Жизнь прекрасна и удивительна

Жизнь диктует свои законы…
Ч.Уэзеррелл[1]

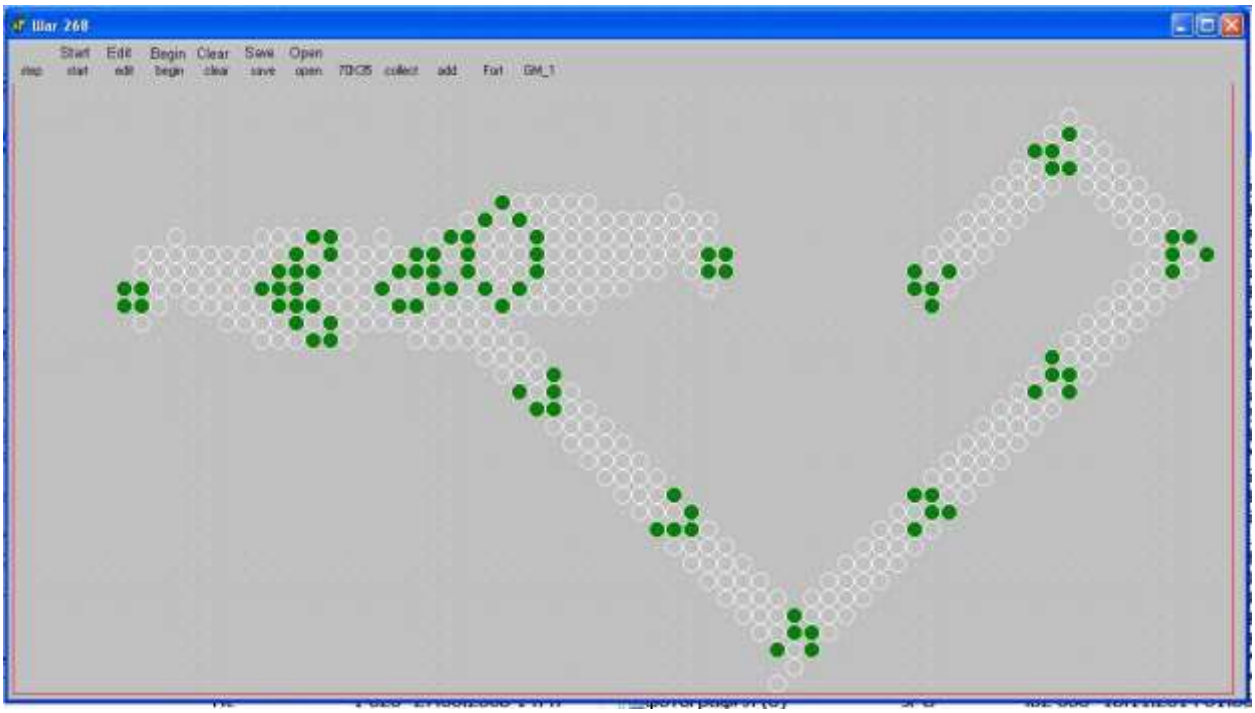

Рис 1. Планерное ружьё в действии

На рисунке 1 экран программы, моделирующей игру «Жизнь» (далее без кавычек): 268-е поколение *планерного(глайдерного) ружья* Госпера. На каждом шаге работы программы строится новое поколение (другая конфигурация клеток, представленных небольшими кружками). *Планер* – это конфигурация из пяти клеток, которая через каждые 4 поколения сдвигается на одну позицию по диагонали. В динамике это выглядит, как полет кувыркающегося планера. Ружье через каждые 30 шагов «выстреливает» новый планер.

Если читатели не слышали об этой игре, то изложим основные сведения. Игра была придумана английским математиком Джоном Конвеем в 1970 году и сразу привлекла внимание математиков и программистов.

Математики и раньше занимались подобными моделями, которые назывались *клеточными автоматами*. Подобно обычным автоматам, они имеют начальное состояние – конфигурацию заполненных и пустых клеток бесконечного, разбитого на прямоугольники (ячейки) поля. С точки зрения общей теории игра Жизнь является примером внешнего *тоталистического* клеточного автомата с набором значений ячеек {0,1}[8].

