ОПЕРАТОР МУТАЦИИ В ЕВОЛЮЦИОННОЙ ТЕХНОЛОГИИ РЕШЕНИЯ ЗАДАЧИ СОСТАВЛЕНИЯ РАСПИСАНИЙ

Елена Сипко

Аннотация: *Статья посвящена исследованию используемого в генетических алгоритмах и задачах оптимизации оператора мутации. Рассмотрены проблемы, возникающие при составлении расписаний учебных занятий. В качестве примера использования случайной мутации приведен код фрагмента программы, применяемый при составлении расписания учебных занятий студентов Черкасского государственного технологического университета.*

Ключевые слова: *расписание, оператор мутации, популяция, кроссовер, инверсия.*

ACM Classification Keywords *I.2 Artificial Intelligence, H.4.1 Office Automation*

Вступление

Генетические алгоритмы (ГА) ориентированы на применение операторов, которые изменяют генотип, или двоичное представление индивида. В зависимости от результатов работы операторов соответственно изменяется и фенотип. Одним из таких операторов является мутация. Мутация в ГА имеет полную аналогию в природе, когда происходит замена одного гена другим в ДНК под воздействием, например, радиоактивности [Воронин, 2004]. Считается, что мутация – это причина эволюции, и благодаря ей появляются новые виды. В ГА мутация способствует защите от преждевременной сходимости. Реализуется она путем выбора случайного гена или группы генов и их изменения по определенным правилам.

Оператор мутации моделирует естественный процесс мутации. Его применение в ГА обусловлено следующими рассуждениями. Начальная популяция, какой бы большой она не была, охватывает ограниченную область пространства поиска. Оператор скрещивания, безусловно, расширяет эту область, но все же до определенной степени, поскольку использует ограниченный набор значений, заданных начальной популяцией [Корнеев, 2003]. Внесение случайных изменений в генотип или фенотип индивида дает возможность отменить это ограничение и иногда значительно сократить время поиска и улучшить качество результата.

В бинарном случае оператор мутации заключается в инвертировании символов в случайно выбранных позициях. В случае работы с конечным алфавитом случайно выбранный символ заменяется на любой, отличный от него. Это может быть перестановка двух элементов в строке, замена элемента строки значением элемента из другой строки, в случае битовой строки может применяться инверсия одного из битов и др. Оператор мутации применяется не к каждому гену представителей в популяции. Обычно сначала задается вероятность мутации P_m и некоторый алгоритм осуществления мутации, например, такой [Кисляков, 2001]. Сначала нумеруются случайным образом все представители начальной популяции. Далее, начиная с первого гена первой хромосомы, пересматриваются элементы всей популяции, при этом выбираются случайные числа из интервала [0, 1). Если на некотором шаге выбранное число оказывается меньше вероятности P_m, то текущий ген подвергается мутации.

Конкретное значение P_m зависит от решаемой задачи, но чаще всего вероятность мутации имеет достаточно малое значение. Так, в литературе встречаются оценки $P_m \approx 0{,}001$ в работе [De Jong, 1975], $P_m \in [0{,}005, 0{,}01]$ в работе [Schaffler, 1989], $P_m \approx 0{,}01$ в работе [Grefenstette, 1986].

Выбор метода осуществления мутации

Обычно при реализации ГА к скрещиваемым индивидам сначала применяют оператор скрещивания, а потом оператор мутации, хотя возможны и другие варианты. Существует мнение, что оператор мутации является основным поисковым оператором и известны алгоритмы, не использующие других операторов (кроссинговер, инверсия и т.д.), кроме мутации.

Исследования показали, что в простых задачах, используя ГА с мутацией (и без кроссовера), находят решение быстрее. Также для такого метода требуется меньший размер популяции. Если имеют место сложные многоэкстремальные функции, то лучше использовать ГА с кроссовером, поскольку этот метод более надежен, хотя и требует большего размера популяции.

