CIS 559 PROJECT 2: PARALLEL FOOTBALL

IAN SIBNER, MICHAEL MOLISANI, AND LAURA KINGSLEY

20 October 2015

CONTENTS

LIST OF FIGURES

1 INTRODUCTION

Parallel Soccer is a game in which four teams of P players (where P can vary from 1 to 250) compete to kick soccer balls into their teams' goal. The goals are arranged in each corner of a 32×32 grid, and the rest of the squares on the board are initialized to contain one ball each, for a total of $1,020$ balls on the board to start off with. In the first step of the game, each team can place each of their P players anywhere on the board; multiple players can potentially occupy the same square. After that, each player (with full knowledge of the board) must choose to either *kick* a ball in their square up to K distance away (with K being a constant whose value may vary between one and 45), or *move* up to 1 square in any direction (including diagonals). When a ball is kicked into a team's goal, that team is awarded a point and the ball is removed from play. The game ends when all balls are removed from play, and the team with the most points wins.

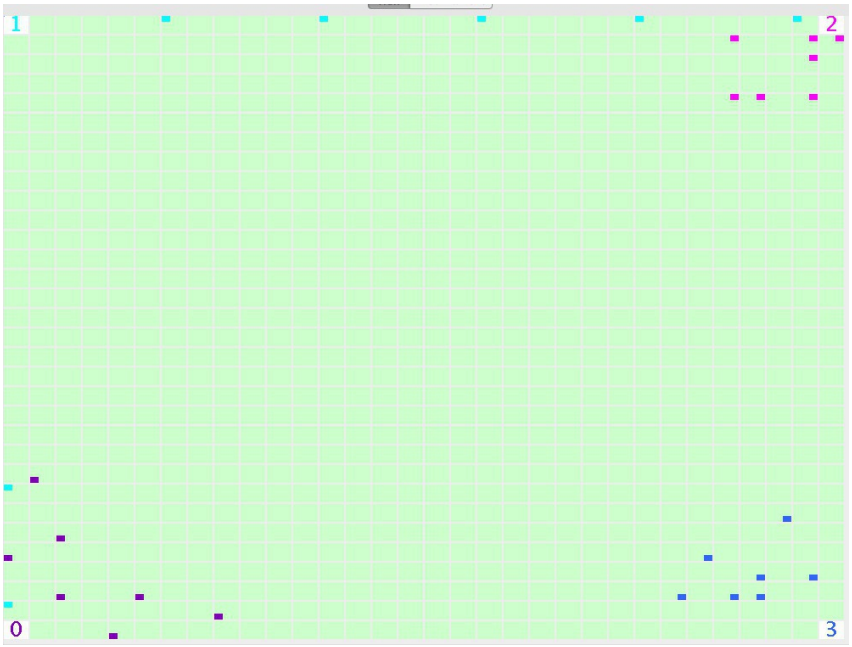

Figure 1: An initial positioning of the board where $P = 7$

2 INITIAL INSIGHTS AND OBSERVATIONS

Immediately, our team observed several key features about the game.

1. The information available to organisms would be very limited. They can only 'see' in four directions, and could only detect food (or lack of food) and other organisms' states. So strategies that required non-local knowledge or multi-organism coordination would be difficult to implement.
2. Staying put is much cheaper than moving since $v \geqslant 1$. Also, for the majority of the setups we considered, we had $v \geqslant 20$.
3. For all setups, we had $q \geqslant p$. This means that food tended to 'collect' in certain squares, rather than being widely distributed on empty squares.
4. There is a great deal of inherent randomness, especially in the beginning
5. Strategies may work well for certain combinations of P and K and not others. A smart player would change its strategy depending on what value these constants took on.
6. Certain strategies may work particularly well in the early phase of the game, when all the balls are on the board and there are not clusters, while others may work better in the middle phase (where there are fewer balls and more clusters) or in the end game (when there are only a few balls which every team is competing for).
7. Efficient strategies (i.e. those which required to fewest moves to kick a ball into the goal) would generally favor kicking over moving, since $K \geqslant 1$. This naturally leads

to a sort of "bucket brigade" where players kick balls close to one another in order to "pass" them towards the goal.

These insights guided our strategy throughout and allowed us to focus on the most important aspects of the game.

