

- ▶ **Markov Processes** are state-transition processes that satisfy the *Markov Property*
 - ▶ **Definition:** the Markov property is the property that the probability distribution over the next possible states of a system depends only on the current state
 - ▶ The Markov Property is satisfied by many interesting systems, e.g.
 - ▶ Cellular Automata
 - ▶ Finite State Automata
 - ▶ Genetic Algorithms
 - ▶ A GA satisfies the Markov Property if we specify each possible population as a separate state
 - ▶ Then the probability of generating any population is a function only of the current population, and the operators used

- ▶ A Markov Process can be defined by a *transition matrix*
- ▶ The transition matrix defines the probability of reaching each possible state, from each possible state
- ▶ So for a transition matrix

$$Q = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

- ▶ the entry $Q_{i,j}$ is the probability that row state i is reached from column state j
 - ▶ E.g. $Q_{1,0} = \frac{1}{2}$
- ▶ Each column is a probability distribution over states
 - ▶ The transition matrix is a *stochastic matrix*
 - ▶ The entries must all be non-negative, and columns must sum to one

Markov Chains

- ▶ A sequence of states in a Markov Process is referred to as a *Markov Chain*
- ▶ For any state at time t , the probability distribution over states at time $t+1$ will be given by the transition matrix
- ▶ In general we can calculate the probability distribution at time t from any probability distribution over states at time $t-1$, through simple matrix multiplication $p(t) = Qp(t-1)$
 - ▶ $p(t)$ is the column vector giving the probability distribution over states
 - ▶ Knowing which state we are in is just a special kind of vector $p(t)$ in which one entry is 1 and all others are 0
- ▶ In fact we can calculate the probability distribution over states at time t in the future, given some initial probability distribution $p(0)$

$$p(t) = Q^t p(0)$$

- ▶ This is a version of the Chapman-Kolmogorov equation

Absorbing States - Limiting Transition Matrix

- ▶ **Definition:** a transition matrix is *reducible* if its states can be reordered to give a transition matrix of the form

$$Q = \begin{bmatrix} I & R \\ 0 & S \end{bmatrix}$$

- ▶ where I is the identity matrix, and 0 is the zeroes matrix
- ▶ **Theorem (Matrix Reducibility):** for a reducible transition matrix Q

$$\lim_{t \rightarrow \infty} Q^t = \begin{bmatrix} I & R(I-S)^{-1} \\ 0 & 0 \end{bmatrix}$$

- ▶ So for a reducible matrix we can calculate the limiting transition matrix
- ▶ The matrix $(I-S)^{-1}$ is called the *fundamental matrix* and is very useful
 - ▶ $N.B.$ the order of the identity matrix I used in calculating the fundamental matrix may differ from that in the decomposition of Q above

Limiting Distributions

- ▶ If the transition matrix for a Markov Process is *primitive*, we can calculate the limiting probability distribution over the different states
 - ▶ **Definition:** A transition matrix is *primitive* (or *regular*) if Q^t has all non-zero entries for some $t \geq 0$
 - ▶ I.e. there is a non-zero probability of reaching every possible state at some arbitrary point far enough into the future
 - ▶ **Theorem (Perron-Frobenius):** For any primitive stochastic matrix Q , $Q^\infty = \lim_{t \rightarrow \infty} Q^t$ exists, where each column of Q^∞ is the unique probability vector q s.t. $Qq = q$ (i.e. q is an eigenvector of Q with eigenvalue 1). $q = \lim_{t \rightarrow \infty} Q^t p(0)$ for any $p(0)$, and all entries of q are non-negative.
 - ▶ I.e. in the limit of infinite time we can calculate the expected distribution over the different possible states the process can be in

Absorbing States

- ▶ If a transition matrix is not primitive it may have absorbing states
 - ▶ An absorbing state is one which, once entered by the process, will never be left
 - ▶ They are identified by a 1 on the diagonal of the transition matrix
- ▶ If an absorbing state can ultimately be reached from any state in the process, the Markov Chain will eventually reach an absorbing state and stay in it
- ▶ In such a case we can calculate the limiting transition matrix Q^∞

