Глава 15
БИОЛОГИЧЕСКИЕ И ЭКОЛОГИЧЕСКИЕ СИСТЕМЫ

Имитационное моделирование помогает решить широкий круг научных задач, связанных с изучением бактериальных сообществ, различных механизмов коллективного и адаптивного поведения самообучающихся особей, эволюции популяций [4; 8], построением "организменных" моделей, созданием модели мозга [19]. Кроме того, моделирование био- и экосистем позволяет спрогнозировать развитие сельского хозяйства и среды обитания животных и человека [1; 18]. В настоящей главе рассмотрены сравнительно простые модели развития популяций, основанные на решении системы дифференциальных уравнений [14], применении метода клеточных автоматов и мультиагентного подхода, а также модели, использующие генетические алгоритмы [9; 15; 18].

15.1. Дискретные модели развития отдельной популяции

Первая математическая модель динамики популяции была изучена Леонардо Пизанским, известным как Фибоначчи, в «Трактате о счете» (1202 г.). Он проанализировал развитие популяции кроликов, исходя из следующих допущений: 1) изначально имеется новорожденная пара – самец и самка; 2) начиная со второго месяца после своего рождения, самец и самка спариваются и каждый месяц производят новую пару кроликов; 3) кролики никогда не умирают. Необходимо определить, как будет увеличиваться количество пар кроликов с течением времени.

Понятно, что количество пар кроликов F_i в конце i-го месяца будет складываться из числа пар F_{i-1} кроликов в предыдущем месяце и числа

только что появившихся пар, которое равно количеству пар F_{i-2} два месяца назад. Получается: $F_n = F_{n-2} - F_{n-1}$. Несложно составить программу, вычисляющую **ряд Фибоначчи**: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144 ... или промоделировать эту систему в электронных таблицах Excel.

Рассмотрим еще одну дискретную модель. Допустим, на изолированном острове имеются насекомые, которые размножаются один раз в год, например, летом. Введем целочисленную переменную t, соответствующую номеру года. Если численность популяции в момент t равна $N(t)$, то в момент времени $t+1$ она составит $N(t+1) = rN(t)$. Можно записать следующее:

$$N(1) = rN(0) = rN_0, \quad N(2) = rN(1) = r^2 N(0) = r^2 N_0,$$

$$N(3) = rN(2) = r^3 N(0) = r^3 N_0, \ldots, \ \ldots \ N(t) = r^t N(0) = r^t N_0.$$

Получаем уравнение: $N(t) = r^t N_0$, где $N_0 = N(0)$. Эта модель не учитывает **внутривидовой конкуренции**, которая состоит в использовании энергетического ресурса (еды) организмом, что уменьшает доступность этого ресурса другим организмом того же вида.

При увеличении численности популяции N из-за внутривидовой конкуренции количество ресурса, приходящееся на одну особь, уменьшается, рождаемость R снижается. Если считать, что коэффициент размножения $R(N) = R_0 /(1 + (aN(t))^b)$, то численность популяции

$$N = R_0 N_t /(1 + (aN_t)^b) = R(N)N(t)$$

стремится к некоторому пределу.

Широко известна еще одна дискретная модель **популяции с неперекрывающимися поколениями**. Размножение насекомых на изолированном острове описывается квадратичным отображением $x^{t+1} = ax^t(1 - x^t)$, где $0 < x < 1$ [14, с. 89–92]. В самом деле, если численность насекомых невелика, то внутривидовая конкуренция отсутствует и численность $N(t+1)$ в следующем $(t+1)$-м поколении пропорциональна числу насекомых $N(t)$ в предыдущем t-м поколении. С ростом $N(t)$ насекомые начинают испы-

тывать нехватку энергетического ресурса (еды) и их коэффициент размножения уменьшается. При достижении численности насекомых $N(t)$ своего предельного значения A коэффициент размножения уменьшается до 0. Получаем следующие уравнения: $N(t+1) = rN(t)(A - N(t))$, или $x^{t+1} = ax^t(1 - x^t)$, где $x^t = N(t)/A$, $a = rA$.

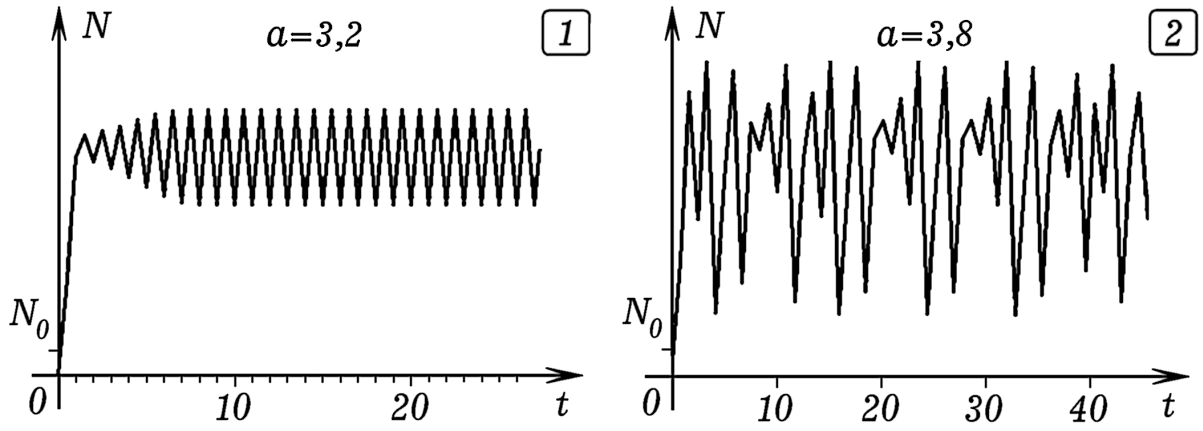

Рис. 15.1. Дискретная модель размножения насекомых $a = 3{,}2$ и $3{,}8$

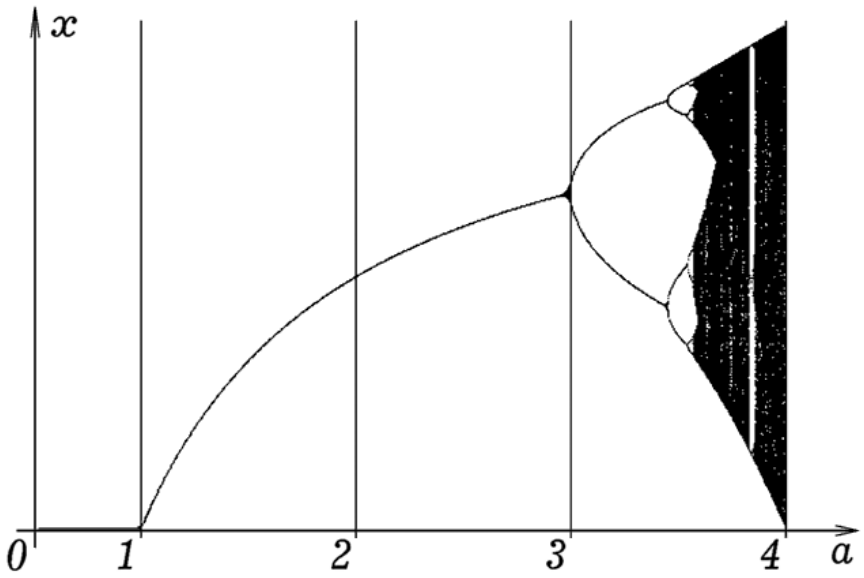

Рис. 15.2. Переход от детерминированного поведения к хаотическому

Результаты вычислений $N(t)$ ($t = 1,\ 2, \ldots$) при $a = 3{,}2$ и $3{,}8$ приведены на рис. 15.1. Видно, что при увеличении коэффициента размножения a система начинает совершать **хаотические колебания**. Чтобы изучить поведение этой функции в зависимости от параметра a, построим график

$N^t = N^t(a)$ при $t > 100$. Используется программа ПР-1, которая многократно вычисляет значения N^t при целочисленном t в интервале от 100 до 1000 и различных значениях параметра a, изменяющихся от 0 до 4. Результаты ее работы представлены на рис. 15.2. Видно, что пока a меньше 3, каждому значению параметра a соответствует одно значение N^t. Дальнейшее увеличение a приводит к **каскаду бифуркаций удвоения периода**: сначала одному значению a отвечают два значения N^t, затем 4, затем 8 и т. д. Последовательность бифуркаций, начинаясь слева при $a = 3$, переходит в хаос справа, которому соответствует уже не непрерывная линия, а **фрактальное множество точек**. Полосы хаоса перемежаются с "окнами периодичности". Все это согласуется с графиками на рис. 15.1.

15.2. Непрерывные модели роста популяции

Кроме дискретных моделей, описывающих размножение организмов с неперекрывающимися поколениями, для изучения биологических процессов применяются **непрерывные модели**. При этом предполагается следующее [13, с. 29]: 1) во всех точках территории плотность особей (насекомых, бактерий) данного вида одинакова; 2) всеми возрастными, половыми и генетическими различиями можно пренебречь; 3) изменения численности популяции описываются детерминистическими уравнениями; 4) контакты особей одного и различных видов длятся пренебрежимо малый промежуток времени. Иногда пренебрегают временем вынашивания потомства. По **закону Мальтуса**, для популяции с непрерывным размножением скорость dN/dt роста ее численности пропорциональна N:

$$\frac{dN(t)}{dt} = rN(t), \qquad \int_{N_0}^{N(t)} \frac{dN}{N} = \int_0^t r dt, \qquad N(t) = N_0 e^{rt}.$$

Эта модель описывает популяцию, в которой нет конкуренции и скорость размножения r постоянна [14, с. 36–37]. Если $r > 0$, то численность бесконечно увеличивается, если $r < 0$, то она уменьшается до 0.

Допустим, имеет место внутривидовая конкуренция, состоящая в использовании энергетического ресурса одним животным, что снижает доступность этого ресурса для другого животного того же вида. Это приводит к тому, что с ростом численности N коэффициент рождаемости уменьшается, а когда N приближается к предельному значению K, рождаемость падает до 0. Получается **уравнение Ферхюльста** [14, с. 37–39]:

$$\frac{dN(t)}{dt} = rN(t)\frac{K - N(t)}{K} = rN(t)(1 - aN(t)), \quad a = 1/K \approx 10^{-2} - 10^{-5}.$$

Здесь K – максимальное количество особей, которое может прокормить среда. При увеличении N величина aN приближается к 1, при этом скорость роста численности уменьшается из-за нехватки ресурса (пищи). Зависимость $N(t)$ подчиняется **логистическому закону**:

$$N(t) = \frac{KN_0}{N_0 + (K - N_0)e^{-rt}}.$$

При $t \to \infty$ численность популяции N стремится к K. Для моделирования используется программа ПР-2; результаты – на рис. 15.3.

