МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ОБРАЗОВАНИЯ МНОГОКЛЕТОЧНЫХ СТРУКТУР

В. Э. Баскин

Процесс развития исходной клетки в организм имеет информационную основу в виде последовательности нуклеотидов цепей ДНК. Однако пока даже в общих чертах не ясно, как развертывается эта информация, приводя к построению весьма сложных клеточных структур, способных к исправлению повреждений и самовоспроизведению. Некоторые возможности для исследований в этой области дают математические модели типа конечных автоматов. Построение такой модели составляет предмет настоящей работы (см. В. Э. Баскин. — ДАН СССР, 1982, т. 262, № 2, с. 462—466).

Рассмотрим математический объект (развивающийся автомат), состоящий из двух взаимосвязанных частей — программы и фигуры. Фигурой будем называть множество замкнутых непересекающихся односвязных областей (клеток), которые соприкасаются друг с другом частями своих границ. Под программой подразумевается последовательность символов принятого алфавита, посредством которых обозначены проверяемые в клетках условия и предписания (инструкции) о проведении элементарных действий. Клетка имеет память, где хранится копия одинаковой для всех клеток программы. В оперативную часть этой памяти могут быть по указаниям программы записаны последовательности символов алфавита (коды). Программа не только строит структуру клеток, но и имеет с ней обратную связь, когда расположение клеток в структуре влияет на прохождение программы. Этим программа может контролировать нежелательные, например внесенные извне, изменения в структуре и предпринимать действия по их исправлению.

Для полного определения развивающегося автомата необходимо задать язык его программы. Ограничимся случаем плоской фигуры из прямоугольных клеток.

Обозначим стороны клетки символами (метками) A, B, C, D, причем взаимно дополнительные метки A и B или C и D поставим у противоположных сторон. Параллельной сторонам перегородкой клетка может быть разделена на две равные дочерние клетки, причем сторонам или частям сторон, переходящим от родительской клетки к дочерним, сохраним бывшие у них метки, а сторонам, входящим в перегородку, дадим метки, дополнительные к меткам противоположных сторон. Введем теперь следующие элементы языка.

К о д. Состоит из последовательности меток, например AB, ABX, AXX. Два кода считаются при сравнении совпадающими при различии в них лишь тех разрядов, в которых стоит особая метка X. Так, код AXX совпадает с ABX, но отличен от AB. Один из хранимых в клетке кодов, служащий для ее опознания программой и направления предназначенных ей инструкций, назовем адресом клетки, остальные — информационными признаками.

С п и с о к а д р е с о в. Обозначается заключенной в круглые скобки последовательностью адресов, разделенных запятыми. Пример: (AB, ACA, IXX, J).

И н с т р у к ц и я. Описывает выполняемые по указанию программы действия. Обозначается символом в фигурных скобках, например:

$\{!ABC\}$ — клетке дается новый адрес ABC (переадресация);

$\{.\delta\}$ — клетке дается некоторый информационный признак δ;

$\{\delta\ .\}$ — признак δ снимается;

$\{..\delta\}$ — клетке дается признак δ, оставляемый ее потомкам;

$\{:A\}$ — клетка с некоторым адресом R делится на две клетки перемычкой, параллельной стороне A. При этом примыкающая к этой стороне дочерняя клетка получает адрес RB, а ее новая сторона метку B. Другая дочерняя клетка получает адрес RA, а ее новая сторона — метку A. Инструкция $\{:C\}$ определена также, но вместо A надо поставить C, а вместо B — D. $\{!*\delta.\}$ — отыскивается любая клетка (реперная), которая примыкает стороной к рассматриваемой и не имеет признака δ. Сторона рассматриваемой клетки получает метку, дополнительную к метке контактирующей с ней стороны реперной клетки, а остальные метки ставятся в прежнем порядке их следования при обходе контура клетки. Если реперная примыкает к той стороне рассматриваемой, которая имеет метку A (B), то в адресе реперной клетки A заменяется единицей, B — нулем, остальные метки игнорируются, и к полученному так двоичному числу арифметически прибавляется единица. После этого цифры по тому же правилу обратно заменяются буквами и полученный код становится адресом рассматриваемой клетки. Если реперная клетка примыкает к той стороне рассматриваемой, которая имеет метку C (D), то делается аналогичная операция с заменой $C = 1$, $D = 0$.