Time to Absorption

- ▶ Using the fundamental matrix we can also calculate the expected time to reach an absorbing state from any non-absorbing state
- ▶ **Theorem (Time to Absorption):** Given a reducible transition matrix Q , the times to absorption from each of the states k are the entries a_k in the vector

$$a = 1(I-S)^{-1}$$
- ▶ where 1 is the ones matrix

Markov Chain Analysis of GAs

- ▶ To apply the techniques of Markov Chain analysis to a GA, we need to
 - 1 Identify the states of the process
 - 2 Calculate the transition matrix
- ▶ The first of these is quite straightforward
 - ▶ The states of the process are all the possible populations
 - ▶ Given that a population can contain duplicates, for population size N and search space ℓ the number of possible states is

$$\binom{\ell + N - 1}{N}$$

- ▶ With all possible populations as states, we have a Markov Process
 - ▶ The next state (population) depends (stochastically) only on the current population
- ▶ We shall assume that the transition matrix does not change over time
 - ▶ Hence we have a *homogenous* Markov Process
- ▶ With the states identified, we can now calculate the transition probabilities between states, i.e. the transition matrix

- ▶ We begin by representing a population as a vector

$$v = (v_0, v_1, \dots, v_{|\mathcal{I}|-1})$$

- ▶ where each v_k is the number of copies of point k in the search space, and

$$\sum_{k=0}^{n-1} v_k = N$$

- ▶ The for fitness proportional selection, the probability of an individual representing point i in the search space being selected in a particular population v is

$$P(i|v)_1 = \frac{v_i f(i)}{\sum_{j \in \mathcal{I}} v_j f(j)}$$

- ▶ Given the probability of selection $P(i|v)$ we can calculate the probability of generating a population u from population v , by using the multinomial distribution

$$P(u|v) = N! \prod_{i \in \mathcal{I}} \frac{P(i|v)_i^{u_i}}{u_i!}$$

- ▶ Next we extend the probability expression $P(i|v)_1$ to take account of the mutation and crossover operators
- ▶ To do this we shall first need to construct mixing matrices for these operators...

Mixing Matrices

- ▶ A *mixing matrix* gives the probability of producing each possible offspring chromosome for every possible parent chromosome(s)
- ▶ A mixing matrix is thus an $n \times n$ -dimensional matrix, where n is the number of parents involved in producing an offspring
- ▶ Hence the mixing matrix M for the mutation operator is 2-dimensional
 - ▶ M_{ij} gives the probability that the operator applied to chromosome j produces chromosome i
- ▶ E.g. for standard mutation on binary strings of length $\ell = 2$ we can calculate

$$M = \begin{pmatrix} (1-\mu)^2 & \mu(1-\mu) & \mu(1-\mu) & \mu^2 \\ \mu(1-\mu) & (1-\mu)^2 & \mu^2 & \mu(1-\mu) \\ \mu(1-\mu) & \mu^2 & (1-\mu)^2 & \mu(1-\mu) \\ \mu^2 & \mu(1-\mu) & \mu(1-\mu) & (1-\mu)^2 \end{pmatrix}$$

Mixing Matrices

- ▶ For crossover, the number of parents involved is 2
- ▶ Assuming 1 offspring is produced, we thus need a 3-dimensional matrix M where $M_{i,j,k}$ is the probability that parents i and j produce offspring k through crossover
- ▶ However we can simplify by calculating the probability that an arbitrary chromosome is produced

$$M(0) = \begin{pmatrix} \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \end{pmatrix}$$

- ▶ where $M_{i,j}(0)$ is the probability that chromosomes i and j produce the all-zeroes chromosome through application of the crossover operator
- ▶ Now we can calculate any point in the full 3-dimensional matrix through a permutation

$$M_{i,j}(k) = M_{\oplus i, \oplus j}(0)$$

where i, j and k are the binary representations of the corresponding matrix indices, and \oplus is XOR