На каждом шаге работы автомата по определенным правилам меняется конфигурация. Судьба клетки, останется ли она заполненной или станет пустой, зависит от её окрестности, в которую входят соседние клеток. Для клеточного автомата Конвея правила очень просты: 1) клетка становится пустой (погибает), если рядом с ней либо меньше двух соседей, либо больше трех 2) пустая клетка становится заполненной (рождается), если рядом с ней ровно три «живых» клетки. Соседними считаются 8 клеток: соседние по горизонтали, вертикали или по диагонали.

Жизнь Конвея сразу завоевала популярность, и даже стал выходить ежеквартальный журнал, посвященный Жизни и родственным темам. Игра оказалась интересной и потому, что она породила много интересных задач. Сотни, а может быть и тысячи любители стали играть, находя интересные колонии, некоторые из которых получили имена изобретателей. Например, то же глайдерное ружье, найденное Биллом Госпером в 1970 году, явилось решением задачи о нахождении колонии,

растущей до бесконечности. Таких интересных объектов, получивших свои названия, сотни (в 25-й версии словаря Жизни [6] 861 статья!).

Советские читатели узнали об игре сначала из журнала «Квант» (№9, 1974 г.), а затем из книг Ч. Уэзереолла [1] и М. Гарднера [2]. Читателям «Кванта» предлагалось использовать разграфленный лист бумаги и фишки двух цветов. Можете представить, как было утомительно прослеживать историю колоний. Настоящий расцвет игра получила с появлением компьютерных программ.

Для программистов игра Жизнь интересна с точки зрения реализации, так как позволяет проявить способности, как в организации памяти, так и в эффективных алгоритмах. Еще в 1978 году такая задача была поставлена в книге Ч. Уэзерелл [1], где давались рекомендации по реализации и развитие темы.

В Интернете можно найти несколько реализаций Жизни [3-5], среди которых непревзойденным лидером является программа Golly (на русский это переводится как «обалдеть!»). Программа действительно имеет фантастические возможности.

Я (Л.Ч.) также не остался в стороне, и в 2001-2005 годах занимался разработкой своей программы, назвав её NewLife. В этой программе реализовано несколько возможностей, которых я не встречал в других реализациях.

Первое, что я считаю главным достижением, видно на первом рисунке: глайдеры отражаются от границ! Это позволяет моделировать «ограниченные» вселенные, где движущиеся частицы ведут себя естественным образом. Для этого пришлось модифицировать правила Жизни около границ. В обычной жизни это тоже бывает, например, поведение молекул в поверхностном слое воды.

(*Здесь пропущен абзац, поясняющий граничные правила. Опубликую позже – ноу-хау :*)

На рисунке 2 показана панель для задания параметров игры. Для каждой из границ задается одна из трех стратегий обработки: поглощение, отражение и склейка. Склейка позволяет задавать поле в виде цилиндра, когда противоположные границы «склеиваются» или тора, когда склеиваются и горизонтальные границы и вертикальные. В системе Golly размеры поля автоматически расширяются при необходимости. В нашей программе размеры поля задаются как параметры. Также задаются форма и цвет клетки, цвета границ и фона. Программа фиксирует стабильные (неизменяющиеся) и циклические (с повторяющимися конфигурациями) колонии. Время работы ограничивается либо числом шагов, либо временем в миллисекундах. Также регулируется скорость работы.

В классической жизни ячейки имеют два состояния: пустая ячейка или ячейка с живой клеткой. В программе NewLife можно отслеживать время жизни каждой клетки, задавая число поколений и их цвета.

В главном меню имеются пункты для редактирования в разных формах начальных колоний, в том числе копирование и перемещение групп клеток, вставки стандартных конфигурация. Также допускается возврат на начало, очистка поля и сохранение/восстановление конфигураций.