Используя теорему шаблонов, можно утверждать [Исаев, 2003], что мутация только вредит росту количества представителей приемлемых шаблонов, поскольку лишний раз их разрушает. Однако мутация просто необходима для ГА с малым размером популяции. Сущность этого утверждения в том, что для малочисленных популяций свойственна преждевременная сходимость (premature convergence). Это та ситуация, когда в некоторых позициях все индивиды имеют один и тот же бит, но такой набор битов не соответствует глобальному экстремуму. При этом кроссовер практически не изменяет популяции, т.к. все индивиды почти одинаковы. В этом случае мутация способна инвертировать «застрявший» бит у одного из индивидов и вновь расширить пространство поиска.

В качестве оператора мутации наибольшее распространение получили случайная и неравномерная мутация Михалевича [Michalewicz, 1995].

При случайной мутации ген, подлежащий изменению, принимает случайное значение из интервала своей области определения. В неравномерной мутации значение гена после оператора мутации рассчитывается по формуле:

$$c_i^* = \begin{cases} c_i + \delta(t, b_i - c_i) & \textit{при} \quad \chi = 0, \\ c_i - \delta(t, c_i - a_i) & \textit{при} \quad \chi = 1, \end{cases}$$

$$\delta(t, y) = y\left(1 - r^{\left(1 - \frac{t}{\varepsilon_{\max}}\right)^b}\right),$$

где χ — целое случайное число, принимающее значение 0 или 1; $r \in [0,1]$ — случайное вещественное число; ε_{max} — максимальное количество эпох алгоритма; b — параметр, задаваемый исследователем.

Кроме того, если на протяжении достаточно большого числа поколений не происходит увеличения приспособленности, то применяются «малая» и «большая» мутации поколения. При «малой» мутации поколения ко всем особям, кроме 10% лучших, применяется оператор мутации. При «большой» мутации каждая особь либо мутирует, либо заменяется на случайно сгенерированную.

Число поколений до реализации «малой» и «большой» мутации во время работы алгоритма является постоянным. Также оператор мутации можно применять только в том случае, если к данной паре родительских особей не был применен оператор скрещивания. Можно выбирать некоторое количество точек в

хромосоме для инверсии, причем их число также может быть случайным. Допускается инвертирование сразу некоторой группы подряд идущих точек.

При увеличении вероятности мутации и при уменьшении влияния результатов отбора (например, за счет использования других стратегий отбора) размножение представителей приспособленных особей замедляется, но зато происходит интенсивный поиск других особей. Обратно, уменьшение вероятности мутации и увеличение влияния отбора ведет к интенсивному использованию найденных хороших особей, но тогда меньше внимания уделяется поиску новых [Holland, 1994].

Случайная мутация в задаче составления расписания учебных занятий

В задаче составления расписаний решением является оптимальный в некотором смысле вариант расписания. Оператор мутации предназначен для модификации одного из таких вариантов. Поскольку начальные расписания генерируются случайным образом, то такая генерация не обязательно будет обеспечивать поиск по всему пространству решений. Оператор мутации предназначен именно для того, чтобы внести в популяцию случайные решения, к которым трудно или вообще невозможно прийти, применяя только оператор скрещивания.

Поскольку в результате мутации генерируются новые решения и отсутствует внесение хаоса в популяцию, в исследовании используется вероятность возникновения мутации менее 0,01. Реализация мутации осуществляется с использованием нормального распределения: когда вероятность мутации оказывается менее 0,01 – в избранной позиции число изменяется на любое другое, но из определенного интервала [Бойко, 2006].

