УДК 681.51:57

С.Я. МАЙСТРЕНКО

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ ПРОЦЕССА АВТОМАТИЧЕСКОЙ ИДЕНТИФИКАЦИИ И КОРРЕКЦИИ ТИПОВЫХ ОШИБОК ПОЛЬЗОВАТЕЛЯ ПО СЛОВАРЮ

Abstract: *The imitation model (IM) of the process to automatic identification and correction (AIC) of standard error of r on dictionary are built. The analysis factor, influencing upon efficiency of the software implementation of the methods AIC are organized. Two schemes IM: with shaping the dictionary in ambience AIC and DBMS ORACLE are investigated. Quantitative estimations to efficiency algorithm are received. Practical recommendations about software implementation of the method are given.*
Key words: *imitation modeling, automatic identification, quantitative estimations.*

Анотація: *Побудовано імітаційну модель (ІМ) процесу автоматичної ідентифікації та корекції (АІК) типових помилок користувача за словником. Проведено аналіз факторів, що впливають на ефективність програмної реалізації методів АІК. Досліджено дві схеми ІМ: з формуванням і обробкою словника в середовищі АІК та середовищі СУБД ORACLE. Отримано кількісні оцінки ефективності алгоритмів. Надано практичні рекомендації щодо програмної реалізації методу.*
Ключові слова: *імітаційне моделювання, автоматична ідентифікація, кількісні оцінки.*

Аннотация: *Построена имитационная модель (ИМ) процесса автоматической идентификации и коррекции (АИК) типовых ошибок пользователя по словарю. Проведен анализ факторов, влияющих на эффективность программной реализации методов АИК. Исследованы две схемы ИМ: с формированием и обработкой словаря в среде АИК и среде СУБД ORACLE. Получены количественные оценки эффективности алгоритмов. Даны практические рекомендации относительно программной реализации метода.*
Ключевые слова: *имитационное моделирование, автоматическая идентификация, количественные оценки.*

1. Введение

В [1] рассмотрены алгоритмы и модели автоматической идентификации и коррекции (АИК) типовых ошибок пользователя на основе словаря разрешенных (допустимых) значений. Полученные аналитические соотношения и иллюстрирующие их табличные данные определяют вероятностные характеристики, т.е. зависимости вероятностей правильной, ложной и «ручной» коррекции различных ошибок от избыточности представления слов словаря. Реальные значения быстродействия алгоритмов, определяющие целесообразные области их практического применения (и общие методы АИК в целом) зависят от многих факторов, трудно поддающихся аналитическому учету [1]. В связи с этим с целью получения соответствующих оценок ставится задача построения имитационной модели (ИМ) и экспериментального исследования процесса АИК.

2. Имитационная модель

Для описания процесса имитационного моделирования и его результатов введем следующие понятия и обозначения, заимствованные из [1]: $\mathrm{A_x}$, $\mathrm{A_y}$, $\mathrm{B_x}$ – соответственно правильное слово входного сообщения, искаженное слово, в котором обнаружена ошибка, и некоторое слово словаря, для которого выполняется $\mathrm{A_x} \equiv \mathrm{B_x}$; q – алфавит представления слов словаря; n и N – соответственно длина слова и количество слов в словаре; $\tilde{\mathrm{A}}^{\mathrm{k}}$ – вариация, под которой понимается некоторое «обратное» изменение слова $\mathrm{A_y}$ в пределах k-го класса ошибок $\mathrm{E_k}$ (однократных транскрипций E_1, вставок E_2, пропусков E_3, транспозиций E_4, двукратных транскрипций E_5); V_k – количество вариаций для k-го класса ошибок.

Далее уточним задачи и свойства ИМ. Рассмотренные в [1] алгоритмы отличаются один от другого решениями при нахождении в словаре одной или нескольких вариаций $\tilde{\mathrm{A}}^{\mathrm{k}} = \mathrm{B_x}$, т.е., по существу, стратегиями устранения неоднозначности совпадений. В остальном схемы их работы подобны и включают,

в частности, такие «времяемкие» этапы, как генерация вариаций и их поиск в словаре. Следовательно, исходными переменными для ИМ должны быть следующие факторы:

1) ансамбли $\mathrm{K(k_j)}$ (от этого при прочих равных условиях зависит количество генерируемых исходных вариаций);

2) значения n и N;

3) метод обработки словаря (поиска вариаций $\tilde{\mathrm{A}}^{\mathrm{k}} = \mathrm{B_x}$).

С точки зрения первого фактора, примем вариант ансамбля $K(1,2,3,4,5)$. Это наиболее «тяжелый» случай, соответствующий максимальному значению количества вариаций $V = \sum_k V_k$.

