Project 3: Getting to Know You

Group 1
Andrey Kutser (avk2001@ columbia.edu)
Chris Murphy (cdm6@ columbia.edu)
Becky Plummer (rp2176@ columbia.edu)

November 15, 2004

Introduction

Let's Get Together is a game where the object is to allow one organism to obtain every piece of information. This game is similar to the Organisms II game in structure. However, the goal is achieved through cooperation, not competition. All the organisms wander around on the board only able to view friends in neighboring cells or the trail of information that has passed through an adjacent square. An organism may move in any direction, one square every time step. In order to exchange information, organisms must be on the same square. Each organism knows the size of the board, the number of players, and the information it possesses. Therefore, when an organism sees a friend it can check to see if it already has obtained that friend's piece of the puzzle. However, if a friend has obtained other friends' information cells, the organism may only tell by the trail that appears on the square. In other words, it doesn't know, from a neighboring square, exactly what other information a friend contains. Therefore the organism must make an educated guess whether or not to jump into the friend's square. The goal of the game is cooperation to communicate all the information to one player as fast as possible.

This document starts off by explaining our initial thoughts about this problem, based on our discussions as a group and in class. It then lists some of the approaches and strategies that we considered and prototyped, before discussing the implementation that we ultimately settled on. Finally, this report summarizes our results in the various user evaluation tests that were performed, and finishes with some closing thoughts and ideas on how we could have improved.

Summary of our approach

Initially, some of the ideas were similar to the strategies from the Organisms II game, such as farming, herding, and random movement. However a major difference between Organisms II and Let's Get Together is the absence of reproduction and health. Players do not age or reproduce in Let's Get Together; thus, certain strategies, such as farming, will not work because each team only has one player contributing to the community. Each team must develop a player that performs well in an environment containing many of its own players and in a game with other groups' players. Therefore a major challenge was determining how to locate and communicate with other players of friendly species.

Our first player was designed for a multiplayer world. The player moved in diagonal patterns in order to locate other players quickly. Many groups in class had discussed using horizontal and vertical moving players. We thought that a diagonal moving player would encounter these other species quickly and traverse the board faster. The player was designed to move in one of the four diagonal directions and after wrapping around the board without encountering another player, it would pick a different random direction to try. This player was robust because it did not think very much about its moves. It did have some intelligent movement, because it would calculate how long it had been since a contact and change direction and it had some intelligence in how to move onto another player. This player was quite robust in multiplayer games.

In our group discussions we decided to try something more interesting. We started thinking about board coverage and laying down trails. We implemented a strategy that would traverse the board so that it made the minimum number of moves possible while still viewing every square on the board. This is possible because an organism can see the eight squares around it without moving onto the squares. Therefore, we could move so that we would only go onto three out of every eight squares. However, this strategy was very dependant on the board dimensions and whether the dimensions were relatively prime numbers. This strategy made us start thinking bigger and we came up with something even more interesting.

In discussion of board coverage, we started to wonder if by marking trails if we could tell where an organism had started its trail. We discovered a simple strategy where each organism would create a ring in the minimum of the two dimensions. Then it would traverse once in the perpendicular direction. This allows each organism to pick up a coordinate of all the other players relative to its home square. Then it would traverse the minimum direction again to pick up the other coordinate. This strategy depends on knowing how many other players are in the game and the dimensions of the board. After gaining this information, it was our intention to calculate a meeting square. This strategy is very robust and converges very quickly; however it only works in a single player environment.

Our final step was to make a player that could determine as quickly as possible whether the game is single or multiplayer and switch strategies accordingly. We implemented a detection system that would check if we had information before we had expected to and, if information had been encountered ahead of schedule, we would switch to multiplayer. Our multiplayer strategy is the diagonal mover and collector. We have found this player to be robust in all situations.

In some cases, the player does not perform as well as expected. These cases occur when more than one area is the same size. In these cases, all of the organisms do not always meet up, but they are able to realize that they have not met. They attempt to resolve the conflict, but if this resolving does not occur, they move into the multiplayer strategy. With more time we would have liked to refine the flexible player to work in all cases and not have to resort to the diagonal movement in a single player world.

