the generalization process on their own.

### 3 Overview of the paper

In this paper we focus on teaching concepts from Information Theory, as started by Shannon [20] in 1950. For a recent publication that gives an overview of all topics we discuss in this paper, see for example Cover [1].

The main contribution of this paper is a set of didactic games and activities that can be used to introduce, illustrate, or even teach various concepts in Information Theory. All of the games and activities we present were “field-tested” under various conditions – mostly on camps for talented secondary school students in Slovakia. For each game we include a section where we describe our practical experience with the game, and another section where we discuss the scientific background behind the particular game.

The games and activities in this paper illustrate the following main concepts:

- **The definition of information.** Information we get is equal to our surprise. The less probable the observation, the more information we get.
- **Information can be measured.** Asking a yes/no question may give us a bit of information – but we have to ask “smart” questions.

- **Information is relative.** Same event may give different amounts of information to different people.
- **Transmission channel.** Used to send information “from  $A$  to  $B$ ”.
- **Data coding.** When transferring data, we often need to pre-process it into a form that can be sent via the transmission channel.
- **Data compression.** Reasons, methods, lower bound = information content.
- **Redundancy of languages.** In the natural language each letter carries less than 2 bits of information. Why is this a good thing?
- **Error correction.** Hamming distance: To guarantee that we can correct transmission errors, no two possible transmissions may be too similar.
- **Covert channels.** Misusing environment to transfer information.

In the following Section we give a short note with our variation on the well-known “Twenty questions” game. Each of the remaining Sections describes one of our games and activities.

## 4 Guess the sentence

**Concepts:** The definition of information. Information can be measured. Information is relative.

Bell et al. [2] nicely explain how the common “Twenty questions” game<sup>3</sup> can be modified to introduce the basics of Information Theory. However, we must point out two issues that we feel are not handled correctly in their version.

First, when guessing an entire sentence, Bell et al. give the instruction: “the sentence should be guessed one letter at a time, from left to right”. Clearly, the motivation for this has been Shannon’s original experiment [19]. However, that choice is not appropriate in our context, where our primary goal is to measure information content. By introducing such arbitrary restrictions we are influencing the outcome of our measurements.

We can nicely show this on an example from our practical experience: We had players guessing sentences in the “letter by letter” way. This game was often frustrating: a word was clear after guessing its first few letters, and the player felt that she is wasting questions by being forced to confirm each and every letter in order. These questions gave the player no new information, hence the number of questions did not correspond to the information amount.

The version in which arbitrary questions are allowed is more precise in measuring information content; and also more entertaining from the player’s point of view, as it offers a wide spectrum of strategies – semantic, syntactic, or combined in various ways.

The second point missing in the presentation by Bell et al. is the relativity of information. When playing this game, we often have several players guess the same sentence independently of each other, and compare their scores. In such

---

<sup>3</sup> The activity is called “Twenty guesses” in their book.

situations we must keep in mind that information is relative – and make the secret sentence “equally improbable” for each of the players.

Alternately, we may use this game in the opposite way: to illustrate the point that information is relative. For example, we had players guess the sentence “Cuban rum is made from sugar cane”.<sup>4</sup> On average, players who did not know this fact needed about twice as many questions as those who did.

## 5 Soy-Sugar-Glue

**Concepts:** Data coding. Transmission channel. Covert channels.

### 5.1 Rules

The game is played by two players, called *the robot* and *the navigator*. During the actual game the robot has eyes covered by a blindfold, and he is not allowed to speak. Apart from that, the robot is allowed to do anything he wishes to – but he does not know the goal he is supposed to achieve. On the other hand, the navigator is told the goal when the game starts, but she is only allowed to say three different words. We customarily use the words “soy”, “sugar”, and “glue” (hence the name of the game), but any three distinct words will do.

Multiple robot-navigator pairs can be given the same goal, and then compete which of them can achieve it first.

### 5.2 Materials and preparation

After the rules are explained to the players, and robot-navigator pairs are formed, the players need to be given time to prepare – they need to devise a way how the navigator can manipulate the robot using just those three words. The exact preparation time varies with age, for the age group 15 to 18 usually 30 minutes were sufficient.

It is helpful to give the players some simple example goals, so that they have something to practice on during the preparation time. A list containing some goals we used (including their approximate difficulty) is given in Appendix A.