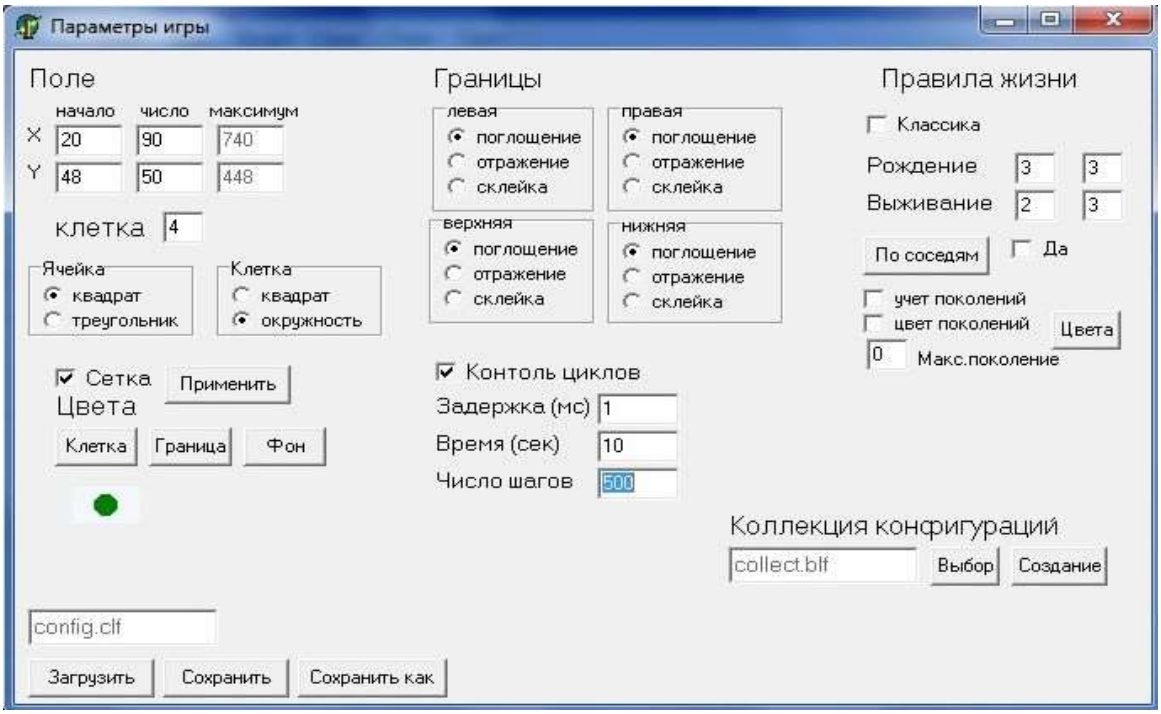

Рис 2. Параметры игры NewLife.

Интересной возможностью является задания правил игры (рис.3). В принципе существует бесконечно много правил Жизни, так как можно учитывать состояние не только ближайших соседей, учитывать время жизни соседних клеток, менять правила в процессе смены поколений и т.п.

Сколько существует вариантов правил для восьми соседей и как их описывать или именовать? В своей знаменитой книге «A New Kind of Science» Стивен Вольфрам разделил все клеточные автоматы на 4 класса и предложил их классификацию. Для простейших одномерных автоматов правила можно задать следующей таблицей (0 – пустая клетка, 1 – живая):

Текущее состояние трех клеток	111	110	101	100	011	010	001	000
Новое состояние средней клетки	0	0	0	1	1	1	1	0

В первых двух случаях клетка погибает, в третьем и пятом – рождается. Число во второй строке равно 30 в двоичной записи и правилу присваивается этот номер. Таким же образом нумеруются все 256 правил. Наиболее интересными оказались правила 30, 110, 161, 184.

Для двумерного случая, каковым является Жизнь Конвея, состояние клетки и ее соседей задается девятью бинарными цифрами, поэтому число правил 2^{512} ! Большинство из этих правил порождают неинтересные миры, в которых либо все колонии быстро погибают, либо заполняют все пространство. По-видимому, должен быть баланс между порождающими и убивающими конфигурациями в правилах. Проще определять правила, в которых учитывается только число соседей. Такие правила задаются формулами B3S23 (это сама Жизнь), где после B (*born*) перечисляются количества соседей, при котором клетка рождается, а после S (*save*) – выживают. В проекте Golly рассматриваются также игры с правилами (в скобках их названия): B36/S23 [HighLife], B345/S5 [LongLife], B3678/S34678 [Day & Night], B35678/S5678 [Diamoeba], B2 [Seeds], B234 [Serviettes or Persian Rug].

