УДК 519.7

С.В. Лазаренко, А.С. Макаренко

ВЫЧИСЛИТЕЛЬНЫЕ СЛОЖНОСТИ МОДЕЛИРОВАНИЯ ДИНАМИЧЕСКИХ СИСТЕМ С АНТИЦИПАЦИЕЙ

Ключевые слова: вычислительная сложность, дискретные динамические системы, оператор Хатчинсона, периодическая траектория.

Введение

Моделирование динамических систем, операторы эволюции которых предусматривают многозначность решений, — относительно новое направление теоретической кибернетики. Актуальность и значительный прикладной потенциал заключаются в построении новых моделей, которые, в некоторой степени, будут лучше описывать и формализовать ряд сложных явлений и процессов, так как предполагают некую рефлексивную составляющую. К числу таких процессов, несомненно, можно отнести такие социально-экономические рефлексивные процессы: общественное и индивидуальное сознание [1], моделирование транспортных потоков [2] (являющееся важной задачей управления инфраструктурой современных крупных городов) и др.

В ходе моделирования систем с многозначными операторами эволюции так или иначе сталкиваются с проблемами объемов вычислений и нелинейным ростом использования машинной памяти. Поэтому именно эти принципиальные аспекты моделирования динамических систем с антиципацией (ДСА) находятся в фокусе данной статьи. В частности, рассматриваются вопросы вычислительных сложностей процедур поиска периода циклических траекторий при моделировании таких систем.

1. Математическая модель

Для начала введем необходимые определения и принятые обозначения. С более детальной терминологией антиципационных систем можно ознакомиться в работе [3] и по ссылкам в ней. В качестве примера рассмотрим исходную систему, описываемую законом сильной антиципации первого порядка:

$$x_{i+1} = f(x_i, x_{i+1}, \Lambda),\ x_i \in R^n,\ i = 0, 1, 2, \ldots, \tag{1}$$

где $\Lambda = (\lambda; \alpha) \in R^2$ — управляющий параметр, $f : R^n \times R^n \times R^2 \to R^n$ — оператор связи, x_i — состояние неявной системы.

Нас интересует случай, когда f предполагает возможную многозначность решений. Исходя из этих соображений, вводим следующие определения. Часто под состоянием антиципационной динамической системы (ДС) понимают именно x_i в (1), однако в контексте нашей задачи по поиску периодических траекторий будем придерживаться следующего определения.

Определение 1. Состояниями ДС в явном виде в дискретные моменты $i = 0, 1, \ldots$, будем называть такие множества: $X_i = \bigcup\limits_k \{x_k^i\} \in S_n$. Здесь S_n — подмножество множества всех возможных подмножеств R^n ($S_n \subset 2^{R^n}$).

Пусть оператор f в (1) предусматривает многозначность решений, и его можно представить отображением $F: S_n \times R^{n^{\dim(\Lambda)}} \to S_n$ в виде

$$X_i = F(X_{i-1}, \Lambda) = \bigcup_{x \in X_{i-1}} \bigcup_{k} f_k(x, \Lambda), \qquad (2)$$

здесь $X_i \in S_n$, $x \in R^n$, $\Lambda = (\lambda_1, \lambda_2, \ldots, \lambda_{\dim(\Lambda)})$, $\lambda_i \in R^n$, $i = \overline{1, \dim(\Lambda)}$.

С необходимыми понятиями теории многозначных отображений и многозначного анализа, которые используются в данной статье, можно ознакомиться, например, в [4; 5]. При таком представлении нашей антиципационной ДС (2) через явную зависимость текущего ее состояния от предыдущих (по времени) состояний оператор $F(\cdot)$ будем называть оператором эволюции ДС с антиципацией.

Определение 2. Траекторией ДС, заданной правилом смены состояний (2) под действием оператора $F(\cdot)$, начиная с состояния в момент i, будем называть последовательность состояний $X_i, X_{i+1}, X_{i+2}, \ldots, X_{i+k}$. Если k — конечное, то речь пойдет о конечной части траектории.

Определим на S_n метрику Хаусдорфа

$$d_H(X, Y) = \max\{\sup_{x \in X} \inf_{y \in Y} \rho(x, y), \ \sup_{y \in Y} \inf_{x \in X} \rho(x, y)\}, \qquad (3)$$

где $X, Y \in S_n$, $\rho(,)$ — евклидова метрика.

Заранее зададим достаточно малое значение ε, которое будет представлять точность расчетов периодических орбит нашей ДС.

Определение 3. Под неподвижной точкой периода p антиципационной ДС будем понимать набор последовательных состояний $X_i, X_{i+1}, \ldots, X_{i+p-1}$ таких, что p — наименьше положительное, при котором

$$d_H(X_i, F(X_{i+p-1}, \Lambda)) \xrightarrow[i \to \infty]{} 0.$$

Практически для определения периода будем использовать условие $d_H(X_i, F(X_{i+p-1}, \Lambda)) < \varepsilon$.

