CodeGambler: A Web-based Game to Understand the Emergence of Linguistic Categories

A thesis submitted in partial fulfilment of the requirements for the degree of

Master of Technology

in

Computer Science and Engineering

by

Rajneesh Agrawal
11CS60R10

Under the supervision of

Prof. Animesh Mukherjee

Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur

West Bengal 721302, India

Declaration by the Author of the Project Report

I, **Rajneesh Agrawal**, Roll No **11CS60R10**, registered as a student of M.Tech program in the **Department of Computer Science and Engineering**, Indian Institute of Technology, Kharagpur, India (hereinafter referred to as the 'Institute') do hereby submit my project report, titled: **CodeGambler: A Web-based Game to Understand the Emergence of Linguistic Categories** (hereinafter referred to as 'my thesis') in a printed as well as in an electronic version for holding in the library of record of the Institute.
I hereby declare that:

1. The electronic version of my thesis submitted herewith on CDROM is in PDF Format.

2. My thesis is my original work of which the copyright vests in me and my thesis does not infringe or violate the rights of anyone else.

3. The contents of the electronic version of my thesis submitted herewith are the same as that submitted as final hard copy of my thesis after my viva voce and adjudication of my thesis in April 2013.

4. I agree to abide by the terms and conditions of the Institute Policy and Intellectual Property (hereinafter Policy) currently in effect, as approved by the competent authority of the Institute.

5. I agree to allow the Institute to make available the abstract of my thesis in both hard copy (printed) and electronic form.

6. For the Institute's own, non commercial, academic use I grant to the Institute the non-exclusive license to make limited copies of my thesis in whole or in part and to loan such copies at the Institute's discretion to academic persons and bodies approved of from time to time by the Institute for non-commercial academic use. All usage under this clause will be governed by the relevant fair use provisions in the Policy and by the Indian Copyright Act in force at the time of submission of the thesis.

7. Furthermore,

   (a) I agree to allow the Institute to place such copies of the electronic version of my thesis on the private Intranet maintained by the Institute for its own academic community.

(b) I agree to allow the Institute to publish such copies of the electronic version of my thesis on a public access website of the Internet should it so desire.

8. That in keeping with the said Policy of the Institute I agree to assign to the Institute (or its Designee/s) according to the following categories all rights in inventions, discoveries or rights of patent and/or similar property rights derived from my thesis where my thesis has been completed.

   (a) With use of Institute-supported resources as defined by the Policy and revisions thereof,

   (b) With support, in part or whole, from a sponsored project or program, vide clause 6(m) of the Policy. I further recognize that:

   (c) All rights in intellectual property described in my thesis where my work does not qualify under sub-clauses 8(a) and/or 8(b) remain with me.

9. The Institute will evaluate my thesis under clause 6(b1) of the Policy. If intellectual property described in my thesis qualifies under clause 6(b1) (ii) as Institute-owned intellectual property, the Institute will proceed for commercialization of the property under clause 6(b4) of the policy. I agree to maintain confidentiality as per clause 6(b4) of the Policy.

10. If the Institute does not wish to file a patent based on my thesis, and it is my opinion that my thesis describes patentable intellectual property to which I wish to restrict access, I agree to notify the Institute to that effect. In such a case no part of my thesis may be disclosed by the Institute to any person(s) without my written authorization for one year after the date of submission of the thesis or the period necessary for sealing the patent, whichever is earlier.

Rajneesh Agrawal	Prof. Animesh Mukherjee
________________	______________________

____

Signature of the Head of the Department

Computer Science And Engineering

Certificate

This is to certify that the report entitled "*CodeGambler: A Web-based Game to Understand the Emergence of Linguistic Categories*", submitted by **Rajneesh Agrawal**, in the Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur, India, for the award of the degree of Master of Technology, is a bonafide record of an original research work carried out by him under my supervision and guidance. The report fulfils all the requirements as per the regulations of this institute. Neither this report nor any part of it has been submitted for any degree or academic award elsewhere to the best of my knowledge.

Prof. Animesh Mukherjee
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
Kharagpur, India 721302

Acknowledgement

I would like to express my gratitude towards my supervisor Prof. Animesh Mukherjee for his immense help, motivation, and guidance to carry out this work. In addition, I would also like to thank Monojit Choudhury (Microsoft Research, India), Vittorio Loreto (ISI Foundations, Italy), Saverio Caminiti and Suman Kalyan Maity for their continued support in the form of discussion, constructive criticism, and clear understanding has helped me to proceed in the right direction. I am really indebted to Prof. Animesh Mukherjee for giving me a wide exposure to the field of linguistic categorisation. Without his invaluable advice and supervision, it would not be possible to complete this work. Finally, I would like to thank all professors, lab administrators and friends for their continuous and unconditional support.

Rajneesh Agrawal

Contents

List of Figures

Abstract

Linguistic category is a lexicon of word-meaning associations that is shared between users existing in an environment. These categories emerge from the complex interactions of human beings amongst themselves and with the environment. Individual from different cultural backgrounds may perceive, and even conceptualize their environment in a variety of ways, each subtly different from the other. For successful communication, they need to agree upon certain linguistic ontologies to understand each other. There exist some conclusions about the emergence of linguistic categories, which have been generated by category games played as computer simulations. We now face the question that, when real human users play the category games, do we see the emergence of similar linguistic categories? We try to address this issue with our web-based game called CodeGambler. In CodeGambler, we apply a simple and easy to follow negotiation scheme, that makes use of memory and feedback. Our experimental results show that our scheme is sufficient to ensure the emergence of a communication system, which the users have organized with their understanding of the environment. This communication system is able to discriminate objects in the environment and it requires only a small set of words.

Chapter 1

Introduction

1.1 Categories

Categories are fundamentals for recognizing, differentiating and understanding the different objects present in the environment. Categories play the basic role in all the above activities. Categories are the conventions that are shared by a given group. These conventions depend on the culture followed by the group. In this perspective, there are some questions which have to be answered:

- How such categories are accepted at a global level without any central coordination?
- Whether categories simply follow an underlying structure of the nature or instead emerge from the complex interactions of human beings among themselves and with the environment.

Answers to these questions lie in the communication among human beings.

In this thesis, we address these questions by modelling a population of individuals who co-evolve their own system of words and meanings by playing our web-based game CodeGambler. The central result of our experiment is the emergence of a hierarchical category structure. This structure is divided into two distinct levels:

- A basic layer, that is responsible for fine discrimination of the environment.

- A shared linguistic layer that groups together perceptions. These perceptions are responsible for communicative success.

Linguistic Category

Linguistic category is a lexicon of word-meaning associations that is shared between users existing in an environment ([7, 8]). Individual from different cultural backgrounds may perceive, and even conceptualize their environment in a variety of ways, each subtly different from the other. For successful communication, they need to agree upon certain linguistic ontologies to understand each other.

Recent years have shown that the approach of self-organization can be easily applied in multi agent models for the emergence of language. In the past there have been many computational and mathematical studies about form-meaning associations. These methods address the learning procedures for form-meaning associations. In this context, a community of language users is viewed as a complex dynamical system. The aim of this system is to develop a shared communication system. Through the use of this system, agents will be able to communicate with each other. There is a still open question: why the words used for communication are very small in numbers for an almost infinite number of different meanings. For example, the few basic color terms, present in natural languages, for an almost infinite number of different colors ([4, 5, 6]).

1.2 The Category Game Model

The computational game model used in our experiment, introduced in [1], is a category game model. This model involves a population of N artificial agents. The model starts from scratch and without any predefined categories. Agents play a number of games, dynamically generating a pattern of linguistic categories. This pattern is shared in the whole population. The Category Game model has the advantage of incorporating an extremely low number of parameters, basically the number of agents N and the JND (described in section 1.2.1) curve $d_{min}(x)$. It gives a very rich and realistic output as compared to the inputs given to the model.

1.2.1 The Just Noticeable Difference (JND)

Humans view the world in a non-uniform way; they have different perceptual precisions for stimuli with different wavelengths from a given continuous hue space [9]. The perceptive resolution power of the individuals limits their ability to distinguish objects/stimuli that are too close to each other in the perceptual space. Psycho-physiologists define the JND as a function of wavelength to describe the minimum distance at which two stimuli from the same scene can be discriminated. In principle, this parameter can either be taken as constant across the whole perceptual interval or be modulated to account for the regions with different resolution powers.

1.2.2 Categorisation

In category games, players have to categories the environment presented to them. For the sake of simplicity and not losing the generality for the purpose of analysis, wheel categorisation can be reduced to a single analogical continuous perceptual channel. Each stimulus presented on the wheel can be assumed to be a real number in the interval [0, 1]. Thus the wheel categorisation can be reduced to the partitioning of the interval [0, 1] into subintervals or perceptual categories.

Each individual has an inventory of form-meaning associations. This inventory links all the perceptual categories (meanings) to the corresponding words (forms). This inventory basically represents all the perceptual categories and their linguistic counterpart (words). The words in our category game are built by using the symbols available in the game. The inventory dynamically expands as the game progresses. Perceptual categories and words associated to them co-evolve dynamically through a sequence of elementary communication interactions, simply referred to as games. All players are initialized with only the trivial perceptual category [0, 1], with no name associated to it initially.

For every game round, a pair of individuals is chosen randomly from the population. In this pair, one player plays as speaker and the other one plays as hearer. In each game round the pair is presented with a new “scene”. In category game model, a scene is defined as a set of $M \geq 2$ objects (stimuli). Based on the stimuli presented in the game, the speaker discriminates the scene. Then he names one of the stimuli (the topic), using the symbols available for communication. If required, he adds new category boundaries to isolate the topic. Then the speaker sends the name to the hearer. The hearer tries to guess the topic with the help of received name. The word to name the object is chosen by the speaker among those associated to the category containing the object. While naming the object he gives a preference for the one that has been successfully used in the most recent game involving that category. A correct guess by the hearer makes the game successful.