3 STRATEGIES AND CONCEPTS

3.1 Movement & Benefit Function

In order for a player to decide which cell was the best move for any given board state, we created a benefit function that assigned a score to each cell for every given player. Since we had access to the entire board, we used many factors in this function, including the number of balls in a cell (and surrounding cells), distance from the cell to our goal, distance from the cell to the player, and the number of opponents around the cell. Given the loose efficiency constraints given the problem, we decided that a globally optimal benefit function would be both feasible and desirable. The primary gain from this decision was a team optimized for late gameplay, where our players were able to seek out and steal balls toward the end of the game. The reason was that once we and other teams cleared the balls near our respective goals, balls became scarce. If any team had accumulated balls, even if they were far from our players, they would seek them out and attempt to score with them.

On the other hand, as teams improved, the game time grew shorter and quicker. In such a game, it is possible that a globally optimal zone will change into a depleted zone before a far-away team can reach it. It is likely that our model did not fully capture the constantly changing state of the game board. We addressed this by heavily weighting our own goal in benefit calculations.

3.2 Benefit Function

The benefit function was inspired by what we thought a human player would consider while deciding where best to move. We realized that given a point on the board, factors considered while deciding whether or not to move towards that point included not only the Euclidean distance to that point, but also the number of opponents, concentration of balls in the general vicinity of that point, and distance of that point from our home goal. Accounting for the proximity of opponents was particularly important becuase it gave us a measure of how susceptible to change the benefit of a particular cell was: the nearer the opponents, and the higher the number of opponents, the more likely it was for balls in the cell in question to be kicked out. This allowed us to take into consideration not just how the board was laid out now, but also gave us a proxy for how each cell might change in the near future.

The following is the benefit function for each cell that was calculated from the perspective of each player on the board so as to allow for the player to move to the cell that offered him the greatest benefit:

$$B_{i,j} = G_{i,j} - D_{i,j}$$

where $G_{i,j}$ is the gain offered by that cell and its neighboring 4×4 cluster of cells to the player and $D_{i,j}$ is the distance penalty for that cell given the position of the player and the cell.

$$G_{i,j} = \sum_{a:|a-i|\leqslant 2} \sum_{b:|b-j|\leqslant 2} \alpha \times \mathrm{numBalls}(\mathrm{cell}_{i,j}) - \beta \times \mathrm{numOpponents}(\mathrm{cell}_{i,j})$$

and

$$D_{i,j} = \gamma \times \mathrm{cellDistanceFromPlayer}(\mathrm{cell}_{i,j}) + \delta \times \mathrm{numKicksFromCellToOurHomeGoal}(\mathrm{cell}_{i,j})$$

where $\alpha = 1.2$, $\beta = 0.4$, $\gamma = 1$ and $\delta = 5$ are weights. These weights were assigned based on the optimal values we found through emperical observations of our performance against other teams over a range of weights.

3.3 Cell clustering

On the second iteration of our benefit function, we accounted for ball and opponent clustering by keeping track of the number of balls and opponents surrounding a particular cell. This allowed our players to seek out areas of high benefit, rather than single cells.

After initial trials with a large radius, we settled on a radius of 4. We observed that larger radii caused the players to move back and forth somewhat inefficiently, and hypothesize that these larger zones changed too quickly for players to capitalize on them.

3.4 Kicking

Initially, our players would kick the maximal distance towards the goal. Although this will guarantee that the ball will be as close to the goal as possible, it does not necessarily mean that the number of steps to kick that ball in the goal is minimized when there are teamates around. With this insight, we improved the kicking algorithm with a simple calculation. A player will look at all possible cells it can kick to and minimize the number of steps to kick that ball into the goal for anyone on its team. What this translates to in practice is that a player will now pass the ball as close to a teammate as possible if the teammate is closer to the goal than itself. We observed that after implementing this kicking algorithm, the players naturally form a bucket brigade to minimize the number of steps a player needs to go to for the ball to score.

3.5 Placement

We experimented with several placement strategies. These ranged from clustering near the home goal, near the center, and near opponent goals. We also attempted two hard-

coded early-game strategies called *pincer* and *fanned*. Although we eventually opted for a simple clustering around the home goal, we outline some other attempts below.

