**Unit Technologies Identification**

Team: The Good, The Bad, and The Ugly

**Representation**

Board Representation:

The chessboard is presented as a 2D array that contains characters representing different pieces and their locations. For example, a black pawn is “p” and a while pawn is ‘P’. Each step creates a new board configuration.

Move Generator:

This function takes the current board configuration and output a vector of the pointers of all possible configurations, that will be passed to different algorithms described below.

GUI:

We will start with a 2D graphic representation of the chessboard and pieces, where the user can control the movement of each piece by mouse click. If time permits, ideally we can expand the GUI to 3D.

**Responsible Team Members**: Prasanth Palli, Shannon Govekar and Xiaorui Li

**Data Structure** – Trees

We plan to use a tree structure to specify different possibilities and levels of prediction. The root is the current board configuration, and nodes are different possible configurations according to depth/height.

## Algorithms:

1) **Negamax-Search:**

This is the strategy which we want our chess-engine to use while playing against a user. It uses a mini-max criterion to choose the action which *minimizes* the reward of the opponent’s *best* action. It evaluates all the possible successor configurations which is usually time-intensive.

**Responsible Team Members**: Xiaobin Ran, Shannon Govekar

2) **Alpha-Beta Search:**

Alpha-Beta search tries to reduce the search space by deleting branches which could not result in the best-move. This results in a speed-up and thereby allows the chess-engine to look deeper into the tree.

**Responsible Team Members**: Prasanth Palli, Matthew Tyler Prelich, Xiaorui Li

3) **Principal Variational Search with Move Ordering**:
Principal Variational Search further tries to speed up the search by sequencing the moves to explore. For example, piece captures are given preference over other moves.
**Responsible Team Members**:Shannon Govekar, Xiaobin Ran, Xiaorui Li

4) **Iterative Deepening:**
We will be implementing iterative deepening search to search the tree level-by-level. By doing this, the chess-engine will always have the best possible action till that depth level.
**Responsible Team Members**: Prasanth Palli, Matthew Tyler Prelich

**Stretch Goals:**
*NULL Move Pruning* is simple heuristic technique used to enhance the speed of alpha-beta pruning algorithm. It is designed to guess cutoffs with less effort than otherwise required.
**Responsible Team Members**: TBD

*Futility pruning* discards moves that do not contribute to the alpha value in Alpha Beta Search algorithm by adding a cost of called futility margin to the current position.
**Responsible Team Members**: TBD

**External Library**
fssimplewindow
3D Chess: https://code.google.com/p/divs/