Based on the outcomes of the game, individuals may update their category boundaries and the inventory of the associated words.

- In a successful game, both players erase competing words in the category containing the topic, keeping only the word used in that game.
- In failed games, the speaker points out the topic and the hearer proceeds to discriminate it. Then he adds the spoken word to its inventory for that category.

New names (symbolic terms) are invented every time every time when there is a new category for the purpose of discrimination. These new terms are distributed through the population in successive games. All individuals start from the scratch having only the perceptual category [0, 1] with no associated name.

As the game starts, The first phase of evolution comes. During this phase, players have the pressure of discrimination which makes the number of perceptual categories increase. Also in this phase, many different words are used by different agents for some similar categories. Different words for a same category are called synonymy. Initially this kind of synonymy grows and reaches a peak value. When competing words for a same category are erased, the synonymy falls down very quickly. Second phase of the evolution starts from this point i.e. When on average only one word is recognized by the whole population for each perceptual category. During this phase, words expand their reference across adjacent perceptual categories. Words for the neighbouring perceptual categories start joining these categories to form unique "linguistic categories". The coarsening of these categories becomes slower and slower.

1.2.3 JND in the Model

To take JND into account, we define a threshold d_{min}, inversely proportional to their resolution power. In a given scene, the M stimuli are chosen to be at a distance larger than this threshold; i.e., $|o_i - o_j| > d_{min}$ for every pair (i, j). Any two of these M stimuli cannot appear at a distance smaller than $d_{min}(x)$, where x is the value of either of the two. In this way, the JND is implemented in the model. Nevertheless, objects presented in different games may be closer than d_{min}.

1.3 Aim of the Present Work

In order to show the emergence of linguistic categories practically we have developed a communicational game which we call **CodeGambler**. Through the use of our category game we want to show that when an assembly of individuals is given some basic communication rules and an initially empty set of categories and these individuals don't have any external supervision, then they can achieve a non trivial communicating system. This communication system is characterized by only a few linguistic categories. Through the use of our web-based game we want to practically prove the hypothesis that a basic communication which leads to the cultural exchange between individual is sufficient for the emergence of linguistic categories. The individuals play elementary language games. The rules of these games constitute the only knowledge initially shared by the population. They are also capable of perceiving stimuli and communicating with each others.

The goal of our work is also to investigate why the continuous space of perceivable meanings in the world is organized, in language, in a finite and small number of subsets with different names. Apart from the evident example of the partition of the continuous light spectrum in a small number of "basic color terms" this phenomenon is widespread in language.

We already have all the above results about the emergence of linguistic categories [1] but all these results are generated by simulations. These results are discussed in chapter 2. With the help of an extensive and systematic series of simulations it has been shown that if agents in the category game are given the mechanism of memory and feedback, they will form a self organized communication system. The memory and feedback mechanism is sufficient to guarantee the emergence of a self-organized communication system. This communication system is able to discriminate objects in the world. It will only require a small number of words.

The results also show that individuals alone have the ability of forming perceptual categories. Cultural interaction among them is responsible for the emergence and alignment of linguistic categories. Another available result is a typical feature of natural languages: despite a very high resolution power, the number of linguistic categories is very small. For instance, in many human languages, the number of "basic color terms" used to categorize colors usually amounts to ≈ 10 ([4, 5, 6]), in European languages it fluctuates between 6 and 12, depending on gender, level of education, and social class, while the light spectrum resolution power of our eyes is evidently much higher.

We question whether similar results are generated when humans play games similar to Category Game. Through the use of our web based category game, we want to get an answer to this question. We want to know what happens if human players play category games. We want to verify the results generated by simulations to the result generated by a real world category game.

1.4 Organization of the Thesis

The thesis is divided into several chapters. In Chapter 2, we talk about survey of all the earlier works on this field. In Chapter 3, we talk about the web game CodeGambler. In Chapter 4, we show the experimental results. In Chapter 5, we talk about the conclusion of our experiment.

Chapter 2

Background Study

In [1], all the results about emergence of linguistic categories are shown. All these results are generated by simulations. In this chapter, we are going to discuss these results in detail. Our work is to verify these results when human players play category games. In [1], it has been shown that the emergence of linguistic categories occurs in two main stages:

- First stage is a stage where players do not understand each other. They do not have any categories at all.
- Second stage is a stage where communication has reached to very high success. This is due to the emergence of a self organized communication channel.

In spite of this high success in the second stage, there are still evolving perceptual categories and a finite fraction of failures. The reasons for these failures are slightly unaligned categories and ambiguities.

2.1 Synonymy

When at the same time, many different words are used by different agents for some similar categories; all those words are called synonymy. Figure 2.1 shows the growth and decline of synonymy. Growth of synonymy indicates first stage and decline indicates second stage. Synonymy, in the context of the "naming game" (a single object to be named by the use of available symbols), already has been studied [10]. All individuals, when they do not have any associated name for the category, create new words with zero probability of repetition. This leads to the initial growth of the words assigned to each perceptual category. So the synonymy graph rises at the first stage. New words are spread through the population in later games. Whenever a word is understood by both players; other competing words for the same category are forgotten. This eventually leads to only one word per category and also the reason of decline of synonymy.

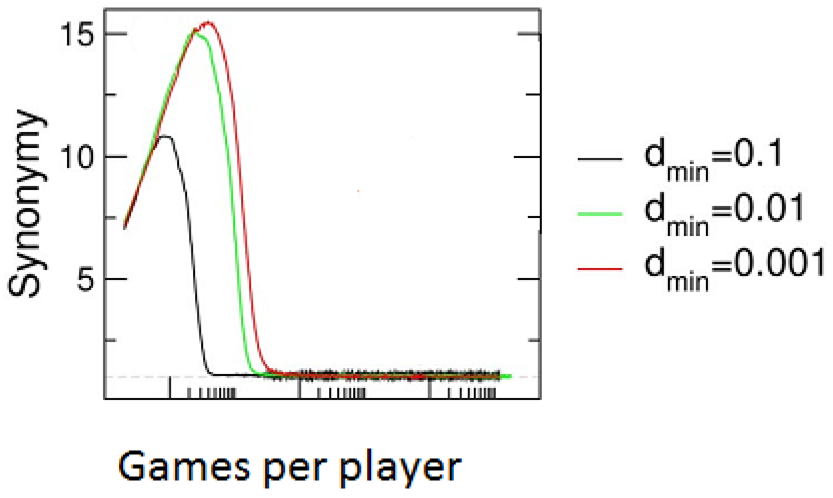


Fig 2.1: Synonymy graph
Courtesy of Puglisi A., Baronchelli A. and Loreto V., "Cultural route to the emergence of linguistic categories", Proceedings of The National Academy of Sciences (PNAS). [1]

2.2 Success Rate

Figure 2.2 shows the graph between success rate and games per player. During the growth of the dictionary, the success rate is very small. The subsequent reduction of the dictionary corresponds to a growing success rate that reaches its maximum value after all the synonymy has disappeared.

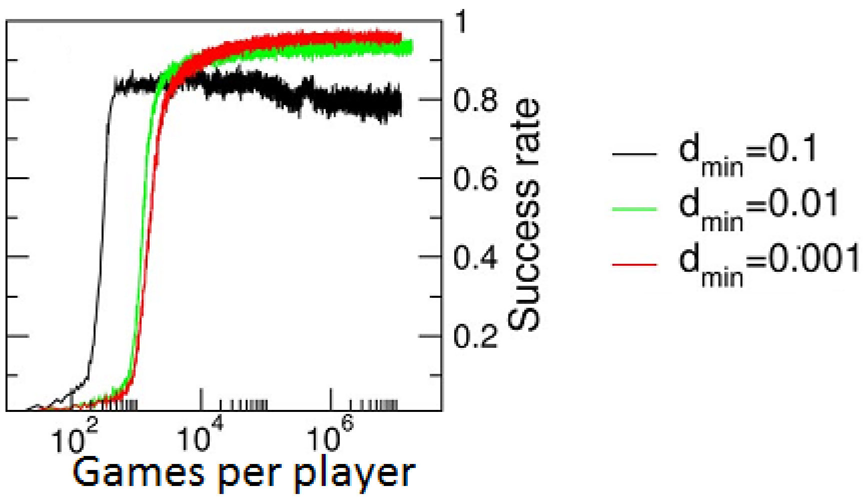


Fig 2.2: Success rate graph
Courtesy of Puglisi A., Baronchelli A. and Loreto V., “Cultural route to the emergence of linguistic categories”, Proceedings of The National Academy of Sciences (PNAS). [1]

2.3 Categories

Figure 2.3 shows the graph between categories and games per player. The first step of each game is, in fact, the discrimination stage. In this stage the speaker (possibly followed by the hearer) may refine his category inventory to distinguish the topic from the other objects. This causes the growth in the number of perceptual categories. The growth of the number of perceptual categories n_{perc} is denoted by dashed lines in Figure 2.3. The growth in n_{perc} for each individual is limited by the resolution power. Resolution power means in a game, two objects cannot appear at a distance smaller than d_{min}. The average number of perceptual categories per individual grows logarithmically, and for many practical purposes it can be considered constant.

The emergence of linguistic categories proves that individuals are able communicate with each other. It also proves that there exists a coordination of the population on a higher hierarchical level. Linguistic categories emerge as totally self-organized and is the product of the (cultural) negotiation process among the individuals. The average number of linguistic categories per individual (Figure 2.3, solid curves) increases during the first stage (where communicative success is still lacking), then decreases and stabilizes to a much lower value.