Разумным ограничением для других правил является принцип симметрии. Если начальную колонию повернуть на угол, кратный 90 градусов, или зеркально отобразить, то ее дальнейшая судьба не будет изменена. В программе NewLife удобно задавать правила и по количеству, и по принципу симметрии, и для произвольного случая. На экране определения правил изображаются все возможные комбинации соседей. Черной рамочкой обведены конфигурации, когда центральная клетка погибает, а желтой (на рисунке может быть не видна) – когда клетка рождается. Если в

комбинации в центре кружок, это комбинация выживания. Выбирая режим задания правил, по клику на комбинации одновременно отмечаются несколько комбинаций (с тем же числом соседей или симметричных). Правила можно запомнить под определенным именем и загрузить для проведения игры.

Упражнение. Определите, по какому принципу на панели размещены конфигурации и подсчитайте количество правил по числу соседей и симметричных.

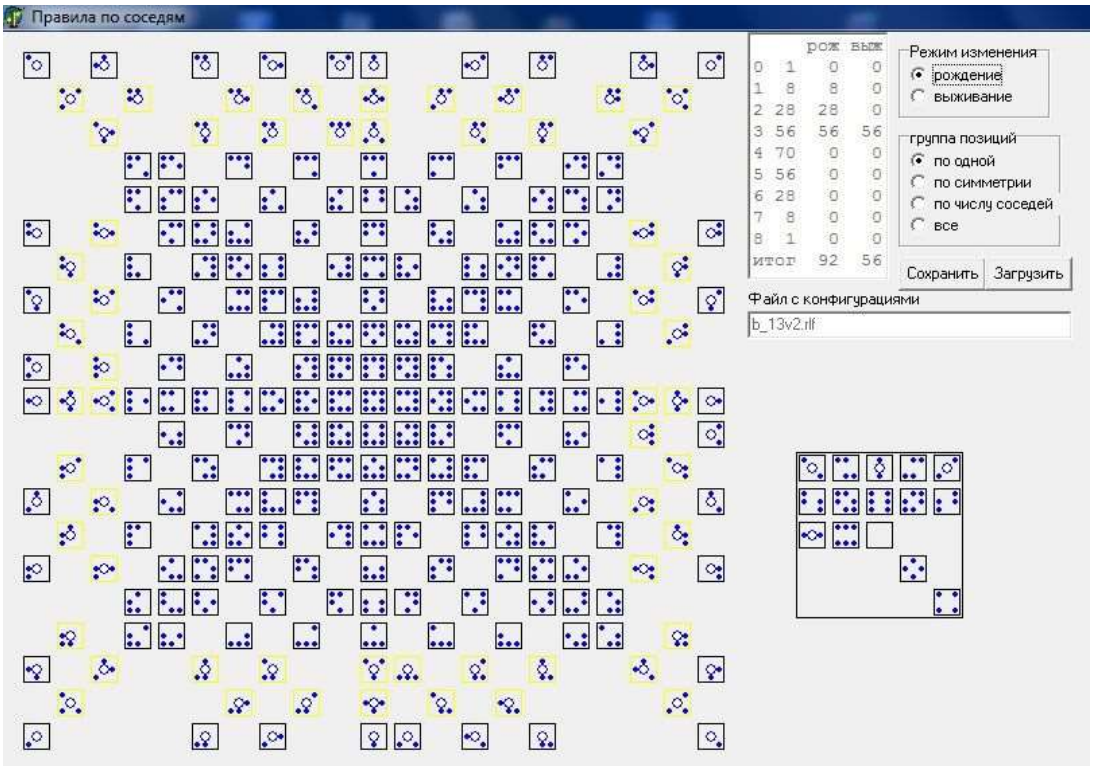

Рис 2. Задание правил игры NewLife.