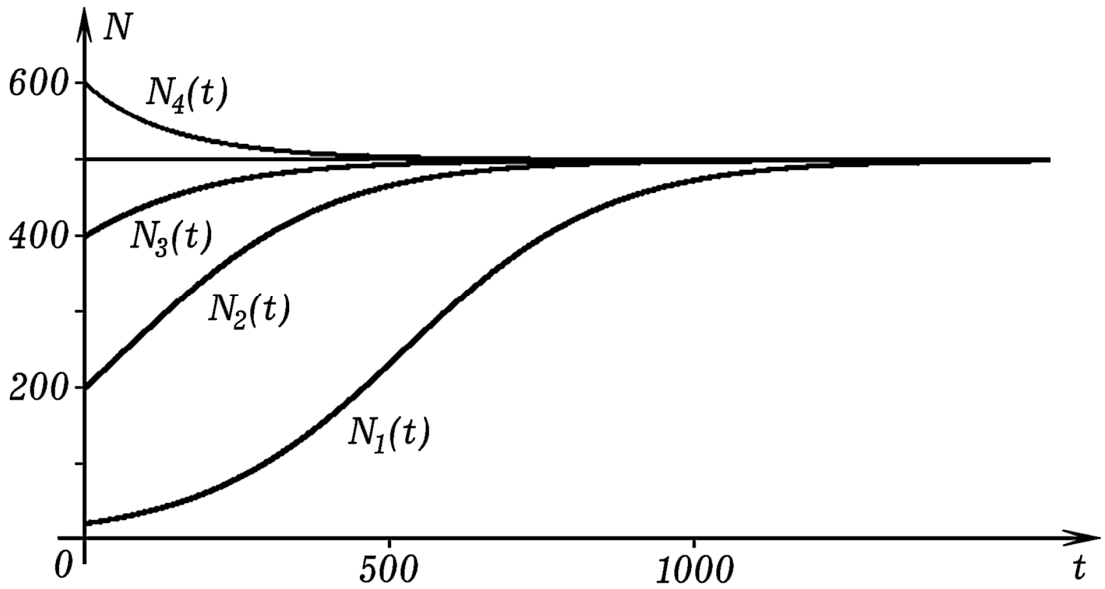

Рис 15.3. Рост популяции с внутривидовой конкуренцией

Теперь рассчитаем изменение численности популяции с течением времени в случае, когда смертность на некоторое время увеличивается в k раз (например, вследствие эпидемии). Чтобы учесть уменьшение числа животных из-за роста смертности, введем слагаемое $-sN$, где s – коэффициент смертности (обычно $0 < s < 0{,}1$). Отсюда следует:

$$\frac{dN}{dt} = rN\frac{K-N}{K} - sN \quad \text{или} \quad N^{t+1} = N^t + r(1 - N^t / K)\Delta\tau - sN^t\Delta\tau .$$

Так как поколение, родившееся в момент t, начинает приносить потомство через время t', то на скорость роста популяции в момент t влияет ее численность в момент $t - t'$. Получаем **уравнение Хатчинсона** [13, с. 54]:

$$\frac{dN}{dt} = rN(t-t')\left(1 - \frac{N(t-t')}{K}\right) - s(t)N(t).$$

Запишем его в конечных разностях:

$$N^{t+1} = N^t + (rN^{t-t'}(K - N^{t-t'})/K - s^t N^t)\Delta\tau .$$

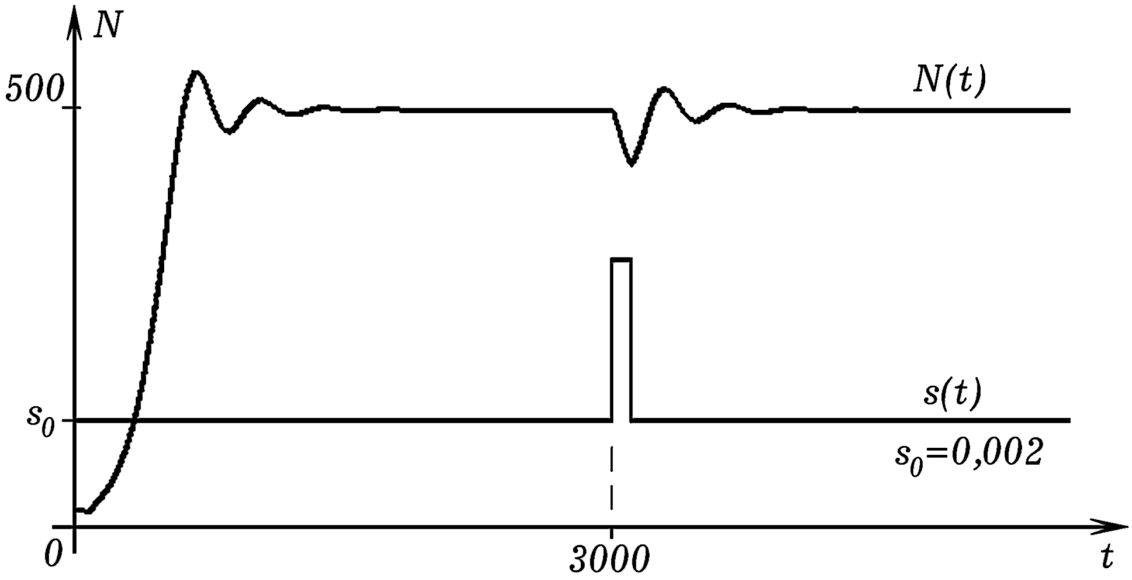

Рис. 15.4. Изменение численности популяции с течением времени

Для численного решения этого уравнения используется программа ПР-3, результаты – на рис. 15.4. Сначала численность N растет и, совершив несколько затухающих колебаний, стабилизируется на величине $N' = 500$. В момент $t_1 = 3000$ смертность s увеличивается в $2{,}5$ раза на не-

продолжительное время (допустим, начинается эпидемия). Количество особей уменьшается, а когда s возвращается к своему первоначальному значению s_0, опять возникают затухающие колебания. Их период определяется средним временем τ между рождением особи и появлением у нее потомства.

15.3. Матричное моделирование популяции

Суть матричного метода состоит в следующем [13, с. 60-61; 18, с. 60–71]. Популяцию разделяют на m дискретных возрастных классов, которые имеют различные коэффициенты рождаемости и смертности. Допустим, в момент t популяция представлена в виде вектора (одномерной матрицы) (N_0^t, N_1^t, N_2^t, ..., N_m^t). Тогда структура популяции в следующий дискретный момент $t+1$ может быть определена так:

$$N_{i+1}^{t+1} = p_i N_i^t \text{ или } \begin{pmatrix} N_0^{t+1} \\ N_1^{t+1} \\ \vdots \\ N_m^{t+1} \end{pmatrix} = \begin{pmatrix} r_0 & r_1 & \dots & r_{m-1} & r_m \\ p_0 & 0 & 0 & \dots & 0 & 0 \\ 0 & p_1 & \dots & 0 & 0 & 0 \\ 0 & 0 & \dots & p_{m-1} & 0 \end{pmatrix} \cdot \begin{pmatrix} N_0^t \\ N_1^t \\ \vdots \\ N_m^t \end{pmatrix},$$

где r_0, r_1, ..., r_m – плодовитость самок различного возраста, p_i – вероятность того, что самка t-го возраста доживет до $t+1$ возраста. Вероятности p_t связаны с коэффициентом смертности s_t так: $p_t = 1 - s_t$.

В качестве примера применения **матричного метода** рассмотрим программу ПР-4. Вначале задается исходное распределение числа особей по возрастам: $N_t = 1000\exp(-0{,}01 \cdot t)$, где t – возраст в годах (месяцах, днях). Допустим, максимальная продолжительность жизни $T = 30$ лет и при $t \le h = 12$ особи не могут производить потомство, то есть коэффициент рождаемости $r = 0$. При $t > h$ коэффициент рождаемости быстро увеличивается, достигает максимума, а затем медленно убывает в соответствии с уравнением:

$$r(t) = \frac{0{,}7(1-\exp(-0{,}4(t-h)))\exp(-0{,}2(t-h))}{1+S_N/10^4}, \text{ где } S_N = \sum_{t=1}^{m} N_t .$$

Формула учитывает, что коэффициент рождаемости при увеличении биомассы (количества особей в популяции S_N) уменьшается из-за внутривидовой конкуренции. Зависимость коэффициента смертности от возраста t будем аппроксимировать логистическим уравнением:

$$s(t) = \frac{0{,}32}{1+\exp(0{,}5(20-t))} + 0{,}02 .$$

При $t = 20$ коэффициент смертности $s \approx 0{,}18$. Количество родившихся особей в $t+1$ году определяется так:

$$N_0^{t+1} = \sum_{i=h}^{m} r_i^t N_i^t .$$

Программа пересчитывает количество особей каждой возрастной категории $N_i^{t+1} = N_i^t(1-s_i)$, результаты записывает в массив. Сначала численность популяции растет, средний коэффициент рождаемости снижается, система стремится к динамическому равновесию, когда средние рождаемость и смертность примерно равны. На рис. 15.5 представлены установившееся распределение количества особей по возрастным категориям $N = N(i)$ при $t \to \infty$, а также графики $r = r(i)$ и $s = s(i)$.

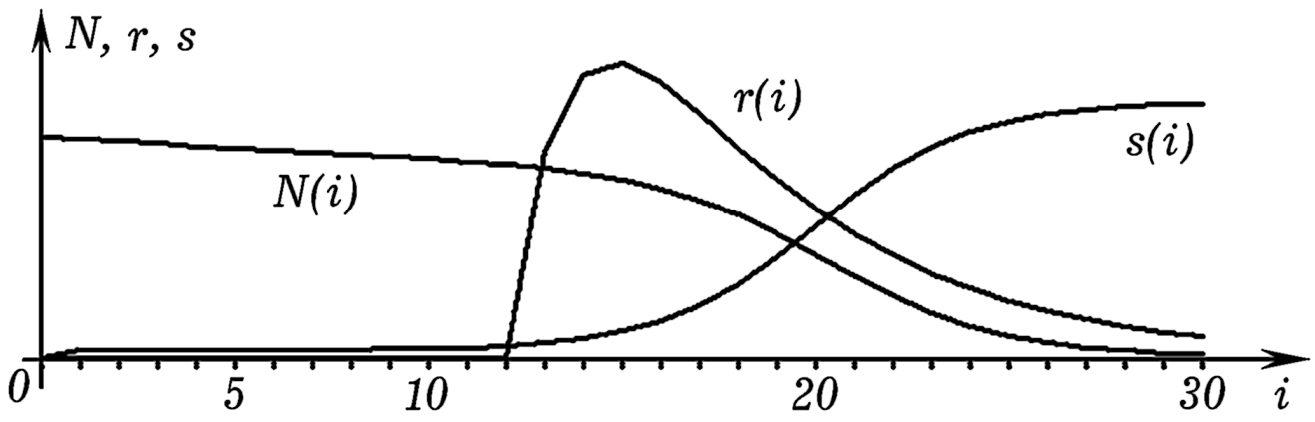

Рис 15.5. Распределение N, r, s по возрастным категориям популяции

15.4. Модели, учитывающие межвидовую конкуренцию

Допустим, на изолированном острове живут две популяции с численностями $N_1(t)$ и $N_2(t)$, потребляющие один и тот же энергетический ресурс. Чтобы учесть межвидовую конкуренцию, используют **модель Лотки-Вольтерры** [3; 12; 18]:

$$\begin{cases} \dfrac{dN_1}{dt} = r_1 N_1 \dfrac{K_1 - N_1 - a_{12}N_2}{K_1}, \\ \dfrac{dN_2}{dt} = r_2 N_2 \dfrac{K_2 - N_2 - a_{21}N_1}{K_2}. \end{cases}$$

Коэффициенты a_{12}, a_{21} позволяют учесть то, как численность N_1 первой популяции уменьшает скорость роста второй популяции N_2 и наоборот. При этом возможно изучить устойчивое сосуществование двух популяций (при $a_{12} \cdot a_{21} < 1$), либо ситуацию, когда одна популяция подавляет другую [7, с. 756-758].

