2. Тузов В.А. Компьютерная семантика русского языка // СПб: Изд-во СПбГУ, 2004. С. 400.
3. Леонтьев Н. Н. Автоматическое понимание текстов: системы, модели, ресурсы: учеб. пособие для студентов лингвистических факультетов вузов. М.: Издательский центр «Академия », 2006. С. 304.
4. Душкин Р. В., Мовчан Д. А. Искусственный интеллект // М.: ДМКПресс, 2019. С. 280.

УДК 681.3:553.98(574.4)

М.М. Чуриев[1], М.А. Гельдыева[1], Ы. Керимов[1], Б.С. Гафуров[2]
[1]Международный университет нефти и газа имени Ягшыгельди Какаева
Ашхабад, Туркменистан
[2]"Dragon Oil" (Turkmenistan) Ltd
Ашхабад, Туркменистан

СОЗДАНИЕ ПРОГРАММНЫХ КОДОВ НИЗКОГО УРОВНЯ ДЛЯ «УСКОРЕНИЯ» АЛГОРИТМОВ

Аннотация. *В данной работе рассматривается возможность применения гибридных программных кодов для ускоренной обработки информации, которую в свою очередь можно успешно применить для реализации определенной деятельности. Авторы предлагают способы «ускорения» программных кодов на основе внедрения кодов языка низкоуровенего программирования Ассемблера.*

M.M. Churiev[1], M.A. Geldiyeva[1], Y. Kerimov[1], B.S. Gafurov[2]
[1]Yagshigeldi Kakaev International University of Oil and Gas
Ashgabat, Turkmenistan
[2]"Dragon Oil" (Turkmenistan) Ltd.
Ashgabat, Turkmenistan

CREATING LOW-LEVEL PROGRAMMING CODES TO «SPEED UP» ALGORITHMS

Abstract. *In this paper, we consider the possibility of using hybrid program codes for accelerated information processing, which in turn can be successfully applied to the implementation of certain activities. The authors of the work propose ways to «accelerate» program codes based on the introduction of low-level programming language codes of Assembler.*

В эпоху всеобщей информатизации было создано множество программ, открыты новые направления, связанные с применением информационно-коммуникационных технологий (далее - ИКТ),

произошла компьютеризация деятельности, автоматизация которой еще 15-20 лет назад казалось просто немыслимым.

В данной работе рассматривается возможность применения гибридных программных кодов для ускоренной обработки информации, которую в свою очередь можно успешно применить для реализации определенной деятельности. В данном случае под определенной деятельностью будем подразумевать биотехнологию, а еще конкретно биоинформатику.

Главная цель Биоинформатики — способствовать пониманию биологических процессов. Отличие Биоинформатики от других подходов состоит в том, что она фокусируется на создании и применении интенсивных вычислительных методов для достижения этой цели. Одной и важнейших частей Биоинформатики является Биоинформатика последовательностей.

Этот раздел биоинформатики занимается анализом нуклеотидных и белковых последовательностей. В настоящее время разработаны эффективные экспериментальные методы определения данных последовательностей [1,2].

Представим себе геном небольшой бактерии – это непрерывная строка длиной в 1-10 миллионов символов, и далеко не вся ДНК кодирует белки. Первый тип биоинформатической задачи – это задачи поиска в нуклеотидных последовательностях особых участков, участков, кодирующих белки, участков, кодирующих РНК [3,4].

И это не всегда простые задачи, например, гены некоторых организмов состоят из чередующихся «осмысленных» и «бессмысленных» фрагментов, и расстояние между «осмысленными» фрагментами может достигать тысяч нуклеотидов [5].

Сравнение последовательностей является важнейшей задачей биоинформатики. Трудно найти современного биолога, ниразу не использовавшего программы Blastp и ClustalX, появление этих программ – уже крупный успех биоинформатики. Но современные биоинформатики недовольны. Их недовольство можно понять - анализируемая информация огромна, результатов сравнения иногда ждут часами, днями ...

Например, в одной из баз данных биоинформатики хранится более 100 миллионов документов с описанием нуклеотидных последовательностей, содержащих в целом более 150 миллиардов символов, что соответствует примерно библиотеке более 100 толстых томов. Найти нужный ген в базе, это все равно, что найти цитату или предложение в такой библиотеке.

Итак, пришло время сформулировать основную цель нашей

работы – разработать код или алгоритм сравнения, который бы позволил за приемлемое время сравнить огромное количество цепочек, которые в свою очередь состоят из огромного количества символов. На сегодняшний день разработаны и применены множество математических приемов и алгоритмов быстрого поиска или анализа, однако даже они не способны существенно сэкономить время. Поэтому будем исходить от скорости непосредственной обработки информации, а точнее операции. За операцию будем считать одно сравнение определенного количества символов двух последовательностей.

Для наглядности и удобства рассмотрим самый простой способ сравнения символов последовательностей – перебор символов пароля.

Рассмотрим конкретный пример с полным перебором 6 символьного пароля, созданного строчными английскими буквами (от последовательности аааааа до последовательности zzzzzz).

Чтобы определить количество последовательностей, достаточно из теории чисел представить следующую формулу:

$$N = X^Y \quad (1)$$

где,

N - количество последовательностей;

X - количество символов применяемого алфавита (в даном случае имеем X = 26);

Y – длина (количество символов) пароля (в данном случае Y=6);

Таким образом,

$$N=26^6= 308\ 915\ 776$$

Переходим ко второй части нашей работы – разработке кода по перебору данного количества последовательностей. Заранее оговорим, что были созданы несколько программ на таких языках программирования, как Delphi, C++, Assembler. В нижеследующей таблице для сравнения приведены результаты работы этих программ на одном и том же статистическом компьютере.