Еще одно достоинство программы – встроенный интерпретатор Форта (см. раздел…). Это облегчает исследования и поиск интересных ситуаций. Основные дополнительные слова Форта следующие:

```
B      ( X1 Y1 ---> )      рождение клетки
D      ( X1 Y1 ---> )      смерть клетки
B?     ( X Y ---> stat )   есть ли жизнь в ячейке?
Start  ( ---> )            запустить жизнь
step   ( ---> )            выполнить один шаг
0XY    ( X Y  --->)        задать начало координат
stat   ( ---> nStep )      определить предел жизни
clear  ( ---> )            очистить все
```

Кроме того имеются слова, позволяющие протоколировать исследования. Пример исследования: бомбардировка глайдерами какой-либо колонии с разных позиций. На Форте программируется перебор начальных позиций, запускается игра и фиксируются результат. Вот простой пример:

```
: GL !2 !1   ( X Y ---> определения глайдера в заданной позиции )
@1     @2     B
@1 1 + @2     B
@1 2 + @2     B
@1     @2 1 + B
@1 1 + @2 2 + B ;
: mg clear    ( запуск глайдера с фиксацией результата )
    0 0 MIG over xg ! dup yg ! GL start
result ;
var k  var x
```

```
: Gxy  5 k !    ( перебор пяти вариантов )
BEGIN k @ 2 div 2 - x !
    x @ k @ 2 mod 5 + + x @
    DO I  k @ I - mg  LOOP
    k @ 1 + dup k !
    6 >
UNTIL ;
80 85 0XY Gxy    ( установка начала и запуск программы Gxy)
```

С помощью этого скриптового языка можно было бы определять и более сложные правила, учитывающие, например, несколько состояний клеток или большее число соседей (это пока не реализовано).

Вернемся к общим проблемам клеточных автоматов вообще и Жизни в частности. Несмотря на простоту клеточных автоматов, они породили целое направление в науке (например, книга Вольфрама). Исследователи даже стали задаваться вопросом, а не является ли наша вселенная клеточным автоматом. В физике возникло направление, названное *цифровой физикой*.

Одна из давно поставленных задач: могут ли клеточные автоматы производить реальные вычисления. Оказалось, что да. 11-го ноября 2002 года Пауль Чепмен построил образец Жизни, эквивалентной машине Тьюринга. Первая версия образца была большой (268096 живых клеток на площади 4558 x 21469 ячеек). Таким образом, в игре Жизнь можно выполнить любой алгоритм, который можно реализовать на современном компьютере. Было также доказано (в 1998 году), что это может делать одномерный автомат, работающий по правилу 110.

Ниже приведена программа на Питоне, моделирующая автомат 110. Удивительно, что эта программа может решить любую задачу! Надо только правильно задать начальную конфигурацию и число итераций.

```
R={'111':'0','110':'1','101':'1','100':'0','011':'1','010':'1','001':'1','000':'0'}
s = "1"; n = 5  # начальная конфигурация и число поколений
for i in range(n):
  s1 = "0"+s+"0"; s = ""
  for i in range(len(s1)-2): s += R[s1[i:i+3]]
  print(s)
```

Упражнение. Сделай свою жизнь!

1. Уэзерелл. Этюды для программистов : Пер. с англ. – М. : Мир, 1982.
2. Гарднер М. Крестики-нолики. : Пер. с англ. -- М. : Мир, 1988.
3. Проект Golly. http://golly.sourceforge.net/.
4. Программа для моделирования игры «Жизнь». http://life.written.ru/ .
5. Наука и «Жизнь». Ресурсы о «Жизнь». Программа Fam Life. http://www.famlife.narod.ru/.
6. Словарь Жизни. Русский перевод (оригинал в программе Golly) http://beluch.narod.ru/lifelex/lexr.htm .
7. Программа NewLife. Подборка ссылок о «Жизни». http://www.chernyshov.com/Life/.
8. Клеточные автоматы. https://ru.wikipedia.org/wiki/Клеточный_автомат

необработанное дополнение ….

