1. Lecture 8: Natural Algorithms

1.1 Swarms, Flocks, PSOs, Boids

- These swarms, flocks, boids, etc. can be used to (a) study how real flocking, herding, crowd movement occurs; (b) simulate such things for graphics, games, etc. (c) solve desired optimization problems.
- This paper used several principles to update the velocities of the “boids” in the flock:
  1. Collision Avoidance: update your velocity to point away from nearby birds
  2. Velocity Matching: update your velocity to the average of nearby birds
  3. Flock Centering: update your velocity to point towards the center of mass of the flock
- Leaderless; velocities are updated by the influence of each individual’s neighbors over neighbors
- Coordination appears at several levels depending on the density of birds and amount of noise in their update rule. Seminal work on the subject is Vicsek et al. Novel Type of Phase Transition in a System of Self-Driven Particles, Physical Review Letters 75(6):1226-1229 (1995), from which this figure is taken:

(a) random positions; (b) low density, low noise; (c) high density, high noise; (d) high density, low noise (⇒ coordination).

1.2 Particle Swarms For Optimization

- Minimize an objective function $f$.
- A population of particles is maintained (swarm)
• A particle has a position that represents a candidate solution to the problem. \( f(x_k(t)) \) is the “fitness” of that candidate solution. The particle also has a current velocity \( v_k(t) \).

• Each particle position is updated according to a weighted sum of (a) its current velocity, (b) the velocity that would take it towards the best solution seen by any particle in the swarm and (c) the best solution seen by any particle in the local neighborhood.

Hence: particles have an inertia to continue on their current path. They will be swayed away from that by a point with good fitness either among those found by the whole swarm or just local neighborhood. Being pulled towards the global best point so far, draws the particles to good areas of the search space. Keeping an effect of the local neighborhood of particles, however, avoids everyone rushing to a local optimum.

1.3 Natural Algorithms

• The next few sections are based on:

• Flocking network: \( G_t = (V_t, E_t) \) represents the contacts between birds at time \( t \). \( V_t \) is always the set of birds. \( \{u,v\} \in E_t \) if \( d(u,v) \leq 1 \) at time \( t \).

1.4 Model K


• Simplified model: \( n \) birds, with 2D positions at each time step: \( x_1(t), \ldots, x_n(t) \).

• Constant \( \sigma \) velocity birds, only headings can change.

• Each bird has a direction it is flying in at each time step: \( \theta_k(t) \) (this is an angle between 0 and \( 2\pi \))

• Dynamics:
  1. Positions are updated as:
     \[ x_k(t) = x_k(t-1) + \sigma e^{i\theta_k(t)} \] (1)
     where the positions \( x_k(t) \) are considered to be complex numbers \( x_k(t) = a_k + ib_k \).
  2. Flight headings are updated as:
     \[ \theta_k(t+1) = \frac{1}{d_k(t) + 1} \left( \theta_k(t) + \sum_{\{k,j\} \in G_t} \theta_j(t) \right) \] (2)
     where \( d_k(t) \) is the number of neighbors of \( k \) at time \( t \) in \( G_t \). Thus, the new flight heading is the average of the headings of neighboring birds.

1.5 Model D

• Simplified model: \( n \) birds, each with a 2D position at each time step: \( x_1(t), \ldots, x_n(t) \).

• Each bird has a velocity vector: \( v_k(t) \).
Dynamics:

1. Positions are updated as:
   \[ x_k(t) = x_k(t - 1) + v_i(t) \] (3)

2. Velocities are updated as:
   \[ v_k(t + 1) = v_k(t) + c_k(t) \sum_{(k,j) \in G_t} (v_j(t) - v_k(t)) \] (4)
   where \( c_k(t) \) is a given “constant” of proportionality. Here, \( v_k(t + 1) - v_k(t) \approx \text{acceleration} \). Using \( F = ma \) with \( m = 1 \), the interpretation of this new velocity is that each bird has forces on it proportional to the difference between its velocities and each of its neighbors.

1.6 Convergence

- **Theorem** (Chazelle): Both models K and D converge to a steady state in \( \leq 2 \uparrow\uparrow O(n) \) steps, where \( 2 \uparrow\uparrow a \) is a tower of \( a \) 2s.
- Conversely, there is an initial condition such that it takes \( 2 \uparrow\uparrow O(\log n) \) steps for the flock to converge to a steady state.
- Technicalities:
  1. Steady state means the network of neighbors \( G_t \) no longer changes.
  2. Use a hysteresis rule: a change in an edge in \( G_t \) requires a change in distance of more than a very small amount \( (n^{-n^4}) \). This so you can’t get trapped making very small changes in velocities that lead to changes in \( G_t \).
  3. Assume all inputs can be encoded as \( O(\log n) \) bit numbers to avoid really tiny velocities and angles.

1.7 Multiagent agreement systems

- Representations: \( n \) agents, each with a location \( x_k(t) \) in \( \mathbb{R}^d \), and an infinite series of graphs \( G_1, \ldots, \) where the node set of each \( G_i \) are the \( n \) agents.
- General update rule: Each agent moves within the convex hull of its neighbors, but not too close to the boundary. In one-dimension:
   \[ (1 - \rho)m_{k,t} + \rho M_{k,t} \leq x_k(t + 1) \leq \rho m_{k,t} + (1 - \rho)M_{k,t}. \] (5)
   where \( m_{k,t} \) is the minimum position of a neighbor of \( k \) at time \( t \) and \( M_{k,t} \) is the analogous maximum, and where \( \rho \) is a constant such that \( 0 < \rho \leq 1/2 \). This is Equation (3) in Chazelle, The Total s-Energy of a Multiagent System.
- Specific instances of systems can be designed that obey these rules (see below for an example).
- **Definition:** The system \( \epsilon \)-coversges if the agents end up in disjoint intervals of length \( \leq \epsilon \), and all future interactions always take place within these intervals. \( C_\epsilon(n) \) is the number of steps in which some edge of \( G_t \) has length \( > \epsilon \) (the number of non-small steps).
- **Theorem** (Chazelle):
  \[ C_\epsilon(n) \leq \min \left\{ \frac{1}{\epsilon}\rho^{-O(n)}, (\log \frac{1}{\epsilon})^{n-1}\rho^{-n^2-O(1)} \right\} \] (6)
  This is Theorem 1.4 in Chazelle The Total s-Energy of a Multiagent System. Unfortunately, the proof is beyond the scope of this class.
1.8 Krause opinion dynamics model

- Idea: opinions of people represented as a position $x_k(t)$. At each time step, $x_k(t + 1)$ is set to the average of the opinions of the neighbors of $k$: $x_k(t + 1) = \frac{1}{|N_k(t)|} \sum_{j \in N_k(t)} x_j(t)$.
- By Chazelle’s theorem, will converge in $n^{O(n)}$. 