Analysing the Simple GA - Crossover

- ▶ If we have the crossover mixing matrix $M(0)$ and permutations on it as previously described, we can calculate the probability of producing chromosome k from chromosomes i and j using mutation, selection and crossover

$$P(k|v)_3 = \sum_{i,j \in \mathcal{I}} M_{i,j}(k) P(i|v)_2 P(j|v)_2$$

- ▶ As any reasonable crossover operator should be able to produce any chromosome k given suitable parents i and j , the sum in the above expression will be greater than zero
- ▶ Also we have already shown $P(i|v)_2 > 0$ for all i
- ▶ So the transition matrix is now

$$P(u|v) = N! \prod_{i \in \mathcal{I}} \frac{P(i|v)_i^{u_i}}{u_i!}$$

- ▶ and is primitive

Analysing the Simple GA

- ▶ We have shown that a GA with or without crossover is primitive
 - ▶ It has no absorbing states, hence does not converge
 - ▶ Its limiting distribution is calculable
- ▶ *Theorem (Davis-Principe)*: If Q is the primitive transition-matrix for a GA with non-zero mutation rate, the limiting distribution is given by

$$q_v = \frac{|Q_v - I|}{\sum_u |Q_u - I|}$$

- ▶ where q_v is the probability of population v , and the matrix Q_u is the matrix Q with the u th column replaced by zeroes
- ▶ This formalises the intuition that mutation acts to prevent population convergence

Analysing the Simple GA - Mutation

- ▶ Given our mixing matrix, which we label U , the probability of generating individual i through selection then mutation is

$$P(i|v)_2 = \sum_{j \in \mathcal{I}} U_{i,j} P(j|v)_1$$

- ▶ So the two-operator transition matrix is defined by

$$P(u|v) = N! \prod_{i \in \mathcal{I}} \frac{P(i|v)_i^{u_i}}{u_i!}$$

- ▶ Suppose that U contains only non-zero entries, as it would for standard bitwise mutation with $\mu > 0$
 - ▶ If there are no zero entries in U , the matrix is primitive and there can be no absorbing states
 - ▶ The GA does not converge, and we can calculate the limiting probability distribution over the states with the Perron-Frobenius theorem

GA Non-Convergence

- ▶ Consider the example of a search space $\{00, 01, 10, 11\}$ denoted $\mathcal{C} = \{0, 1, 2, 3\}$
- ▶ The fitness function is $f(x) = x + 1$
 - ▶ I.e. $f(0) = 1, f(3) = 4$, etc.
- ▶ The mixing matrix for standard mutation with $\mu = 0.1$ is

$$U = \begin{pmatrix} 0.81 & 0.09 & 0.09 & 0.01 \\ 0.09 & 0.81 & 0.01 & 0.09 \\ 0.09 & 0.01 & 0.81 & 0.09 \\ 0.01 & 0.09 & 0.09 & 0.81 \end{pmatrix}$$

- ▶ For population size $N = 4$ there are $\binom{4}{i} = 35$ possible populations
- ▶ We can calculate the 35×35 transition matrix for fitness proportional selection and mutation

- Applying the Davis-Principe theorem gives us the limiting distribution over all possible populations
- The highest probability populations in this limiting distribution are

Population	Probability
(0,0,0,4)	0.143
(0,0,1,3)	0.124
(0,1,0,3)	0.102

- i.e. the most probable population is that containing only copies of the optimal solution
- But, 15 possible populations contain no copies of the optimal solution, and the cumulative probability of these populations is approximately 17%
- The GA will not converge, regardless of how long it is run for
 - This can be demonstrated empirically
 - Exercise: Run multiple replicates of a simple GA without crossover on the above fitness function, and compare the proportion of optimal and suboptimal populations with the theoretical limits

- We have proved that a GA with mutation does not converge
- Convergence *non-proofs* are less compelling arguments for using a search heuristic than convergence *proofs*!
- How can we modify the GA so that we can prove convergence?
 - We might *anneal* the mutation rate
 - As in simulated annealing, we progressively reduce the mutation rate to restrict the search
 - However there is no guarantee the GA will converge to the optimal population

- Elitism can also give us provable convergence
- Elitism guarantees that for the best individual k in the population at time t , then at time $t + 1$
 - k remains in the population, or
 - A better solution than k is in the population
- Now the fitness of the best individual in the population is a monotonically increasing sequence with an upper-bound given by the optimal fitness value in the search space