As the resolution power is increased i.e., as d_{min} is diminished the asymptotic

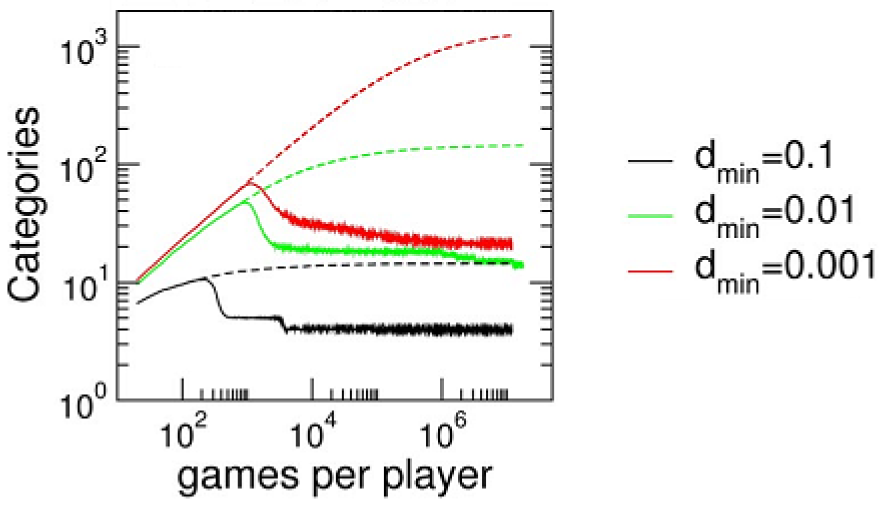


Fig 2.3: Categories graph
Courtesy of Puglisi A., Baronchelli A. and Loreto V., "Cultural route to the emergence of linguistic categories", Proceedings of The National Academy of Sciences (PNAS). [1]

number of linguistic categories increases. This observation shows that the number of linguistic categories n_{ling} is trivially constrained by d_{min}, with a relation of the kind $n_{ling} \propto 1/d_{min}$, implying a divergence of n_{ling} with the resolution power. The number of linguistic categories is inversely proportional to the resolution power.

2.4 Overlap

The success rate is expected to depend on the alignment of the category inventory among different individuals. In the first stage there are unaligned category boundaries so the success rate is low. As the game progresses alignment increases so the success rate also increases. The degree of alignment of category boundaries can be measured by an overlap function O. This function will return a value proportional to the degree of alignment of the two category inventories, initially 0 and reaching its maximum unitary value when they exactly coincide. Figure 2.4 shows the graph between overlap and games per player. The curves of Figure 2.4 display the overlap function. It shows that the alignment of category boundaries grow with time.

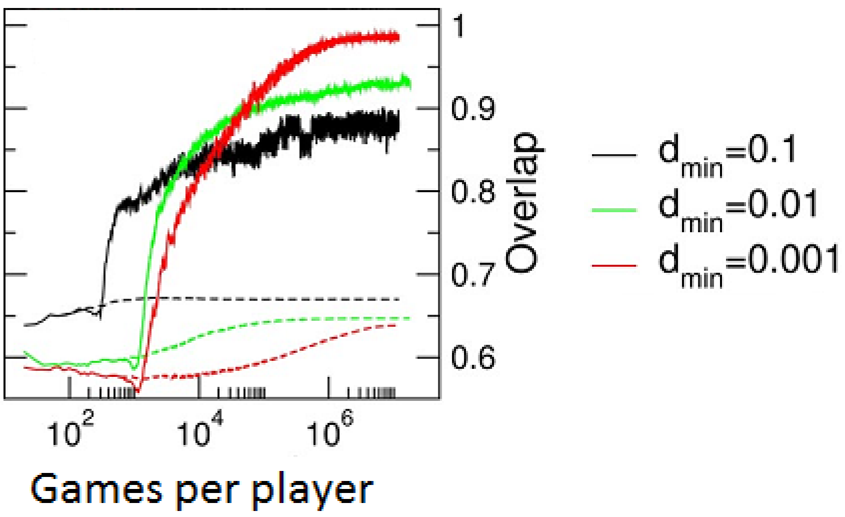


Fig 2.4: Overlap graph
Courtesy of Puglisi A., Baronchelli A. and Loreto V., “Cultural route to the emergence of linguistic categories”, Proceedings of The National Academy of Sciences (PNAS). [1]

Chapter 3

CodeGambler

We propose a coordination game exploiting the web in order to understand the emergence of linguistic categories. CodeGambler in a client server game. The coordination game should be interesting enough to attract subjects to play the game so that we can gather a large amount of data that would reflect how humans agree on naming categories.

Players will play it for fun but in the background, the game server will record all the statistics related to the game. The client side of the game is developed in flash builder. So The game will run in all platforms and in all major browsers, but to run the game it will be necessary to install Adobe Flash Player 11. Game server is implemented in JAVA 1.7. We are also using MySql 5.5 for recoding user data in the database. See Appendix A for more details about the interfacing of flash and java.

3.1 Introduction to the Game

Players have to register themselves to the game server to start playing the game. Registration window is shown in Figure 3.1. Once they have registered they can login to the game and start playing.

Game Server will select any two players at random and will pair them. Since its a two player game. One player cannot start the game until he is paired with some other player. The players do not know who they are paired against. The players do not know the score/rank of the player they are playing against. They have no ways to communicate with each other apart from the game messages.

The pair will play a round of games, or as we call them, “bets”, in our game. During each bet, one of the player will be randomly assigned the role of a gambler, and the other one the role of the seer.

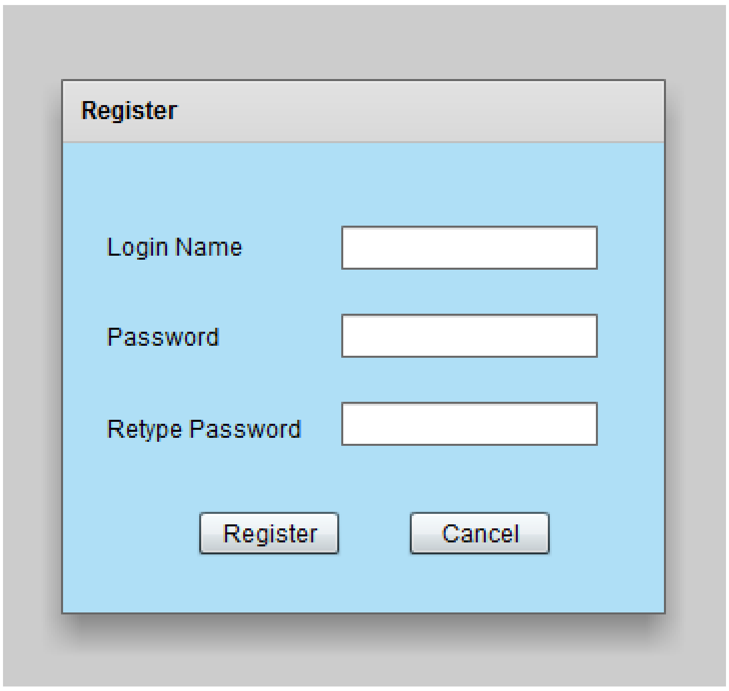


Fig 3.1: Registration window

Login Window

The login window of CodeGambler in shown in Figure 3.2. Once registered, a user can use the login window to join and play the game.