П р и м е р: реперная клетка с адресом $BCAD$ примыкает к стороне рассматриваемой, имеющей метку A. Инструкция $\{!*\delta.\}$ даст рассматриваемой клетке адрес $ACBD$, ибо $01 + 1 = 10$ (числа двоичные).

У с л о в и е. Обозначается символом в квадратных скобках. Может выполняться или нет. Конкретные условия:

$[.\delta]$ — выполнено, если в клетке есть код δ;

$[\delta.]$ — выполнено, если в клетке нет кода δ;

$[*\ \delta.]$ — выполнено, если хоть в одной соседней клетке нет кода δ;

[** δ .] — выполнено, если во всех соседних нет кода δ;

[? * .] — выполнено, если при принятии каждой клетки, контактирующей с данной клеткой, за реперную и получении по адресу реперной нового адреса путем действий инструкции {! * δ .} окажется, что этот адрес отличен от имеющегося в данной клетке или контактирующие стороны данной и реперной клеток имеют не взаимно дополнительные метки.

Простейший программный сегмент состоит из последовательно расположенных списка адресов, условий и инструкций.

Такт работы программы состоит в следующем. В каждой клетке ее адрес сравнивается с адресами всех списков. В простейших сегментах, где адрес клетки совпадает с адресом из списка, происходит проверка условий. Затем в тех простейших сегментах, где все условия выполнены, реализуются инструкции.

Пример 1. Рассмотрим программу:

$S_1' \equiv (I)\,\{:A\};$

$S_2' \equiv (IA, IB)\,\{:C\};$

$S_5' \equiv (IAC, IAD, IBC)\,(!\,J)\,(IBD)\,(!\,L);$

$S_6 \equiv (J, JXX)\,\{:A\}\,(JX, JXXX)\,\{:C\};$

$S_7 \equiv (L, LXX)\,\{:A\}\,(LX, LXXX)\,\{:C\};$

$S_8 \equiv (JA, JAC, JACB, JACAD, JACBC, JBCAD, JADBC)\,\{.\pi\};$

$S_9 \equiv (LBD, LBDA, LACAC, LACBD, LBDAC, LBDBD)\,\{.\pi\};$

$S_{10} \equiv (JACBD)\,\{!\,I\}.$

При запуске ее в одной исходной клетке с адресом I (и метками A и C соответственно у правой и нижней сторон) произойдет следующее. В такте *1* сегмент S_1', в списке которого есть I, поделит клетку вертикалью, образовав две дочерние клетки с адресами IA, IB. В такте *2* сегмент S_2' поделит каждую из этих клеток горизонталью, образовав четыре клетки с адресами IAC, IAD, IBC, IBD (рис. 56). В такте *3* сегмент S_5' присвоит трем из них один и тот же адрес J, а четвертой — адрес L. Сегменты S_6, S_7 в тактах $4 \div 7$ разовьют каждую из стволовых клеток с адресом J и L в 16 клеток. Сегменты S_8, S_9 присвоят некоторым из клеток получившейся 64-клеточной фигуры признак π. Сегмент S_{10} приведет три клетки фигуры в состояние исходной клетки, так что после увеличения каждая из них может быть той же программой развита в тождественную предыдущей фигуру. Последовательные стадии развития фигуры с указанием адресов клеток и окраской клеток с признаком π в темный цвет показаны на рис. 56, 57,*а*. Клетки, позволяющие вновь воспроизвести фигуру, отмечены символом X.