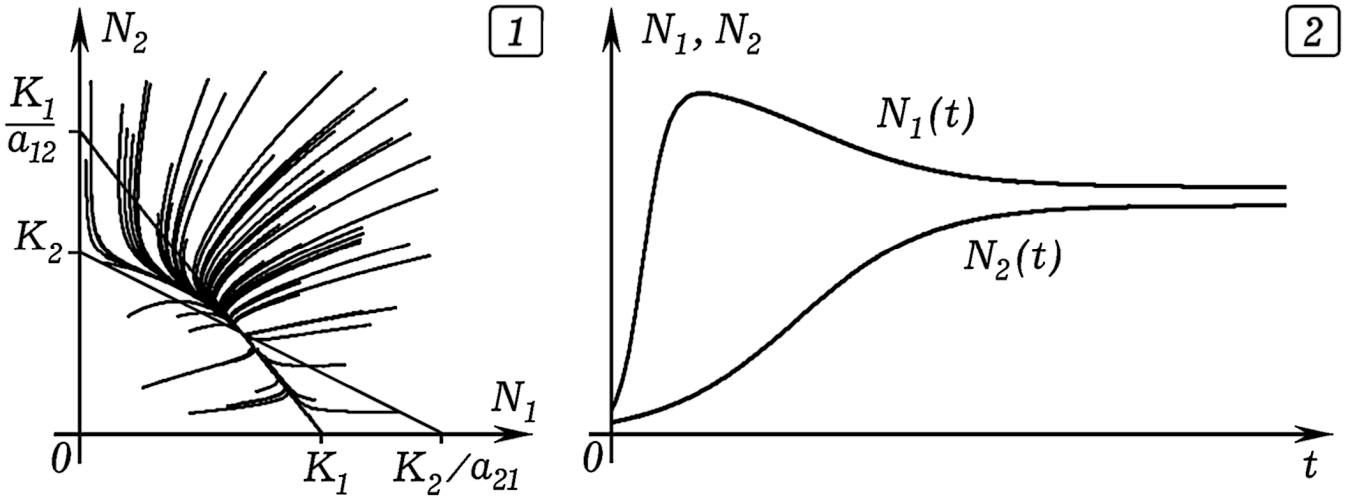

Рис. 15.6. Фазовая кривая и графики $N_1(t)$ и $N_2(t)$

Для моделирования применяется программа ПР-5. Чтобы определить условия роста численности той или иной популяции используют диаграммы, на которых численность одного вида откладывают по оси абсцисс, а другого – по оси ординат. При параметрах модели $K_1 = 200$,

$r_1 = 0{,}002$, $\alpha_{12} = 0{,}05$, $K_2 = 300$, $r_2 = 0{,}01$, $\alpha_{21} = 0{,}5$ будем случайным образом изменять исходные значения N_1 и N_2. Из получающихся фазовых кривых (рис. 15.6.1) видно, что при данном наборе параметров выполняется условие устойчивого сосуществования популяций и все фазовые кривые стремятся к некоторой точке, соответствующей динамическому равновесию системы. На рис. 15.6.2 представлены графики $N_1(t)$ и $N_2(t)$, соответствующие начальным численностям $N_{10} = 20$ и $N_{20} = 10$.

15.5. Модель “хищник-жертва”

Иногда организмы одного вида питаются организмами другого вида. Пусть на некоторой ограниченной территории живут волки и зайцы (хищники и жертвы). Волки поедают зайцев, а те питаются растительной пищей, запасы которой очень велики. Численность популяции зайцев и волков обозначим через N и C соответственно. Быстрота увеличения числа зайцев dN/dt тем больше, чем больше их количество N, чем выше коэффициент рождаемости r и чем меньше частота встреч волков и зайцев. Скорость увеличения численности волков dC/dt пропорциональна частоте их “встреч” с зайцами. Если учесть, что частота встреч волков и зайцев на ограниченной территории пропорциональна произведению NC, то получаем систему [12; 18]:

$$\begin{cases} \dfrac{dN}{dt} = rN - aNC, \\ \dfrac{dC}{dt} = faNC - qC. \end{cases}$$

Здесь q - коэффициент смертности волков, a, f – коэффициенты, учитывающие эффективность процессов поиска зайца и перехода пищи в потомство волков. Программа ПР-6 позволяет вычислить численность популяции волков и зайцев в последовательные моменты времени и построить фазовую кривую системы (рис. 15.7).

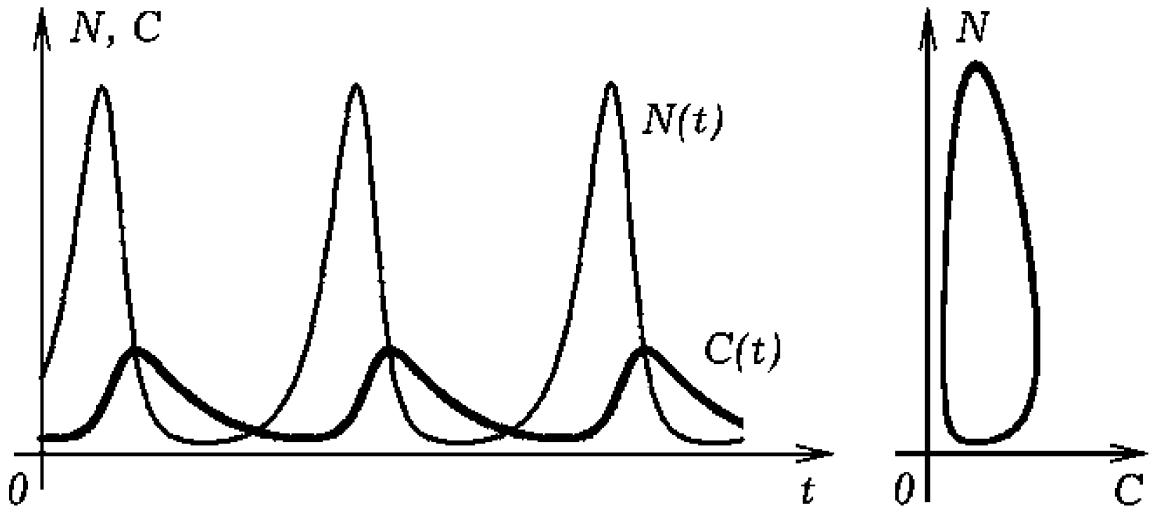

Рис. 15.7. Динамика численности популяции волков и зайцев

В реальной ситуации плотности распределения волков и зайцев зависят не только от времени t, но и от координат x и y: $N = N(x, y, t)$ и $C = C(x, y, t)$. Биологическая система "хищник-жертва" с учетом пространственного распространения обоих видов в одномерном случае описывается системой уравнений [2, с. 162-165]:

$$\frac{\partial N}{\partial t} = R(N - L)\frac{K - N}{K} - bNC + D_N \frac{\partial^2 N}{\partial x^2},$$

$$\frac{\partial C}{\partial t} = -sC + ebNC + D_C \frac{\partial^2 C}{\partial x^2}.$$

Здесь $N(x,t)$ и $C(x,t)$ – линейные плотности (или численности) жертвы и хищника (в м$^{-1}$); b и e – коэффициенты выедания и переработки хищником биомассы жертвы (с превращением ее в потомство); s – коэффициент смертности хищников; K – емкость среды; R – коэффициент размножения жертвы; L – нижняя критическая численность. Коэффициенты диффузии D_N и D_C характеризуют быстроту рассеивания стаи, обусловленную хаотическим движением особей. Задача решается тем же методом, что и при моделировании диффузии и автоволн (глава 10). Используется программа ПР-7, результаты моделирования – на рис. 15.8. При соответствующем задании параметров модели и начальных условий изме-

няется распределение зайцев $N(t)$ и образуется "волна преследования" волков $C(t)$.

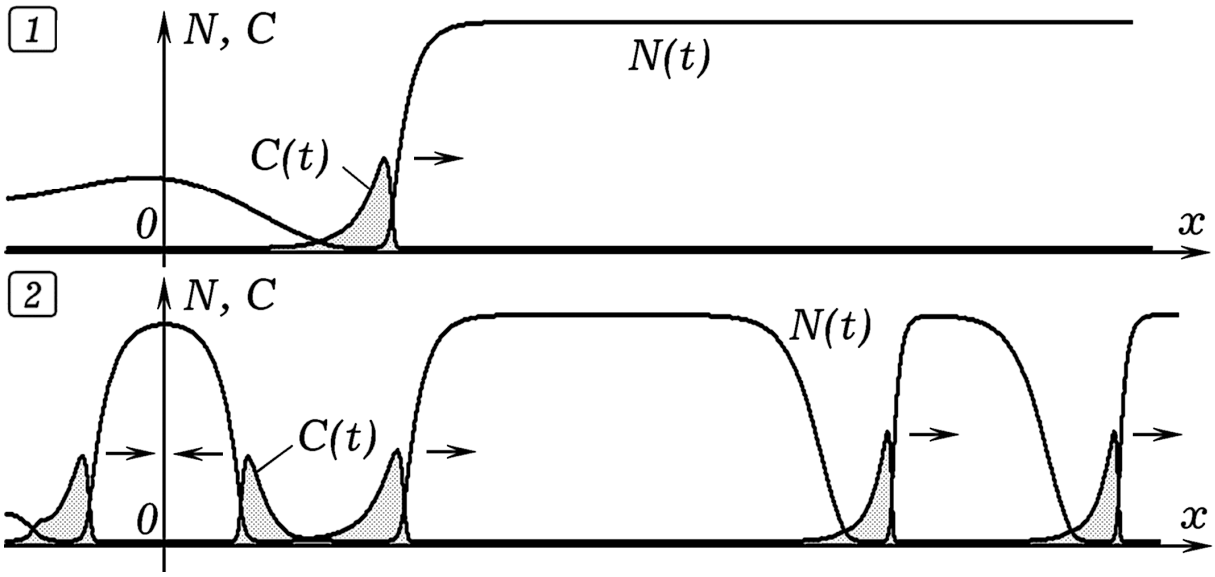

Рис. 15.8. Решение задачи в одномерном случае

При определенных параметрах модели можно получить **автомодельное решение** рассматриваемой системы уравнений (рис 15.9) [12, с. 299-302]. Возникает автоволновой процесс: в направлении оси Ox бежит "волна" зайцев, которую преследует "волна" волков, причем форма обеих "волн" с течением времени практически не изменяется (программа ПР-8).