Background and Initial Findings

Based on our initial discussions as a group and in class, we felt the following were the most important ideas to focus on in this project:

Horizontal and vertical motion

Initially, this strategy was discussed as an optimal solution to the game; however, the game was then changed to be more interesting. As a group we thought this idea was still a very good idea even if it was no longer the optimal solution. This strategy was adapted to be a way of collecting information without knowing exactly which players are moving in which direction. Many class members volunteered the idea of having some players move horizontally and others move vertically, while one player would pause on the trail of the other when new information was detected. We decided to use the idea of directional motion as multiplayer strategy where our player would move diagonally to collect the data, because it would traverse the trails of the other players more swiftly.

Communication standards

This game is about cooperation among different species in order to succeed. Therefore, a major concern was the actual communication of information amongst organisms. We immediately determined in discussion that if each organism tries to jump onto a friend, they would just continually swap places. Thus communication protocols were discussed. One idea was to give each position a number such that each player would calculate the number of the friend and of itself from the friend's perspective. Then the organism with the higher position would move onto the one with the lower position. Another idea was splitting the eight neighboring squares into two L shapes and determining who should move based on general position. The possibility of deadlock was an issue in communication protocols and what to do when another species does not follow the protocol. Communication protocols were not enforced on the class: each team had to choose a strategy on its own and try to make it as compatible with other species as possible. This complication made the presence of deadlocks more obvious. Therefore we adopted a time out in our adaptation of the higher number communication protocol.

Chasing and following trails

Another interesting piece of this project is the case when a player sees a trail of an organism with information that it does not yet have. Many people talked about what to do when the player picked up a trail of information. One idea was to chase the organism that is leaving the trail by following the path of the organism. This also occurs when a player encounters an organism as a neighbor and it has essential information, but does not follow any conventions and moves away. In this instance a major question is "to chase or not to chase". The answer to this question was up to each group to determine individually. We decided not to implement chasing in our organism. We determined that we would possibly see the same organism again or another one which had obtained the information. The same decision holds for following trails.

Board coverage

Some discussion was sparked by the idea of leaving many trails on the board in coverage. Not only allowing other species to pick up a trail, but also to study the board and try to make an intelligent decision based on the flow of data. This idea was rebuffed by some, because of the non-decaying trails and therefore massive board clutter that could confuse certain organisms. We started to implement a board coverage strategy, but dropped the idea because of the peculiarities of traversing the board in a diagonal motion.

Flowers and bees

Like in the Organisms II project, some class members determined that a good strategy is to have some organisms sit around and not move much, or leave a trail, and others move around a lot to find them. This strategy was named Flowers and Bees after the pollinating motion that bees take. This strategy is nice, because it allows for some organisms to be easily found and communicate a lot of information quickly. Some groups discussed enhancing this strategy so that the “flowers” moved a little bit to leave a trail so that they could be more easily found especially on a large board. In our player, we decided to allow it to remain a bee-like creature in a multiplayer environment. We did implement many of these ideas in our multiplayer stages, such as bees and communication protocols, but decided to do something a bit more interesting with our single player strategy.

Strategies and Concepts

This section describes some of the concepts and ideas that we experimented with and discussed past the initial stages. Though some of them were ultimately thrown out, we feel that a future project may want to explore some of these further.

Moving diagonally

After some initial experimentation and analysis of our first player, we realized that moving in a diagonal fashion has several advantages over simply horizontal or vertical movement. One of the advantages is that the number of squares discovered at each step is maximized. For horizontal or vertical movement, three new squares are discovered, while diagonal movement allows the player to see five. Since the game relies on sharing all information in the minimum number of rounds, maximizing information found per turn is important. The other advantage of this movement strategy is the fact that the likelihood of finding another player is also increased. In cases when the board dimensions are relatively prime, players moving in one of the diagonal directions will get to see every square of the board in a shorter time than a similar vertical or horizontal strategy.

The strategy does not fare so well when the board dimensions are multiples of one another, resulting in the player looping in the same diagonal without actually covering new portions of the board. However, this case is easy to detect and account for by moving off the diagonal once it has been fully traversed and resuming the diagonal motion. We implemented these ideas in the multiplayer portion of our code.

History tracking

To complement the diagonal movement strategy, a history of the player's moves is helpful in keeping track of the information units that the player has seen at a given round. The units seen at any given point in time are stored in a history structure, which can then be used to identify the squares that players with information required by our player have visited. Having done a discovery of the board, our player can make intelligent decisions about which sector of the board players are likely to be in, thus minimizing the search time of the player.