Определение 4. Возмущенным состоянием ДС с антиципацией относительно состояния $X_i \in S_n$ будем называть такое состояние $X_i' \in S_n$, что $d_H(X_i, X_i') = \| \tilde{x}_i \|$, где $\tilde{x}_i \in R^n$ — возмущение X_i.

Такое определение задает неоднозначность X_i'. Понятно, что, возмущая X_i разными способами с помощью $\tilde{x}_i$ так, чтобы $d_H(X_i, X_i') = \| \tilde{x}_i \|$, можно получить различные результаты в ходе итерирования (2). Для большей определенности будем возмущать X_i следующим образом: $X_i' = X_i + \tilde{x}_i = \{x \mid x - \tilde{x}_i \in X_i\}$.

2. Теория алгоритмов и вычислительные модели

Хотя само понятие алгоритма кажется довольно очевидным, его формализацию заложили относительно недавно — в первой половине XX века. Прежде всего этим мы обязаны диссертации А.М. Тьюринга, работам А. Черча, Е.Л. Поста [6], А.Н. Колмогорова, А.А. Маркова и др. [7, 8].

В настоящее время практическое применение и актуальность теории алгоритмов не вызывает никаких сомнений, поскольку работа практически всей тех-

ники базируется на основах этой теории. Именно работы Тьюринга и Дж. фон Неймана дали основной теоретический толчок к созданию ЭВМ [9]. В настоящее время теория алгоритмов охватывает значительный спектр вопросов: начиная от первичных, формировавших ее, — вопрос вычислимости функций, вычислительных моделей, сложности вычислений и отношение между их классами; и заканчивая построением новых типов алгоритмов и их классификации — управляющие, вычислительные и тому подобное. С прикладной точки зрения, в теории алгоритмов, прежде всего, стоят задачи анализа трудоемкости алгоритмов, их классификация по сложностям и определение и оптимизация их ресурсоемкости. Так, среди основных прикладных достижений теории алгоритмов есть наработки практических рекомендаций по проектированию и разработке программных систем, что наряду с усовершенствованием аппаратного обеспечения обусловливают решение ряда задач математического моделирования, решение которых до недавних пор было невозможным. Несмотря на постоянно растущую вычислительную мощность машин, вопрос сложности алгоритмов актуален и поныне. Поэтому именно этот вопрос, так или иначе, будет проходить через всю статью как основной.

Алгоритм [10, с. 99; 11, с. 135] есть предписание, однозначно определяющее ход некоторых конструктивных процессов. Он задает переход от одного состояния процесса в другое под действием некоторого оператора «непосредственной переработки» состояний. Этот оператор задается конечным набором правил. Один и тот же алгоритм может задавать различные процессы, которые отличаются лишь начальным состоянием. Алгоритмический процесс состоит из отдельных шагов наперед заданной сложности. Для построения алгоритма необходимо иметь алфавит, с помощью которого представлены состояния процесса.

Формально алгоритм можно представить как функцию $f : B^* \to B^*$, которая превращает одно конечное слово в другое из того же алфавита B (в качестве алфавита, как правило, берут $B = \{0,1\}$). B^* обозначает множество всех конечных последовательностей номеров из B, т.е. $B^* = \bigcup_{i \geq 0} B^i$ при всех конечных i, где B^i — множество всех конечных последовательностей из B длиной i. Расчет такой функции f осуществляется на некоторой вычислительной модели. Эта модель задается фиксированным набором правил, при последовательном применении их можно рассчитать $f(x)$ для любого входного слова $x \in B^*$. Каждое правило представляет собой комбинацию базовых операций этой модели. Например, такими операциями для машины Тьюринга могут быть следующие: прочитать символ с входной ленты, прочитать символ из рабочей области памяти, записать символ на ленту, остановка расчета или выбор следующей операции. Кроме базовых операций в определение модели также включена и цена (cost) этих операций для определения сложности данного алгоритма. Из самых распространенных вычислительных моделей отметим [12, с. 5]: Машина Тьюринга, лямбда-исчисление, регистровые машины. Сейчас существует достаточно большое количество различных модификаций и производных моделей: недетерминированные машины Тьюринга, клеточные автоматы и тому подобное [13].

3. Теория сложности вычислений

На множестве алгоритмов вводят такие характеристики, как понятие мер сложности для сравнения алгоритмов и выбора «лучшего» по заданному кри-

терию. Эта мера сложности описывается количеством вычислительных ресурсов, необходимых для выполнения данного алгоритма. Основными мерами являются время выполнения (временная сложность) алгоритма на данной вычислительной модели (его скорость выполнения) и объем памяти (пространственная сложность), который нужен для того, чтобы выполнить алгоритм на той же вычислительной модели.