### 5.3 Strategies and practical experience

There is no clear optimal strategy to this game. There are plenty of ways in which the commands can be mapped to various actions. Usually, the best strategies are those that are flexible enough (i.e., give the navigator a sufficient freedom), but at the same time are systematic enough for the robot to actually remember them. When devising a strategy, it is useful to visualize the robot as a string puppet – we need to be able to make his various body parts move in various directions.

---

<sup>4</sup> In Slovak: Kubánsky rum sa robí z cukrovej trstiny.

One common example of a good strategy is to use commands starting with “soy” to describe body parts, and commands starting with “sugar” to describe movements. For most goals given in Appendix A, it is sufficient to be able to describe the basic body parts (entire person, head, body, left/right arm/leg), and the six basic directions (up/down, left/right, forward/back).

We often find that the players’ designs mimic existing interfaces that they encountered in practice. It is quite common to see an “undo” command that reverses the last action. One particularly lovely interface developed by a pair of players was a special command that put the robot into a “select body part” mode. The robot was using his right index finger to point at various body parts, and the navigator was able to select and confirm the right one. In this way, just a few commands were sufficient to be able to select virtually any body part.

On some occasions we had a pair of players attempting to “cheat” – instead of mapping commands to the robot’s actions, they mapped commands to the letters of the alphabet (e. g., using Morse code or binary). Then, instead of navigating the robot, the navigator simply sent him the goal using their chosen encoding. However, this type of “cheating” is easily prevented. Note that the robot is blindfolded, and therefore unaware of its surroundings. Any task that requires orientation in the surroundings can not be accomplished in this way.

A different (and in some settings acceptable) way of “cheating” is to use intonation, volume, speed of speech and various other effects in order to communicate more than just the meaning of the command – dismay in the navigator’s voice can signalize that the robot is doing something wrong, speed of speech can be reflected in the speed (and amount) of the robot’s movements, etc.

#### 5.4 Background and insights

The rules of this game were developed by the organizers of camps for talented youth, probably around the year 1990. This entertaining game nicely illustrates the concept of a transmission channel, and the need to encode information. The information is the actual content we want to transmit, whereas the transmission channel is something we have available. Information coding is a necessity – we have to process the information we have into a form that can be sent along the transmission channel.

Additionally, there are other, more subtle aspects related to Information Theory in this game. A prominent one is the concept of covert channels (see Lampson, [13]). This is precisely what is happening in the last type of “cheating” we mentioned – in addition to the official transmission channel, there is a possibility to transmit more information using a different, hidden channel that originally was not supposed to be used in this way.

## 6 Reconstructing damaged text

**Concepts:** Redundancy of languages. Error correction.

## 6.1 Rules

- *Non-interactive version 1* (word completion):  
The teacher picks a suitable paragraph of text, damages it using one of the methods outlined below, and then highlights about ten words. Each of the students is given a copy of the resulting text. The students are then given a fixed amount of time (we use 10 minutes) to fill in as many of the highlighted words as they can. Only exact matches count. (Deciphering other words is not required, but it may help to understand the text.)
- *Non-interactive version 2* (question answering):  
Instead of highlighting words, the teacher prepares a set of questions about the original text. The students try to answer as many of them as possible.
- *Interactive version*:  
If this activity is performed in a computer lab, we can use its interactive version. Again, the student is given a text to decipher and understand, only this time the text is displayed in an application that allows the student to control the amount of damage to the text. Initially, this amount is set to maximum. The student’s goal is to decrease the amount of damage until he can read most of the text (and possibly prove it in the same way as in one of the non-interactive versions).

## 6.2 Materials and preparation

**An online script.** We implemented a simple script that can be used to play the interactive version, and also to prepare materials for the interactive version. Currently it supports English and Slovak. Location: <http://ksp.sk/~misof/damage/>.

**Different ways to damage text.** First of all, we note that in all the activities in this section we only consider effects that damage letters – spaces and punctuation remain untouched.<sup>5</sup> When damaging a text we have to choose two different things: how to select the letters to damage, and how to damage them.

The basic choices for selection of letters are *periodic* (every  $n$ -th letter is damaged) and *random* (each letter is damaged with the same probability). Another significant choice is to damage *vowels only*.

The letters we pick up to be damaged can either be *hidden* (replaced by a new “blank” symbol, e. g. ‘\_’), or *corrupted* (replaced by a different letter). There are multiple ways how to pick the new letter to replace the old one; we considered two of them: *uniform* (each letter equally likely) and *natural frequencies* (each letter as likely as it normally is in the given language).