Реализация оператора мутации осуществляется посредством следующей процедуры:

```
procedure mutation;
begin
 for n:=0 to a-1 do begin
   for j:=0 to (5*5*t_g-1) do begin
    for i:=0 to 1 do begin
      if Random<0,01 then rez1[i,j,n]:=IntToStr(random(5)+1);
    end;
    if Random<0,01 then begin
      temp1[2]:=IntToStr(random(t_g)+1);
      if StrToInt(temp1[2])<10 then temp1[2]:='0'+temp1[2];
      temp1[3]:=IntToStr(random(t_v)+1);
      if StrToInt(temp1[3])<10 then temp1[3]:='0'+temp1[3];
      temp1[4]:=IntToStr(random(t_d)+1);
      if StrToInt(temp1[4])<10 then temp1[4]:='0'+temp1[4];
      rez1[2,j,n]:=temp1[2];
      rez1[3,j,n]:=temp1[3];
      rez1[4,j,n]:=temp1[4];
    end;
```

```
    if Random<0,01 then begin       //
      rez1[5,j,n]:=IntToStr(random(t_a)+1);
      if StrToInt(rez1[5,j,n])<10 then rez1[5,j,n]:='0'+rez1[5,j,n];
      if (t_a>=100) then
        if StrToInt(rez1[5,j,n])<100 then rez1[5,j,n]:='0'+rez1[5,j,n];
    end;    end;
  for j:=0 to (5*5*t_g-1) do begin
    for i:=0 to 1 do begin
      if Random<0,01 then rez2[i,j,n]:=IntToStr(random(5)+1);         end;
    if Random<0,01 then begin
      temp2[2]:=IntToStr(random(t_g)+1);
      if StrToInt(temp2[2])<10 then temp2[2]:='0'+temp2[2];
      temp2[3]:=IntToStr(random(t_v)+1);
      if StrToInt(temp2[3])<10 then temp2[3]:='0'+temp2[3];
      temp2[4]:=IntToStr(random(t_d)+1);
      if StrToInt(temp2[4])<10 then temp2[4]:='0'+temp2[4];
      rez2[2,j,n]:=temp2[2];
      rez2[3,j,n]:=temp2[3];
      rez2[4,j,n]:=temp2[4];
    end;
    if Random<0.01 then begin
      rez2[5,j,n]:=IntToStr(random(t_a)+1);
      if StrToInt(rez2[5,j,n])<10 then rez2[5,j,n]:='0'+rez2[5,j,n];
      if (t_a>=100) then
        if StrToInt(rez2[5,j,n])<100 then rez2[5,j,n]:='0'+rez2[5,j,n];
    end;    end;
  for i:=0 to (5*5*t_g-1) do
    for j:=0 to 5 do begin
      new_hrom1:=new_hrom1+rez1[j,i,n];
      new_hrom2:=new_hrom2+rez2[j,i,n];
      y[j,i,2*n]:=rez1[j,i,n];
      y[j,i,2*n+1]:=rez2[j,i,n];
    end;
  new_pop:=new_pop+new_hrom1+new_hrom2;
end; end;
```

Предложенный алгоритм иллюстрирует случайную мутацию с вероятностью появления меньше 0,01 для создания расписания занятий в группе, причем предполагается, что оно составляется на пять дней и каждый день группа имеет пять учебных занятий.

Применение эволюционных алгоритмов при решении задачи составления расписаний сопряжено с необходимостью учета «проклятия размерности», поскольку среднестатистическое количество групп в среднестатистическом высшем учебном заведении составляет от 50 до 200. Заметим, что в таких алгоритмах выполняется значительное количество операций, не ведущих к нахождению оптимального решения, и их практическое тестирование и реализация наталкивается на проблемы, связанные со временем выполнения.

Сократить количество операций можно, используя некоторые дополнительные процедуры, оптимизирующие процесс поиска решения. Определенным образом они связаны с реализацией оператора мутации. Традиционно использовалась мутация, базирующаяся на разыгрывании случайной величины, имеющей равномерное распределение. Результаты проведенных экспериментов указывают на то, что «равномерный» выбор фрагмента хромосомы и «равномерная» его мутация не направлены на сокращение времени поиска оптимального решения, поскольку одинаковые шансы для модификации имеют как «перспективные», так и «неперспективные» решения. Причем мутация в первых может приводить к появлению «неперспективных» решений, а вторых – к появлению «перспективных», что указывает на случайный ненаправленный характер поиска оптимального решения.