In addition, we considered a decaying history in which the trail information of the surrounding nine square block visible to the player at each turn was initially set to a constant value proportional to the larger of the board dimensions, and then decremented at each round thereafter. The result is a trail that allows the player to prioritize squares that it has not visited yet and also to prevent the revisiting of squares which it has recently seen, since they are less likely to have been visited since we last saw them. The idea behind this is that ideally we would want to revisit any squares that have been marked in our absence and not bias against locations we have seen before. Another observation we made was that as the number of rounds increases, the board becomes filled with old path information and makes analysis of the trail information difficult, if not impossible without an auxiliary history.

Complete board coverage

From the diagonal movement, we realized that discovering every square of the board also meant discovering the locations where all the other players relative to our own starting position. Since the real coordinates on the display board are not known, we assumed for simplicity's sake that our player starts at the Cartesian coordinate (0, 0) every time, and all the resulting locations of information are simply offsets from the player's starting point. Having gathered that knowledge after traversing the entire board, each can player then determine, in terms of offsets from one's own location the distance required to move to any other player's last location seen. Initially we had considered simply traversing all the squares of the board, but that resulted in our players taking `L * W` rounds to discover the entire board. We looked to our diagonal motion analysis for quicker board coverage, and found that not every square of the board had to actually be visited. Since the player can see one square to either side of itself, only half of the board needed to be traversed. Thus, diagonal motion, in conjunction moving 2 squares parallel to previously traveled diagonals results in complete board coverage in the minimum number of steps required.

Congregation

We found the diagonal movement strategy for board coverage cumbersome to implement. However it led us on to a more promising idea for single player mode, which did not actually require seeing the entire board to locate our players. We realized that even though we cannot identify "player zero" and try to converge with it, as per Bogdan's suggestion in class, we can determine each player's actual location at the end of a fixed number of rounds, determined by the formula below:

```
2 * MIN(L, W) + MAX(L, W)
```

where `L` and `W` are the board parameters. This does not require visiting every square on the board, thus becoming more and more effective as the board dimensions get larger, but it does need all of the players to be ours to work. Our idea was to have each player then move to the same location that all players calculate heuristically, thus guaranteeing convergence and also complete information sharing in a deterministic fashion. The players would determine who is above and who is below them by looking at their coordinate and if it was greater than half the board, setting them as being above the given player, and below otherwise. A similar computation was done for the horizontal coordinate, resulting in a grid centered on the player, describing the locations of all the other players relative to it.

Initially we had attempted to implement this by computing the midpoint between all of the players and then moving to that location. This worked very well for two players, but broke down in cases with a larger number of players. In theory, having each player move to that point would have resulted in all of them ending up in the same location on the actual game board. In reality, due to the relative nature of the calculations, a single point was not enough to pinpoint the ultimate gathering location, and often resulted in players choosing the wrong direction to move in if the player was in the middle of either the length or width of the board, congregating at different points on the board.

After some discussion in class, we modified this strategy by instead choosing the smallest area formed by two of the players on the board and then moving into this area. This worked very well since there was less room for error in terms of choosing the wrong direction, but players still needed to look for one another. Since we wanted a deterministic solution to this problem on average, we enhanced the strategy slightly by congregating in the middle of the minimum area. This had excellent results and this is the idea we'd implemented for single player.

Conflict resolution with other players

Though congregating at the center of the minimum area is very effective, there were still conflicts that needed to be resolved. The exact coordinate will be rounded due to integer division if it is odd, thus a pair of players might end up right next to one another if they traveled in opposite directions, but not on the exact same spot as we would have liked. To resolve this case, we simply use our standard deadlock resolution based on the relative positions of the players to decide which of the two should stay, and which should move to share information. We use the integer values of the movement directions to make the movement decision. In the example below, Player 1 would stay and Player 2 will move. That is, the player who sees a smaller direction value than its mate will stay, and the other will move.

5	3	6
1	0	2
7	4	8

Player 1’s view of Player 2

5	3	6
1	0	2
7	4	8

Player 2’s view of Player 1

Any additional conflicts between two players are resolved by moving back and forth on the horizontal, vertical or diagonal between the two players depending on whether there exist horizontal, vertical, or both conflicts, respectively, due to the wrapping of the board. The two players are given the same distance to travel, but their direction is set randomly, so that, assuming an unbiased distribution of random numbers, the convergence of the two players will happen in most cases. Since the conflict case does not happen often, we felt that some randomization in the uncommon case is acceptable, as it won’t affect the average case.

Implementation

This section explains our algorithms for implementing the major features of the program that we submitted. The first part describes how our player acts in "single player mode", i.e. when ours is the only type of player on the board. The second describes our implementation in "multiplayer mode", i.e. when there are other types of players on the board. Both modes are encapsulated in the same player, but the player is only in one mode at a time.