**Preparing suitable materials.** For this activity we usually picked a paragraph of text (about 1000 characters long). Care should be taken to avoid “unfair” texts that contain a significant amount of information only known to some of the

---

<sup>5</sup> Note that as a consequence the word lengths are always preserved.

students. For the non-interactive version 1, we recommend to pick the 10 words by hand, and only to do so after the text is damaged. For the non-interactive version 2, when phrasing the questions it is important to keep two goals in mind: First, the questions should be as unambiguous as possible, ideally with single-word correct answers. Second, the questions we ask must reveal some information about the damaged text; care must be taken to minimize this amount of information.

### 6.3 Strategies and practical experience

An entertaining way to do this activity is to pick one fixed type of text damage, and try to discover what is the largest amount of damage you can have if you want to be able to reconstruct (most of) the original text. The interactive version is intentionally designed with this purpose in mind.

The non-interactive versions can be seen as an “offline approximation” of the interactive version. A good way to achieve a similar effect as in the interactive version is to play the game with 2 to 5 different texts, all damaged using the same method, but with different amounts of damage.

We had a test group of 23 solvers play the non-interactive versions of the game with 5 different Slovak texts. The outcomes are summarized below.

In the non-interactive version 1 (word completion) the most successful solvers actually tried to understand as much of the text as possible. However, there was a significantly large group of successful contestants that only focused on filling in the highlighted words (and their surroundings, if necessary). This behavior may have been influenced by their perception of the time constraint.

In the non-interactive version 2 (question answering) only those people who managed to understand a substantial portion of the text had a chance to actually answer some of the questions. However, despite our best efforts we often received comments from the solvers that some particular questions helped them slightly in trying to understand the text.

If only vowels were damaged, most of our solvers were able to reconstruct almost the entire original text, even if all vowels were hidden or corrupted. If we allow arbitrary letters to be hidden, the amount of damage we can afford becomes decreases. In our test group, a major part of a text with 40% letters hidden was successfully restored by most solvers, but another text with 55% letters missing already proved too challenging for many (but not all!) of the solvers. For the non-interactive version 2 we used a text in which 35% of letters were replaced by new letters (with the same probability distribution as in the Slovak language). This proved to be by far the most difficult text to restore. Summary data on all the performances is given in Appendix B.

### 6.4 Background and insights

Redundancy of natural languages evolved as people needed to understand each other even in the presence of noise. It is precisely the redundancy that allows our

brain to correct minor mistakes, and fill in the missing parts in the speech we hear. This feature of languages is not only well understood, but also exploited. For example, in inherently noisy environments such as aviation [7], intelligibility of spoken communication is increased using standardized phraseology – such as *negative* instead of *no*.

Error correction codes in computer science work in very much the same way: redundancy is what makes them possible. However, redundancy is just a necessary condition, not a sufficient one. This can also be shown in the context of our game: Consider the sequence: **Peter has a \_at**. What exactly does he have? A cat? A hat? Something else? We can not be sure. And in this way we can discover the sufficient condition for an error correcting code: In order to be able to correct errors in the transmission, every two possible messages must have a sufficiently large edit distance.

For many languages most of the information in written text is carried by consonants – and in some languages vowels are commonly omitted in the written form. This can also be shown using our game: vowels account for approximately 40% of letters in English text, but an English text with all vowels removed is easier to reconstruct than an English text with random 40% of letters removed.

## 7 Text message game

**Concepts:** Data compression. Redundancy of languages. Error correction.

### 7.1 Rules

This game is played by several competing pairs of students. Before playing the game, the teacher has to choose a suitable paragraph of text.

- *Version 1* (blackening letters)

From each pair, one student is given a copy of the teacher's text, typeset in a monospace font (i.e., all letters have the same width). The student now may blacken as many letters as she wants to. For each such letter her team gains a point. Letters must be blackened completely, so that they are not readable. Once the first student is finished, she hands over the edited text to her partner. He has to reconstruct the original text. For each incorrect word, his team loses 20 points. The team with most points in the end wins.

- *Version 2* (text message shorthand)

As above, one student is given a copy of the teacher's text. She has to rewrite it onto a clean sheet of paper, using only uppercase letters and spaces.<sup>6</sup> Her goal is to use as few symbols as possible. Once she is done, the provisional score of her team is calculated as the number of characters she saved. Again, as above, her partner then has to reconstruct the original, losing 20 points for each incorrect word.