Уменьшим количество «ложных» шагов алгоритма, используя место равномерного распределения нормального, что и предлагается в настоящей статье. Реализация такого подхода имеет следующие этапы. Вначале определяются участок потенциальных решений, который является одинаковым в каждом из них и которому соответствует максимальное или близкое к нему значение функции принадлежности. Напомним, что фенотип этого участка представляет собой действительное число, принадлежащее интервалу, который указывается в начале процесса кодирования элементов начальной выборки или начальной популяции решений [Бойко, 2006]. На следующем этапе разыгрывается случайное число, имеющее нормальное распределение со средним, соответствующем фенотипу участка, и среднеквадратическим отклонением, таким, чтобы интервал $(m-3\sigma, m+3\delta)$, где m- фенотип соответствующего участка, σ- среднеквадратическое отклонение, совпадал с априорным интервалом изменения фенотипа. Этот интервал играет определенную роль при расчете величины мутации. Отметим также, что среднеквадратическое отклонение в процессе поиска решения изменяется, а именно, уменьшается, что указывает на то, что по мере приближения к оптимальному или квазиоптимальному решению вероятность значительных мутаций уменьшается.

Результаты экспериментов указывают на 30-50% сокращение времени поиска решения по сравнению с использованием равномерного распределения.

Заключение

Предложенная технология реализации оператора мутации при решении задачи составления расписаний направлена на сокращение времени поиска ее решения. Отметим, что выбор параметров реализации как алгоритма, так и операции мутации имеет значительное влияние на сходимость и скорость сходимости алгоритма, поэтому требует проведения дальнейших исследований и проведения экспериментов.

Библиография

[Воронин, 2004] Воронин О., Дьюдни А. Дарвинизм в программировании // Мой компьютер. – 2004. – № 35 (310). – С. 3-12.

[Корнеев, 2003] Корнеев В.В., Гареев А.Ф., Васютин С.В., Райх В.В. Базы данных. Интеллектуальная обработка информации. – М: изд-во „Нолидж". – 2003 г. – 400 с.

[Кисляков, 2001] Кисляков А.В. Генетические алгоритмы: операторы скрещивания и мутации // Информационные технологии. – 2001. – №1. – С. 29-34.

[De Jong, 1975] De Jong, K.A. An analysis of the behavior of a class of genetic adaptive systems // Doctoral thesis. Dept Computer and Communication Sciences. – University of Michigan, Ann Arbor. – 1975.

[Schaffler, 1989] Schaffler J.D., Caruana R.A., Escheman L.J., Das R. A study of control parameters affecting online performance of genetic algorithm for function optimisation // Proceedings of the Third International Conference on Genetic Algorithms and their Applications. – San Mateo, CA: Morgan Kaufmann Publishers. – 1989. – Pp. 51-60.

[Grefenstette, 1986] Grefenstette J.J. Optimization of control parameters for genetic algorithms // IEEE Transaction on systems, man end cybernetics. SMC-16(1). – 1986. – Pp. 122-128.

[Исаев, 2003] Исаев С.А. Популярно о генетических алгоритмах (http://algolist.manual.ru/ai/ga/ga1.php).

[Michalewicz, 1995] Michalewicz Z. Genetic Algorithms, Numerical Optimization and Constraints, Proceedings of the 6th International Conference on Genetic Algorithms, Pittsburgh, July 15-19, 1995. - P. 151-158.

[Holland, 1994] Holland J. H. Adaptation in natural and artificial systems. An introductory analysis with application to biology, control, and artificial intelligence. – London: Bradford book edition, 1994 – 211 p.

[Бойко, 2006] Бойко О.М. Еволюційна технологія розв'язування задачі складання розкладів навчальних занять // Искусственный интеллект. – 2006. – №3. – С. 341-348.

Информация об авторе

Елена Сипко *– аспирант, Черкасский государственный технологический университет, бул. Шевченко, 460, Черкассы,18006, Украина; e-mail: boyko-e-n@mail.ru*