клеточный автомат, придуманный Конвеем и известный у нас под названием "игра 'Жизнь', может моделировать игру "машина Тьюринга". Это означает, что в этой клеточной структуре можно

запрограммировать и выполнить любое вычисление, которое может выполнять машина Тьюринга. Мне встречалась как модель машины Тьюринга, так и вычисление последовательности простых чисел, не моделирующее, однако, машину Тьюринга.

в "Жизни" можно создать конфигурации, которые моделируют саму игру "Жизнь" на более высоком уровне. Вот в этой программе я нашёл паттерн http://www.mirwoj.opus.chelm.pl/ca/

Интересно в нашем мире такое возможно?

А вот как определить насколько сложные структуры могут образоваться в "Жизни"? Никто не читал вот эту книгу? http://www.ilachinski.com/ca_book.htm

В игре "Жизнь" может существовать не только машина Тьюринга (универсальный компьютер), но и Универсальный конструктор (УК)!

Из словаря "Жизни":

"универсальный конструктор Конфигурация, которая является способной к строительству почти любого образца, поддающегося глайдерному синтезу. Это определение немного неопределенно. Точное определение кажется невозможным, потому что не было доказано, что конструируемы все возможные глайдерные флоты. В любом случае, универсальный конструктор, чтобы считаться таковым, должен быть способен строить самого себя. Схема доказательства Конуэя существования такого образца может быть найдена в Winning Ways, а также в Рекурсивной Вселенной. Универсальный конструктор, разработанный этим способом, может также функционировать как универсальный деструктор - конфигурация, которая может удалить почти любой образец, который может быть удален глайдерами.

В мае 2004 года Пол Чепмен и Дейв Грин создали прототип программируемого универсального конструктора. Он способен строить объекты путем последовательного глайдерного строительства. Вероятно, он мог бы быть запрограммирован на строительство самого себя, но необходимая для этого программа будет очень большой; кроме того, для этого был бы необходим дополнительный механизм, копирующий программу.

Универсальный конструктор наиболее полезен в комбинации с универсальным компьютером, который может быть запрограммирован для управления конструктором, чтобы произвести глайдерный синтез желательного образца. В дальнейшем я буду предполагать, что универсальный конструктор всегда включает в себя этот компьютер.

Существование универсального конструктора/деструктора имеет множество теоретических последствий. Например, конструктор можно запрограммировать на изготовление собственных копий. Это – репликатор.

Конструктор может даже быть запрограммирован, чтобы изготовить единственную копию себя, передвинутого на некоторое расстояние, а затем удалить себя. Это был бы (очень большой, с очень высоким периодом) космический корабль. Возможно любое смещение (за исключением того, что оно не должно быть слишком маленьким), так, чтобы космический корабль мог путешествовать в любом направлении. Он может также двигаться со скоростью более медленной, чем любая данная скорость, поскольку перед копированием мы можем запрограммировать выполнение некоторой задачи бесполезной траты времени (типа повторяемого строительства и удаления блока). Конечно, мы можем также захотеть, чтобы он оставлял некоторые развалины, делая таким образом паровоз.

Также возможно показать, что существование универсального конструктора подразумевает существование устойчивых отражателей. Это доказательство, однако, не настолько легко и не имеет большого значения теперь, когда явные примеры таких отражателей известны."

Существование универсального конструктора/деструктора имеет множество теоретических последствий.

Ведь здесь перечислены не все последствия?

Вот какая мысль пришла ко мне в связи с Универсальным Конструктором:

В настоящей физической реальности кроме вещества есть поля, которые могут переносить информацию, а что если к "Жизни" добавить ещё некоторые правила какого-нибудь ещё клеточного автомата, способного моделировать такие поля, по-моему такие автоматы есть ( ) , тогда эти самые Универсальные конструкторы (если взять их сразу несколько штук) смогли бы обмениваться друг с другом информацией через это поле. Или ещё мысль: говорится, что УК может быть "космическим кораблём", который может двигаться в любую сторону с любой скоростью, тогда: несколько таких УК могут моделировать классическую динамику движущихся и сталкивающихся друг с другом тел.

http://www.cs.sjsu.edu/faculty/rucker/boppers.htm

http://www.ventrella.com/Darwin/darwin.html

http://dxdy.ru/post16976.html