Single Player Mode

The player always starts in single player mode, and stays in this mode until it realizes that there are other types of players on the board (see below). The strategy for single player mode is based on the "congregation" idea described above, and we believe it sets us apart from every other group, since we are the only ones who have attempted to implement this strategy (well, as of this writing, that is!).

Each player begins by moving in one direction (either east or south, depending on which dimension is shorter, and east if they are equal), laying down its "trail". When it returns to its original point, it moves in a perpendicular direction, detecting the trail of other players and learning one of their x-y coordinates. Upon returning to its starting point, it again moves around the board (in the original direction from the first sweep), picking up the other coordinate. By the time it returns to the starting point, it has made three round trips (twice in the shorter direction, once in the longer), and it will be able to know the coordinates of every other player on the board.

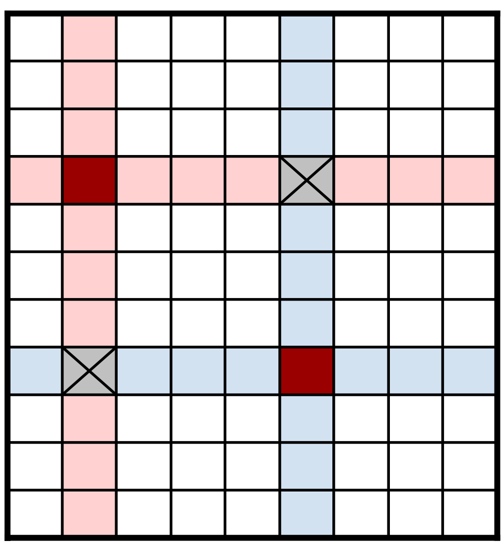

Figure 1. The two players (shown as red boxes) after having laid down their trails. The intersection points (marked by X) can be used to calculate the x- or y-coordinate of the other player.

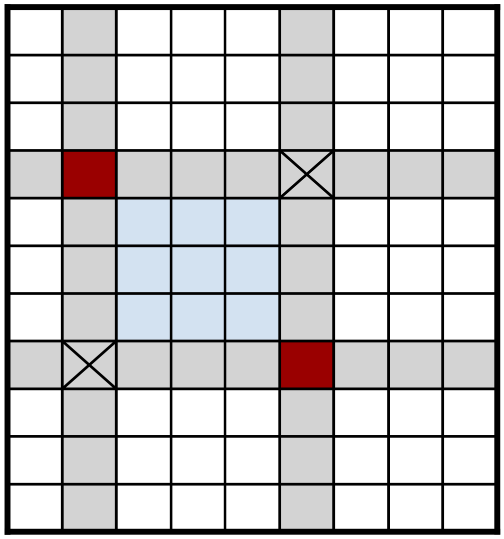

Figure 2. Now that the players know where the other players are, they can find the region with the smallest area, here shown in blue. The players will now attempt to meet at the center point (or approximate center) of the region.

At any given point, though, the player may realize that it is in a multiplayer environment. This would occur if any of the following conditions were met:

1. After its second sweep of the board, the player is missing both the x- and y-coordinates of another player. This would mean that the other player is not moving in the same way ours is.
2. After completing all three sweeps of the board, the player is missing either the x- or y-coordinate of another player. This would also mean that the other player is not moving the same way ours does.
3. At any point before completing three sweeps of the board, the player has a piece of information other than its own. This would mean that it had encountered another player and shared information, but this could not happen in single-player mode because all players would consistently be moving in parallel.

We believe that there are probably other ways to detect the multiplayer environment sooner (for instance, by observing the motion of or the trails left by players in adjacent squares while making the sweeps), but found that players typically detected such an environment for reasons #1 or #3 before completing all three sweeps.

If the player does, in fact, make it to the end of its third sweep, and has calculated the location of all other players, it would then calculate the "meeting point". It does so by evaluating all of the regions that are formed by the intersections of different players' trails, and looking for the one with the smallest area. It then calculates the approximate center point of the region, and determines the shortest path to get to that point.

Assuming that all players choose the same meeting point, the players will eventually be at the same space (or in an adjacent space), and all of the information would be shared. In most situations, the number of rounds in which the players will converge is

```
2.5 x MIN(L, W) + 1.5 x MAX(L, W)
```

since, after making its initial three sweeps, the player would, at worst, need to move half the board in each direction in order to reach the meeting point.

