

## Multi-Agent Systems

### Project proposal: Strategies for playing symmetric Hangman

6 march 2011

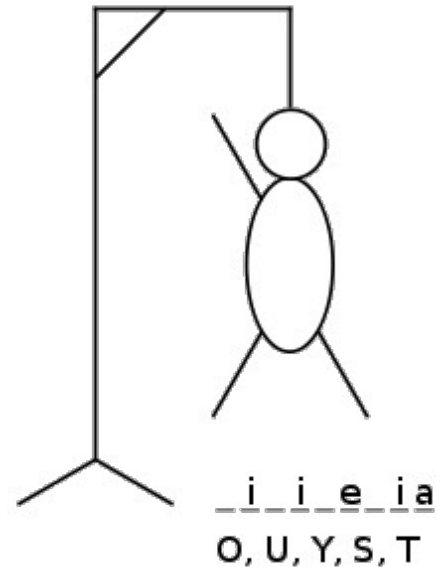
Jefta Saija, s1612727

Ruud Henken, s1634097

In the original game of Hangman one player (player 1) thinks of a word and the other player (player 2) is to guess the word. The word is represented by dashes. Player 2 is to guess the word by asking single letters. If the letter occurs in the word, player 2 gets to know the positions of the occurrences in the word. If the letter does not occur in the word, player 1 draws an element of the hangman diagram (see figure to the right). The game ends if:

- \* Player 1 completes the Hangman-diagram, player 1 wins
- \* Player 2 completes the word, player 2 wins

This is not a trivial game since the alphabet has more letters than the diagram contains elements. Therefore player 2 is to choose the letters carefully.



We would like to model a symmetric version of the game. In this adapted version both player 1 and player 2 think up a word. In each turn one of the players guesses a letter, but both players have to indicate the occurrences of the letter in their word. The game ends if:

- \* One of the players completes his word, that player wins
- \* The hangman-diagram of one player is completed, that player loses

Constraints: 1. both players need to choose a word of the same length, this ensures that the difficulty of guessing a word is for both players more equal.

2. both players must chose a different word

We would like to determine the best strategy for winning the symmetric version of Hangman. Would it, for example, be more effective to only ask for letters which do not occur in your own word? Or would it be most effective to guess only vowels first? And what if you find out that your opponent is using the same strategy?

When this game is implemented there will be a choice function from which can be determined whether it is the case of (1) a human player versus a CPU, (2) CPU versus CPU or (3) a human player versus another human player. Situations (1) and (2) are most interesting for this course and therefore we will only further discuss these two situations in this proposal.

Situation (1):

A human player playing the game of symmetric Hangman, will use its knowledge of words (in a particular predetermined language) as part of its strategy to guess letters to determine its word. This knowledge of words is also necessary for a CPU and therefore it should be necessary to implement this part of the knowledge as a vocabulary that contains words of the predetermined language that is accessible to the CPU.

At the start of the game with human player vs CPU, the human player will have to determine what word the CPU has to guess and vice versa. To ensure that both players know the word that one player assigns to the other player it is necessary to let both players each pick one word from the vocabulary. The human player must first determine what the length of the words will be, then he can choose one word from the presented list of words of that length. The CPU will choose one word at random from this list. This way the CPU always has the ability to determine the word, but if the vocabulary for example contains over 200.000 words then it is almost impossible for the human player to have the ability to determine the word. Therefore should the vocabulary not contain too difficult words and not be too large, but also not that small that the human player can determine the word by guessing the first letter.

It should also be the case that each player can see the other player's word progress. Seeing the word progress of the other player is knowledge that you can use in your strategy. If the CPU sees that the human player doesn't know its own word, but only has 1 gap in its word like "b\_at" and the CPU determines that the word must be "boat", then the CPU knows that he must not ask for the letter o, because then the human player will know its own word. So the CPU must also have knowledge of the word progress of the human player to determine which letter to ask for.

Also the hangman is a factor that determines which strategy to choose. Every time a player asks for a letter that is not in its own word a piece of the player's hangman will be drawn, once the whole player's hangman has been drawn the player loses. So the CPU must carefully consider which letters to ask for. At the beginning of the game, the CPU does not know any letters of both words, so he will have to guess a letter. In this case human players sometimes ask for a letter that appears frequently in words of that language, in Dutch for example the letter 'e' is used frequently. This strategy can also be used for the CPU, the CPU can check which letters are frequently used in the words of the vocabulary and can ask such a letter at the beginning of the game when no knowledge is yet available.

#### Situation (2):

This is the same as situation (1), but now one CPU must reason about the other CPU and both CPU's must choose one word (for the other CPU to determine) at random from the list of a length that will be determined at random by the game. But the tricky part is that we are now dealing with two CPU's that share the same word knowledge from the vocabulary and will have the same strategies to apply. This makes that two perfect logicians are playing against each other. This makes playing according to a strategy even more interesting.