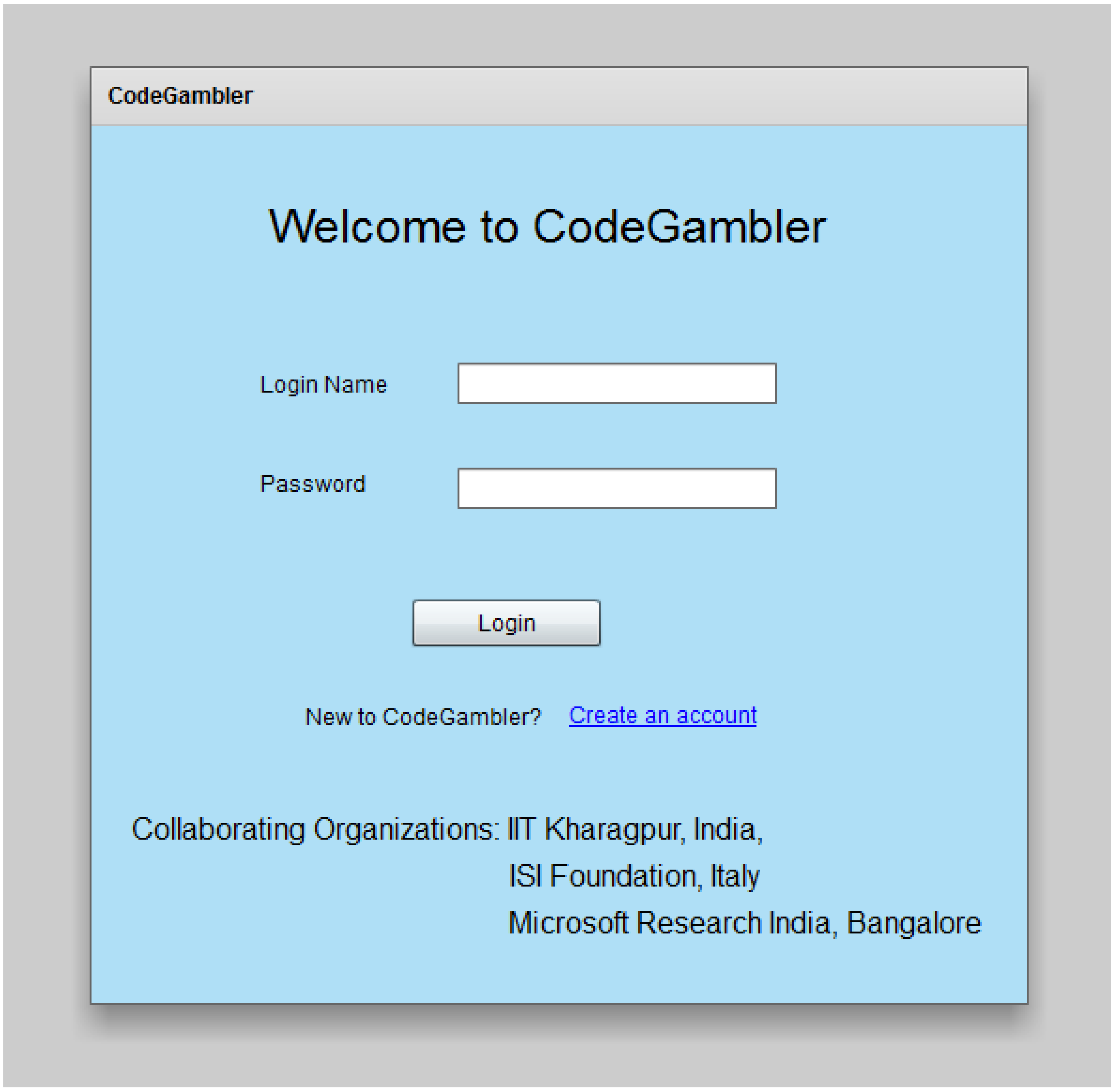
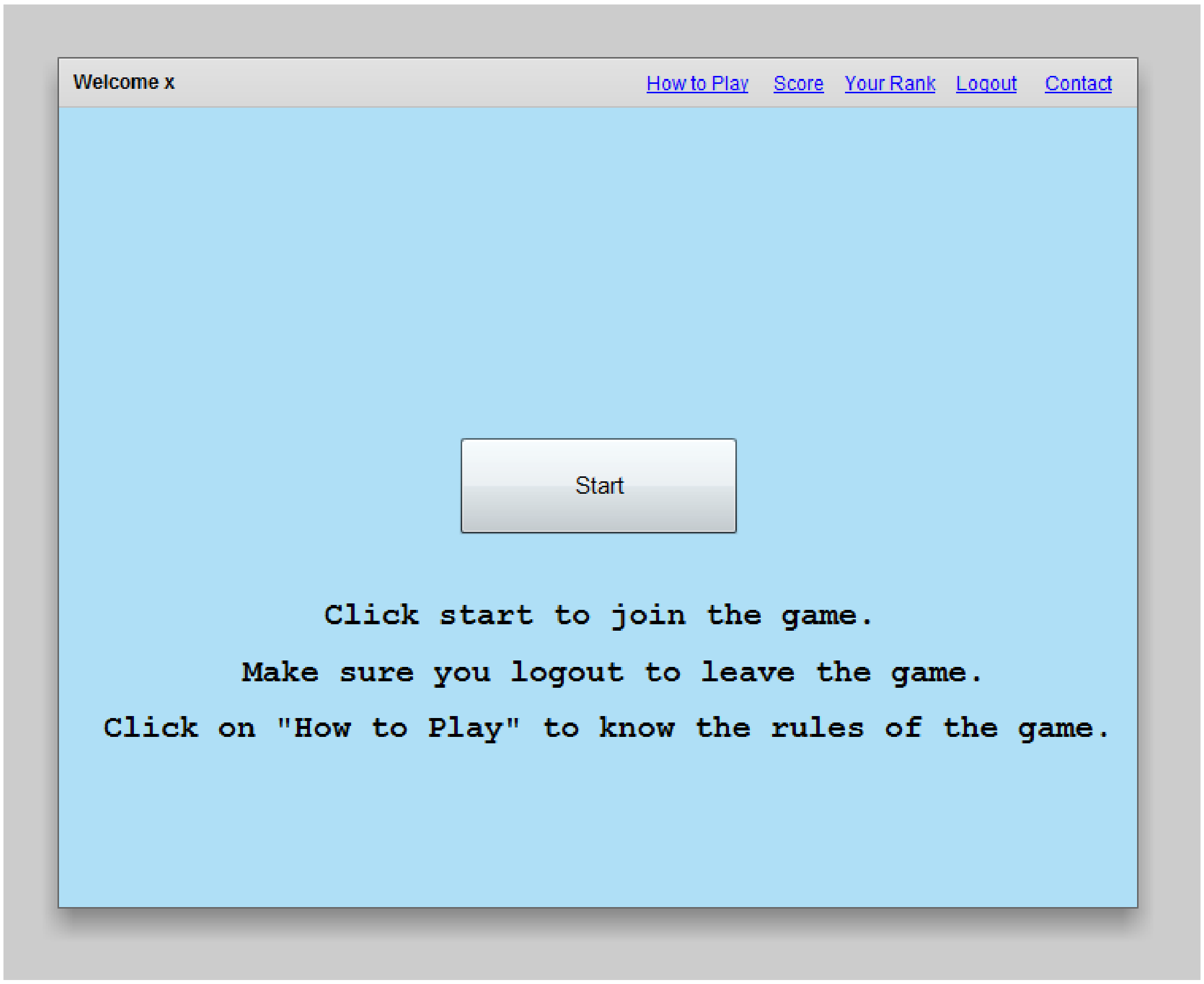


Fig 3.2: Login window

Start Window

The following figure shows the start window of CodeGambler. A user is directed to this page, when he first logs in and unless another player joins the game, the user must wait at the start window, for a pairing to occur.



Click start to join the game.

Make sure you logout to leave the game.

Click on "How to Play" to know the rules of the game.

Fig 3.3: Start window

3.2 Seer

The seer will press a button “generate” on his screen to foresee final position of the Roulette wheel for that bet. This is shown in the Figure 5. There is a phase angle assigned to each player (remains fixed for the particular player throughout). Note that the players have no way to find out what their respective phase angle is.

The seer has to communicate the gambler through the “gambler’s code” (constructed out of solar system symbols) about the location where the gambler should place the bet to get maximum profit. In the Figure 3.4, this is the location with 97 [topic]. If the gambler places the bet on other locations, he might make a small profit (as in 6 and 7). The topic cell is highlighted with different colour (Cyan) to inform the seer that it is a topic cell. The other cells are highlighted with the colour (blue) different from the topic cell. Seer will send a message to indicate the topic cell. Game messages are constructed using the available communicating symbols. Here, in our category game the value of M (no of object present in the scene) is 3.

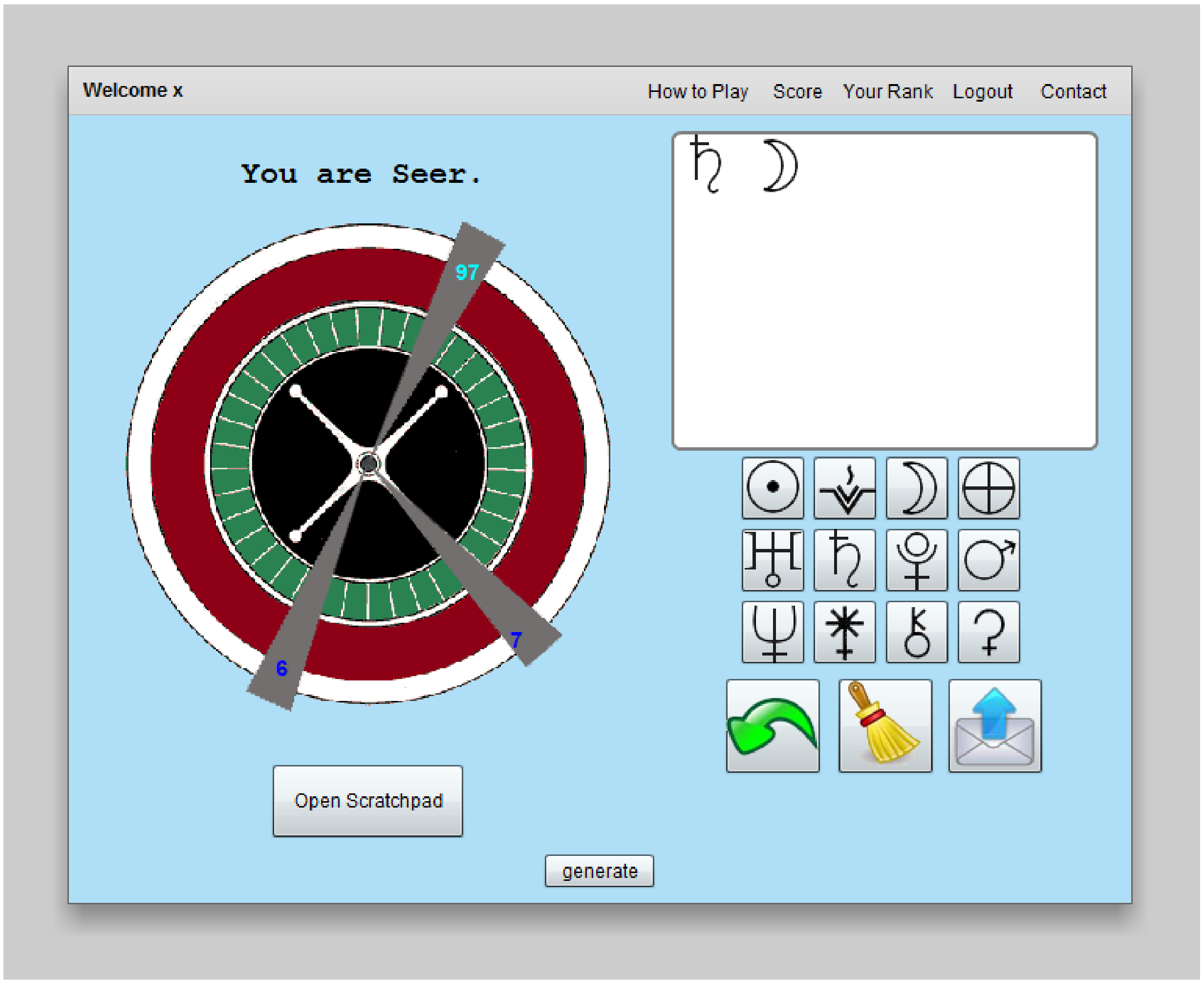


Fig 3.4: Seer window

3.3 Gambler

The gambler sees a Roulette wheel on his screen which has the topic cell, plus a few more cells some of which may or may not be shown to the seer. The values on the cells are not shown to the gambler until he bets on one of the cells. This is shown in the Figure 3.5. Note that the wheel is rotated by an angle theta which is the phase angle for the gambler. In our example, phase angle of the gambler is 110 degree. So the wheel in the Figure 3.4 is rotated by 110 degree resulting a wheel of Figure 3.5. Again the phase angle is not known to the gambler, but it remains fixed for him/her throughout.