Временем выполнения алгоритма $T(n)$ для данной вычислительной модели является количество базовых операций, которые выполняются для преобразования входного слова x длиной n. Временная сложность алгоритма в худшем случае определяется как

$$t_{\max}(n) = \max_{|x| \le n} t(x).$$

При вычислении точного значения использованных ресурсов для выполнения алгоритма в нем фигурируют постоянные составляющие — затраты на выполнение базовых операций, которые будут зависеть от конкретной вычислительной модели. Однако с ростом размера входных данных (длины входного слова x) их влияние нивелируется вместе со слагаемыми низших порядков. Поэтому в теории вычислительных сложностей расчет сложности алгоритма проводится асимптотически, т.е. оцениваются затраты ресурсов (времени или памяти) в предельном случае — когда размер входных данных стремится к бесконечности.

В асимптотическом анализе приняты следующие определения [12, с. 8]. Пусть некоторый алгоритм имеет вычислительную сложность $f(n)$.

Определение 5. Оценка сложности алгоритма $\Theta(g(n))$ задает множество таких функций $f(n)$, если существуют некоторые константы $c_1, c_2 > 0$ и некоторый номер n_0, $\forall n > n_0$, для которых будет выполняться $0 \le c_1 \cdot g(n) \le f(n) \le c_2 \cdot g(n)$. В таком случае говорят, что $g(n)$ — асимптотически точная оценка $f(n)$.

Аналогично вводят понятие асимптотически верхней и нижней оценок.

Определение 6. Оценка сложности алгоритма $O(g(n))$ задает множество таких функций $f(n)$, если существуют некоторая константа $c > 0$ и некоторый номер n_0, $\forall n > n_0$, для которых будет выполняться $0 \le f(n) \le c \cdot g(n)$. Полагаем, что $g(n)$ — асимптотически верхняя оценка $f(n)$.

4. Расчет вычислительных сложностей ДСА

Рассмотрим главный вопрос данной статьи: какие необходимые вычислительные затраты мы понесем в ходе численного моделирования ДСА для поиска периодических траекторий? Для данной задачи выделим следующие подзадачи по расчету вычислительных сложностей: построение состояния ДС, сравнение нескольких таких состояний, построение последовательности состояний. Прежде всего введем ряд необходимых обозначений. Пусть имеем траекторию, дискретную ДСА $X_0, X_1, X_2, \ldots$. Поскольку ДСА часто можно представить в явном виде оператором Хатчинсона от предыдущих состояний, то кардинальные числа $|X_i|$, в худшем случае (например, без образования циклов) будут расти по показательному закону

$$|X_i| = |X_0| \cdot N^i, \; i = 0, 1, 2, \ldots, \tag{4}$$

где N — количество селекторов в операторе Хатчинсона (2). Из-за этой особенности ДСА при их моделировании вопрос вычислительных затрат становится

особенно острым. Рассчитаем вычислительные сложности процедур поиска периодических траекторий. На их основе сможем скорректировать и улучшить предложенные процедуры с точки зрения минимизации вычислительной сложности. Будем придерживаться следующих необходимых обозначений:

- c_n — вычислительные затраты на сравнение двух чисел в R^n;
- m_n — вычислительные затраты на расчет функции, действующей в R^n;
- p — максимальная длина цикла, которую рассматриваем.

4.1. Сложность построения состояния ДСА. Каждая итерация эволюции ДСА сопровождается построением очередного состояния системы, поэтому рассмотрим вычислительные затраты на каждой итерации. Имеет место следующая теорема.

Теорема. Вычислительная сложность построения каждого состояния X_i, $i = 0, 1, 2, \ldots,$ в худшем случае будет $O(N^{2i})$.

Доказательство. Пусть состояние $X_i = \{x_1^i, x_2^i, \ldots, x_{|X_i|}^i\}$ построено на основе предыдущего $X_{i-1} = \{x_1^{i-1}, x_2^{i-1}, \ldots, x_{|X_{i-1}|}^{i-1}\}$. Определим вычислительную сложность построения X_i на множестве конечных кардинальных чисел как отображение $m(k)$ (здесь k — мощность состояния X_i), так как сложность в первую очередь будет зависеть не от номера итерации, а от мощности состояния. Рассмотрим наихудший сценарий в смысле вычислительных затрат, для которого мощность состояний будет расти по показательному закону $|X_i| = N|X_{i-1}|$.