The *pincer* strategy placed players in lines along the sides adjacent to the home goal, and then swept balls toward the main diagonal, before harvesting them. This strategy aimed to harvest as many easy balls as possible in the very early game, while preventing teams who started in our home goal from "stealing" our balls. After initial testing, it turned out that this strategy quickly fell apart as soon as any opponents interfered with the hard-coded pattern, so we did not retain it in the final implementation.

The *fanned* strategy placed players on the edges opposite the home goal corner in a 20×20 grid, and collecting them at "sink points" in increasingly small radii toward the goal. This strategy aimed to fetch 300 of the 400 balls in this grid, assuming thaht 300 was a high enough number to win most matches. The results of this strategy were not great, as we had trouble balancing the tradeoff between systematically sweeping all balls in a zone, and finishing the early-stage sweep quickly enough to switch to a mid-stage strategy. In the end, the configuration was not competitive with our simpler initial placements.

The centered and opponent-goal strategies were somewhat inconclusive in our testing, but generally underperformed the home-goal clustering that we ended up choosing.

The most common opponent strategy was some sort of initial sweep from center-field toward their respective goals. After trying several variants of such strategies, we realized that our players were particularly effective when started from our own goal. Opponents generally swept inwards, carrying some but not all balls from somewhere in the center toward their goal, and then spreading back out to collect what they missed. Starting at the goal, we had a head start at fanning out from the offset and hoped to take advantage of center cells that were left from other team's initial sweeps relatively uncontested.

4 IMPLEMENTATION

4.1 Composability

Because we realized that certain strategies might work very well in the early phase while others might excel in the late phase, we set about building a player that would *compose* other players by switching between its component players' strategies. We met with a lot of success the second week by composing Group 1's `GridPlayer` for the first 200 turns and our own mid-game-optimized `DerickPlayer` for the remainder of the game. This worked about as expected; `GridPlayer` was far more efficient than `DerickPlayer` in the early game, but after about 200 turns, the board had grown sparser and our own player was much better at seeking out balls and scoring them. We believed this was the way forward and set about finding a more intelligent way to determine when to switch (factoring in P, K, and the number of balls left on the board).

However, during the third week of gameplay, the rest of the teams (and our own `DerickBrigadePlayer`) had become so sophisticated that the early stage of the game completely dominated. Essentially, the winner was determined by how well the early phase played out, and the

middle/end phases were nearly nonexistent. Thus we decided to play a pure strategy in the final round, focused on early-game optimization, rather than continuing to compose multiple strategies.

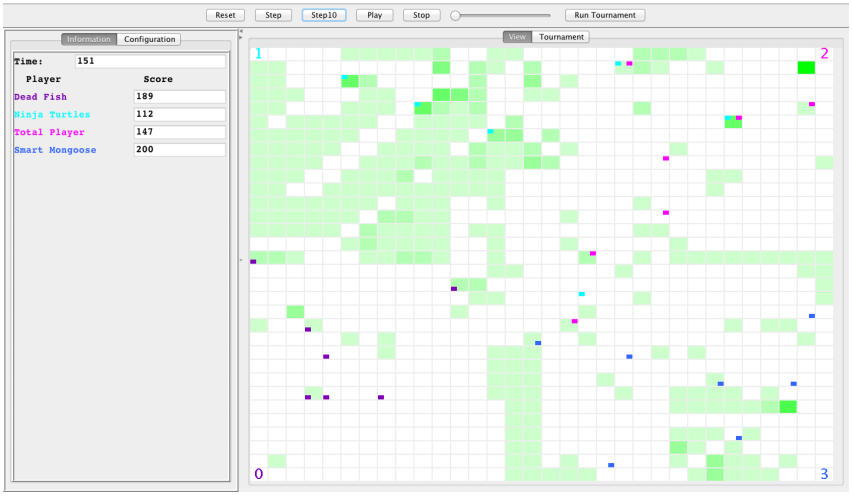

Figure 2: The state of the board after 150 turns with $P = 7$, $K = 6$. Note how over 60% of the balls are already scored; this illustrates the great importance of the early game when sophisticated strategies are placed in competition.

4.2 Placement

In the early phases, we enumerated the different placement strategies and switched according to values of P and K. Our final implementation was fairly simple, and consisted of a random placement of players near the homegoal, with a bounded parameter on k determining their range.