С точки зрения второго и третьего факторов существенным является среда, в которую погружен словарь, и инструмент его обработки. В этом отношении целесообразно рассмотреть две схемы:

1) обрабатываемый словарь находится в среде СУБД и является доменом некоторой таблицы БД. В этом случае поиск вариаций осуществляется штатными методами и инструментами СУБД, такими как:

– выполнение SQL-запросов для поиска каждой из вариаций – Алгоритм № 1.1 (выполнение оператора SELECT ровно столько раз, сколько построено вариаций);

– метод Locate – Алгоритм № 1.2 (перемещение указателя в наборе данных на первую из записей, удовлетворяющую условию, при условии, что такая запись существует);

– фильтрация записей – Алгоритм № 1.3 (построение фильтра для отбора записей, удовлетворяющих условию);

– выполнение SQL-запроса для поиска вариаций, предварительно занесенных в служебную таблицу – Алгоритм №1.4 (выполнение одного оператора SELECT из двух таблиц БД: словаря и списка вариаций);

2) обрабатываемый словарь находится в собственной среде системы АИК. Поиск осуществляется методами:

– дихотомии – Алгоритм № 2.1;

– совместной последовательной обработки синфазно упорядоченных файла вариаций и файла словаря [2] – Алгоритм № 2.2.

В качестве аппаратно-программной платформы для реализации ИМ использовались ЭВМ Intel Celeron-1000 256MB ОЗУ, СУБД ORACLE и система программирования Delphi.

Структура ИМ схемы 1 приведена на рис. 1.

Процесс ИМ включает следующие этапы:

– формирование словаря с заданными параметрами в среде СУБД ORACLE и его «оформление» как домена некоторой таблицы (неучитываемый разовый подготовительный этап);

– выбор произвольного слова и его искажение одной из ошибок ансамбля $\mathrm{K(k_j)}$;

– генерация вариаций в классах ошибок ансамбля $\mathrm{K(k_j)}$;

– поиск вариаций в словаре с помощью заданного штатного метода поиска в среде СУБД ORACLE и нахождение первичного (неискаженного) слова;

– составление и выдача файла отчета о результатах.

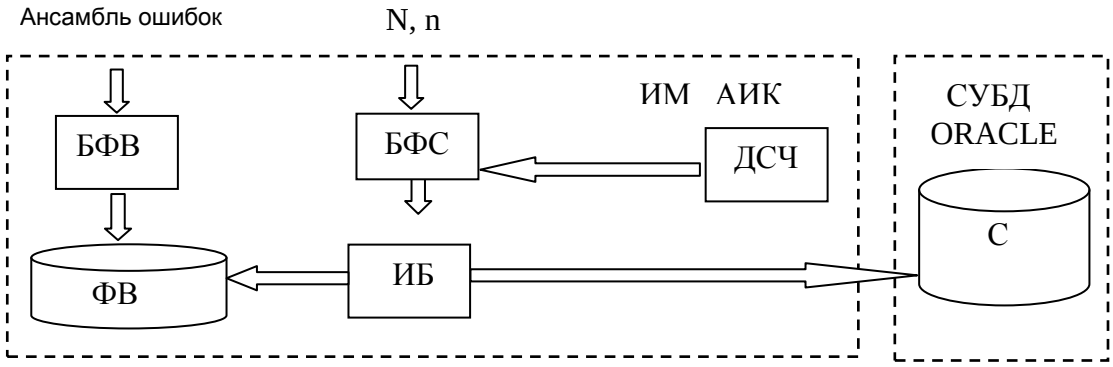

Рис.1. Структура ИМ схемы 1

ИБ – интерфейсный блок;
ФВ – файл вариаций;
С – словарь;
ДСЧ – датчик случайных чисел с равномерным распределением;
БФВ – блок формирования вариаций;
БФС – блок формирования словаря

Структура ИМ схемы 2 приведена на рис. 2.