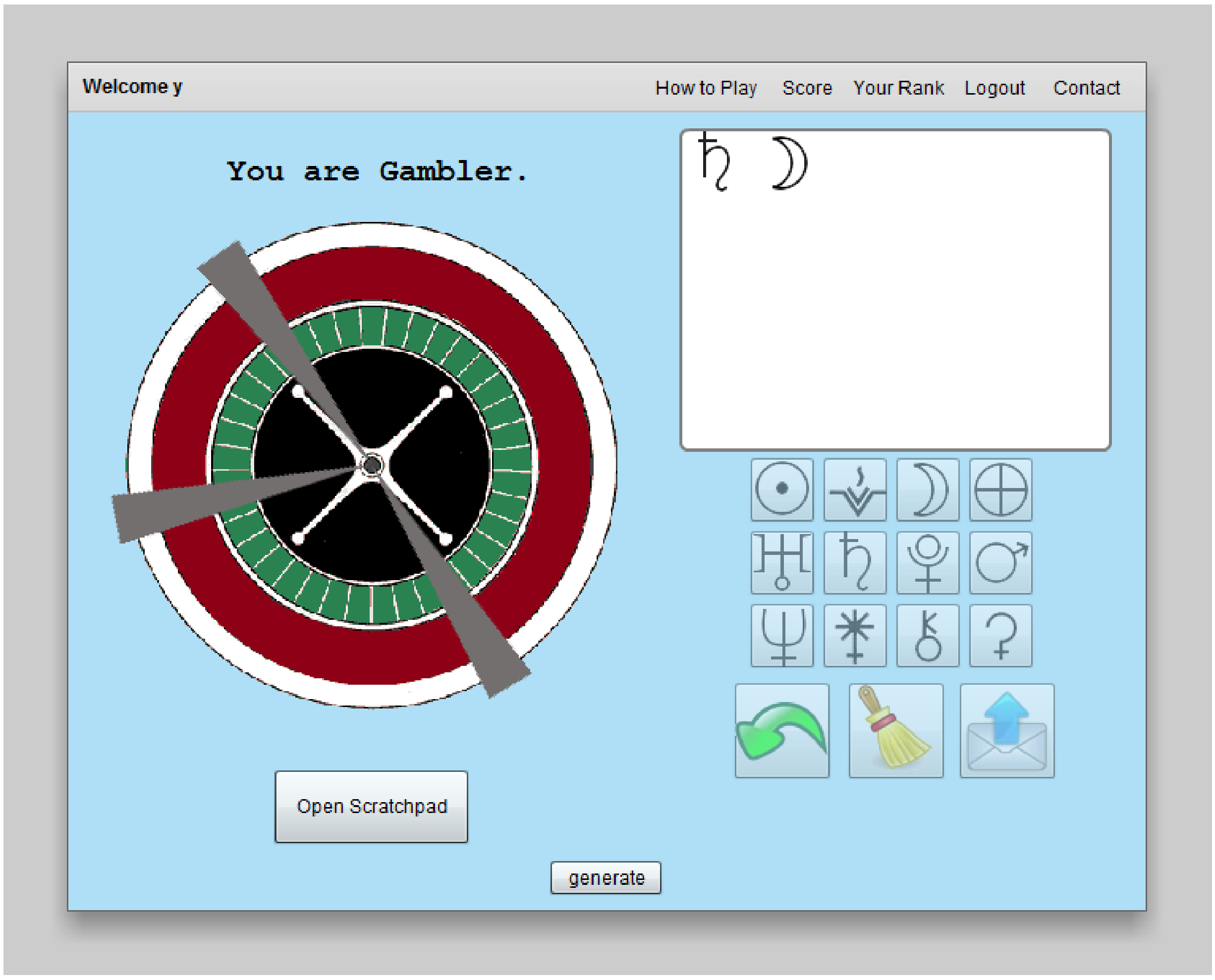


Fig 3.5: Gambler window before bet

Based on the message received, the gambler places the bet on one of the cells by clicking on it. The points associated with all the cells are then displayed to the gambler along with the 'Cyan' background of the topic (to show that it was the topic) and the blue background for other cells. This is shown in Figure 3.6. Both the players get score that is equal to the one displayed on the cell clicked by the gambler. this completes one round of the game.

After one round is completed they move on to the next bet, where again they are assigned new roles, new wheels are displayed, though the phase angles for both the players remain fixed. The final score is cumulatively summed over the bets during a round, and the global score of a player will be the sum of the scores for all rounds he played.

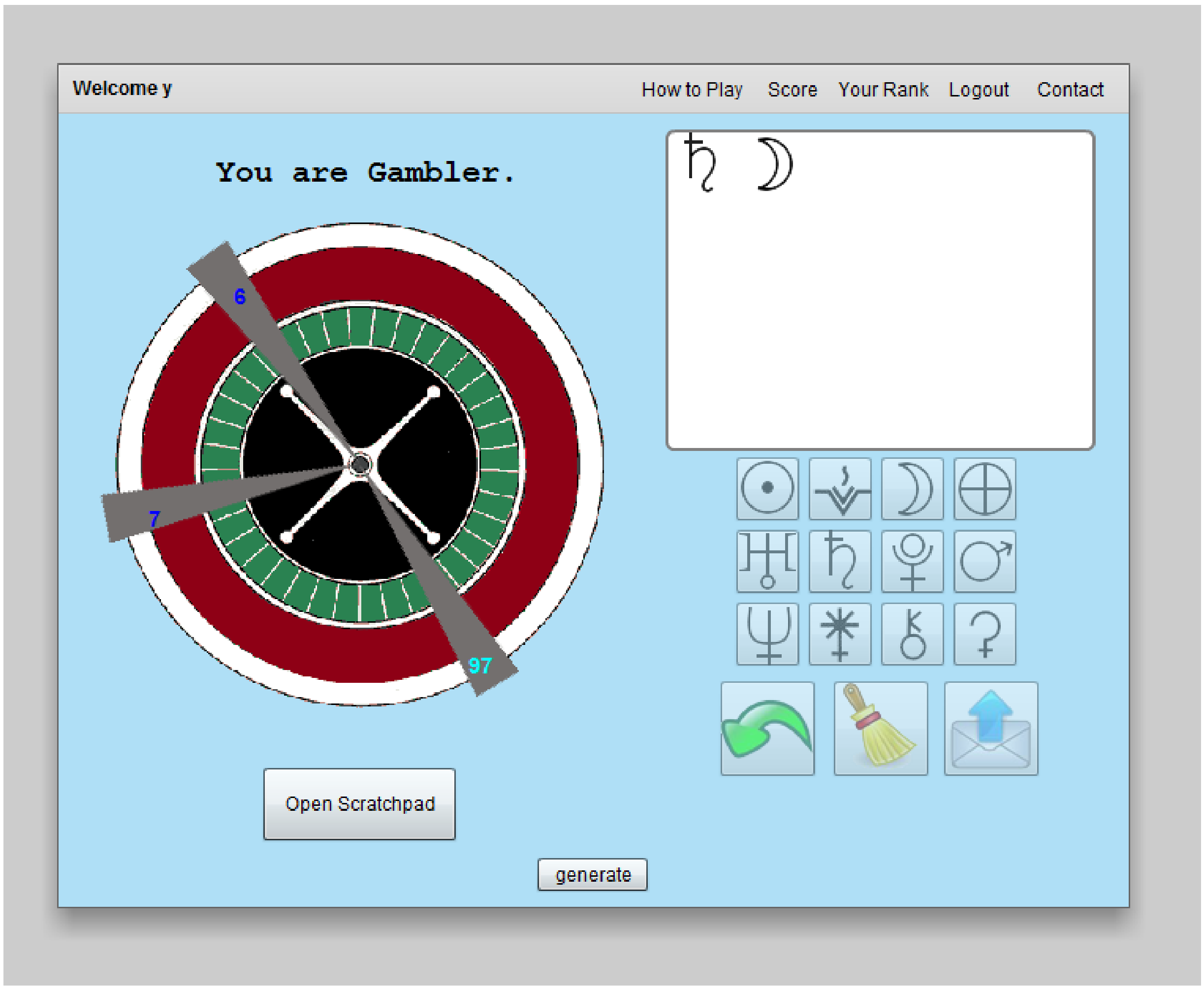


Fig 3.6: Gambler window after bet

3.4 Scratchpad

During the game, at any time the players can consult their respective "memory pad" which is a scratch pad with the Roulette wheel in the background. Players can draw lines on the pad by clicking on the pad. When a player clicks on the pad, a line will be drawn from the position of click to the centre of the Roulette wheel. This line is drawn with the black colour. This is shown in Figure 3.7.

The game will also automatically draw 2 lines for the topic cell displayed on the wheel. The lines are drawn to make it easy for the players to remember what the positions of the cells on the wheel are. Note that these lines are drawn only to suggest the position of the topic cone. Any way he can draw lines where ever he wants. For seer, the lines corresponding to the topic cell are drawn with the colour 'cyan'. This is shown in Figure 3.7.