By starting in a tight cluster bounded by the square from our goal out $\text{MIN}(K, 10)$ spaces, our players would fan out across the board, while the kicking algorithm led to a trail of k-spaced players who formed a bucket brigade. The relatively small square space for player placement was chosen as it balanced the advantages of fanning outward from the goal with scoring an initial k-distanced ball for each player.

4.3 Locking Mechanism

As we found through our trials, using a benefit function to determine where a player should go runs into problems when players want to go to the same cell. The players will eventually converge and go to the same cells the rest of the game. To prevent this major inefficency, we implemented a locking mechanism where only one player can go after a single cell at a time. However, we found that although this works great through most of the game, when the number of balls is very small, it becomes better for multiple players to go after the same ball to have a higher likelyhood of kicking the ball to the goal. As

such, when the number of balls reaches the number of players on the field, we turn off the locking mechanism.

After only locking the cell that the player is going towards, we also lock a radius around that cell. This allows us to create zones such that only one player clearing a zone.Through experimentation, we found that choosing a radius of max(0,3-P/10) works well. With higher number of players, the smaller the zones should become because there is not enough space to create so many zones. Addditionally, it is more likely that opponents will steal balls from a particular zone before a single player can clear that zone out.

4.4 Auxiliary Board

The benefit function we used to guide the movement of each player required us to be able to quickly accesss the not only the positions of our own teammates and the balls, but also the positions of our opponents. Moreover, each player needed to know the next postion of every other teammate to ensure smooth kicking and minimal wastage of moves. We encapsulated all this information in an auxiliary board that we refreshed at every time-step. This additional board was a 32×32 matrix of cells. Each of these cells was an instance of a custom class (Cell) that we created to hold information on the number of opponents, number of teammates now, number of teammates in the next turn and number of balls in the corresponding position on the real board. We found that maintaining this auxiliary board with information specific to our team greatly simplified the code we wrote in other parts of our Player and gave every teammate quick access to decision-critical information.

5 RESULTS

Generally speaking, we had mixed results. Our rank ranged from 1^{st} to 7^{th} place. Our most common standing was 3^{rd} place. We observed that scores followed relative rank, so we chose rank as the proxy for overall performance in the tournaments.

The overall trend in our performance was that we did better than average given low-to-moderate k-values (less than 15), and any p-values. This makes sense in terms of our initial placement strategy. With very high values of k (at least 30), our players simply started too close to our goal, and did not take advantage of the potential for long-range kicking. Teams starting in the middle were able to clear balls in a single kick, so that even when our players reached the middle sections quickly, there were few balls left. Contrarily, our players remained competitive with smaller k. We think that starting players near the goal led to more effective bucket brigading as described in the initial placement section above. Larger values of p did not have a major impact on performance because more players allowed for a more effective covering of the map while maintaining a bucket brigade path back to the goal. It is possible that parameterizing our intitial placement by p as well as k could have improved our scores, but it may have negatively impacted the bucket brigading, and led to similar results.

Two notable configurations are $P, K = 45, 1$ and $P, K = 30, 1$. We placed third and first in these categories, respectively. This was a surprise to us, as we had not optimized for single-player configurations at all. We attribute our success here to the globally-optimal strategy we adopted, so that in slower games with long ranges, our player was able to seek out the most beneficial balls and ball clusters across the entire map. This is in contrast with games with higher p, where the board layout changed so rapidly that a global-optimized strategy did not provide much benefit.

Our results are summarized in the following figures.