Сначала действуем первым селектором оператора $F(\cdot)$ на первую точку в X_{i-1} согласно (2), т.е. $x_1^i = f_1(x_1^{i-1})$. В результате имеем первую точку в X_i, получив вычислительные затраты только на операцию расчета точки в R^n, сложность которой m_n. Для следующей точки $x_2^i = f_1(x_2^{i-1})$ проведем операцию со сложностью $m_n + c_n$, поскольку дополнительно придется проверить, принадлежит ли x_2^i состоянию X_i (сравнив ее с x_1^i). Для всех последующих точек x_j^i (полученных по всем остальным точкам с X_{i-1} и по всем остальным селекторам) будут проведены операции сложности $m_n + (j-1)\cdot c_n$, где $j = |X_i|$. Таким образом, сложность построения состояния X_i

$$m(k) = m_n + (m_n + c_n) + \ldots + (m_n + (k-1)c_n) = \sum_{i=0}^{i=k-1} (m_n + i \cdot c_n) =$$

$$= k \cdot m_n + (k-1) \cdot k \cdot \frac{c_n}{2}.$$

Учитывая $k(i) = |X_i| = N^i |X_0|$, переходим к определению сложности построения состояния в зависимости от номера i этого состояния:

$$m(i) = \frac{c_n |X_0|^2}{2} N^{2i} + \left(m_n - \frac{c_n}{2} \right) \cdot |X_0| \cdot N^i, \qquad (5)$$

она и будет ограничена по отношению к функции N^{2i} при $i \to \infty$.

Теорема доказана.

Как видно из (5), наибольшие вычислительные затраты $O(N^{2i})$ приходятся на операции сравнения, а на вычисление соответствующих точек в R^n — только $O(N^i)$. В этом и заключается одна из самых больших проблем при моделировании ДСА. Ее можно частично обойти за счет представления X_i как мультимножеств (отсутствуют затраты на сравнение $c_n = 0$), тогда в (5) останется только составляющая $O(N^i)$. В таком случае, однако, условие (4) будет иметь место всегда, независимо от того — регулярный или нет режим ДСА. И как следствие — постоянный рост использования памяти по показательному закону N^i.

4.2. Сложность расчета метрики. Так как мы работаем в метрическом пространстве всех непустых компактных подмножеств с метрикой Хаусдорфа $d_H(.,.)$, разумным будет задаться вопросом — какие вычислительные затраты на ее расчет для двух состояний ($X = \{x_i\}_i$ и $Y = \{y_j\}_j$), исходя из определения этой метрики (3)? Как видим, ее расчет состоит из двух видов операций: 1) вычисление $\rho(x, y) \overset{def}{=} |x - y|$ в евклидовой метрике для $x, y \in R^n$ (затраты на ее расчет будем обозначать d_n); 2) сравнение двух значений $\rho(.,.)$, поэтому она потребует c_1 затрат.

Количество расчетов $\rho(x_i, y_j)$ по всем возможным парам (x_i, y_j), очевидно, будет $|X| \cdot |Y|$. Подсчет $\inf\limits_j \rho(x_i, y_j) = \rho_{x_i}$ для каждой точки x_i потребует $|Y| - 1$ операций сравнения $\rho(.,.)$, а для всех x_i их будет $(|Y| - 1) \cdot |X|$. Далее необходимо рассчитать $\sup\limits_i \{\rho_{x_i}\}$, на что потребуется $|\{\rho_{x_i}\}_i| - 1 = (|Y| - 1) \cdot |X| - 1$ сравнений. В силу коммутативности метрики общее количество сравнений $\rho(.,.)$, которые понадобятся для расчета $d_H(X, Y)$, составит $2((|Y| - 1) \cdot |X| - 1) - 1 = 2 \cdot |X| \cdot |Y| - 1$. Общие вычислительные затраты для получения $d_H(X_i, X_j)$, учитывая (4), будут

$$d_{ij} = |X_i| \cdot |X_j| \cdot d_n + (2 \cdot |X_i| \cdot |X_j| - 1) \cdot c_1 = (d_n + 2c_1) N^{i+j} |X_0|^2 - c_1$$

или в *O*-нотации —

$$d_{ij} \in O(N^{i+j}). \tag{6}$$

4.3. Сложность построения траектории ДСА. Следующим шагом после построения состояния ДС является построение их последовательности (траектории). Какой же будет сложность построения траектории длиной p, которая начинается из состояния с индексом i? Обозначим ее $M(i, i+p)$:

$$M(i, i+p) = \sum_{k=i}^{i+p} m(k) = \sum_{k=i}^{i+p} \left(\frac{c_n |X_0|^2}{2} N^{2k} + \left(m_n - \frac{c_n}{2} \right) \cdot |X_0| \cdot N^k \right) =$$

$$= \frac{c_n |X_0|^2}{2} \frac{N^{2i} \cdot (N^{2(p+1)} - 1)}{N^2 - 1} + \left(m_n - \frac{c_n}{2} \right) \cdot |X_0| \cdot \frac{N^i \cdot (N^{p+1} - 1)}{N - 1}$$