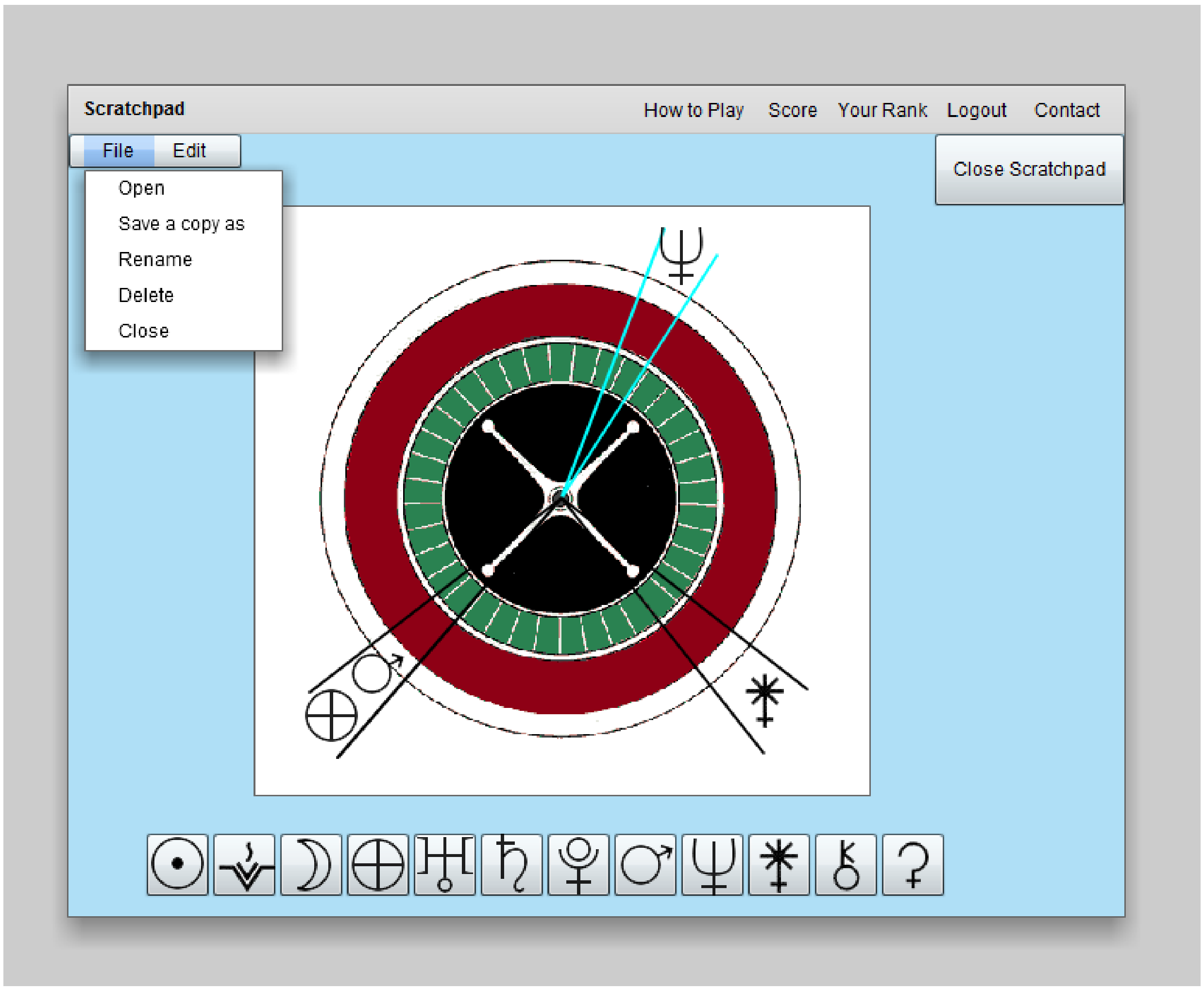


Fig 3.7: Scratchpad

For gambler, the lines for corresponding cells are drawn with the 'cyan' colour. Game will not draw the suggested lines until gambler places a bet on any of the cells. Once he places the bet on one of the cells, corresponding lines for the topic cones are drawn on the scratchpad. On the scratchpad Players can also drag symbols that are available at the bottom of the scratchpad.

There are two options in the edit menu to erase the content of the scratchpad.

‘clear’ option will clean the whole scratchpad and ‘undo’ option will undo the last change player made on the scratchpad. There are five options available in the file menu of the scratchpad; open, save, rename, delete and close. At any point of the game, players can also save the current scratchpad with any name of their choice. He can also open any of the scratchpads he ever saved during any of the game.

If the players logout without saving the current scratchpad, Game will automatically save that. If the player logins then the scratchpad which he was working last will automatically be opened for him. Remember player can always clean the scratchpad at any point of the game.

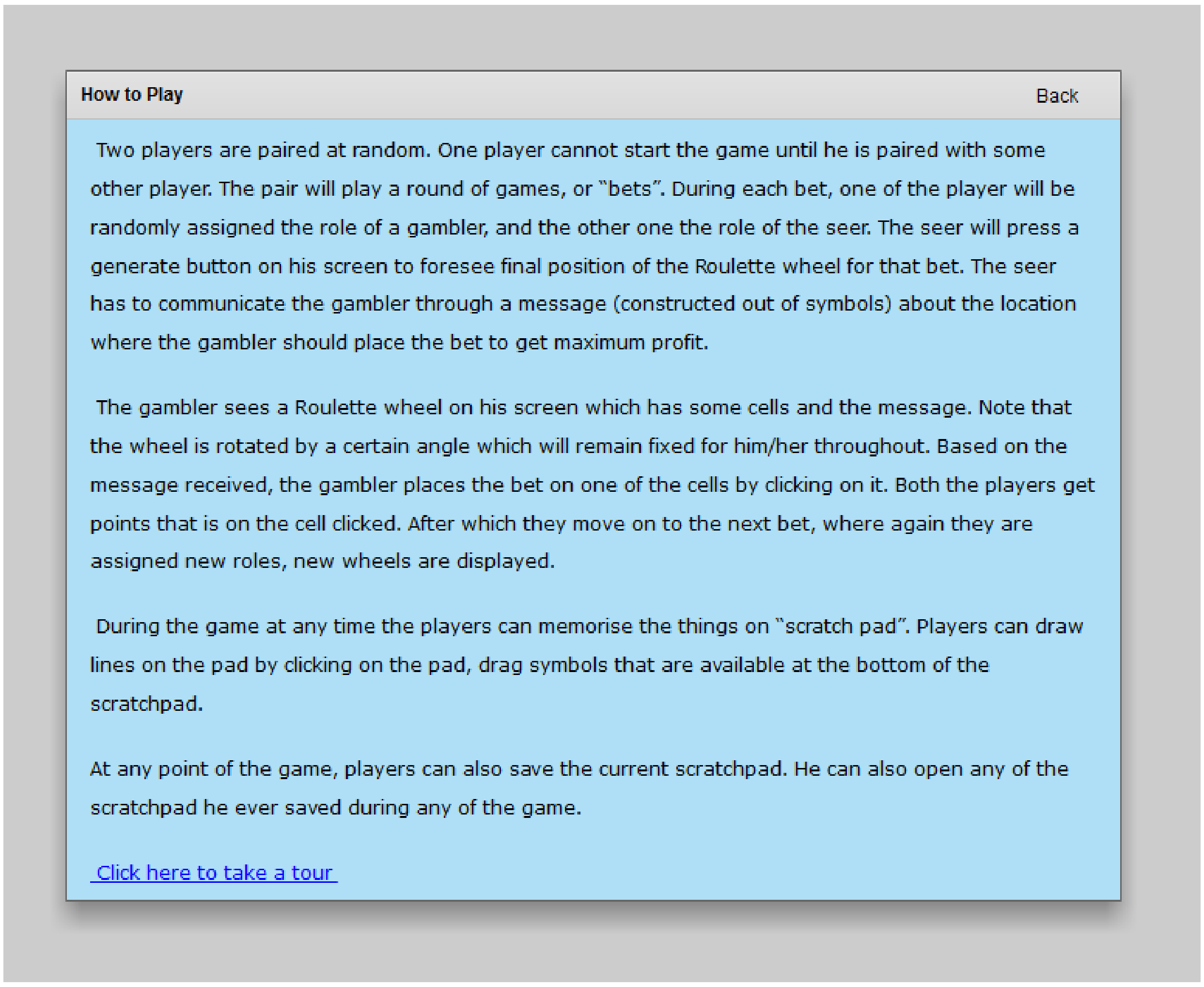
3.5 How to Play

- Once a wheel is generated for the seer, he will open the scratchpad to find out whether he has already assigned a symbol or group of symbols to the part of the wheel which contains the topic cone in any of the previous game.
- If he has already assigned any symbol or group of symbols then he can send the same message to the gambler.
- If he has not assigned any message to the part of the wheel which contains the topic cone in any of the previous game, then he will generate a new message and will assign the same message to that part of the gambling wheel and will send the same message to the gambler.
- Once a message is received by the gambler, he will open his scratchpad to find out whether he has already received the same message in any of the previous game.
- If yes, then he will find out which part of gambling wheel he has assigned the message for. He will click on the bar which is on the same part of the wheel.
- If no, then gambler will select any cone randomly. After he clicks on the cones, values on all the cones will be displayed. Now, he can open the scratchpad and

can assign the same message to the part of the gambling wheel containing the topic cell.

We have made the topic value much higher than the value of the other cells. This will indirectly force the seer to convey the position of the topic to the gambler. We have made the means of the distributions from which the topic score and non-topic scores are sampled very different. Topic value will vary from 91 to 100 and non-topic value will vary from 1 to 10.

At any time player can read the rules of the game by clicking on "How to Play". Rules window is shown in Figure 3.8.