One problem that we struggled with was the “multiple center points” problem. Since the world wraps around, on either axis two players will always have two midpoints between them. We observed that, when one player was quite far away from others (a problem that was magnified on large, sparse boards), the player was likely to miscalculate the midpoint at which to meet. We noticed that it would choose a midpoint which was at a distance of one-half the dimension of the board away from where it was supposed to meet. We fixed this problem by having the player “know” that if the other players were far away, then most likely the meeting point was far away, and that would be the one to meet at.

The last problem that could arise is if multiple regions on the board have the same area, in which the players would need to choose between the different regions, and might not converge. This is especially a problem on a densely populated board, in which it is possible that there will be many small regions, and some will have the same area. We have not solved this particular problem in this version of the player, but a future implementation would need to consider this.

What happens if the player chooses the wrong meeting point, you ask? We implemented a “timeout” in the case that a player gets to the meeting point and realizes that no one else is there and that the game hasn’t ended. To be conservative, we set the timeout to have a value of

```
3 x MIN(L, W) + 2 x MAX(L, W)
```

under the worst-case scenario that, after making its initial three sweeps, a player moves half the distance of the board in each direction, encounters no other players at the “wrong” meeting point, and moves half the distance in each direction *again*, in order to meet the others. After this number of rounds, it is clear that the players are not converging, and so each player will switch to multiplayer mode.

Multiplayer Mode

Our strategy for multiplayer mode is based on moving diagonally, as described above. This was an idea we had from the very beginning of the project, as we realized that moving diagonally would give us more efficient board coverage and would conceivably avoid problems with other players who adopted the more conventional horizontal/vertical approach. The other strategy we use is to constantly be moving. We recognize that, at times, it is beneficial for the player to remain still, but since the multiplayer game is generally dictated by random factors (like initial placement and other players’ strategies), we felt that moving around was a better way to play to that.

Our player in multiplayer mode begins by choosing a random diagonal direction. The player then continues to move in that direction until another player is encountered. If our player does not have the information possessed by the adjacent player, it either moves or waits to be moved upon based on the “conflict resolution” strategy described above.

If our player is waiting for the adjacent player to move and it does not, our player will attempt to move onto it after two rounds. Noting that most other groups chose to break this deadlock after one round, we think this would be a good way to avoid “swapping” if we wait a little longer. On the other hand, if our player is expecting to be the one that moves, and the other player moves away from it, it will “chase” the other player for four steps, before ultimately giving up.

In the case that our player encounters another player but already has its information, the player will look at the information that has passed through that square. If there is information that the player needs, it will try to move onto the player, or wait for it to move, according to the “conflict resolution” strategy. If the adjacent square has no information that is interesting to our player, it will continue to move in its diagonal direction.

If the player has moved a number of squares equal to the minimum of the length or the width and has not yet encountered another player, it will change directions randomly to one of the other diagonals. This prevents it from looping in the same diagonal direction infinitely.

This strategy of moving diagonally works well with other players because we are very unlikely to encounter a deadlock. In the first place, whereas most other teams’ players only move vertically and/or horizontally, ours is moving diagonally, so that our player will not end up “chasing” another one (either by following it or its trail). Additionally, by putting a limit on how long the player will wait for another player to move onto it, and by giving up on a player that is running away from it, our player avoids situations in which it is wasting time pursuing just one player.

Tournament Results

We do not feel that the multiplayer game demonstrates our player strategy, because we focused our player on convergence in single player mode.

In each section below, the "Expected Score" is based on the formula for convergence described above: *2.5 x min(L, W) + 1.5 x max(L, W)*. This would be the ideal situation for convergence, in which players make their three sweeps and then have to move (at most) half the distance of the board in each direction to meet. The "Success" rate is the percentage of simulations in which convergence occurred in fewer moves than the expected score. We expect this to be the measure of how "well" we did in the tournament.

The "Average Score" is the average number of rounds to achieve convergence in 100 simulations. The "Ranking" compares our results to the other twelve groups.

Two players

Board size	Expected Score	Success	Average score	Ranking
L15 W22	70.7	94%	59.61	2
L35 W35	140	100%	111.12	**1**
L76 W50	239	95%	258.0	**1**

Three players

Board size	Expected Score	Success	Average score	Ranking
L15 W22	70.7	83%	72.66	6
L35 W35	140	90%	180.55	6
L76 W50	239	96%	213.36	**1**

Our player did extremely well in the two-player game, with the best ranking based on average score in two of the simulations. In addition, the success rate was very close to perfect in each case. We would actually expect it to be perfect, because when there are only two players, we can accurately calculate the meeting point of the two players and their convergence time should be predictable.