---

<sup>6</sup> Optionally, numbers and basic punctuation may be allowed.

## 7.2 Material

As in Section 6, the teacher has to select a “fair” paragraph of a text, so that extra knowledge of the topic of text does not help the students in better reconstruction. A good text should only consist of letters, spaces and basic punctuation. Punctuation may be removed by the teacher if desired. Note that in version 1 the monospace font is necessary in order to prevent guessing missing letters from their width.

## 7.3 Background and insights

When writing text messages on cell phones, it is natural to “compress” the text in order to fit in more information. In many languages including English, this leads to significant changes in the written form of the language. One significant type of changes is using phonetic equivalents, e. g., “u”, “2”, and “w8” instead of “you”, “to/too”, and “wait”. Another type of changes is simply leaving out unnecessary letters, e. g., “msg” instead of “message”. This game is built upon this process, which should come natural to most students.

The reason why we decided to present two versions is that they offer a trade-off, and it is up to the teacher to pick the more suitable one. The advantage of the first version is that it is purely mechanical, less time consuming, and incredibly easy to evaluate. On the other hand, the second version is more attractive. It is closer to the “original” text messaging, it gives the players an opportunity to be creative and look for ways to compress the text even more. But to do this they will require more time. Also, we have to check whether the sheets produced by the first players are correct.

# 8 Knocking game

**Concepts:** Data coding. Transmission channel. Data compression.

## 8.1 Rules

The game is played by a teacher and a pair of students.

The students are put into two adjacent rooms with a closed door inbetween. The teacher picks a simple short sentence (4-5 words) in the students’ native language, and tells this sentence to the first student. The first student has some time to prepare. Then, at a moment that is announced to the second student, a five minute period is started. During this period, the student that knows the sentence has to transmit it to his friend in the other room. However, the only allowed way of communication is that the first student is allowed to knock at the door.

The obvious first goal is to transmit the sentence without any mistakes. The second, more challenging goal is to do this using as few knocks as possible.

## 8.2 Materials and preparation

No materials are needed and no preparation (other than finding a suitable location to play the game) is required on the teacher's part.

Between explaining rules and actually transmitting the message the students need to be given time to devise the best strategy they can find. Regardless of the setting we estimate that a minimum of 30 minutes is necessary, but we recommend even more. One good scenario is to explain the rules at the end of a lesson and then play the game during the next lesson (which takes place in a day or so).

## 8.3 Strategies and practical experience

The obvious basic strategy is to transmit the sentence character by character.

The first attempt is usually using unary coding: 1 knock is A, 2 is B, etc. Students that start with this strategy usually quickly realize that it is quite straining (and error prone) to make more than 20 knocks just to transmit a single letter that is close to the end of the alphabet.

Sometimes, especially with younger students, we then encountered crude optimizations, such as shifting more frequent letters (e. g., vowels) to the beginning of the alphabet – so that they are represented by less knocks.

The usual next conceptual step the students discover is to use various knocking patterns to decrease the number of knocks. An especially popular case is to use two knocking patterns, clearly inspired by modern computer mice: “knock” and “double-knock”.

By far, the most common variation of this approach is that the students assign the values 1 and 0 to knock and double-knock, and then for each letter in the sentence they transmit its index in the alphabet. A slightly more efficient (but less common) variation is to use Morse code, with knocks and double-knocks representing dots and dashes.

The optimal variation of this approach considers the possible knock/double-knock patterns (or even includes triple- and quadruple-knocks), orders them according to their total number of knocks and assigns them letters in this order.

Additionally, we sometimes encountered crude compression methods. The most popular method was to use text message style abbreviations (see Section 7.3 for details).

Our experience is that only the best secondary school students are able to proceed beyond this point. The following strategies will more commonly be devised by university students.

When actively trying to minimize the number of knocks, one can discover that the double-knocks actually waste knocks. In the same way short and long pulses were used in telegraphs, we can use pauses of various length *between* the knocks. Mathematically speaking, this observation lowers the number of knocks necessary to transmit  $n$  binary symbols from an expected  $3n/2$  to  $n + 1$ .

The best solution we saw students discover so far was performed by two students who were both successful participants in the International Olympiad

in Informatics. The approach they finally developed was based on dictionary coding – they created a list of common Slovak words, assigned them numbers based on their frequency, and then only transmitted those numbers, using the technique from the previous paragraph. Their solution needed less than 50 knocks to transmit an approximately 30-letter sentence.