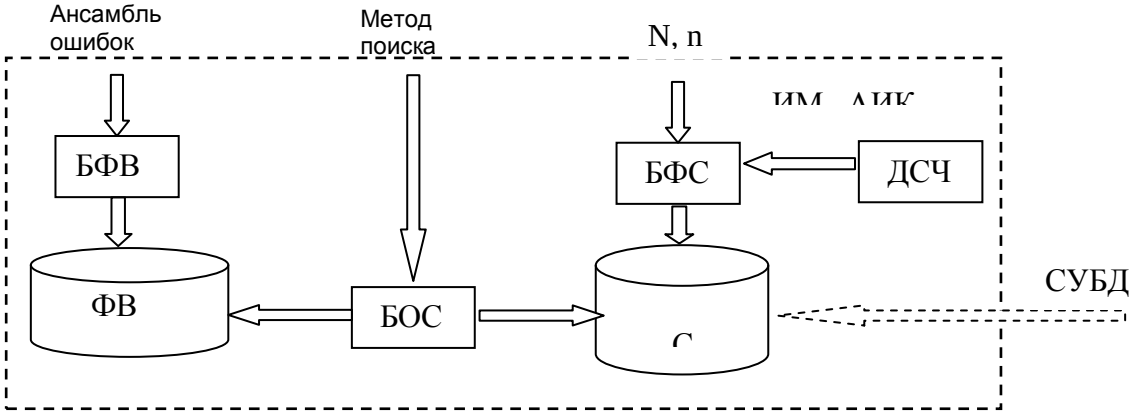

Рис. 2. Структура ИМ схемы 2

БФВ – блок формирования вариаций;
БФС – блок формирования словаря;
БОС – блок обработки словаря;
ДСЧ – датчик случайных чисел с равномерным распределением;
ФВ – файл вариаций;

Отличия процессе ИМ для схемы 2 заключаются в том, что формирование файла словаря производится непосредственно в среде ИМ АИК. В этом случае имитируется процесс переписи словаря из таблицы БД в среду ИМ АИК.

3. Результаты моделирования

3.1. Схема 1

Предварительные испытания схемы 1 показали, что наилучшие временные характеристики обеспечивает алгоритм 1.3. Этот факт иллюстрируют данные табл.1, в которой приведены характеристики в формате мин:сек:мсек для словаря размера $N = 1000$, слов с количеством символов (разрядов) $n = 8, 12, 16$ в алфавите $q = 10$ и набора вариаций $E_1 - E_5$.

Таблица 1. Временные характеристики поиска вариаций для схемы 1

№ Алгоритма	$n=8$	$n=12$	$n=16$
1.1	00:07:982	00:23:364	00:43:022
1.2	00:08:592	00:19:668	00:34:680
1.3	00:07:711	00:17:445	00:31:836
1.4	00:11:216	00:25:056	00:45:766

Тем не менее даже наилучший алгоритм схемы 1 дает для $K(k_j)=(1,\ 2,\ 3,\ 4,\ 5)$ и $N=1000$ (и тем более для $N>1000$) практически неудовлетворительные результаты, особенно для алгоритмов АИК 3, 4 [1], предназначенных для работы on-line. В связи с этим возникает вопрос, при каких ослабленных исходных условиях (уменьшении N и ограничении ансамбля $K(k_j)$) время поиска вариаций в словаре не превышает приемлемого для on-line значения, например, 2 секунды. Исходные данные для ответа на этот вопрос содержатся в табл.2, полученной ИМ для параметров $n=8,\ 12,\ 16$, $q=10$ (десятичные цифры), 36 (латинский алфавит + десятичные цифры).

Как следует из табл. 2, время поиска вариаций для корректировки ошибочного слова монотонно возрастает с ростом N и, с другой стороны, с увеличением q, n и количества классов корректируемых ошибок. Закрашенная область таблицы соответствует совокупностям n, q, N, $K(k_j)$, при которых время поиска вариаций превышает 2 сек. Видно, что корректировка всех пяти классов ошибок при использовании первой схемы практически нереальна.

Таблица 2. Временные характеристики поиска вариаций для алгоритма 3.1 и словарей объема N

$K(k_j)$	N	$q=10$			$q=36$		
		$n=8$	$n=12$	$n=16$	$n=8$	$n=12$	$n=16$
1,4	100	00:00:070	00:00:070	00:00:091	00:00:161	00:00:210	00:00:300
	200	00:00:090	00:00:130	00:00:160	00:00:261	00:00:391	00:00:501
	500	00:00:220	00:00:300	00:00:381	00:00:620	00:00:901	00:01:182
	1000	00:00:501	00:00:591	00:00:731	00:01:231	00:01:812	00:02:373
	2000	00:00:921	00:01:242	00:01:593	00:02:624	00:03:866	00:05:128
1,2,3,4	100	00:00:090	00:00:130	00:00:161	00:00:290	00:00:440	00:00:540
	200	00:00:171	00:00:241	00:00:301	00:00:521	00:00:751	00:00:991
	500	00:00:390	00:00:571	00:00:711	00:01:251	00:01:762	00:02:503
	1000	00:00:751	00:01:091	00:01:382	00:02:464	00:03:555	00:04:737
	2000	00:01:662	00:02:344	00:03:005	00:05:368	00:07:842	00:10:174
1,2,3,4,5	100	00:01:082	00:02:563	00:04:437	00:15:412	00:36:343	01:05:574
	200	00:02:013	00:04:657	00:08:081	00:28:020	01:05:555	01:58:811
	500	00:04:706	00:11:006	00:19:478	01:06:256	02:36:635	04:48:284
	1000	00:09:303	00:21:281	00:38:595	02:16:356	05:20:201	09:35:077
	2000	00:20:169	00:46:758	01:22:719	04:51:449	11:31:034	20:58:800