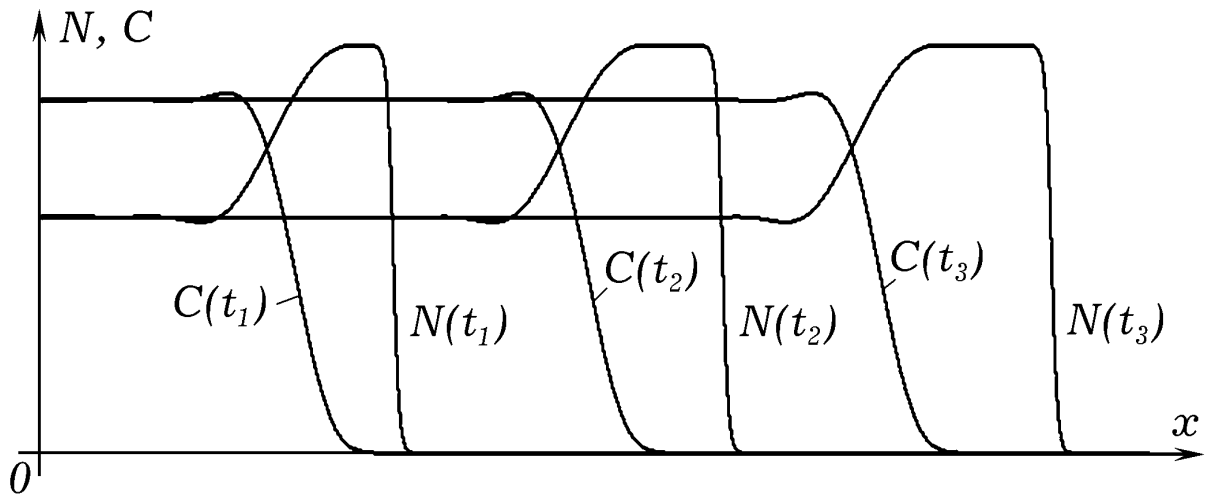

Рис. 15.9. Автомодельное решение для одномерного случая

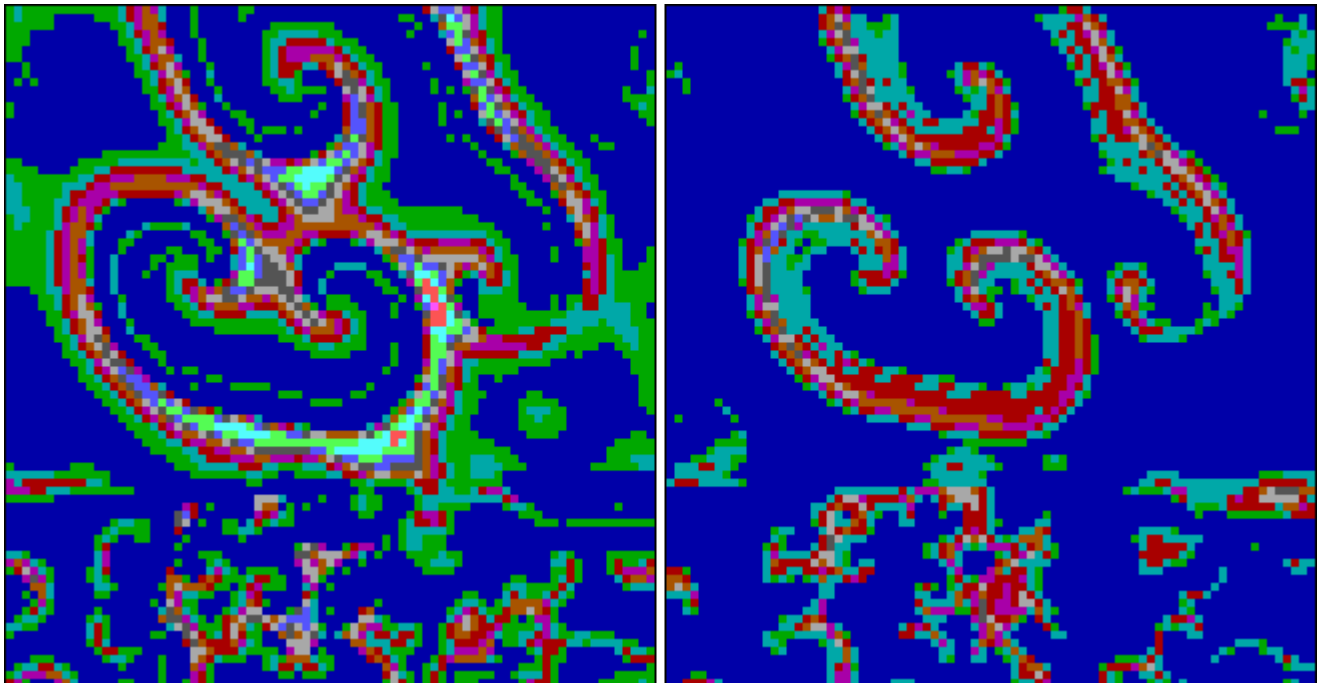

Рис. 15.10. Распределение зайцев и волков на некоторой территории

В двумерном случае распределения хищников $C = C(x, y, t)$ и жертв $N = N(x, y, t)$ описываются уравнениями:

$$\frac{\partial N}{\partial t} = R(N - L)\frac{K - N}{K} - bNC + D_N\left[\frac{\partial^2 N}{\partial x^2} + \frac{\partial^2 N}{\partial y^2}\right],$$

$$\frac{\partial C}{\partial t} = -sC + ebNC + D_C\left[\frac{\partial^2 C}{\partial x^2} + \frac{\partial^2 C}{\partial y^2}\right].$$

Используется программа ПР-9. Можно так подобрать параметры системы, что происходит периодическое образование и распространение двумерных автоволн. Результаты моделирования приведены на рис. 15.11.1 (распределение зайцев) и 15.11.2 (распределение волков).

15.6. Мультиагентный подход к моделированию биологических систем

Сущность мультиагентного подхода заключается в создании достаточно большого числа программных агентов, моделирующих поведение

отдельных особей. При этом может предполагаться следующее [17]: 1) среда обитания состоит из квадратных ячеек с координатами (i, j), внутри которых может находиться одна особь того или иного вида и жизненный ресурс (трава); 2) известны начальные распределения популяций хищников и жертв; 3) состояние системы изменяется дискретно: в каждый следующий момент времени конкретная особь либо остается в текущей ячейке, либо перемещается в соседнюю; 4) все особи на каждом временном шаге уменьшают запасенное количество энергии на некоторую величину; 5) энергия пополняется за счет потребления ресурса (для жертв) или поглощения других особей (для хищников); 6) если энергия особи становится ниже порогового значения $E_{\min}$, она гибнет от голода; 7) особи через определенные промежутки времени воспроизводят потомство.

Этот подход позволяет установить общие закономерности развития популяций хищников и жертв, выявить типичные аттракторы каждой популяции, соответствующие тем или иным наборам параметров. Так, можно обнаружить, что распределение хищников сильно зависит от фрагментации территории и, оказавшись на ограниченных "островах", они вымирают, в то время как численность жертв (животных, питающихся растениями, и которых поедают хищники) стремится к некоторому пределу.

Рассмотрим компьютерную модель "рыбы-акулы". Используемая программа [6] должна содержать цикл по времени t; каждая итерация цикла приводит к увеличению t на 1 УЕВ (условную единицу времени). Модель двумерная и дискретная; поверхность моря разбита на квадратные ячейки длиной 1 УЕД (условная единица длины). Каждой ячейке с координатами (i, j) отвечает элемент массива x_{ij}. Ячейка может либо быть пустой ($x_{ij} = 0$), либо содержать рыбу ($x_{ij} > 0$), либо содержать акулу ($x_{ij} < 0$). Абсолютное значение x_{ij} равно возрасту рыбы или акулы. Если $x_{ij} = 3$, то ячейка содержит рыбу возрастом 3 УЕВ. Если $x_{ij} = -5$, значит в этой ячейке находится акула, возраст которой 5 УЕВ.

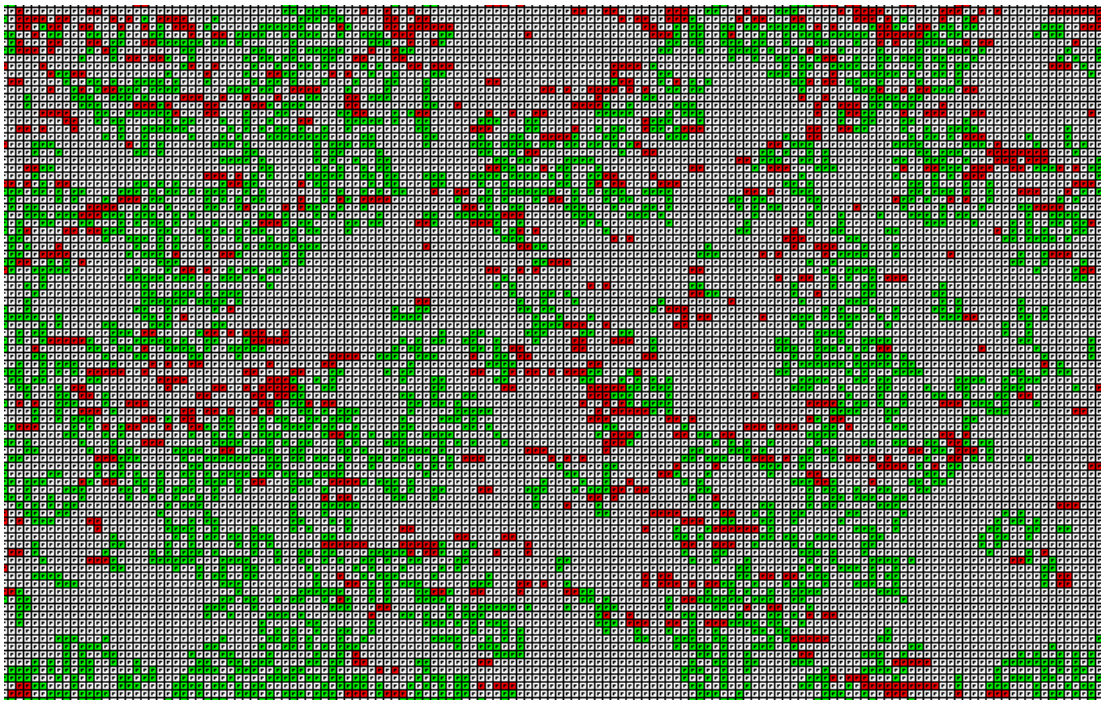

Рис. 15.11. Дискретная модель совместного существования рыб и акул

Допустим, продолжительность жизни для рыб и акул равна live_r $=8$, live_a$=12$ УЕВ. Каждая рыба (акула) через rogd_r$=2$ УЕВ (rogd_a$=3$ УЕВ) может родить одну рыбу (акулу), возраст которой равен 0. Это происходит, когда соседняя случайным образом выбранная ячейка не занята рыбой (акулой). Если ячейка занята, то рождения не происходит из-за перенаселенности. С каждой итерацией возраст всех рыб и акул увеличивается на 1 УЕВ. Когда возраст рыбы (акулы) превышает среднюю продолжительность жизни, то рыба (акула) умирает от старости. Рыбы питаются водорослями и червями, а акулы поедают рыб. Если для какой-то ячейки (i, j), занятой рыбой, количество соседних ячеек, занятых рыбами, больше 6, то эта рыба умирает от недостатка пищи: x[i,j]=0. Если акула не ела в течение t_gol$=5$, то она умирает от голода. За 1 УЕВ акула может переместиться в одну из 8 соседних ячеек (при этом ее координаты i и j изменяются на 1), либо остаться на месте. Если при перемещении акула попадает в ячейку, занятую рыбой, то она съедает ее. При этом переменная gol[i,j] обнуляется. На каждом временном шаге, когда акула не съела

рыбу, величина gol[i,j] увеличивается на 1. Рыба движется медленно и перемещается в соседнюю ячейку за 8 УЕВ.

В начале используемой программы [6] случайным образом задается расположение заданного количества рыб и акул и их возраст. На каждом временном шаге $t = 1, 2, \ldots$ проверяется или рассчитывается: 1) возраст каждой рыбы и акулы; 2) перемещение рыб и акул; 3) логические условия рождения рыб и акул; 4) условия смерти рыб от перенаселенности; 5) условия смерти рыб и акул от голода. При запуске компьютерной модели на экране появляется двумерная сетка (рис. 15.11), по которой перемещаются рыбы (зеленые клетки) и акулы (красные клетки). Белый цвет означает, что ячейка пуста. Для вывода изображения на экран используется процедура Draw. Ее можно закомментировать и вывести на экран график зависимости численностей рыб и акул от времени (рис. 15.12).