## 8.4 Background and insights

The door in this game (the one between the students, used to transmit the message by knocking on it) is actually a model of an *transmission channel*. As you cannot knock letters, there is the need to *encode information* into a form that can be sent by this channel. Furthermore, the rules of the game reward those who can devise an efficient way how to encode this information and *compress* it into as few knocks as possible.

(There is one significant difference between the game and the real life settings: in the game we are *not* trying to minimize transmission time, the compression is measured in a different way.)

The knock/double-knock approach shows how an analog channel can be used to transport a digital signal. The transmissions that use knocks and double-knocks are essentially using a binary alphabet to transmit their message, and timing is used to separate knocks belonging to different symbols.

The idea with pauses between knocks can actually be pursued further – we can introduce pauses of different lengths. When regarded from the theoretical perspective, this corresponds to picking a larger alphabet for the transmission channel,<sup>7</sup> However, this approach has an upper bound on  $k$ : we must be able to transfer the message within the time limit, and this may be impossible for large  $k$  and limited precision of measurements achievable by the receiving student.

See Appendix C for a table with a comparison of the above strategies.

## 9 Conclusion

We presented a set of various games and activities that involve concepts from Information Theory. Most of these activities are original, developed by the authors of this paper. All of these activities are simple enough to be explained to (and played by) 12 years old children – but on the other hand, they have a deep background, and can be challenging even for college students. For each activity, we give our insights into its background, and where possible we provide helpful observations from our practical experience. It is our hope that these activities will, in the future, help the next generations understand Information Theory as something nice, simple and natural.

---

<sup>7</sup> This decreases the number of symbols to transmit by a constant factor. With a  $k$ -symbol alphabet we need approximately  $\log_k 2^n = n/\log_2 k$  symbols to transmit  $n$  bits of information.

## References

1. Cover, T. M., and Thomas, J. A.: Elements of information theory. 2nd Edition, New York: Wiley-Interscience, ISBN 0-471-24195-4 (2006)
2. Bell, T., Fellows, M., and Witten, I.: Computer Science Unplugged... Off-line activities and games for all ages. <http://csunplugged.com/> (1998)
3. Van Emde Boas, P.: Games in the Classroom. OOPSLA'99 workshop (1999)
4. Federation of American Scientists: Harnessing the power of video games for learning. Summit on Educational Games (2006)
5. Ginat, D.: Elaborating heuristic reasoning and rigor with mathematical games. SIGCSE Bull. 39, 32–36 (2007)
6. Hatch, G.: A rationale for the use of games in the mathematics classroom. Topic Issue 19 (Spring 1998) NFER (1998)
7. Hawkins, F. H.: Human Factors in Flight. 2nd edition reprinted by Ashgate (2001)
8. Hejný, M.: Knowledge without understanding. Proceedings of the international symposium on research and development in mathematics education, 63–74 (1988)
9. Hejný, M.: Understanding and Structure. In CERME3 – Conference on European Research in Mathematics Education 3, Group 11 (2003)
10. Hill, J., Ray, C., Blair, J., and Carver, C.: Puzzles and games: Addressing different learning styles in teaching operating systems concepts. SIGCSE 03, 182–186 (2003)
11. Holt, C. A., and Anderson, L.: Classroom Games: Understanding Bayes' Rule. Journal of Economic Perspectives 10(2), 179–187 (1996)
12. Holt, C. A., and Capra, M.: Classroom Experiments: A Prisoner's Dilemma. Journal of Economic Education 31(3), 229–236 (2000)
13. Lampson, B. W.: A Note on the Confinement Problem. CACM 16, 613–615 (1973)
14. Levitin, A., and Papalaskari, M.-A.: Using puzzles in teaching algorithms. SIGCSE Bull. 34, 292–296 (2002)
15. Mahoney, M.: Refining the Estimated Entropy of English by Shannon Game Simulation. <http://cs.fit.edu/~mmahoney/dissertation/entropy1.html> (1999)
16. McFarlane A., Sparrowhawk, A., and Heald Y.: Report on the educational use of games. Shelford Studio, 46 Whittlesford Road, Little Shelford, Cambridge [http://www.teem.org.uk/publications/teem\\_gamesined\\_full.pdf](http://www.teem.org.uk/publications/teem_gamesined_full.pdf) (2002)
17. Rowe, J.: An experiment in the use of games in the teaching of mental arithmetic. Philosophy of Mathematics Education Journal 14, 1–16 (2001)
18. Sandford, R., and Williamson, B.: Games and Learning, A Handbook from Futurelab. Futurelab, Harbourside, Bristol [http://www.futurelab.org.uk/resources/publications\\_reports\\_articles/handbooks/Handbook133](http://www.futurelab.org.uk/resources/publications_reports_articles/handbooks/Handbook133) (2005).
19. Shannon, C. E.: Prediction and Entropy of Printed English. Bell Sys. Tech. J 30, 50–64 (1950)
20. Shannon, C. E.: A Mathematical Theory of Communication. Bell Sys. Tech. J 27, 379–423 and 623–656 (1948)
21. Stur Language Institute (Jazykovedný ústav E. Štúra), Slovak Academy of Sciences: Slovak national language corpus – prim-3.0-public-all. (2007)
22. Sfard, A.: On the dual nature of mathematical conceptions: reflections on processes and objects as different sides of the same coin. Educational Studies in Mathematics 22, 1–36 (1991)
23. Virvou, M., Katsionis, G., and Manos, K.: Combining Software Games with Education: Evaluation of its Educational Effectiveness. Educational Technology & Society 8, 54–65 <http://www.ifets.info/journals/8.2/5.pdf> (2005)
24. Wastiau, P., Kearney, C., and Van den Berghe, W.: How are digital games used in schools? Complete results of the study. European Schoolnet, EUN Partnership AISBL [http://games.eun.org/upload/gis-full\\_report.en.pdf](http://games.eun.org/upload/gis-full_report.en.pdf) (2009)