или в O-нотации — $M(i, i+p) \in O(N^{2i+2p})$, а в случае мультимножеств $(c_n = 0)$ —

$$M(i, i+p) \in O(N^{i+p}). \tag{7}$$

4.4. Сложность поиска периода циклической траектории ДСА. Задача поиска периода траекторий выступает как составляющая, например, такого инстру-

мента анализа ДС, как построение карт динамических режимов ДСА. Детальное описание алгоритма конструирования этих карт можно найти по ссылке [14]. Рассмотрим возможные процедуры определения периода траектории и их вычислительные сложности. Каждую процедуру разобъем на две составляющие: 1) построение состояний ДС в моменты времени $i, i+1, \ldots, j$; 2) сравнение новых состояний между собой и с ранее полученными в метрике $d_H(.,.)$. Вычислительная оптимизация процедуры поиска периода как раз заключается в комбинировании этих составляющих таким образом, чтобы общая сложность была наименьшей.

Рассмотрим следующие возможные процедуры.

Процедура А. Суть ее заключается в том, что на каждом шаге построения траектории мы пытаемся найти цикл, сравнивая последнее полученное состояние ДС с предыдущими в обратном времени. Для начала ограничиваемся поиском цикла длиной p. Пусть в некоторый момент времени мы находимся в точке X_{i-1} ДСА. Строим X_i согласно (2) и сравниваем это состояние с предыдущими $i-1$-м, $i-2$-м и так далее до $i-p$, пока не найдем достаточно близкое (с заранее заданной точностью) состояние в выбранной метрике. Именно такое направление сравнений выбрано не случайно, чтобы избежать ситуации, когда мы сначала нашли кратный цикл. Таким образом, сначала получим «наименьший» цикл (без удвоений, утроений и т.д.). Если найдено такое состояние X_j с $d_H(X_i, X_j) \le \varepsilon$, то дальнейшие действия вполне понятны. В противном случае продолжаем траекторию на одно состояние вперед и повторяем процедуру снова.

Нас интересует вопрос: какой будет сложность действий на каждой итерации? Пусть мы ранее построили состояние $i+k-1$ и не нашли цикл длины $k-1 < p$. Строим $i+k$-е и сравниваем его со всеми ранее построенными в обратном порядке — с $i+k-1, i+k-2, \ldots, i$. Это соответственно потребует таких затрат:

$$m(i+k) + d_{i+k\,i+k-1} + d_{i+k\,i+k-2} + \ldots + d_{i+k\,i}.$$

Поэтому и вся процедура потребует следующих затрат при итерировании из состояния i в состояние $i+p$, замыкающее найденный цикл длины p:

$$m(i) + (m(i+1) + d_{i\,i+1}) + (m(i+2) + d_{i\,i+2} + d_{i+1\,i+2}) + \ldots$$
$$\ldots + (m(i+p) + d_{i\,i+p} + d_{i+1\,i+p} + \ldots + d_{i+p-1\,i+p}) = M(i, i+p) + \sum_{k=i}^{i+p-1} \sum_{j=k+1}^{i+p} d_{kj}. \quad (8)$$

На протяжении всей процедуры необходимо постоянно держать в памяти последние $p+1$ состояний ДС ($X_i, X_{i+1}, \ldots, X_{i+p}$) для расчета метрик между ними (пространственная сложность), т.е. необходимо держать в памяти $\sum_{j=i}^{i+p} |X_j| =$ $= |X_0| \cdot N^i \cdot \frac{N^{p+1}-1}{N-1}$ чисел из R^n или в O-нотации $O(N^{i+p})$.

Процедура В. Из выражения (8) нетрудно показать, что если в качестве состояний ДС выбраны мультимножества ($c_n = 0$), составляющая по вычислению метрик $d_H(,)$ в процедуре поиска цикла требует значительно больше вычислительных ресурсов, чем построение состояний. Для этого возьмем лишь последнее слагаемое в (8) $d_{i+p-1\,i+p}$ и сравним его со сложностью $M(i, i+p)$, учитывая (6) и (7). В результате получим $d_{i+p-1\,i+p} \in O(N^{2i+2p-1})$, $M(i, i+p) \in O(N^{i+p})$.