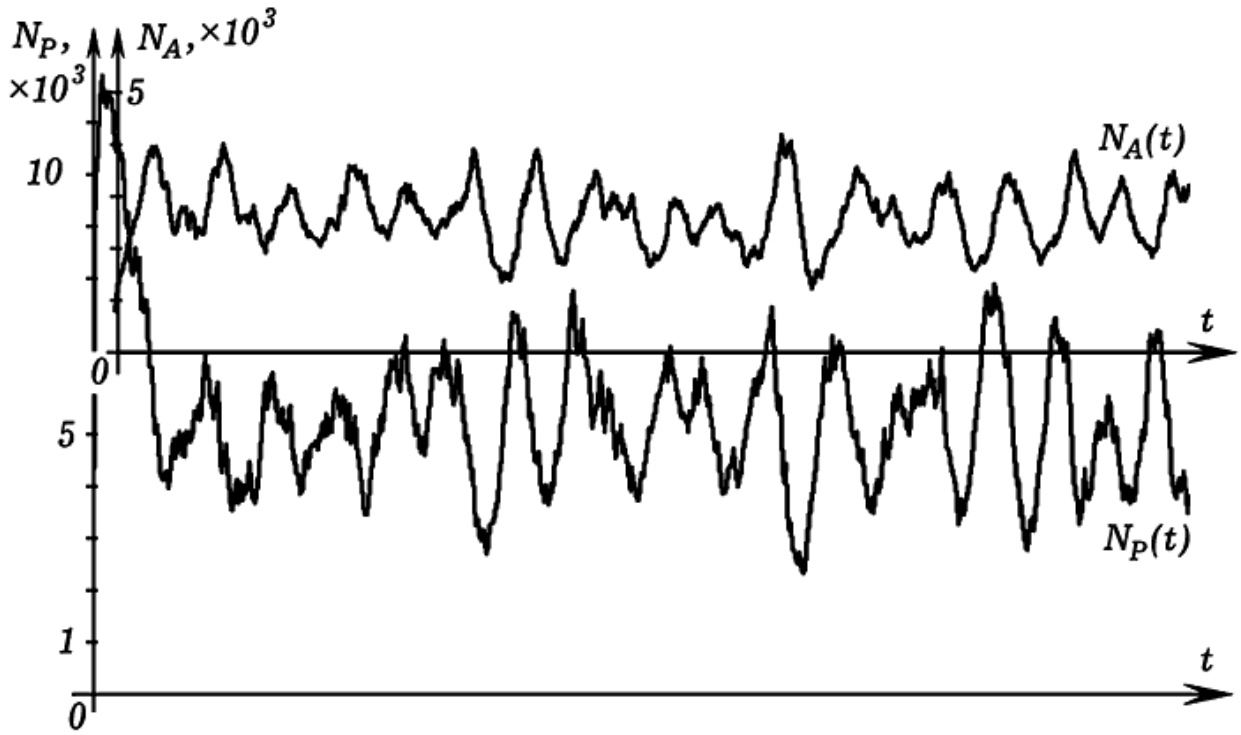

Рис. 15.12. Колебания численности рыб и акул с течением времени

Компьютерная модель позволяет исследовать сосуществование рыб и акул в одной акватории при различных параметрах этой системы. Она, например, позволяет промоделировать ситуацию, когда акулы, зайдя в

область i от 0 до 20, умирают из-за мелководья или охоты. Если уменьшать продолжительность жизни акул, длительность их существования без пищи, увеличивать период появления потомства, то можно добиться того, что акулы вымрут, а рыбы останутся. Можно подобрать такие параметры модели, при которых будут происходить колебания численности рыб и акул (рис. 15.12). Размножение рыб приводит к росту численности акул; так как эти процессы инерционны, система проскакивает положение равновесия. Когда акул становится слишком много, они поедают большую часть рыб, им не хватает пищи, поэтому часть акул умирает от голода. Уменьшение численности акул приводит к увеличению количества рыб, и все повторяется снова.

15.7. Поведение колонии муравьев

Теперь промоделируем поведение сообщества муравьев, которые ползают по земле в поисках пищи, а найдя ее, переносят пищу в муравейник. Известно, что сначала муравьи совершают случайные перемещения, а после нахождения пищи возвращаются в свою колонию, отмечая свой путь особыми веществами - **феромонами**. Когда другие муравьи натыкаются на феромоновые тропы, то они начинают двигаться по ним. Дойдя до источника питания, эти муравьи возвращаются в колонию, при этом они также "поливают" свой путь феромонами. С течением времени феромоны испаряются, вероятность выбора муравьем данного пути уменьшается. Более длинный путь требует большего времени для прохождения от муравейника до пищи и обратно, поэтому при тех же условиях феромоновый след испарится сильнее. По короткому пути прохождение будет быстрым, плотность феромонов высокой. Если какой-либо муравей нашел более короткий путь до источника пищи, то другие муравьи, вероятнее всего, пойдут по его следу и увеличат концентрацию феромонов,

что сделает его еще более привлекательным. В результате более длинный путь прервется, феромоновый след испарится.

В нашей модели [6] роль муравьев будут выполнять программные агенты, каждый из которых находится в некотором состоянии S. Разобьем горизонтальную поверхность на $L \times M$ квадратных ячеек размером 1×1. В каждой ячейке может находиться только один агент–муравей с координатами x_k и y_k ($k = 1, 2, ..., N$). Агенты могут быть в одном из трех состояний $S = 1$, 2 или 3. Находясь в состоянии $S = 1$ (поиск), агенты случайным образом перемещаются по поверхности земли, ища пищу. Наткнувшись на пищу, агент переходит в состояние $S = 2$ и возвращается в муравейник с пищей, оставляя за собой феромоновый след. После того, как агент достиг муравейника и отдал пищу, он переходит в состояние $S = 3$ и начинает двигаться по следу в сторону пищи. Если другой агент в состоянии поиска $S = 1$ наткнется на феромоновый след, то он также перейдет в состояние $S = 3$ и будет двигаться по следу от муравейника. Феромоны с течением времени испаряются, след прерывается. Если агент, находящийся в состоянии $S = 3$, движется по "тропинке" к пище и натыкается на разрыв следа, либо теряет его, то он переходит в состояние $S = 1$ (поиск пищи). Результаты моделирования представлены на рис. 15.13 и 15.14. Это сканы с экрана, из которых видно, как “муравьи” сначала находят “пищу” в одном месте, переносят ее в “муравейник”, затем находят “пищу” в другом месте, даже если она находится за препятствием.

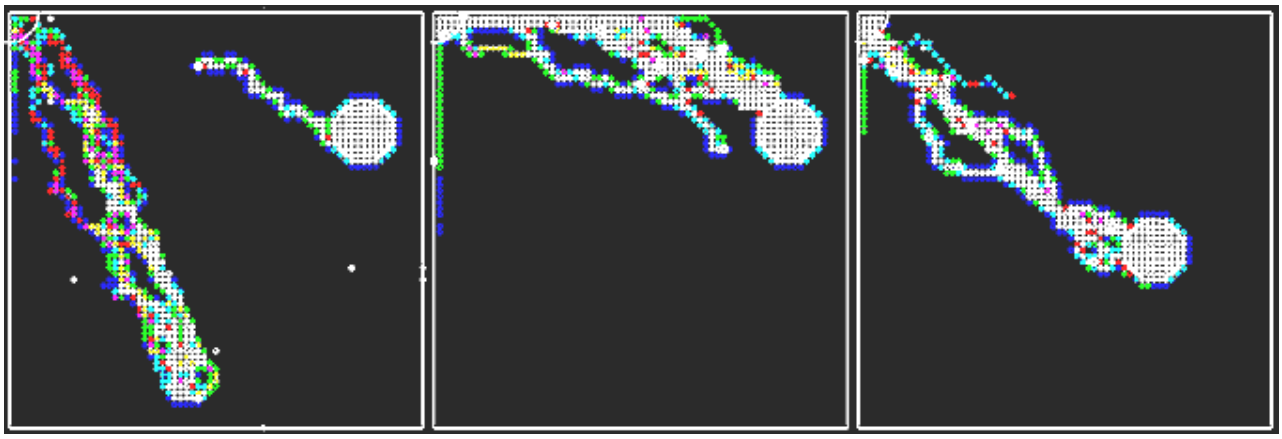

Рис. 15.13. Моделирование поведения колонии муравьев

Решение данной задачи позволило разработать **муравьиный алгоритм**, который является эффективным способом нахождения оптимальных маршрутов на графах. В его основе лежит имитация поведения колонии муравьев, каждый из которых моделируется программным агентом, функционирующим в соответствии с некоторыми простыми правилами. Установлено, что с ростом размерности решаемых задач оптимизации эффективность муравьиных алгоритмов повышается. Они применимы к динамически изменяющимся сетям и графам, так как за счет испарения феромона происходит перестройка оптимального маршрута. Использование мультиагентного подхода приводит к тому, что система не имеет централизованного управления, обмен локальной информацией осуществляется только между отдельными агентами посредством феромона.

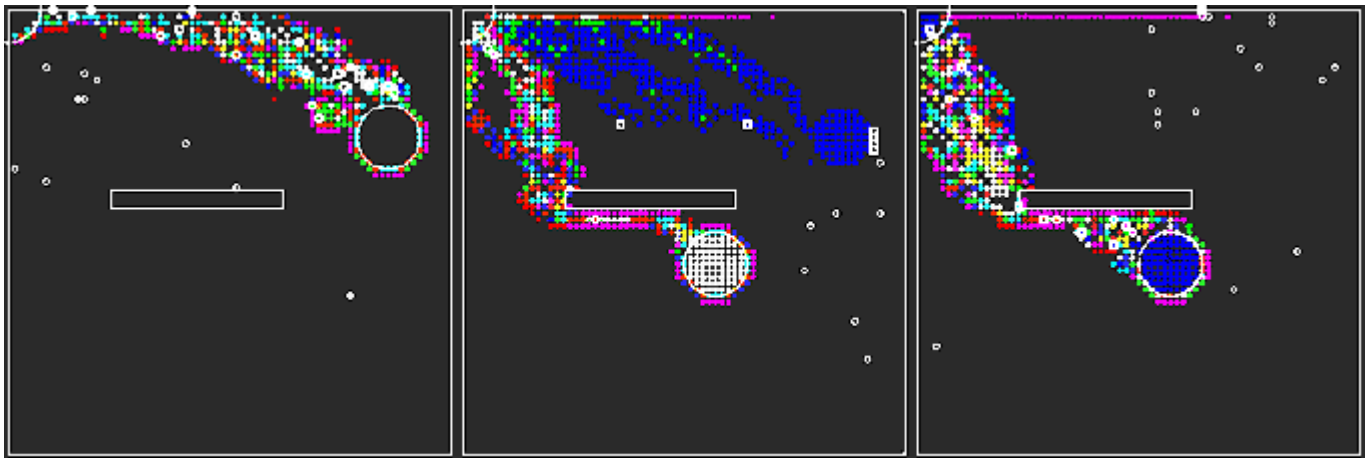

Рис. 15.14. Муравьи находят еду за препятствием

15.8. Моделирование эволюции

Эволюционные модели позволяют исследовать процесс биологической эволюции, состоящей в появлении новых видов растений и животных. Эволюция обусловлена **мутациями** и **естественным отбором**, заключающимся в том, что более приспособленные особи в среднем дают больше потомства, а наименее приспособленные погибают.

Дж. Холанд сумел интерпретировать принципы действия “биологических механизмов” и применить их для решения **проблемы адаптации искусственных систем**. Эволюция может рассматриваться как последовательные изменения хромосом, каждая из которых представляет собой цепочку генов. Моделями хромосом могут служить двоичные цепочки 101101…1 или цепочки из произвольных целых чисел [4; 15, с. 109-128]. При этом происходят следующие процессы: **наследование, мутация, отбор** и **кроссовер**. В результате мутации случайно меняются генетический код той или иной особи, а при скрещивании родители обмениваются частями хромосом с образованием двух потомков.