Пример 2. Пусть в программе примера 1 ее начальная часть (от S_1' до S_5') заменена следующими сегментами:

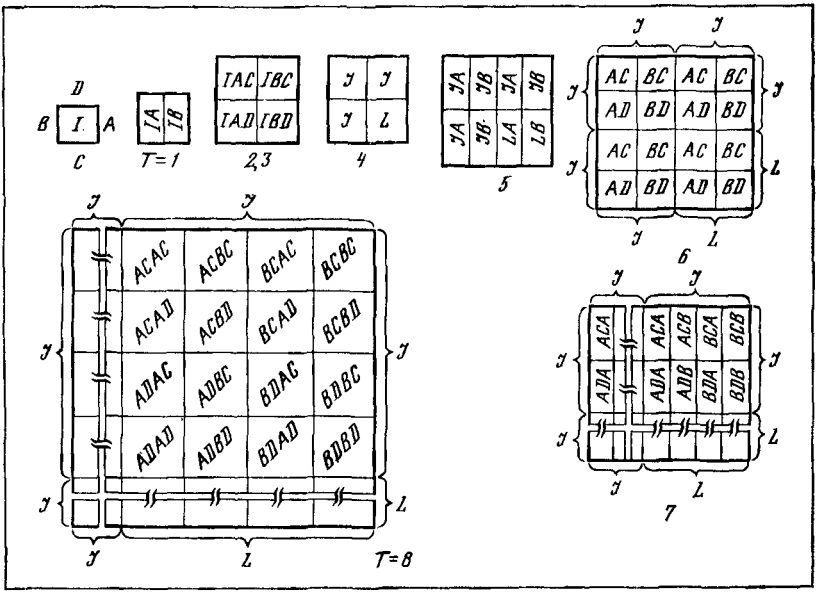

Рис. 56

Объяснение в тексте

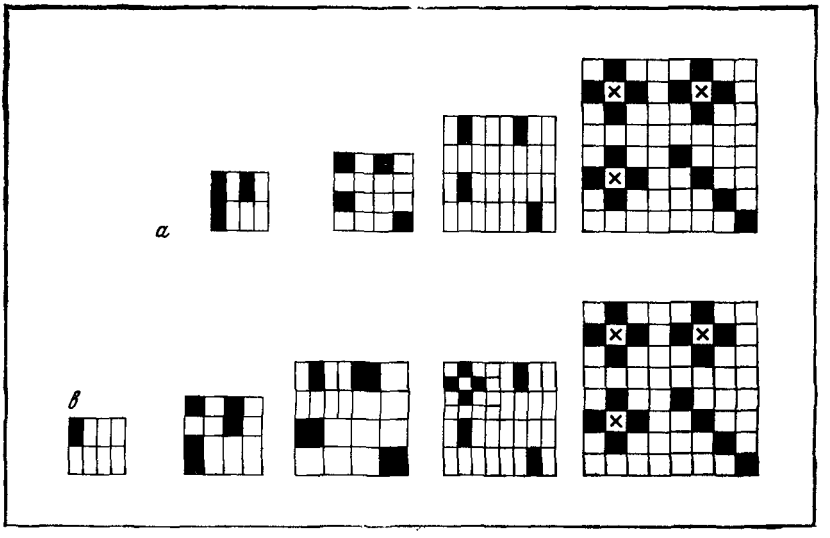

Рис. 57

Объяснение в тексте

$S_1 \equiv (I)\ \{: A\}\ \{.. \varepsilon\}$,

$S_2 \equiv (IA, IB)\ \{: C\}$,

$S_3 \equiv (IXX)\ [. \varepsilon]\ [? * .]\ \{. \delta\}\ (IXX)\ \{\varepsilon.\}$,

$S_4 \equiv (IXX)\ [. \delta]\ [* \delta .]\ \{! * \delta .\}\ \{\delta .\}$,

$S_5 \equiv (IAC, IAD, IBC)\ [\delta .]\ [\varepsilon .]\ [** \delta .]\ \{! J\}\ (IBD)\ [\delta .]\ [\varepsilon .]\ [** \delta .]\ \{! L\}$.