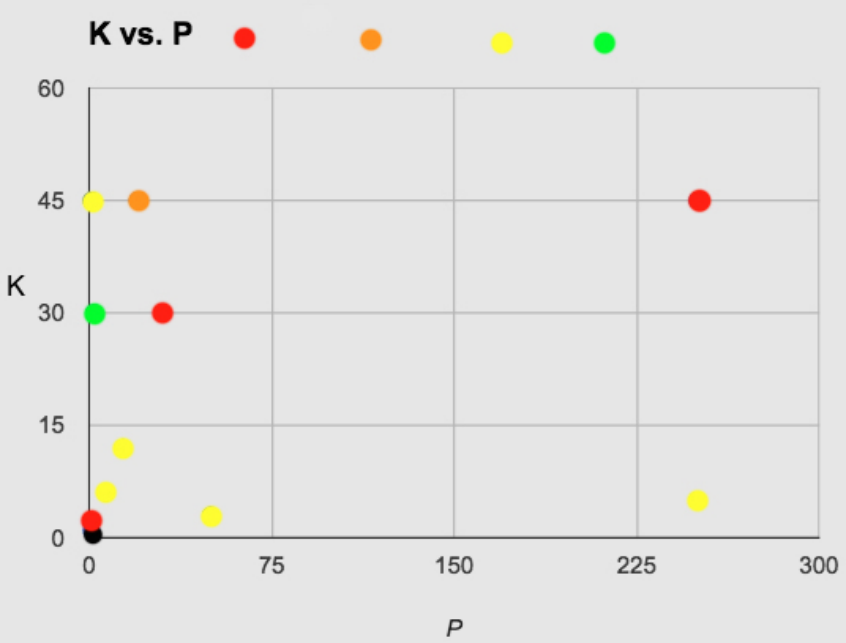

Figure 3: P on the horizontal vs K on the vertical axis. Observe the yellow 3^{rd} place band for low k

P	1	1	1	2	7	14	20	30	50	250	250
K	1	2	45	20	6	12	45	30	3	5	45
Rank	Expt	6	3	1	3	3	5	6	3	3	7

Figure 4: Table of rankings per configuration

6 CONTRIBUTIONS

Just as we adopted many good ideas by other teams (see Acknowledgements section), we contributed two major ideas to the group discussion over the course of the project:

1. **Benefit function -** Modelling how a human player would have made his/her movement decision gave us some key insights, especially with the realization that the runtime efficiency of our decision algorithm was not a constraint since this was only a 32×32 board. We realized that each player on the team could survey the entire board at every time-step and make a player-specific decision of where best to move, taking into account information about ball positions, opponents' positions as well as teammates' current and future positions. In the week after we brought up these insights in class, we saw that many other groups also adopted similar hotspot strategies (i.e. strategies that surveyed the board and picked out the best places to move towards). We were pleased to see that this was an insight that persisted through all further discussions in class and was validated, in different forms, through the strategies of various other teams.
2. **Composable Players and Game Phases -** Although we realized early on that the game would likely be characterized by distinct phases in which certain strategies would dominate, it was not until week 2 when other groups really started to pay attention to this idea. This was largely because our performance that week was extremely dominant after we composed `GridPlayer` in the early game and `DerickPlayer` for the rest of the round! Other groups began to think about how they could tailor their strategy to perform well in all phases of the game, or at leasthold their own. One group even took this idea to the extreme, creating a `ReinforcementLearningPlayer` that essentially composed *all* players using a 1NN algorithm!

7 FUTURE DIRECTIONS AND LIMITATIONS

7.1 Strengthen Early Game Through Sweeping

Among teams that placed highly in the tournament, we noticed that their players would tend to start the game by sweeping balls inwards towards their own goal, maximizing their early-game wins before beginning to spread out across the board. Our players, which started close to our own goal, did not exhibit this behavior. Adapting our placement strategy so that they started farther away might be sufficient to achieve this result; however, we could also look into a composability solution in which a sweeping strategy is used for the very first portion of the game before switching to our current players' behavior. This would require significantly tweaking our composability behavior, particularly the point at which the behavior switch occurred, in order to ensure that our players did not continue sweeping for too long before switching to mid-game behavior.

7.2 Generalized Step Distance

This idea stems from our attempts to take advantage of global-scope "hotspots," or high-density ball clusters that are not near the goal. The existing implementation weights balls close to the home goal particularly high, which is good in the early game, but may not be ideal overall.