On the very large board with two players, there were two cases in which the score was very high, indicating that the players switched to multiplayer mode after waiting for each other; our belief is that this occurred because the players started off in locations that were equidistant in both the horizontal *and* vertical axes.

When there were three players on the board, our player fared slightly less well than with two players. The average score increased possibly because the players started off half a board distance away from each other and, as described previously, they might miscalculate the meeting point, but visit the other one after reaching the first. This would cause a slight increase over the expected score.

Five players

Board size	Expected Score	Success	Average score	Ranking
L15 W22	70.7	72%	73.89	11
L35 W35	140	79%	150.36	4
L76 W50	239	69%	354.44	4

Nine players

Board size	Expected Score	Success	Average score	Ranking
L15 W22	70.7	52%	176.92	12
L35 W35	140	61%	262.78	10
L76 W50	239	53%	489.73	7

When the number of players became very large, our performance dropped significantly. This was expected because we knew that there could be problems when there were multiple regions on the board with the same area. When there are nine players, that possibility increases, and the players will converge in different spots. After waiting an extra number of turns (which further increases the score because the players are stagnant), they will switch to multiplayer mode, and start wandering the board somewhat randomly (see above). In those cases, it is difficult to predict the number of moves in which the players will converge.

Overall Tournament Analysis

Our overall success rate was 93% in the games with two or three players; though obviously we think it should be 100%, we recognize that there are some configurations in which the players can get confused and try to congregate at the wrong spot. Further development of this player would need to focus on the small variants in the board (especially when players start half a distance away from each other) that could lead to these problems. However, considering that we had the best average results for three of the six scenarios that we evaluated, we think that our player was quite successful.

Whereas the conventional wisdom in the class was that the game would be easier with more players, we found that our performance in fact worsened in those situations. A big factor is the possibility of multiple regions having the same area. With five players, there are 25 regions formed; with nine, there are 81. The probability that two regions would have the same area cannot be discounted, but unfortunately our player does not have a suitable way of solving that problem. A future implementation would need to consider this problem as a high priority.

Conclusion and Lessons Learned

Areas for improvement

Though we are extremely proud of what we accomplished in this project, we realize that there are some aspects of our player that could be enhanced.

- The player needs to be more robust in calculating the meeting point if there are more than one region with the same area. Some ideas that were suggested included finding one with the smallest gap around it, or choosing to meet at the second-smallest area or largest area to break the tie (as long as there is some unique area).
- We recognize that it would be possible to detect multiplayer mode sooner from the single-player mode. By observing the motion of adjacent players, we could probably determine if their paths were parallel to ours; if not, then it is a multiplayer game.
- Our multiplayer strategy is quite simple and does not explicitly take advantage of (a) the trails left by other players or (b) the fact that other players' strategies are more or less known to us through discussions with other groups. Though we like the idea of diagonal motion, we feel that we could have developed it a bit more.

Accomplishments

- We demonstrated **predictable performance** that could come from a deterministic single-player mode strategy, regardless of board size or number of players.
- We had **a unique strategy** in which we calculated the location of each other player. No other group attempted this. In fact, we beat the system in a sense. The problem explanation described, "Players do not know their coordinates on the grid. (This excludes certain strategies, such as "everybody meet at (3,5)")". However, we were able to calculate coordinates relative to each player and create a place to meet on the board.
- We created **a specialized player** that was conscious of its surroundings. It could determine whether the environment contained players of the same species or players of foreign species and adapt based on its findings. This strategy (of having two different "modes" of players) is preferable to relying on one generalized player that would do "ok" in several cases.

Acknowledgements

- Bogdan Caprita had the original idea for horizontal and vertical motion in order to determine other players' locations.
- Professor Ross and Sasha Davidev volunteered the idea of having all organisms meet in a region of the board with the smallest area laid down by the organisms' trails.

Group member contributions

- Chris Murphy came up with the idea of the "three board sweeps" to determine other players' locations, and helped resolve the "multiple center points" problem.

- Becky Plummer’s major contribution was in the brainstorming of ideas, implementation of the diagonal wanderer, and in the design, implementation, and debugging of the multiplayer meeting algorithm.
- Andrey Kutser implemented most of the single-player (calculating meeting points, moving to the convergence points, etc.) and conflict resolution code, and developed the history tracking and board coverage ideas.