Моделирование состоит в следующем. Создается множество из n случайно сформированных **хромосом** $(x_{i,1}, x_{i,2}, \ldots, x_{i,m})$ $i = 1,2,\ldots,n$, соответствующее популяции живых объектов; задается **функция приспособленности** (или цены) $F(x_1, x_2, \ldots, x_m)$, позволяющая определить близость объекта к оптимальному состоянию. Она выбирается так, чтобы объекты, соответствующие целям эволюции, имели бы самую высокую цену. С ее помощью определяется приспособленность $F_i(x_j)$ $(i = 1,2,\ldots,n;\ \ j = 1, 2, \ldots, m)$ каждого объекта в популяции, и на следующем итерационном шаге $t+1$ создаются объекты следующего поколения.

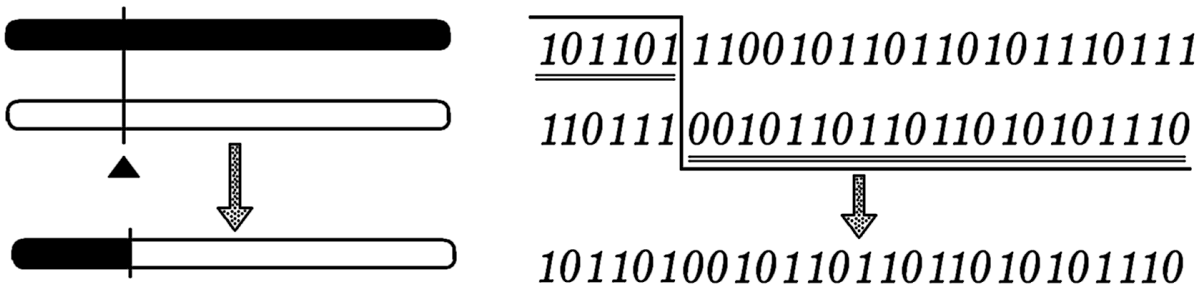

Рис. 15.15. Генетический оператор кроссовер

Новый дочерний объект обычно возникает в результате взаимодействия двух существующих объектов-родителей, которые отдают ему часть своих свойств в результате **скрещивания** или **кроссовера** (рис 15.15). При

скрещивании двух объектов происходит обмен частями родительских хромосом и появляется новая хромосома, которая включает в себя гены своих родителей и наследует их свойства. Программа генерирует случайное число, обозначающее номер гена, на котором происходит разрыв родительских хромосом. Чем выше приспособленность F_i, тем больше вероятность участия i-го объекта в создании следующего поколения, то есть имеет место естественный отбор, в результате которого побеждают более сильные хромосомы. При использовании **стратегии элитизма** лучшие особи (элита) переходят в следующее поколение без изменений.

Важным фактором эволюции являются мутации, при которых некоторые гены изменяются случайным образом. При отсутствии мутаций хромосомы следующего поколения содержали бы в себе только ту информацию, которая была заложена в хромосомы первого поколения. Случайные изменения генов приводят к появлению хромосом, не похожих на хромосомы первого поколения. В результате последовательности итераций, в ходе которых моделируются скрещивание, мутации и естественный отбор, появляются объекты нового поколения, у которых среднее значение приспособленности F_i существенно выше.

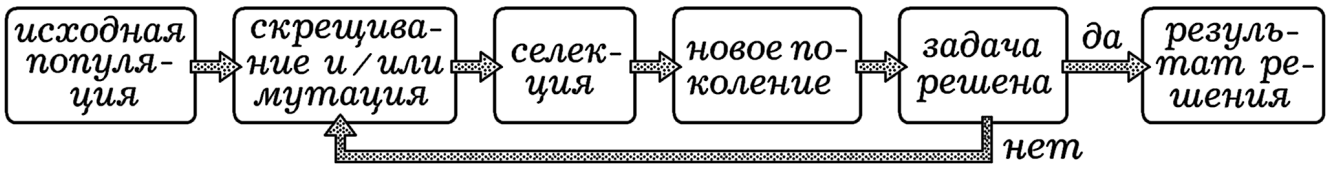

Рис. 15.16. Сущность генетического алгоритма

Для моделирования эволюции используется программа ПР-10. Хромосомы состоят из 64 генов (0 или 1), идеальная хромосома, имеющая максимальную приспособленность, может быть задана произвольно, например так: 32 нуля и 32 единицы. Приспособленность произвольной хромосомы тем выше, чем меньше она отличается от идеальной хромосомы. Первое поколение содержит 40 хромосом, сгенерированных случайным образом. Для каждой хромосомы определяется значение функции

приспособленности $F(x_j)$ и вычисляется среднее F_{cp} для всех хромосом данного поколения t. Программа содержит цикл по времени t, в котором на хромосомы воздействуют операторы мутации и кроссовера. **Оператор мутации** заменяет состояние произвольного гена в хромосоме на противоположное (0 на 1 и наоборот). В результате кроссовера хромосома потомка "собирается" из фрагментов двух родительских хромосом, благодаря чему особи обмениваются генетической информацией. Номер инвертируемого гена при мутации и место разрыва родительских хромосом при кроссовере определяются с помощью генератора случайных чисел.

Важным является то, что перечисленные выше генетические операторы никак не используют информацию о целевой функции или идеальном распределении генов. При переходе в следующее поколение "слабые" хромосомы с $F(x_j)$ ниже среднего F_{cp} отбрасываются (умирают, не принося потомства). "Сильные" (более приспособленные) хромосомы с $F(x_j) > F_{cp}$ повторяют самих себя, а также создают новые хромосомы путем кроссовера. Программа ПР-10 написана так, что на каждом шаге t количество особей в популяции остается неизменным.

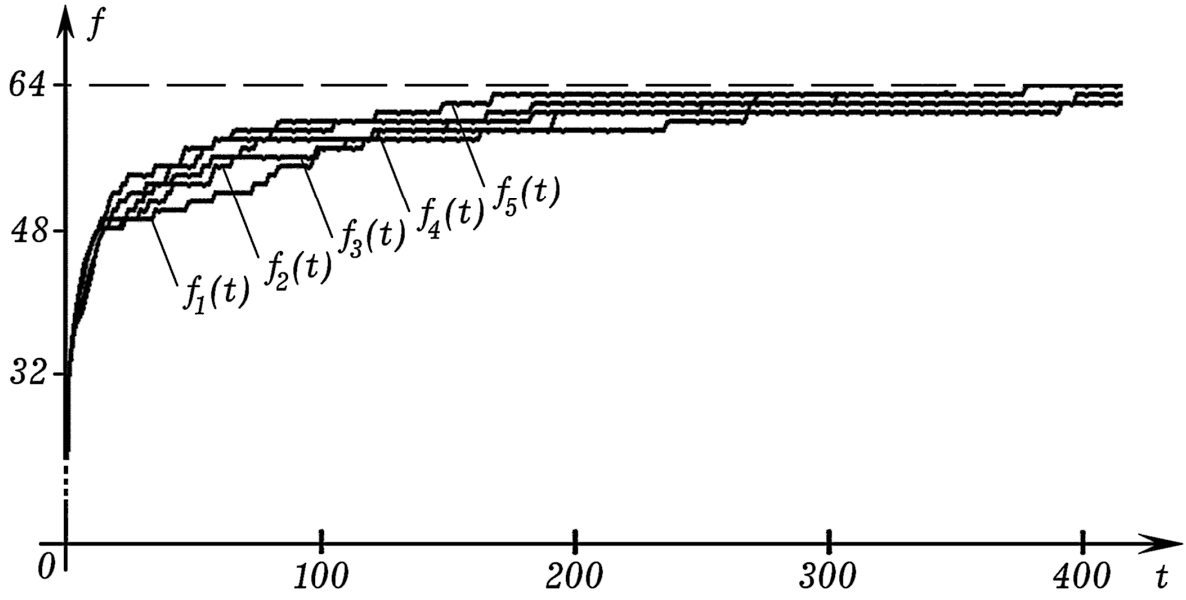

Рис. 15.17. Изменение целевой функции с течением времени

В результате $400-500$ итераций средняя приспособленность $f(x_i)$ по поколению увеличивается от 32 до примерно 64, что является максимумом. График зависимости приспособленности от номера поколения (итерации) t для пяти эволюционных процессов приведен на рис. 15.17. Видно, как средняя приспособленность вида монотонно возрастает, стремясь к своему максимальному значению. Все это моделирует естественный отбор и эволюцию живых организмов. Можно задать функцию приспособленности так, чтобы она имела несколько максимумов, и не накладывать ограничения на размер популяции. Тогда в результате действия генетических операторов произойдет "расщепление" популяции на несколько популяций, что соответствует появлению новых видов животных.

15.9. Генетические алгоритмы и эволюционные вычисления

Программы, моделирующие эволюцию, позволяют понять сущность метода генетических алгоритмов [4; 15, с. 115-128]. С их помощью осуществляется поиск оптимального (экстремального) значения функции. **Бинарные генетические алгоритмы** состоят в следующем: задают структуру объектов, функцию цены $F(x_i)$, являющуюся аналогом приспособляемости некоторой особи к конкретным условиям. Необходимо найти такую комбинацию аргументов $x_i = 1$ или 0 ($i = 1,2,..,n$), при которых функция цены $F(x_i)$ достигает максимума. Создают набор разных особей с различными генетическими кодами x_i, образующих популяцию, и запускают **эволюционный процесс**, в котором особи будут совершенствоваться, повышая функцию цены $F(x_i)$ от поколения к поколению. При этом кроме операторов мутации и кроссовера иногда применяют оператор инверсии, который изменяет порядок следования фрагментов хромосом у

потомка по сравнению с родителем. На каждом шаге t особи с низким значением $F(x_i)$ удаляются из популяции.

При решении некоторых задач используют генетические алгоритмы с непрерывными параметрами. Радклифф в 1991 году предложил алгоритм операции кроссовера (скрещивания) для непрерывных значений. Если i-й параметр в двух хромосомах родительской пары в t-м поколении равен G_1^t и G_2^t, то i-й признак двух дочерних хромосом может быть рассчитан по формуле [15, с. 127]:

$$G_1^{t+1} = \beta G_1^t + (1-\beta)G_2^t, \qquad G_2^{t+1} = (1-\beta)G_1^t + \beta G_2^t,$$

где β случайно выбирается из интервала $[0;1]$. Мутация моделируется путем прибавления к текущему значению G_i^t нового случайного значения, равномерно распределенного в некотором интервале $[-a;a]$.

Таким образом, генетическое программирование представляет собой эвристический метод программирования, помогающий создавать программы для задач, алгоритм решения которых неизвестен. В некоторых случаях популяцией является совокупность программ; для них задают функцию цены, эволюционные операторы и осуществляют последовательность итераций до тех пор, пока не будет получена программа, решающая совокупность задач одного класса.

Эффективным методом решения нестандартных задач являются **эволюционные вычисления**. Они используются, когда задача трудно формализуема, или необходимо получить грубый результат для принятия решения в режиме реального времени. Этот метод позволяет решить задачу коммивояжера, задачу оптимального распределения ресурсов, задачи, связанные с управлением движущимся объектом в реальном времени.

Эволюционные алгоритмы имеют высокий параллелизм, так как каждый объект популяции может обрабатываться отдельно, что при использовании многопроцессорных компьютеров позволяет найти решение за небольшое время. Генетические алгоритмы позволяют решать задачи нахождения глобального экстремума функции большого числа перемен-

ных, что имеет центральное значение в теории управления. Например, при обучении нейронных сетей роль параметров играют весовые коэффициенты ω_i и порог срабатывания Θ_j; размерность задачи, равная числу искомых параметров, велика. Оптимизационная функция цены $F(\omega_i, \Theta_j)$ тем больше, чем выше правильность работы нейронной сети. На каждом шаге в результате мутации и кроссовера изменяются значения ω_i и Θ_j; с помощью функции цены $F(\omega_i, \Theta_j)$ осуществляется отбор оптимальных наборов ω_i и Θ_j.