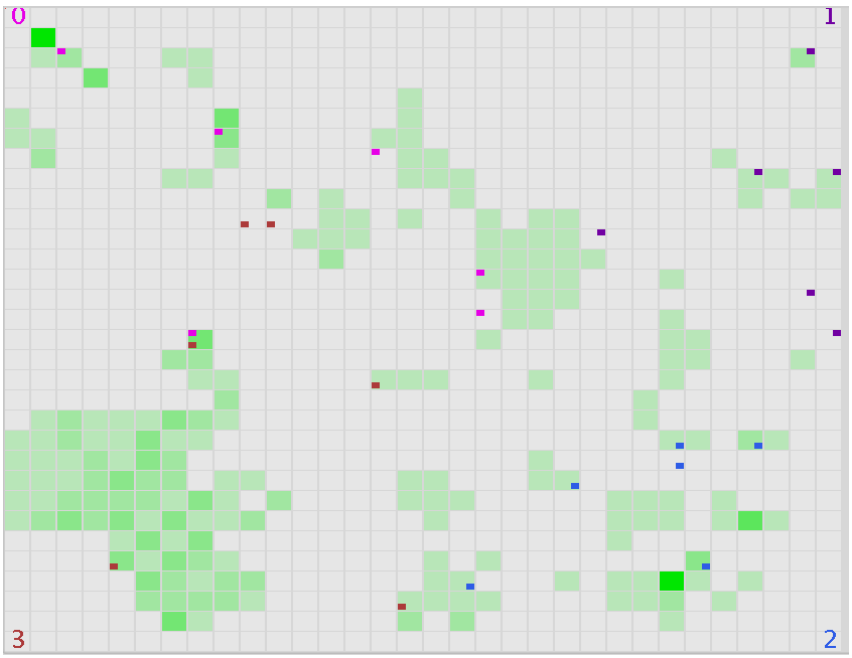

Figure 5: Example of non-local hotspot after 145 turns with $P = 7$, $K = 6$. Note how balls are clustered in top bottom-left corner

As we can see in the figure above, certain strategies tend to cluster balls near an opponent goal, where they stay relatively untouched for several cycles. A smarter global-cluster detection would drop the current strategy

The biggest disadvantage to this approach is that the time it takes to travel to such a cluster may be greater than the the time the cluster exists. Even if the seeking team is able to reach the cluster in time, the balls lost in the transit time may not be worth it.

It would be possible to determine, by factoring in the number of balls left B_{total}, the number of balls in close range B_{nearby}, and the number of balls expected to remain in the cluster $B_{cluster}$. Such a strategy would decide to go for the cluster if and only if:

$$B_{total} - B_{nearby} < B_{cluster}$$

One way of implementing this strategy would be to generalize the `numberOfStepsToGoal()` benefit function to be `numberOfStepsToPosition(p)`, where the latter takes in any valid position `p` as it's argument, causing the cells around it to get a score boost. Then, upon deciding to pursue a cluster, this position would be updated to the cluster center, until a new position was found.

7.3 Pathing

It becomes clear through watching our players that a pathing algorithm could improve our players drastically. We found that our players will often take the same path to their desired locations even though their destinations are distinct. This creates inefficiencies as if there are balls on that path, only one player is needed to clear those balls. In the future, we would look into minimizing the number of cells such that two players would through to reach their destination.

Another problem with our players with pathing is that they arbitrarily choose a minimum path to its destination. So, if two paths are equally minimal, the player will not choose the path with the most balls, thus missing opportunties to score more points. In the same vein, if two paths are slightly different in the number of steps but the longer path has way more balls, the player will choose the shorter path, causing inefficiencies. To fix this problem, The path the player should take should additionally take into consideration how many balls the player can kick on that path.

8 ACKNOWLEDGMENTS

Our progress was largely a result of the class discussions and adopted strategies. Through the course of the project, we adopted starting strategies from Groups 1 (Grid Player), as well as experiments in stealing opponent balls with the opening placements. We would like to give special thanks to Groups 2 and 6 for the important heuristics we adopted from them and eventually used in our final implementation.

Group 2 (The Swarm) was the first group to successfully implement a kind of benefit function with the emergent behavior of a bucket brigade. This function was based on the number of steps from the ball to the goal, and we adopted it as a core feature of our ensemble benefit function.

Group 6 (Bucket Brigade) contributed a smarter kicking function that favored passing to players over kicking as far to the goal as possible. We adopted this idea as a core part of our improved kicking strategy.

9 CONCLUSION

Overall we were happy with the progression of this project and the contributions that our group was able to make to the class, including the benefit function and the composable players. We were also quite pleased with our performance during the second week of the project when our player came out on top!

We hope to take the lessons learned on this project and apply them to later projects in the class. In particular, the idea of optimizing for the early stage of the game (which turned out to be key in parallel football) may prove extremely useful later on as well. Especially

in zero-sum games like this one, it can be difficult to stage a come from behind victory when one team starts out with a convincing lead.