Challenging the Assumptions of Scrabble Static Evaluation

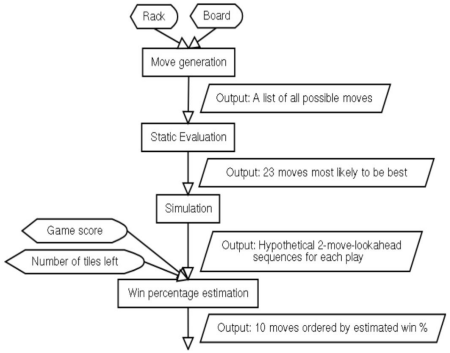
Ethan Mathieu, Computer Science Department, Tim Barron, Yale University

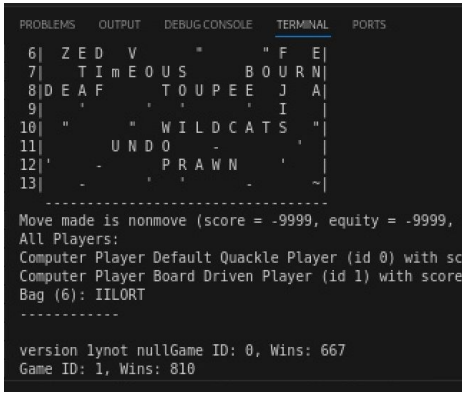
Computational Scrabble Background

Scrabble is a crossword game where players try to maximize their point total by playing high value words.

Scrabble strategy is unique because of
> Imperfect Information (obscured opposing rack)
> Premium Squares (certain
> Random tile draws dictate next available moves.

Quackle is the leading open-source engine for Scrabble. For my project, I challenged its parameter assumptions and designed custom heuristics to bolster its performance.





Experiment Structure

My custom CLI allowed me to run Quackle games in parallel. I edited the codebase to allow Player objects to compete against each other with arbitrary changes and record win statistics.

Quackle uses Truncated Monte Carlo Tree Search to select which *k* possible moves are worthy of simulation. This selection is based on the **equity** of the move.

For each selected move, future outcomes are simulated *p* turns into the future, *i* times, to account for the random rack draws.

I experimented with which quantities of *k*, *p* and *i* are most effective. I also tried different equity heuristics, specifically editing the leave component (*l*) that captures the value of the tiles kept on the rack.

Statistic	Value
Modified Wins	459
Unmodified Wins	535
Number of Games	992
Modified Δ	-98
P-value	0.00130

Table 4.8: l_{modified} vs. $l_{\text{unmodified}}$

$$\text{equity} = \text{intrinsic value} - l$$

$$\gamma = \sum_{j_1 \in \text{Leave}} \sum_{j_2 \in \text{Leave}} p_{j_1} \cdot p_{j_2} \cdot s(j_1, j_2)$$

$$l_{\text{modified}} = s + \text{bt} + \text{vc} - \gamma$$

Statistic	$i = 2r$	$i = \frac{4}{3}r$	$i = \frac{3}{4}r$	$i = \frac{1}{2}r$
P1 (Modified) Wins	532	551	475	448
P2 (Unmodified) Wins	466	446	521	546
Number of Games	992	992	992	992
Modified Δ	+66	+106	-46	-98
P-value	0.03	4.67×10^{-6}	0.07	2.13×10^{-5}

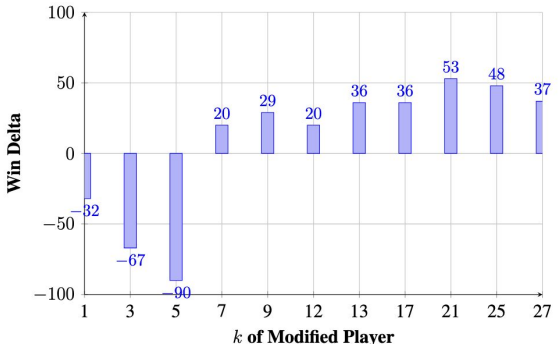
Table 4.6: Summary of Modified Player Statistics for Different i Values

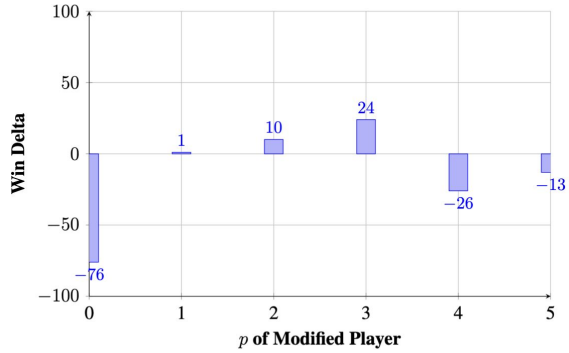
Conclusion

There are areas where the Quackle engine can be improved upon that my research uncovered.

The equity heuristics that the authors crafted are fragile and attempts to edit them to take other game states into consideration greatly skews results.

Further research could be done in using historical gameplay data to craft a more balanced leave value that takes into account game state.





Experiment Results

> Compared to what Quackle currently uses, it could lower its *k*, *p* without seeing a drop off in performance. This would allow it to increase its *i* without dedicating more time and see an increase in performance.

> Increasing these values past what is currently used does not lead to better performance.

> Changing the equity heuristic, specifically the **leave value** to take into account future draws does not lead to better performance.