# Project Summary

Our goal is to find an optimal solution to wordle using logical propositions. We will have a randomly sized dictionary of words to choose a word and have our logical proposition find words to try out and using the clues the logical proposition will search for better words to guess.

# Propositions

Wordle is a 5 letter word guessing game with 6 tries which equates to 30 slots to put letters in. Each of the letters can be put into 4 different states . (x, y) denotes the location on the board.

se(x,y) : If this is true it means the slot is empty, if this is false it means its incorrect, partially incorrect or correct, it looks like this on wordle when true 

si(x,y) : If this is true it means the slot is incorrect, if this is false it means it is empty, partially incorrect or correct, it looks like this on wordle when true 

sp(x,y) : If this is true it means the slot is partially correct, if this is false it means it is empty, incorrect or correct, it looks like this on wordle when true 

sc(x,y) : If this is true it means the slot is correct, if this is false it means it is empty, partially incorrect or incorrect, it looks like this on wordle when true

These 4 propositions we will call the slot series, for each of the 4 slot series propositions, only one can be true at a time for the wordle board.

Lx : This will proposition represents all the letters in the alphabet.

Cx : This will denote the correct word and x will be the letter in the word

# Constraints

Good Guess

Word with all different letters – First guess and subsequent guesses

Exclude words with letters that have si as true

Exclude words with letters that have sp as true in that position and set words that have those letters aside into candidate list

Exclude words with letters that have sc as true and set words that have those letters in the position aside into candidate list

Bad guesses

No repeated words

When guessing words with partially correct letters

When not many characters are correct or partially correct don’t guess words with those letters in those conditions and guess the letters that haven’t been guess.

# Model Exploration

Luck based cases

When there are at least 5 sp and sc together as true it will switch strategies and start to attempt to guess the word

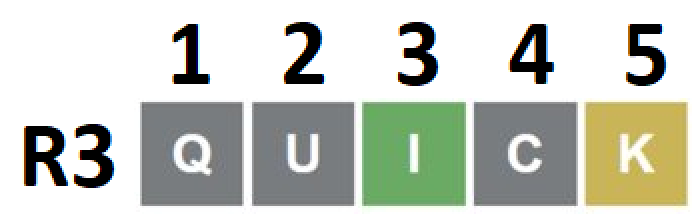
Last guess

The last guess will always try to guess the word while previous guesses will try to gather information

The running out case

Since we are using a randomly generated list we may run out of valid words to guess that have all different letters and such depending on the size of the list. So we may not be able to guess a word with 5 distinct letters not including the letters that were already guessed. This means we would first introduce words with the partially correct letters in different placements along with the rest of the letters we haven't tried out.

# Jape Proof Ideas

*List the ideas you have to build sequents & proofs that relate to your project.*

If there is a correct letter in slot 3, we can deduce that no other letters can go into slot 3

Sc(3,3)→C3→Li⊢(¬La∧¬Lb,...¬Lh∧Li∧¬Lj...¬Ly∧¬Lz)

If there is a partially correct letter in slot 5, we can deduce that the letter must go into one of the other 4 slots.

Sp(3,5)→Lk⊢C1∨C2∨C3∨C4∨¬C5

If there is an incorrect letter in slot 1, we can deduce that the letter is not in the word.

Si(3,1)→Lk⊢¬∃x(Lk→Cx)

# Requested Feedback

Do our constraints fit the project requirements to higher degree of success? If not, what else can be done to further improve them?

More jape proof ideas?

Is the project too difficult?

Should we start with a prefilled board?

Missing propositions from wordle?

Constraints and unexpected possible scenarios that could crop up?

# First-Order Extension

*Describe how you might extend your model to a predicate logic setting, including how both the propositions and constraints would be updated.* ***There is no need to implement this extension!***

**Updated Propositions:**

**E(s):** Means that the slot (s) within the guess is empty

**I(s):** Means that the slot (s) within the guess is incorrect (wrong letter)

**P(s):** Means that the slot (s) within the guess is partially correct (wrong slot, right letter)

**C(s):** Means that the slot (s) within the guess is correct (correct letter in correct slot)

**U(s):** Means that the slot (s) is a unique letter compared to the other slots inside the guess

**W(g):** Means that the guess (g) is the correct word from the word bank

**L(g):** Means that the guess (g) is a word inside the candidate list of possible words

**G(g):** Means that the guess (g) is a “good guess”

**B(g):** Means that the guess (g) is a “bad guess”

**R(g):** Means that the guess (g) is a guess that has already been made

**Updated Constraints:**

– There exists a correct word from the word bank (AKA a possible answer)

– There exists a guess where all slots are correct slots, implying that the guess is the correct one from the word bank

– If a guess is repeated, there are no correct slots or there are slots that are incorrect/empty/partially correct.

U(s) – For all slots, each slot is unique in comparison to the other slots inside the guess and the guess is not considered a “bad guess”. This implies it is a “good guess”.