При запуске этой программы в тактах $T = 1, 2$ сегменты S_1 и S_2, как и в примере 1, построят четыре клетки с адресами IAC, IAD, IBC, IBD и установят в них код ε. В такте *3* сегмент S_3 снимет код ε, не поставив кода δ, ибо условие [? * .] для нормально расположенных четырех клеток не будет выполнено. Сегмент S_4 вследствие отсутствия в клетках кода δ работать не будет, а сегмент S_5 будет, ибо условия [δ .] [ε.] и [** δ .] выполнены. Поэтому развитие фигуры пойдет точно так же, как в примере 1 (см. рис. 57, *а*).

Рассмотрим теперь работу программы примера 2 в условиях эксперимента, когда одна из четырех клеток IAC, IAD, IBC, IBD заменена (например, вместо IBD стоит IBC), а остальные не тронуты. В этом случае условие [? * .] будет выполнено во всех клетках, кроме IAC, и сегмент S_3 установит в них код δ, а также снимет код ε во всех клетках. В такте *4* сегмент S_4, приняв клетку IAC за реперную, присвоит клеткам IAD, IBC эти же адреса и снимет в них код δ. В такте *5* сегмент S_4 таким же путем даст замененной клетке верный адрес IBD, а S_5 присвоит клетке IAC адрес J. Далее развитие пойдет подобно развитию в примере 1, но с опережением деления одних клеток по отношению к другим, так что та же финальная фигура будет получена, но иным путем (см. рис. 57, *в*). Пример моделирует известное в эмбриологии явление эквифинальности.

Введенные элементы языка позволяют строить программы восстановления нормального развития и при заменах групп клеток на более поздних стадиях. Для этого такие группы должны попасть в области с регулярным изменением адресов (показаны на рис. 56 для тактов *7* и *8*), где анализ и исправление нарушений регулярности могут производиться условием [? *.] и инструкцией {!*δ.}.

Рассмотрим соответствующие примеры. Нанесем финальной фигуре примера 1 повреждение путем замены группы из девяти клеток группой, взятой из другого места второй такой же фигуры, что приведет к конфигурации *в*. При восстановлении регулярности поля адресов переставленные клетки получат адреса, соответствующие их новому местоположению, и фигура *b* вернет свой нормальный вид *с*.

Заменим теперь сегмент S_8 примера 1 сегментом

$S_8' \equiv (JA, JAC, JACB, JADAC, JACAD, JACBD, JBCAD, JBDAC)\ \{.\pi\}$.

В этом случае финальная фигура несколько изменится и приобретет вид *d*. Заменим ту же, что и ранее, группу клеток фигу-