3.2. Схема 2

Испытания схемы 2 проводились с целью оценки быстродействия алгоритмов поиска для таких входных параметров ИМ: $n=8,12,16$; $q=10,\ 36$; ансамбль ошибок $K(k_j)=(1,\ 2,\ 3,\ 4,\ 5)$, словари размера $N=10^3,\ 10^4,\ 10^5,\ 2\cdot10^5,\ 5\cdot10^5,\ 10^6,\ 2\cdot10^6$.

Временные характеристики результатов экспериментов в формате сек:мсек приведены в табл. 3.

Таблица 3. Сравнительные временные характеристики работы алгоритмов 2.1 и 2.2 в мсек

N	$q=10$						$q=36$					
	$n=8$		$n=12$		$n=16$		$n=8$		$n=12$		$n=16$	
	2,1	2,2	2,1	2,2	2,1	2,2	2,1	2,2	2,1	2,2	2,1	2,2
10^3	00:016	00:014	00:040	00:032	00:078	00:066	00:254	00:204	00:641	00:534	01:270	01:070
10^4	00:018	00:015	00:041	00:034	00:082	00:064	00:274	00:210	00:687	00:540	01:366	01:086
10^5	00:020	00:034	00:044	00:058	00:088	00:092	00:292	00:236	00:725	00:559	01:416	01:088
$2\cdot10^5$	00:020	00:062	00:046	00:088	00:092	00:120	00:300	00:262	00:743	00:591	01:450	01:146
$5\cdot10^5$	00:020	00:148	00:049	00:170	00:094	00:188	00:306	00:340	00:763	00:685	01:476	01:206
10^6	00:022	00:280	00:050	00:310	00:096	00:351	00:314	00:477	00:784	00:811	01:494	01:338
$2\cdot10^6$	00:024	00:551	00:052	00:567	00:100	00:567	00:322	00:735	00:815	01:100	01:529	01:622

- лучшие результаты показывает алгоритм 2.1;

- лучшие результаты показывает алгоритм 2.2.

Из табл. 3 видно, что схема 2 обеспечивает значения времени поиска вариаций более чем на порядок меньшее по сравнению со схемой 1. Закрашенная область таблицы соответствует лучшим временным характеристикам алгоритма 2.2. Как видно из табл. 3, алгоритм 2.1 дает лучшие результаты для алфавита, состоящего из десятичных цифр и словарей N>=10^5, а для алгоритма 2.2 – для алфавита, состоящего из латинских букв + десятичные цифры. Хотя в целом разница в результативности алгоритмов не слишком велика.

4. Выводы

В результате проведенных экспериментов установлено, что применение схемы 2 обеспечивает возможность автоматической и полуавтоматической (с подтверждением пользователя) корректировки полного ансамбля ошибок E_1-E_5 при весьма больших объемах словаря. Время поиска и исправления заметно меньше, чем время коррекции ошибки в подчеркнутом слове, свойственном SpellChecker'ам [3]. Следовательно, практических ограничений со стороны возможностей даже «среднего» компьютера для реализации метода АИК не имеется. Таким образом, полученные данные дополняют результаты [1] в отношении оценок быстродействия описанных алгоритмов АИК.

Схема 2 имитационной модели в перспективе может быть положена в основу программной реализации полнофункциональной подсистемы автоматического исправления ошибок ввода, подобной, например, AfterScan [3], но ориентированной на типовые ошибки пользователя.

СПИСОК ЛИТЕРАТУРЫ

1. Кузьменко Г.Є., Литвинов В.А., Майстренко С.Я., Ходак В.І. Алгоритми і моделі автоматичної ідентифікації та корекції типових помилок користувача на основі природної надмірності // Математичні машини і системи. – 2004. – № 2. – С. 134 – 148.
2. Литвинов В.А. Алгоритм произвольно-последовательной обработки файлов // Кибернетика. – 1975. – № 6. –С. 25 – 28.
3. AfterScan. http://www/afterscan.com/ru.