Поэтому следующая альтернативная процедура предусматривает минимизацию именно этих вычислений. Для простоты описания данной процедуры рассмотрим некоторые упрощения и локальные обозначения. Пусть ε — наперед за-

данная точность поиска периодической траектории. Имеем D — область определения ДСА, которая является объединением предельного множества (цикл периода не более p) и области его притяжения. Исходное состояние ДС — $X_{i-1} \in D$. Начинаем строить траекторию с первой серии итераций $X_i, X_{i+1}, \ldots, X_{i+p-1}$ ($s = 0$). Заранее задаем длину этих серий, равную p. Расстояние между такими сериями $t \geq 1$ определяется эмпирически и зависит от особенностей динамики системы (чем быстрее траектория приближается к предельному множеству, тем меньшим выбираем t). Такое разбиение процесса итерирования на состояния в сериях и вне их обусловлено тем, что операции сравнения состояний будут осуществляться только для состояний, имеющихся в сериях, сокращая тем самым вычислительные затраты. Структура траектории согласно этой процедуре представлена на рис. 1.

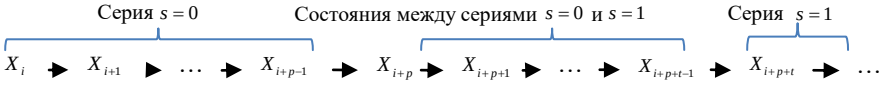

Рис. 1

Таким образом, получим серии, начинающиеся с $i + s(p+t)$ и заканчивающиеся в $i + s(p+t) + p - 1$, при номерах $s = 0, 1, \ldots$, поэтому и сложность построения этих частей траектории будет

$$M(i + s(p+t),\ i + s(p+t) + p - 1) \text{ для } s = 0, 1, \ldots$$

Между сериями будут только затраты на построение траекторий:

$$M(i + s(p+t) - t,\ i + s(p+t) - 1) \text{ при } s = 1, 2, \ldots$$

Для каждой серии необходимы дополнительные затраты на сравнение всех ее состояний с первым состоянием, которое следует за этой серии (для серии $s = 0$ это состояние X_{i+p} приведено на рис. 1). Итак, строим X_{i+p}. Рассчитываем отклонения между состояниями текущей серии и X_{i+p}:

$\bar{d}^{(s)} = (d_1^{(s)},\ d_2^{(s)},\ \ldots,\ d_p^{(s)})$, где $d_j^{(s)} = d_H(X_{i+p+s(p+t)},\ X_{i+p+s(p+t)-j})$

согласно иллюстрации на рис. 2.

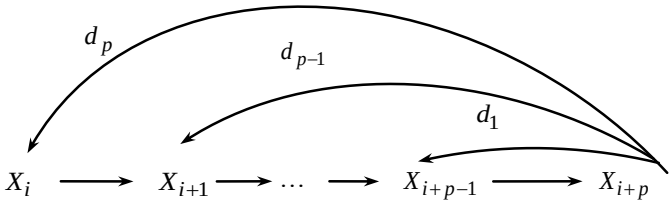

Рис. 2

В таком случае расходы для $s = 0$ будут $\sum_{j=1}^{p} d_{i+p,\, i+p-j}$, и для каждого $s = 0, 1, \ldots$:

$$\sum_{j=1}^{p} d_{i+s(p+t)+p,\, i+s(p+t)+p-j}.$$

Если найдено такое $d_k^{(s)} \leq \varepsilon$ (понятно, что затраты на сравнение значений метрик здесь будут pc_1), то процедура поиска цикла завершена и считаем, что ДС имеет период k. Если таких

$$d_{k_1}^{(s)},\ d_{k_2}^{(s)},\ \ldots \leq \varepsilon \qquad (9)$$

несколько, то выбираем тот, у которого наименьший индекс $\min\limits_i(k_i)$, чтобы заранее избежать кратных циклов и сфокусироваться на «наименьшем». Вычислительные расходы на сравнение всех $d_k^{(s)}$ (равны pc_1) и поиска наименьшего из них ($(p-1)c_1$) будут

$$pc_1 + (p-1)c_1 = (2p-1)c_1.$$

И наконец, третий вариант: если такое $d_k^{(s)} \le \varepsilon$ не найдено, то увеличиваем $s \leftarrow s+1$ и осуществляем следующую серию итераций. Такие серии итерирований $s = 0, 1, \ldots$ проводим до тех пор, пока $m^{(s)} = \min(d_1^{(s)}, d_2^{(s)}, \ldots, d_p^{(s)})$ убывает, но все еще превышает ε. Подытожим затраты этой процедуры:

$$M(i,\ i + s(p+t) + p - 1) + m(i + s(p+t) + p) +$$
$$+ \sum_{k=0}^{s} \sum_{j=1}^{p} d_{i+k(p+t)+p,\, i+k(p+t)+p-j} + (2p-1)c_1.$$