Two players are paired at random. One player cannot start the game until he is paired with some other player. The pair will play a round of games, or "bets". During each bet, one of the player will be randomly assigned the role of a gambler, and the other one the role of the seer. The seer will press a generate button on his screen to foresee final position of the Roulette wheel for that bet. The seer has to communicate the gambler through a message (constructed out of symbols) about the location where the gambler should place the bet to get maximum profit.

The gambler sees a Roulette wheel on his screen which has some cells and the message. Note that the wheel is rotated by a certain angle which will remain fixed for him/her throughout. Based on the message received, the gambler places the bet on one of the cells by clicking on it. Both the players get points that is on the cell clicked. After which they move on to the next bet, where again they are assigned new roles, new wheels are displayed.

During the game at any time the players can memorise the things on "scratch pad". Players can draw lines on the pad by clicking on the pad, drag symbols that are available at the bottom of the scratchpad.

At any point of the game, players can also save the current scratchpad. He can also open any of the scratchpad he ever saved during any of the game.

Click here to take a tour

Fig 3.8: Rules of the game

We have also highlighted the topic in a different colour (cyan in the example) so that the seer doesn't miss it. We do not tell the players who they are playing against, not even reveal the other players previous scores, so that one doesn't gain anything by being strategist.

We have also given a screen-cast of two rounds of the game for better understanding of the game. To make the game more interesting, Game server will rank the players according to their score. Player can see his current rank by clicking on "Your rank".

Chapter 4

Experimental Results

This chapter shows all the results obtained by our experiment. We distributed our web based game CodeGambler to the web and several groups. Many people played the game and we collected a lot of data from them. Here in this chapter, we are plotting few graphs obtained by analysing the data of 35 players.

4.1 Players Statistics

Figure 4.1 shows the graph between number of games and fraction of players who played that number of games. Number of games are on X-axis and fraction of players are drawn on Y-axis. This graph shows that a player plays an average 20 games.

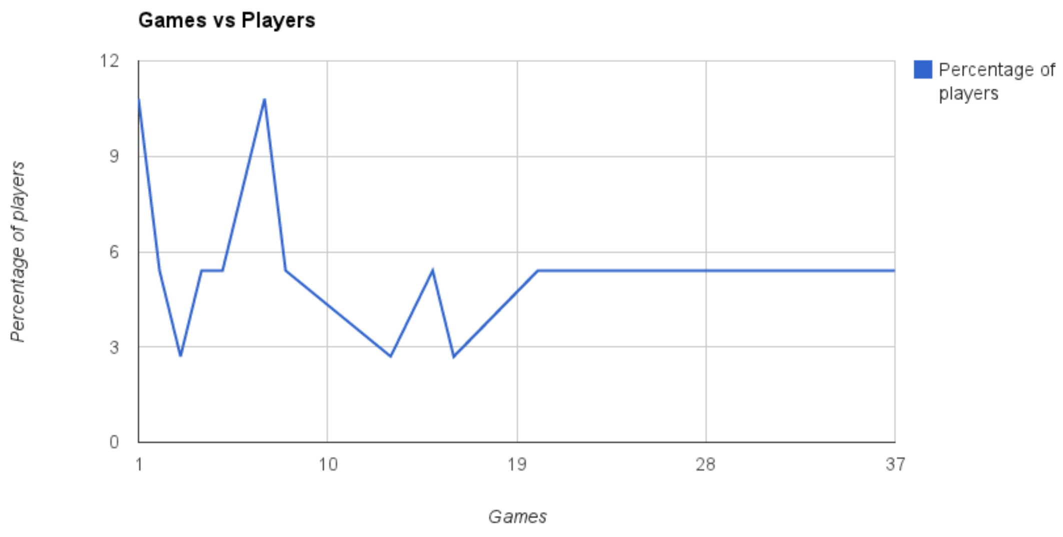


Fig 4.1: Player statistics graph

4.2 Messages Sent by Users

In our category game messages are formed by combining the symbols available in the keypad of the game. This keypad is shown in Figure 4.3. Figure 4.2 shows the symbol statistics. From the graph we can deduce the following points:

- Players were not much interested in using the symbols available at the last line of the the keypad.
- The very first symbol of the keypad was used the most number of times.

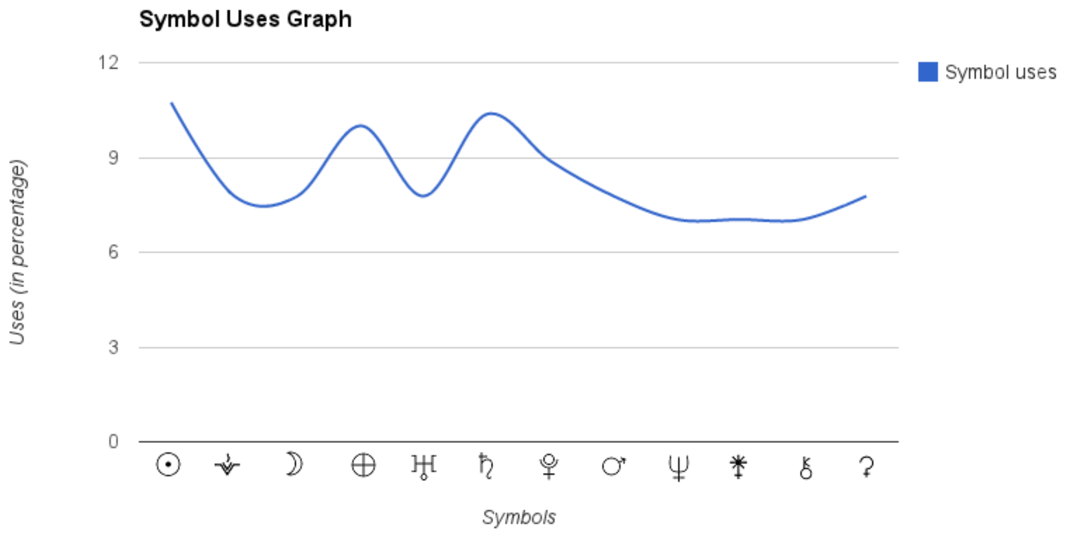


Fig 4.2: Symbols uses graph

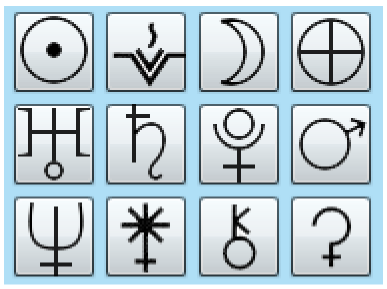


Fig 4.3: CodeGambler keypad

4.3 Success Rate

Figure 4.4 shows the success rate. X-axis shows the number of games per player in chunks of three. First value shows the fraction of success achieved in first three rounds by all the players. Second value on the graph shows the fraction of success in next three rounds of the game i.e. round 4,5 and 6 by all the players. In the graph X-value ranges upto 13 means that the graph is plotted for total 39 rounds. We can deduce the following points from the graph:

- Success rate is minimum around 34% for the first three rounds. Success rate then increases as the game progresses and reaches 100% for the last 6 rounds.
- Our Success Rate graph is very similar to the graph shown in Figure 2.2.

- Increasing success rate shows the cordination among the players.
- It takes average 30 games for the players to reach the consensus.

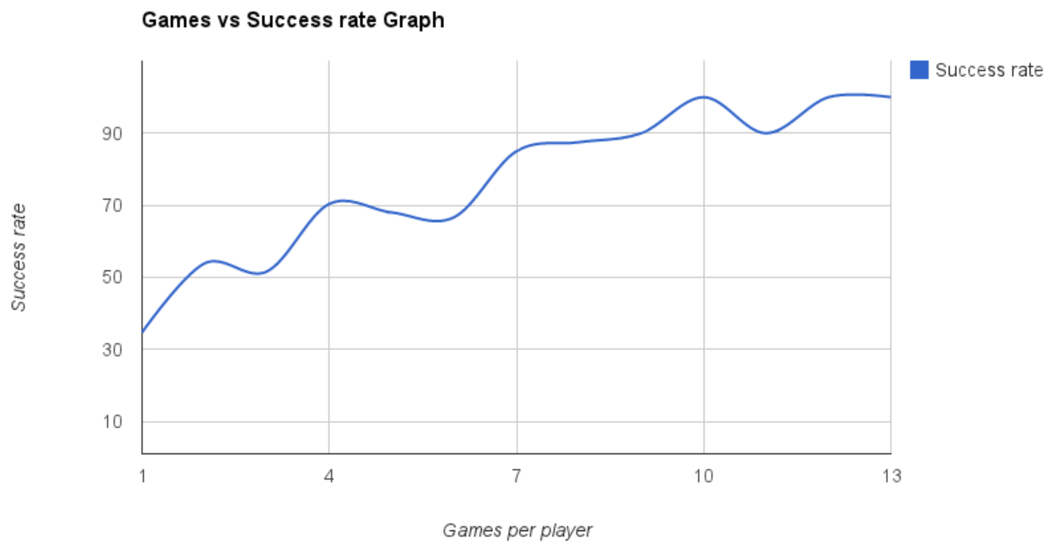


Fig 4.4: Success rate graph

4.4 Categories Statistics

Figure 2.3 shows the graph for number of linguistic categories and number of perceptual categories. This graph shows the following points:

- This graph shows that the number of linguistic categories remains to be finite.
- In the first few games, the number of linguistic categories and number of perceptual categories are very less. The reason for this is that the first step of each game is the discrimination stage, where the speaker followed by the hearer may refine his category inventory to distinguish the topic from the other objects.
- Initially when the success rate in very low(Figure 4.4), the no of linguistic categories increases. After that when success rate becomes high, the number of linguistic categories stabilises to a certain value.