15.10. Основные направления кибернетического моделирования “искусственной жизни”

Методы компьютерного и кибернетического моделирования находят широкое применение при изучении биологических систем. К одним из перспективных направлений кибернетических исследований относятся изучение искусственной жизни и адаптивного поведения животных, возникшие еще в конце XX века [8]. Их основой являются работы М. Л. Цетлина [16], исследовавшего модели автоматов, приспосабливающихся к окружающей среде, и их коллективное поведение, а также работы М. М. Бонгарда, создавшего кибернетическую модель "Животное" - искусственный организм, способный адаптироваться и конкурировать с себе подобными.

Перечислим основные направления исследований в этой области [8, с. 41]: 1) динамика жизнеподобных структур в клеточных автоматах (К. Лангтон); 2) компьютерная модель искусственных организмов Поли-Мир, в которой каждый организм имеет нейронную сеть, обладает зрением, может двигаться, питаться, скрещиваться и бороться друг с другом (Л. Ягеря); 3) модели Тьерра и Авида, позволяющие изучить эволюцию

самовоспроизводящихся компьютерных программ (Т. Рэй, К. Адами); 4) модель Дж. Холланда, описывающая эволюцию организмов, которые размножаются, борются и торгуют друг с другом; 5) модель сосуществования и эволюции двух популяций "паразит-хозяин", включающая в себя популяцию программ, решающих некоторую конкретную проблему (задачу сортировки), и популяцию программ, которые постепенно усложняют проблему (Д. Хиллис); 6) эволюционные модели клеточных автоматов, которые конструируют КА, способный выполнять простые вычисления (М. Митчел); 7) модель "муравьиная ферма", имитирующая поведение муравьев при поиске пищи с учетом эволюции (Р. Коллинз, Д. Джефферсон); 8) модель эволюции когнитивного процесса (Дж. Холланд).

Ученые также обсуждают возможности **моделирования искусственной жизни**, в которой изучается развитие популяции программных агентов, живущих в компьютерах Интернета [8, с. 54]. Каждый агент имеет внутренний энергетический ресурс, выполняет действия, расходуя энергию, а израсходовав ее полностью, умирает. Агенты решают определенные задачи, за это контролирующая система их поощряет или наказывает. Они могут обмениваться информацией, передавая жизненный опыт и накапливая знания о “мире”. Агенты, скрещиваясь, рождают новых агентов, получающих энергию и хромосомы родителей. При этом каждый агент имеет две нейросети, управляющие их поведением: одна - для выбора следующего действия, другая - для решения предлагаемых задач. Геном агента содержит две хромосомы, в которых закодированы веса обеих нейросетей. В результате отбора, мутаций и кроссовера родительских хромосом происходит эволюция первой нейросети агентов. Вторая нейросеть развивается в результате индивидуального обучения и эволюции. У агентов имеются **мотивации** M_i - количественные параметры, характеризующие основные потребности к накоплению энергии и знаний.

П. К. Анохиным разработана **модель эволюционного возникновения целенаправленного адаптивного поведения**, в которой используется понятие мотивации [8, с. 53]. В ней изучается развитие популяции **анима-**

тов, имеющих потребность получения энергии и потребность размножения. Популяция обитает на поверхности (в двумерной среде), разбитой на ячейки. В среде может вырастать трава, являющаяся пищей для агентов. При уменьшении запаса энергии у анимата появляется мотивация найти пищу, периодически возникает мотивация размножения; величина мотивации выражается в числовом виде. Для управления поведением каждого анимата используется нейросеть, на некоторые входы которой подаются значения мотивации M_1 и M_2. Это приводит к целенаправленному поведению агентов. Параметры нейросети составляют геном агентов и изменяются в результате эволюции аниматов. Имитационное моделирование показывает, что наличие мотиваций приводит к эволюционному возникновению целенаправленности. При этом переход от аниматов без мотивации к управлению поведением, учитывающим мотивацию, можно рассматривать как появление нового уровня иерархии управления, связанного с переходом от простых рефлексов к сложным.

15.11. Другие примеры компьютерного моделирования биологических систем

Ученые Университета Ватерлоо с помощью компьютерного нейросимулятора Nengo промоделировали работу человеческого мозга [19]. **Виртуальная модель SPAUN** (Semantic Pointer Architecture Unified Network) состояла из $2{,}5$ миллионов нейронов и включала в себя цифровой глаз с разрешением 28×28 пикселей, через который осуществлялся ввод информации. Глаз различал цифры и знаки, некоторые знаки он воспринимал как команды и осуществлял соответствующие действия. Результаты своих "размышлений" система записывает "механической рукой". Нейробиологи запрограммировали работу симулятора мозга так, чтобы он обрабатывал информацию подобно человеческому мозгу. В частности, он не может сохранить в оперативной памяти длинную последовательность

знаков. Ученые изучают механизм самообучения и самопрограммирования системы. **Виртуальный мозг** SPAUN создавался исключительно в исследовательских целях, а не для решения практических задач, как это было в случае с моделью IBM Watson. Известны и другие модели, например, модели мозга Darwin 1 и Darwin 2.

Используя компьютерные модели, ученые изучают нейроэволюцию возникновения языка при адаптивном поведении, коллективное и социальное поведение, модели поискового адаптивного поведения, нейросетевые модели поведения роботов, создание искусственного интеллекта, адаптивное поведение на основе эволюционных и нейросетевых методов. Имитационное моделирование позволяет решать различные практические задачи, например, изучать динамику рыбного стада, поглощение кислорода листьями, исследовать различные системы организма (мозг, сердце, почки, печень, желудочно-кишечный тракт), продукционный процесс водных экосистем, живых систем (организмов и растений), биологические продукционные процессы физиологии растений, влияние человеческой цивилизации на экологию Земли в целом.

При изучении перечисленных выше проблем наряду с детерминированными методами используются **статистические методы Монте-Карло**. Они оказываются эффективными при недостатке информации об объекте, когда приходится рандомизировать параметры модели и внешние факторы, влияющие на изучаемые процессы. Применение метода Монте-Карло позволяет внести элемент случайности и получить лишь вероятностные оценки поведения био- или экосистемы.

Приложение к главе 15

В приложении представлены тексты программ, которые позволяют промоделировать рассмотренные выше биологические процессы. Они написаны в средах Borland Pascal 7.0 и Free Pascal 1.0.10.

Программа ПР-1

```
program Diskretn_model;
Uses crt, graph;
Var DV,MV,i,j: integer; a,x : real;
BEGIN DV:=Detect; InitGraph(DV,MV,'c:\bp\bgi');
Repeat a:=a+0.003; x:=0.01;
  For i:=1 to 1000 do begin x:=a*x*(1-x);
  If i>100 then putpixel(round(150*a),
                 450-round(x*350),15); end;
until a>3.99; Repeat until KeyPressed; CloseGraph;
END.
```

Программа ПР-2

```
program Neprerivn_model;
uses crt, graph;
var t,s,N,C,dN,dC: real; Gd,Gm: integer;
const r=0.03; K=100; dt=0.1;
BEGIN Gd:= Detect; InitGraph(Gd, Gm, 'c:\bp\bgi\');
Line(0,400,640,400); Line(20,0,20,480); N:=2;
Repeat  t:=t+dt;
  If (t>1000)and(t<1500) then s:=0.01 else s:=0.002;
  dN:=(r*N*(K-N)/K-s*N)*dt; N:=N+dN; t:=t+dt;
  circle(20+round(t/4),400-round(N*3),1);
until t>3000;
Repeat until keypressed; CloseGraph;
END.
```

Программа ПР-3

```
program Model_Hatchinsona;
uses crt, graph; const M=200;
var t,s,N,dN: real; i,Gd,Gm: integer;
x: array[0..M+1] of real;
const r=0.03; K=500; dt=0.2;
BEGIN Gd:= Detect; InitGraph(Gd,Gm,'c:\bp\bgi\');
Line(0,400,640,400); Line(20,0,20,480); N:=20;
Repeat t:=t+dt;
  If (t>3000)and(t<3100) then s:=0.005 else s:=0.002;
  N:=N+(r*x[1]*(K-x[1])/K-s*N)*dt;  t:=t+dt;
  For i:=1 to M do x[i]:=x[i+1]; x[M]:=N;
```

```
  circle(20+round(t/10),400-round(N/2),1);
  circle(20+round(t/10),400-round(s*3E+4),1);
until t>6000;
Repeat until keypressed; CloseGraph;
END.
```

Программа ПР-4

```
program Matrichn_model;
uses crt, graph; const M=30; h=12;
var Gd,Gm,i,j,k: integer; SN: real;
N,r,s: array[0..M]of real;
BEGIN Gd:=Detect; InitGraph(Gd,Gm,'c:\bp\bgi');
Randomize;
For i:=1 to M do N[i]:=1E+1*exp(-0.01*i);
Repeat inc(k);
For i:=1 to M do begin
  s[i]:=0.32/(1+exp(0.5*(20-i)))+0.01;
  If i>h then r[i]:=0.7*(1-exp(0.4*(h-i)))*
    exp(0.2*(h-i))/(1+SN/1E+4);
If r[i]<0 then r[i]:=0; end; N[0]:=0;
For i:=11 to M do N[0]:=N[0]+r[i]*N[i];
For i:=1 to M do begin N[i]:=N[i-1]*(1-s[i]);
If N[i]<0 then N[i]:=0; end; SN:=0;
For i:=0 to M do begin SN:=SN+N[i]; delay(1); end;
If k mod 1=0 then begin cleardevice;
For i:=1 to M do begin line(i*15,400-round(N[i]),
(i-1)*15,400-round(N[i-1])); line(i*15,400-round(r[i]*
1E+3),(i-1)*15,400-round(r[i-1]*1E+3)); line(i*15,400-
round(s[i]*500),(i-1)*15,400-round(s[i-1]*500)); end; end;
until KeyPressed; CloseGraph;
END.
```

Программа ПР-5

```
program Mezhvid_konkurenciya;
uses crt, graph;
const K1=200; K2=150; r1=0.02; r2=0.01; dt=0.05;
a12=0.8; a21=0.5;
var t,s,N1,N2,NN: real; i,Gd,Gm: integer;
BEGIN Gd:= Detect; InitGraph(Gd, Gm, 'c:\bp\bgi\');
```

```
Line(0,400,640,400); Line(20,0,20,400); Randomize;
line(20+round(K2/a21),400,20,400-K2);
line(20+K1,400,20,400-round(K1/a12));
Repeat N1:=random(300); N2:=random(300); t:=0;
Repeat t:=t+dt; {delay(1);} NN:=N1;
  N1:=N1+r1*N1*(K1-N1-a12*N2)*dt/K1;
  N2:=N2+r2*N2*(K2-N2-a21*N1)*dt/K2;
  circle(20+round(N1),400-round(N2),1);
 {circle(20+round(t),400-round(N1),1);}
until (KeyPressed)or(t>400);
until keypressed; CloseGraph;
END.
```

Программа ПР-6

```
program Volki_zaici;
uses crt, graph;
var t,N,C,dN,dC : real; Gd,Gm : integer;
const r=0.03; q=0.01; a=0.00001; f=0.15; dt=0.1;
BEGIN N:=5000; C:=1000;
Gd:=Detect; InitGraph(Gd,Gm,'c:\bp\bgi\');
line(0,400,640,400); line(20,0,20,480);
Repeat t:=t+dt; dN:=(r*N-a*N*C)*dt; {zaici}
  dC:=(-q*C+f*a*N*C)*dt; {volki}
  N:=N+dN; C:=C+dC; t:=t+dt;
  circle(20+round(t/5),400-round(N/100),1);
  circle(20+round(t/5),400-round(C/100),2);
 {circle(20+round(N/50),400-round(C/50),2);}
until t>3000; Repeat until keypressed; CloseGraph;
END.
```