Если стоит задача — убедиться, что мы приблизились к орбите периода именно k_1, то рассматриваем $m^{(s)} = d_{k_1}^{(s)} \le \varepsilon$. В таком случае необходимо проводить оценку компонентов вектора

$$\overline{\Delta}_{k_1}^{(s+1)} = \begin{bmatrix} d_1^{(s+1)} \\ d_2^{(s+1)} \\ \vdots \\ d_{k_1}^{(s+1)} \end{bmatrix} - \begin{bmatrix} d_1^{(s)} \\ d_2^{(s)} \\ \vdots \\ d_{k_1}^{(s)} \end{bmatrix}. \qquad (10)$$

Перейдем к сериям итерированний длиной k_1. В случае k_1-периодической орбиты необходимо, чтобы выполнялось условие $\overline{\Delta}_{k_1}^{(s+1)} \xrightarrow[s\to\infty]{} \overline{0}$.

При его нарушении нужно выбрать следующего кандидата $k_2 \ne k_1$ из соотношения (9), отличного от всех ранее рассмотренных из набора $\min(d_1^{(s+1)}, d_2^{(s+1)}, \ldots, d_p^{(s+1)}) \le \varepsilon$. Если таковых не осталось, то мы имеем дело либо с орбитой периода больше рассматриваемого p, либо мы попали в область квазипериодичности или хаоса. Такой подход дает возможность более точно определить, имеем мы дело с орбитой именно такого периода или перед нами более сложная структура.

Расходы памяти в ходе проведения процедуры близки к расходам процедуры А. Кроме последних $p+1$ состояний ДС в памяти необходимо держать $2p$ чисел для расчета (10). Поэтому пространственная сложность равна $|X_0| \cdot N^i \cdot \cdot \dfrac{N^{p+1}-1}{N-1} + \left\lceil \dfrac{2p}{n} \right\rceil$ чисел из R^n или в O-нотации — $O(N^{i+p})$.

Для рассмотренных процедур, в случае если в процессе итерирования траектория прерывается (множество действительных значений состояний пустое), то возмущаем X_{i-1} и, начиная с него, снова проводим выбранную процедуру поиска периодической орбиты.

Заключение

В настоящей статье рассматривается один из принципиальных вопросов, возникающий при моделировании дискретно-временных динамических систем с антиципацией — вычислительные затраты. Подробно представлены проце-

дуры расчета периода циклических траекторий для ДСА и описаны их сложности. Предложен и обоснован переход описания состояний ДСА на мультимножества для понижения вычислительных затрат в ходе поиска циклических траекторий.

Поскольку проблема использования вычислительных ресурсов при моделировании ДСА — одна из наиболее значимых, то в дальнейшем, в целях ее минимизации в ходе поиска периодических траекторий (например, для построения карт динамических режимов), нужно сосредоточиться на сравнении предложенных процедур (с точки зрения вычислительных сложностей). Это сравнение следует проводить с учетом структуры фазового пространства (например, распределения периодических режимов).

С.В. Лазаренко, О.С. Макаренко

ОБЧИСЛЮВАЛЬНІ СКЛАДНОСТІ МОДЕЛЮВАННЯ ДИНАМІЧНИХ СИСТЕМ З АНТИЦИПАЦІЄЮ

Поняття антиципації передбачає залежність майбутніх станів не лише від минулих, а й від самих майбутніх станів. Одна з основних причин, що обумовлює актуальність дослідження систем із антиципацією, — це надресурсоємність задачі моделювання систем із множинними сценаріями розвитку, оскільки антиципаційні системи часто передбачають багатозначність розв'язків. Невелика кількість робіт в даній області інформатики також обумовлена часто некоректністю постановки задачі в силу існування кількох можливих розв'язків. Тим самим системи із антиципацією представляють новий напрям в кібернетиці та моделі на основі антиципації більш точно можуть формально описувати велику кількість існуючих систем та процесів порівняно з класичними моделями із запізненням. Розглянуто такі нелінійні дискретні динамічні системи із сильною антиципацією, у яких майбутні стани можна представити явною залежністю від минулих за допомогою оператора Хатчинсона. Еволюція таких динамічних систем здійснюється в хаусдорфовому метричному просторі. Розглянуто принципову проблему моделювання таких систем — обсяг використання обчислювальних ресурсів. Введено ряд означень для дослідження динаміки систем із антиципацією. Представлено необхідні поняття теорії обчислювальних складностей. Важливий інструмент дослідження динаміки систем — карта динамічних режимів, побудова якої вимагає адаптації процедур пошуку періодичних траєкторій для систем такого типу. Запропоновано та детально описано процедури пошуку періодичних траєкторій динамічних систем із антиципацією. Послідовно отримано часові та просторові складності побудови станів, траєкторій та цих процедур в цілому. З метою мінімізації часової складності в ході симуляції систем із антиципацією обґрунтовано представлення станів відповідних динамічних систем за допомогою мультимножин. З метою подальшої оптимізації обчислювальних витрат слід враховувати структуру фазового простору динамічної системи із антиципацією, комбінуючи запропоновані процедури.