GA Convergence - Elitism

- Convergence can be proved by showing that the sequence is a *supermartingale*
 - Definition: For a stochastic sequence X_0, X_1, X_2, \dots and function D mapping members of this sequence to the real numbers, a new stochastic sequence D_0, D_1, D_2, \dots can be generated by $D_t = D(X_t)$. If the function D is bounded and $E(D_{t+1}|X_t) \leq D_t$, then the sequence D_t is a *supermartingale*
- Non-negative supermartingales converge almost certainly to some limit, so we can prove
 - Theorem (Rudolph): If X_t is a population at time t and $f(X_t)$ is the fitness of the fittest individual in X_t . Then for an arbitrary elitist GA the sequence $D_t = f^* - f(X_t)$, where f^* is the fitness of the best possible solution, forms a non-negative supermartingale which converges almost certainly to zero.
 - This is an intuitive idea, but it is nice to have a proof
 - We can also approximate the expected time to convergence using the absorbing states theorem on a simplified Markov process

Calculating with Markov Chains

- Markov Chain analysis can be used to analyse GA behaviour without explicitly manipulating transition matrices for realistic GAs
- However, the number of possible populations ($\binom{N+K-1}{N}$) rapidly becomes unmanageable even for comparatively small N and K
- To make the resulting transition matrices manageable we can apply a *state aggregation* technique
 - Identify states sufficiently similar to be aggregated
 - Recalculate the transition matrix to take account of the new aggregated states
 - Recalculate the probability of entering the aggregated states from all other states
 - Recalculate the probability of reaching all other states from the aggregated states
 - Calculate the probability of staying within the aggregated states

Calculating with Markov Chains - State Aggregation

- We seek states that have an identical probability distribution over the next state to be visited in the chain
 - Once in one of these states, it is irrelevant which of the states we are actually in
 - These states can be aggregated into a single state without changing the behaviour of the Markov Chain
 - They can be identified by having identical columns in the transition matrix
- Finding states with identical distributions is unlikely, so we can aggregate other states that are in some way similar
 - We can compare the sum of squares difference between two matrix columns and aggregate if it is under some threshold ϵ , i.e. if for two matrix columns u and v

$$\|u - v\|^2 < \epsilon$$

$$\|u - v\|^2 = \sum_x (P(k|u) - P(k|v))^2$$

where

Calculating with Markov Chains - Matrix Recalculation

- As we aggregate states, we must recalculate the transition probabilities they are involved in
- The probability of entering the newly aggregated state is simply calculated as

$$P(\{u, v\}|k) = P(u|k) + P(v|k)$$

Calculating with Markov Chains - Matrix Recalculation

- The probability of going to another state from the newly aggregated state is calculated as the weighted mean probability of the transitions from each of the original states
 - The weights are estimates of the probabilities of entering each of the original states, obtained by summing the probabilities in the rows of the transition matrix for each of those states

$$w_u = \sum_j P(u|j)$$

$$w_v = \sum_j P(v|j)$$

- So the the column in the transition matrix for the newly aggregated state is given by

$$P(k|\{u, v\}) = \frac{w_u P(k|u) + w_v P(k|v)}{w_u + w_v}$$

Calculating with Markov Chains - Matrix Recalculation

- Finally, we calculate the probability of staying within the newly aggregated state
- As for the probability of leaving the aggregated state, we must estimate the probability we are in each of the original states
- The probability of staying within the new state is again a weighted mean

$$P(\{u, v\}|\{u, v\}) = \frac{w_u(P(u|u) + P(v|u)) + w_v(P(u|v) + P(v|v))}{w_u + w_v}$$

- ▶ With these approximation tools we can simplify our transition matrix in two main ways
 - One pass, in which we only aggregate original states
 - Multipass, in which we aggregate aggregated states
- ▶ By varying ϵ we have a speed-accuracy trade-off
 - High epsilon aggregates more states, hence increases calculation speed with the transition matrix...
 - ...but at the expense of accuracy of the approximation
 - And vice-versa