Программа ПР-7

```
program Volki_zaici_odnomern;
{$N+}Uses crt, graph; const M=640; h=1; dt=0.01;
L=0; K=100; b=0.45; e=0.6; sN=0.2;
var kk,i,j,DV,MV : integer;
N,C: array[0..M+1] of single;
sC,t,R,DN,DC,ZZ : single;
Procedure Raschet;
begin DN:=40; DC:=3;
```

```
If (i>400) then begin DN:=10; DC:=1; end;
If (abs(i-M/2)<12) then begin DN:=5; DC:=0.8; end;
If i>260 then Sc:=0.2+0.1*sin(t/20) else sC:=0.2;
If C[i]>2 then ZZ:=b*(C[i]-2)*N[i]*dt else ZZ:=0;
N[i]:=N[i]+DN*(N[i+1]-2*N[i]+N[i-1])*dt/(h*h)-ZZ+
R*(N[i]-L)*(K-N[i])/K*dt-sN*N[i]*dt; If N[i]<L then
N[i]:=L; C[i]:=C[i]+DC*(C[i+1]-2*C[i]+C[i-1])*dt/(h*h)
+e*ZZ-sC*C[i]*dt; If C[i]<0 then C[i]:=0; end;
BEGIN R:=0.4;
For i:=1 to M-1 do begin N[i]:=20; end;
For i:=1 to 12 do begin C[i]:=0; end;
DV:=Detect; InitGraph(DV,MV,'c:\bp\bgi');
Repeat kk:=kk+1; t:=t+dt;
If round(t)=5 then C[100]:=10;
For i:=2 to M-1 do begin Raschet; end;
For i:=M-1 downto 2 do begin Raschet; end;
N[1]:=N[2]; C[1]:=C[2]; N[M]:=N[M-1]; C[M]:=C[M-1];
If kk mod 30=0 then begin cleardevice;
line(0,400,800,400);
For i:=2 to M-1 do begin circle(i,400-round(3*N[i]),2);
circle(i,400-round(3*C[i]),1); end; end;
until KeyPressed; CloseGraph;
END.
```

Программа ПР-8

```
program Avtomodeln_reshen;
{$N+}uses crt, graph; const M=800; h=1; dt=0.03;
L=0; K=100; b=0.002; e=0.4; cx=0.12; cy=0.045;
var  v,kk,i,j,DV,MV:integer; N,C:array[0..M+1] of
single; t,R,DN,DC,Z: single;
Procedure Raschet;
begin DC:=5; DN:=1.5;
If C[i]>0.1 then Z:=b*(C[i]-0.1)*N[i]*dt else Z:=0;
N[i]:=N[i]+DN*(N[i+1]-2*N[i]+N[i-1])*dt/(h*h)-Z+
R*N[i]*N[i]*(K-N[i])/K*dt-cx*N[i]*dt; C[i]:=C[i]+
DC*(C[i+1]-2*C[i]+C[i-1])*dt/(h*h)+e*Z-cy*C[i]*dt;
If C[i]<0 then C[i]:=0; If N[i]<0 then N[i]:=0; end;
BEGIN R:=0.01;
For i:=1 to 60 do begin N[i]:=60*exp(-0.07*i); end;
```

```
For i:=1 to 12 do begin C[i]:=0; end;
DV:=Detect; InitGraph(DV,MV,'c:\bp\bgi');
Repeat kk:=kk+1; t:=t+dt;
If round(t)=20 then C[10]:=80;
For i:=2 to M-1 do begin Raschet; end;
For i:=M-1 downto 2 do begin Raschet; end;
N[1]:=N[2]; C[1]:=C[2]; N[M]:=N[M-1]; C[M]:=C[M-1];
If kk mod 500=0 then begin line(0,400,800,400);
For i:=2 to M-1 do begin
line(i,400-round(3*N[i]),i-1,400-round(3*N[i-1]));
line(i,400-round(3*C[i]),i-1,400-round(3*C[i-1]));
end; end; until KeyPressed; CloseGraph;
END.
```

Программа ПР-9

```
program Volki_zaici_dvumern;
{$N+}uses crt, graph;
const L=70; M=100; h=1; dt=0.1;
R=1.5; K=200; b=0.5; e=0.4; Cs=0.4;
var kk,i,j,v,DV,MV: integer; DN,DC,q,ZZ,a: single;
N,C: array[0..L+1,0..M+1] of single;
Procedure Raschet;
begin DC:=1.3; DN:=3;
If (j>50) then begin DC:=0.6; DN:=1; end;
ZZ:=b*C[i,j]*N[i,j]*dt;
N[i,j]:=N[i,j]+DN*(N[i,j+1]-2*N[i,j]+N[i,j-1])*dt
/(h*h)+DN*(N[i+1,j]-2*N[i,j]+N[i-1,j])*dt/(h*h)-ZZ+
R*N[i,j]*(K-N[i,j])/K; If N[i,j]<0 then N[i,j]:=0;
C[i,j]:=C[i,j]+DC*(C[i,j+1]-2*C[i,j]+C[i,j-1])*dt/
(h*h)+DC*(C[i+1,j]-2*C[i,j]+C[i-1,j])*dt/(h*h)+e*ZZ-
Cs*C[i,j]; If C[i,j]<0 then C[i,j]:=0; end;
BEGIN For i:=2 to L-1 do For j:=2 to M-1 do begin
N[i,j]:=100; end;
DV:=Detect; InitGraph(DV,MV,'c:\bp\bgi');
Repeat kk:=kk+1; If abs(kk-10)<3 then C[20,20]:=10;
For i:=2 to L-1 do For j:=2 to M-1 do begin Raschet; end;
For i:=L-1 downto 2 do For j:=M-1 downto 2 do begin
                                                Raschet; end;
If kk mod 5=0 then For i:=2 to L-1 do For j:=2 to M-1 do
```

```
  begin setcolor(abs(round(N[i,j]/20)+1));
  rectangle(i*4+10,j*4,i*4+13,j*4+3);
  rectangle(i*4+11,j*4,i*4+12,j*4+2);
  If C[i,j]<25 then v:=0 else v:=abs(round(C[i,j]/20));
  setcolor(v+1); rectangle(i*4+340,j*4,i*4+343,j*4+3);
  rectangle(i*4+341,j*4,i*4+342,j*4+2); end;
until KeyPressed; CloseGraph;
END.
```

Программа ПР-10

```
program Genetich_algoritm;
uses crt, graph; const N=64; M=40;
var Gd,Gm,i,j,k,kk,sl,n1,n2: integer;
s,t: array[1..M]of string; ideal,r: string;
f: array[1..M]of word; V: text; SF,SFF: real;
Procedure Funk;
begin  SF:=0; For j:=1 to M do begin f[j]:=0; r:=s[j];
For i:=1 to N do If r[i]=ideal[i] then f[j]:=f[j]+1;
SF:=SF+f[j]/M; end; writeln(V,SF); end;
Procedure RND;
begin For j:=1 to M do begin r:=s[j];
i:=1+round((N)*random(100)/100);
If r[i]='0' then r[i]:='1' else r[i]:='0'; end; s[j]:=r;
end;
BEGIN Gd:=Detect; InitGraph(Gd,Gm,'c:\bp\bgi');
Assign(V,'C:\result.txt'); Rewrite(V); Randomize;
ideal:='1111111111111111111111111111111100000000000
0000000000000000000';
For j:=1 to M do For i:=1 to N do If random(1000)>500 then
s[j]:=s[j]+'1' else s[j]:=s[j]+'0';
Repeat inc(kk); Funk; i:=0;
For j:=1 to M do begin If f[j]>=SF then
   begin inc(i); t[i]:=s[j]; end; end;
For k:=i+1 to M do begin sl:=round(random(310)/10);
  n1:=1+round((i-1)*random(100)/100);
  n2:=1+round((i-1)*random(100)/100);
  If n1=0 then n1:=1; If n2=0 then n2:=1;
  t[k]:=copy(t[n1],1,sl)+copy(t[n2],sl+1,N); end;
For j:=1 to M do begin
```

```
  s[j]:=t[j]; writeln(V,s[j]); end;
IF kk mod 5=0 then begin RND; RND; RND; end; RND;
line(10+2*kk,450-round(SF*6),8+2*kk,450-round(SFF*6));
circle(10+2*kk,450-round(64*6),1); SFF:=SF;
until (KeyPressed)or(kk>800); Readkey; Close(V); CloseGraph;
END.
```

Список литературы

1. Авдин В. В. Математическое моделирование экосистем: учеб. пособие. Челябинск: Изд-во ЮУрГУ, 2004. 80 с.
2. Базыкин А. Д. Математическая биофизика взаимодействующих популяций. М.: Наука, 1985. 181 с.
3. Вольтерра В. Математическая теория борьбы за существование. М.: Наука, 1976. 286 с.
4. Вороновский Г. К., Махотило К. В., Петрашев С. Н. и др. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности. Х.: ОСНОВА, 1997. 112 с.
5. Гаузе Г. Ф. Борьба за существование. Институт компьютерных исследований, 1999. 160 с.
6. Майер Р. В. Задачи, алгоритмы, программы: электронное учеб. пособие. Глазов: Глазов. гос. пед. ин-т, 2012. URL: http://maier-rv.glazov.net
7. Могилев А. В., Пак Н. И., Хеннер Е. К. Информатика: учеб. пособие для студ. пед. вузов. М.: Издательский центр "Академия", 2003. 816 с.
8. Редько В. Г. Эволюционная кибернетика. Лекции по нейроинформатике. Ч. 1 // IV Всероссийская научно-техническая конференция "Нейро-информатика-2002". М.: МИФИ, 2002. 164 с.
9. Ризниченко Г. Ю., Рубин А. Б. Математические модели биологических продукционных процессов: учеб. пособие. М.: Изд-во МГУ, 1993. 302 с.

10. Робертс Ф. С. Дискретные математические модели с приложениями к социальным, биологическим и экологическим задачам. М.: Наука, Гл. ред. физ.–мат. лит., 1986. 496 с.
11. Рубанов В. Г., Филатов А. Г. Моделирование систем: учеб. пособие. Белгород: Изд–во БГТУ, 2006. 349 с.
12. Самарский А. А., Михайлов А. П. Математическое моделирование: Идеи. Методы. Примеры. М.: Физматлит, 2001. 320 с.
13. Смит Дж. М. Модели в экологии. М.: Мир, 1978. 184 с.
14. Тарасевич Ю. Ю. Математическое и компьютерное моделирование. Вводный курс: учеб. пособие. М.: Едиториал УРСС, 2004. 152 с.
15. Филатов А. Г. Имитационное моделирование процессов [Электронный ресурс]. URL: http://www.twirpx.com/file/26400/
16. Цетлин М. Л. Исследования по теории автоматов и моделированию биологических систем. М.: Наука, 1969. 316 с.
17. Юдин Е. Б., Задорожный В. Н. Агентная модель “хищник-жертва” на статистически однородных структурах // Информационные технологии и автоматизация управления: материалы науч.-практ. конф. ОмГТУ, 20-24 апреля 2009 года. Омск: Изд-во ОмГТУ, 2009. С. 205-207. URL: https://sites.google.com/site/yudinasu/internal-resume
18. Gillman M. An Introduction to Mathematical Models in Ecology and Evolution. Time and Space. Wiley-Blackwell, 2007. 157 p.
19. Лахман К. Модель мозга SPAUN: связывая сложную архитектуру со сложным поведением. URL: http://polit.ru/article/2012/12/06/SPAUN/

ВВЕРХ