Ключові слова: обчислювальна складність, дискретні динамічні системи, оператор Хатчинсона, періодична траєкторія.

S.V. Lazarenko, A.S. Makarenko

THE COMPUTATION COMPLEXITY OF MODELLING OF ANTICIPATORY DYNAMICAL SYSTEMS

The concept of anticipation stands for the dependence of future states not only on the past, but also on themselves. One of the main reasons for the relevance of the study of the anticipatory systems is the over-complexity of modeling of the systems with

multiple possible scenarios, since anticipatory systems often imply multiple solutions. A non-prevalence of this field of computer science is also often caused by not well-posing of the problem due to non-uniqueness of the solutions. Thus, systems with anticipation represent a new direction in cybernetics and the models based on anticipation can more formally describe a large number of existing systems and processes, compared with classical models with delay. In current paper, we consider such nonlinear discrete dynamical systems with strong anticipation, in which future states can be represented by an explicit dependence on the past with the help of the Hutchinson operator. The evolution of such dynamical systems is carried out in a Hausdorff metric space. The article focuses on the fundamental problem of modeling such systems – the amount of use of computing resources. A number of definitions have been introduced to study the dynamics of anticipatory systems. The necessary concepts of the theory of computational complexity are presented. An important tool for studying the dynamics of systems is the Atlas of Charts of dynamic regimes. The construction of which requires the adaptation of procedures for finding periodic trajectories for systems of this type. The article proposes and describes in detail the procedures for searching for periodic trajectories of dynamical systems with anticipation. Time and spatial complexities are obtained for the construction of states, trajectories, and these procedures in general. In order to minimize the time complexity during the simulation of anticipatory systems, the presentation of the states of the corresponding dynamical systems with the help of multi-sets is justified. In order to optimize further computational costs, one should take into account the structure of the phase space of a dynamical system with an anticipation, thereby combining the proposed procedures.

Keywords: computational complexity, discrete dynamical systems, Hutchinson operator, periodic trajectory.

1. Паутова Л.А., Гуц А.К. Использование теории хаоса и странных аттракторов в исследованиях индивидуального и социального сознания. *Математические структуры и моделирование.* 2004. № 13. С.126–131.
2. Семенов В.В. Математическое моделирование динамики транспортных потоков мегаполиса. М., 2004. №34. 44 с. (Препр. Ин-та прикладной математики им. М.В. Келдыша).
3. Лазаренко С.В., Макаренко О.С. Аналіз логістичного антиципаційного рівняння із сильною антиципацією. *Наукові вісті НТУУ "КПІ"*. 2012. № 4. С. 91–96.
4. Борисович Ю. Г., Гельман Б. Д., Мышкис А. Д., Обуховский В. В. Введение в теорию многозначных отображений и дифференциальных включений. М., 2011. 224 с.
5. Половинкин Е.С. Многозначный анализ и дифференциальные включения. М.: Физматлит, 2014. 522 с.
6. Díaz Josep, Torras Carme. A personal account of Turing's imprint on the development of computer science. *Computer Science Review*. 2012. **6.** P. 225–234. DOI: http://dx.doi.org/10.1016/j.cosrev.2012.11.001.
7. M. Cover Thomas, Gacs Peter, Gray Robert. Kolmogorov's Contributions to information theory and algorithmic complexity. the annals of probability. 1989. **17.** 48 p. DOI: http://dx.doi.org/10.1214/aop/1176991250.
8. Guy Leonard Kouemou. History and theoretical basics of hidden Markov models. 2011. 26 p. DOI: http://dx.doi.org/10.5772/15205.
9. Eigenmann Rudolf, Lilja David. Von Neumann computers. 1998. DOI: http://dx.doi.org/10.1002/047134608X.W1704.
10. Колмогоров А.Н. Теория информации и теория алгоритмов. М. : Наука, 1987. 304 с.
11. Марков А.А., Нагорный Н.М. Теория алгорифмов. М. : Наука, 1984. 432 с.
12. James Aspnes. Notes on computational complexity theory CPSC 468/568. Spring, 2017. 162 p.
13. Mitchell Melanie. Computation in cellular automata: A Selected Review. 1996. DOI: http://dx.doi.org/10.1002/3527602968.ch4.
14. Лазаренко С.В., Макаренко О.С. Багатопоточні комп'ютерні обчислення у дослідженні нелінійних динамічних систем. *Проблеми програмування*. 2013. № 3. С. 109–116.

Получено 05.10.2018
После доработки 04.12.2018