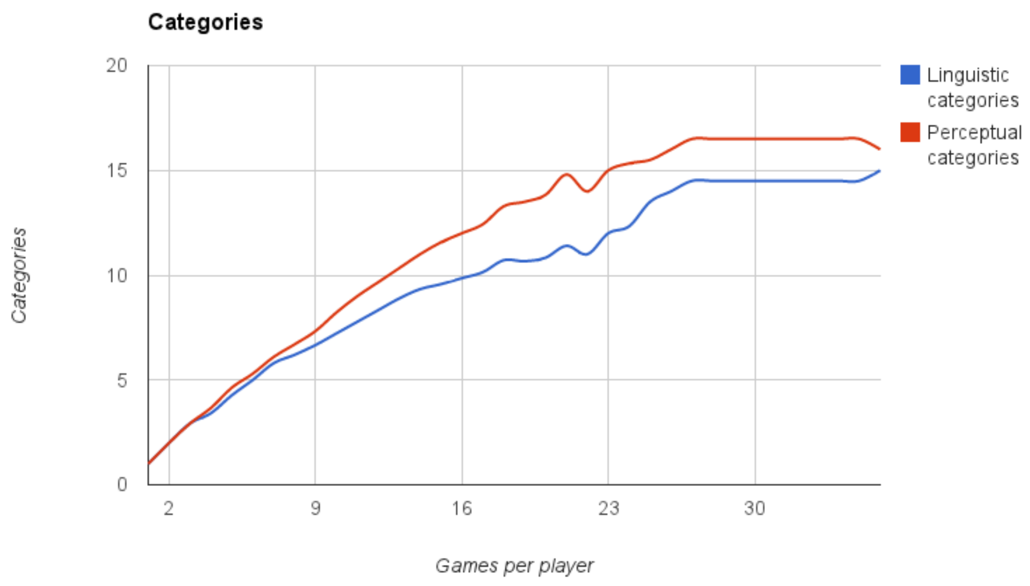


Fig 4.5: Categories graph

Chapter 5

Conclusion

With the help of our web game CodeGambler, we have shown that a simple negotiation scheme, based on memory and feedback, is sufficient to guarantee the emergence of a self-organized communication system that is able to discriminate objects in the world, requiring only a small set of words. Individuals alone have got the ability of forming perceptual categories, while cultural interaction among them is responsible for the emergence and alignment of linguistic categories. Our work shows that Even if the environment to be categorised is almost infinite and the resolution power is very high, the number of linguistic categories is very small. Our experimental results are very similar to those shown in [1].

Finally, we believe that these results could be important both from the point of view of language evolution theories. These results will possibly lead to a quantitative comparison with real data ([11, 12]) and suggesting new experiments (e.g., different populations sizes and ages), and from the point view of applications [e.g., emergence of new communication systems in biological, social, and technological contexts].

Bibliography

[1] Puglisi A., Baronchelli A. and Loreto V., “Cultural route to the emergence of linguistic categories”, Proceedings of The National Academy of Sciences (PNAS), vol. 105, no. 23, 7936-7940, 2008.

[2] Loreto V., Mukherjee A. and Tria F, “On the origin of the hierarchy of color names”, Proceedings of The National Academy of Sciences (PNAS), vol. 109, no. 23, 6819—6824, 2008.

[3] Baronchelli A., Gong T., Puglisi A. and Loreto V., “Modeling the emergence of universality in color naming patterns”, Proceedings of The National Academy of Sciences (PNAS), vol. 107, no. 6, 2403-2407, 2010.

[4] Berlin B., Kay P., “Basic Color Terms: Their Universality and Evolution”, Univ of California Press, Berkeley, 1969, reprinted (1991).

[5] Lindsey D. T., Brown A. M., “Universality of color names”, Proceedings of The National Academy of Sciences (PNAS), vol. 103, 16608-16613, 2006.

[6] Saunders BAC, van Brakel J., “Are there non-trivial constraints on colour categorization?”, Behav Brain Sci, vol. 20, 167-179, 1997.

[7] Steels L., “The origin of linguistic categories. The Evolution of Language”, Selected papers from the 2nd International Conference on the Evolution of Language, London, 1998.

[8] Taylor J. R., “Linguistic Categorization: Prototypes in Linguistic Theory”, Oxford Univ Press, Oxford, 1995.

[9] Long F., Yang Z., Purves D., “Spectral statistics in natural scenes predict hue, saturation, and brightness”, Proceedings of The National Academy of Sciences (PNAS), vol. 103, 6013–6018, 2006.

[10] Baronchelli A., Felici M., Caglioti E., Loreto V., Steels L., “Sharp transition towards shared vocabularies in multi-agent systems”, J Stat Mech, P06014, 2006.

[11] Kay P., Cook R.S., “The World Color Survey”, www.icsi.berkeley.edu/wcs, 2002.

[12] Selten R., Warglien M., “The emergence of simple languages in an experimental coordination game”, Proceedings of The National Academy of Sciences (PNAS), vol. 104, 7361–7366, 2007.

Appendix A

Adobe Flash and Java

A.1 Adobe Flash Player for Client Side

A.1.1 Adobe Flash Platform

The Adobe Flash Platform is what Adobe calls its entire family of technologies used to create, run, and provide data to rich applications (in the form of SWF files).

A.1.2 Flash Platform Runtimes

At the center of the Flash Platform are the client runtimes: Adobe Flash Player for the browser and Adobe AIR for outside the browser. The runtimes render applications created on the Flash Platform (in the form of SWF files).

A.1.3 Adobe Flash Player

The user experience when interacting with traditional web content created with HTML and JavaScript has changed dramatically over the years, progressing from simple links and multiple, discrete pages to single page applications with asynchronous data calls and interactive controls. The experience, however, is still limited by the objects that are available in the browser object model. This is where Flash Player comes in.

Adobe Flash Player is a browser plug-in that has a much richer object model and rendering engine that allows developers to include more highly expressive and interactive content in web applications. To include this richer content, you create you a SWF file (a compiled bytecode file which is what Flash Player can render) using some developer tool and then reference this SWF file in your HTML page. When the HTML page is parsed by the browser, the SWF file is downloaded and run by Flash Player in the browser window. The developer tool used to develop CodeGambler client is Adobe Flash Builder 4.6.

A.1.4 Adobe Flash Builder

Adobe Flash Builder is an Eclipse-based development tool targeted at developers. With this IDE, we use the Flex framework to create SWF files. Flash Builder accelerates Flex application development by providing intelligent code hinting and generation, refactoring, compile-time error checking, interactive step-through debugging, and visual design for laying out and styling user interfaces.

A.1.5 Flex Remote Procedure Calls

Through the use of Flash Remoting requests we can directly call the methods of server-side Java classes that return binary Action Message Format over HTTP.

A.1.6 Flash Remoting

Flash Remoting is a combination of client and server-side functionality that together provides a call-and-response model for accessing server-side objects from Flash Platform applications as if they were local objects. It provides transparent data transfer between ActionScript and server-side data types. Flash Remoting uses client-side functionality built in to Flash Player and server-side functionality that must be installed on the application server.

A.2 The Architecture of Flex and Java

The Java EE Platform is the leading enterprise web server. The Adobe Flash Platform is the leader in the rich Internet application space. So, we have used both to develop the client server to get the benefits of an enterprise back-end solution and a great user experience.

A.2.1 Client/Server Architecture

Flex and Java applications use a multi-tier architecture where the presentation tier is the Flex application, the business or application tier is the Java EE server and code, and the data tier is the database. We can write the back-end code just as we normally would for a Java application, modelling our objects, defining our database and writing the business logic to query and manipulate the database. The business tier must be exposed for access via HTTP from the Flex application and will be used to move the data between the presentation and data tiers.

Typical HTML applications consist of multiple pages and as a user navigates between them, the application data must be passed along so the application itself (the collection of pages and functionality it consists of) can maintain state. In contrast, Flex applications, by nature, are stateful. A Flex application is embedded in a single HTML page that the user does not leave and is rendered by Flash Player. The Flex application can dynamically change views and send and retrieve data asynchronously to the server in the background, updating but never leaving the single application interface (see Figure A.1).

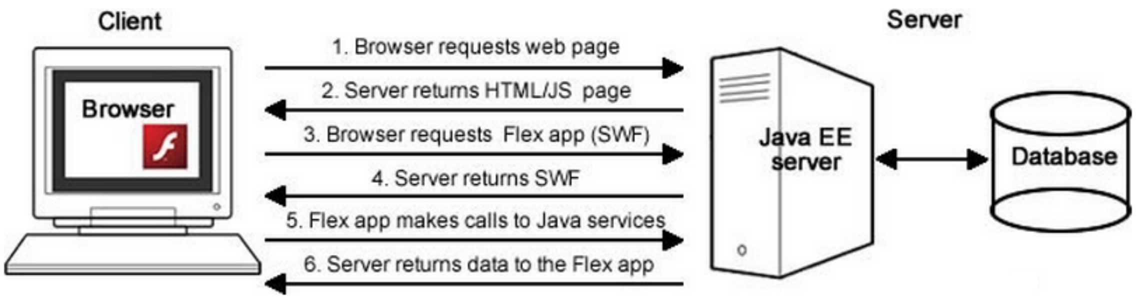

Client
Server
Browser
1. Browser requests web page
2. Server returns HTML/JS page
3. Browser requests Flex app (SWF)
4. Server returns SWF
5. Flex app makes calls to Java services
6. Server returns data to the Flex app
Java EE server
Database

Fig A.1: The client/server architecture

A.2.2 Client/Server Communication

Flex applications can communicate with back-end servers using either direct socket connections or more commonly, through HTTP. The Flex framework has three remote procedure call APIs that communicate with a server over HTTP: HTTPService, WebService, and RemoteObject. All three wrap Flash Player's HTTP connectivity, which in turn, uses the browser's HTTP library. Flex applications cannot connect directly to a remote database. We have used third option RemoteObject for making remote procedure calls in our development. It makes a Flash Remoting request to a method of a server-side Java class that returns binary Action Message Format over HTTP.