Таблица 1

№	Языки программирования	Размер программы	Время полного перебора (сек) аааааа-zzzzzz	Количество операций за 1 сек
	Turbo Assembler	976 байт	261,2	1 182 786
	C++	316 Кбайт	392,2	787 532
	Delphi	400 Кбайт	635,7	485 943

Как видно из таблицы, наилучший результат показал код,

разработанный на языке низкого уровня, близкого машинному коду – на языке программирования Ассемблер.

В ходе работы анализируя результаты и исследовав код, а также среду языков программирования мы пришли к выводу, что нужно совместить легкость и «маневренность» Ассемблера с возможностью объектно-ориентированного языка Delphi, имеющего почти неограниченные возможности в среде Windows. В результате был разработан комбинированный код Ассемблера, команды которого были интегрированы в среду Delphi и получен действительно потрясающий результат:

Комбинация из 308 915 776 последовательностей была “пройдена” всего за 0,77 сек, это где-то около 400 000 000 операций в секунду. И все это на одном и том же компьютере (рис 1.).

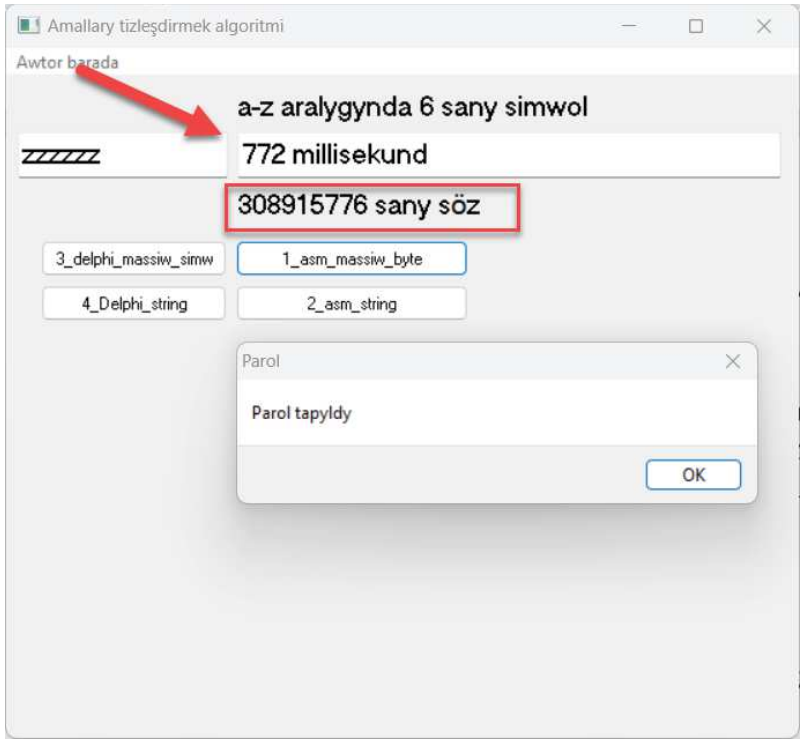

Рис. 1 - Результат работы гибридного кода

Алгоритм данного интегрированного кода, состоял в представлении каждой сравнимаемой последовательности не в качестве строки символов, как обычно это практикутся в языках высокого уровня, а в качестве массива байтов, а реализация осуществлялась не с помощью операторов цикла, а с помощью однобайтовых операций сдвига и сравнения.

Таким образом, был разработан достаточно “быстроходный” алгоритм анализа символьных последовательностей, который можно с успехом применить в биоинформатике при сравнении громоздких

цепочек генов, который позволит за более приемлемое время осущетсвлять поставленные задачи.

Список использованных источников

1. Manual of industrial microbiology and biotechnology. 2nd ed. — Wash., 1999.
2. Патрушев Л. И. Экспрессия генов. — М.: Наука, 2000. — ISBN 5-02-001890-2.
3. Глик Б., Пастернак Дж. Молекулярная биотехнология. Принципы и применение. — М.: Мир, 2002.
4. Егорова Т.А., Клунова С.М., Живу хин Е.А. Основы биотехнологии. — М., 2003.
5. Ivan Y. Torshin Bioinformatics in the Post-Genomic Era: The Role of Biophysics, Novapublishers, 2006, ISBN 1-60021-048-1

УДК 681.3:553.98(574.4)

Д.Д. Чарыева, Л.Р. Новбатова, И.Г. Аширов, И. Какаев
Международный университет нефти и газа имени Ягшыгельди Какаева
Ашхабад, Туркменистан

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ПО ИСПРАВЛЕНИЮ НОСИТЕЛЕЙ ФЛЕШ

***Аннотация**. В статье рассматривается проблема разработки программы для анализа состояния и неисправности некоторых мобильных носителей информации. Как известно, флеш-носитель до сих пор является самым популярным средством для обмена и хранения электронной информации. Программа исправляет ошибки, восстановливает и систематизирует файлы пользователя на флеш-носителе.*

D.J. Charyyeva, L.R. Novbatova, I.G. Ashirov, Y. Kakaev
Yagshigeldi Kakaev International University of Oil and Gas
Ashgabat, Turkmenistan

FLASH MEDIA REPAIR SOFTWARE

***Abstract.** The article examines the problem of developing software for analyzing the state and malfunction of some mobile storage devices. As is known, a flash drive is still the most popular means for exchanging and storing electronic information. The purpose of the program is to correct errors, restore and systematize user files on a flash drive.*