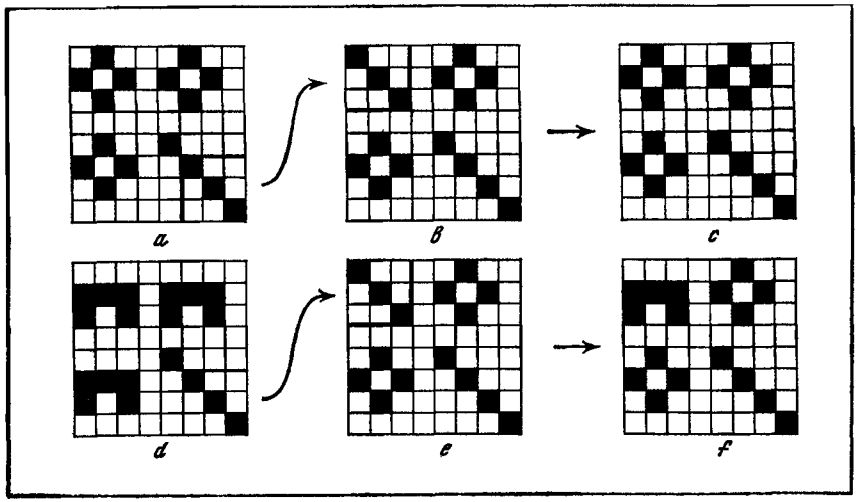

Рис. 58
Объяснение в тексте

ры *в*, взяв их теперь из фигуры *d*, что даст фигуру *e*. Так как переставленные клетки содержат программу с сегментом S'_8, а не S_8, после восстановления регулярных полей адресов из *e* будет получена фигура *f*. Переставленные клетки также развиваются в соответствии с новым местоположением, но теперь уже той фигуры, которой они принадлежали. Этот результат находится в полном соответствии с опытом Шпемана и Шотте, когда пересадка брюшного эпителия зародыша лягушки в область рта зародыша тритона приводила к развитию из этой ткани ротовых структур лягушки.

Как видим, составленные программы описывают развитие геометрических структур клеток, сходное с развитием организма. Из одной клетки развивается дифференцированная, способная к самовоспроизведению клеточная структура, которая контролирует правильность расположения своих частей на раннем этапе развития и может исправлять привнесенные извне нарушения. При этом моделируются такие целостные свойства организма, как эквифинальность и переопределение развития пересаженной эмбриональной ткани в соответствии с новым местоположением.

Как и ген, сегмент либо задает распределение по клеткам некоторого признака, например, окраски (S_8, S_9), либо приводит к развитию фрагмента из ряда клеток (S_7). Если дополнить программу новыми сегментами, то развитие автомата по новой программе будет частично повторять развитие по старой (отображение филогенеза в онтогенезе). Сегменты больших программ, развивающие стволовые клетки, будут содержать многократные повторы частей адресных кодов, совпадающих с адресом стволо-

вой клетки. Подобного рода обширные повторы до 10^2 нуклеотидов действительно существуют у высших организмов. При числе клеток организма порядка 10^{14} длина адреса доходит до 40 символов, так что размещение программы в цепях ДНК (порядка 10^{10} нуклеотидов) возможно лишь при придании многим клеткам одинаковых адресов. Это должно приводить к множеству подобных клеточных конфигураций, что действительно характерно для высших организмов на ряде иерархических уровней (рис. 58).

Строение программных сегментов соответствует модели оперона Георгиева, но с зоной промоторных перекрывающихся сайтов позитивного управления [см.: Ратнер, 1975]. Сложные инструкции представляются реализуемыми связанными оперонными системами. Продуцирование же адресных полимеров может осуществляться особым репликоном, разделяющиеся цепи которого удлиняются под контролем систем оперонов на единицы нуклеотидов или заменяются новыми. Возможность существования у высших организмов РНК-контролируемого синтеза ДНК и нарушение дифференцировки при его усилении в присутствии онкогенных вирусов говорят в пользу такого предположения.

О ПУТЯХ ЛОГИЧЕСКОГО АНАЛИЗА ИНДИВИДУАЛЬНОГО РАЗВИТИЯ

Л. И. Корочкин

После впечатляющих работ Эткина, Бастина и Нойеса, получивших с помощью чисто логических операций все основные физические параметры мира и основные принципы его организации, можно с уверенностью оценить возможности современной символической логики, как в высшей степени обнадеживающие.

Вполне естественно использовать эти возможности для разработки проблем биологии развития, но предварительно требуется выделение основных элементов этой науки, подлежащих логическому анализу. В настоящем сообщении предпринимается попытка выделить эти элементы и наметить некоторые пути их дальнейшего исследования.

Будем исходить из следующих трех постулатов: *во-первых*, всякий онтогенез осуществляется в соответствии с определенным планом развития, который охватывает подчиненную некоторому набору формул феноменологию преобразований и прогрессирующей гетерогенизации развивающейся системы. *Во-вторых*, эта гетерогенизация реализуется в соответствии с определенным планом регуляции развития, который намечается системами, детерминирующими последовательность феноменологических актов и правила их реализации, и который представляет собой в известной степени метаязык плана развития. Наконец, *в-третьих*, из анали-