## A Soy-Sugar-Glue: possible goals

Here we list some possible goals for this game, grouped by difficulty. Easy goals are suitable as examples for practice.

**Easy:** Pick up an object. Rotate on the spot. Get down on all four.

**Medium:** Touch your nose. Clap hands. Score a goal with a football.

**Hard:** Perform a forward roll. Find a bottle, open it and drink from it.

**Very hard:** Jump forward. Give a non-playing person a kiss on the cheek.

## B Reconstructing damaged text: examples, statistics

In Figure 1 we show two short paragraphs of damaged text. In the first paragraph two words are highlighted. For the second paragraph, answer the following question: “What feeling does the player experience?” (Note that the word “player” is clearly readable, hence we are not giving away too much information.)

In Table 1 we present the numbers of successful solvers (out of 23) in our field test of the non-interactive versions of this activity. For version 1, the columns correspond to the 10 words we highlighted, for version 2 these are answers to our 10 questions.

## C Knocking game: comparison of strategies

In Table 2 we present a comparison of different strategies for the English sentence “The rabbit hole went straight on like a tunnel.” (38 letters).

l\_ssr.om g.ms\_and [ct.v.t...s]<sup>1</sup> \_f th\_ typ\_ w\_ pr.sent\_n o.r p.p.r  
 m\_..ply [s.v.r.l]<sup>2</sup> \_mportant r.l.s.

Tois game cas often firsetabng: r ward wvs tlaar after ruetsing  
 ias firnt feu neutirs, and tho player felt trat she is waseini  
 queseires wryn cutssing the rstt of she wyrd.

**Fig. 1.** Damaged texts: 75% vowels missing ; 25% letters replaced.

version	damage type	q1	q2	q3	q4	q5	q6	q7	q8	q9	q10
v. 1	100% vowels replaced	22	23	22	23	22	22	22	22	22	13
v. 1	100% vowels hidden	23	19	21	20	20	22	22	19	23	20
v. 1	40% letters hidden	21	22	20	22	22	22	22	17	10	10
v. 1	55% letters hidden	20	21	20	20	18	13	8	21	20	16
v. 2	35% letters corrupted	22	7	21	15	22	4	0	20	9	12

**Table 1.** Aggregate results of the “Damaged text” activity.

method	knocks
unary coding	442
unary coding based on English letter frequencies	262
plain binary coding of letter indices	206
Morse code	123
optimal binary coding based on letter frequencies	119
plain binary; gap lengths instead of double-knocks	144
dictionary coding; binary using gap lengths	85
dictionary coding; base-4 using gap lengths	50

**Table 2.** Comparison of strategies for the Knocking game.