МЕТОД ПРОГРАММНОГО ХРАНЕНИЯ ИНФОРМАЦИИ С ПОМОЩЬЮ ОТОБРАЖЕНИЯ ЕЕ НА ОПОРНОЙ ГАММЕ ВЫРАБАТЫВАЕМОЙ ГЕНЕРАТОРОМ ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ

УДК 621.96

КОНДРАТЬЕВА Виктория Федоровна,
к.э.н., в.о. зав. каф. информационных технологий МКУ им.Ф.Орлика, г. Николаев,
Научные интересы: методы хранения информации, информационная безопасность.
e-mail: ab011089kvf@gmail.com

ЧАЙКА Дмитрий Остапович
зав. лаб. информационных технологий ППИ НУК им.ад. Макарова
Научные интересы: методы хранения информации, информационная безопасность, алгоритмы определения простых чисел

ПОСТАНОВКА ПРОБЛЕМЫ

Современный мир – это бесконечный поток информации и с каждым днем ее количество только увеличивается. По данным всемирной Сети за последние три десятилетия было произведено больше информации, чем за предшествующие 5000 лет. Ее не просто много, ее слишком много. По этому справедливо задаться вопросом о хранении информации и ее восстановлении.

АНАЛИЗ ПОСЛЕДНИХ ИССЛЕДОВАНИЙ И ПУБЛИКАЦИЙ

В настоящее время, практически все организации сталкиваются с проблемой критического нарастания объемов информации. Рост данных ставит перед техническим персоналом любой компании задачу построения современной системы хранения данных. По данным исследования, проведенного компанией Meta Group (Стэмфорд, шт. Коннектикут), объем информации, аккумулируемой компаниями, удваивается каждые 18 месяцев [11]. Как это ни удивительно, большинство компаний пока относятся к вопросу создания структуры хранения данных не слишком серьезно. "Пока что большинство компаний действуют импульсивно. Осознавая рост объема информации, они просто увеличивают число серверов и расширяют объем дискового пространства. Лишь немногие подходят к проблеме стратегически", - говорит Джефф Хайн, руководитель подразделения профессионального обслуживания компании Berkshire Computer Products (Хопкинтон, шт. Массачусетс) [11].

Согласно с Борисовым Н.А., Лукиным А.А. хранение и накопление информации вызвано многократным ее использованием, применением постоянной информации, необходимостью комплектации первичных данных до их обработки

[3]. Для предотвращения потери информации разрабатываются различные механизмы ее хранения, которые используются на всех этапах работы с ней [1]. Хранение информации включает выбор формы компьютеризации или другого способа сохранения первичных данных для долговременного и многократного использования [2].

Мир современных технологий предлагает две основные возможности хранения информации внешняя и внутренняя память. Альтернативой является «облачное» хранение данных. Александр Яковлев, менеджер по маркетингу продукции FTS, исследует резервное копирование данных и восстановление информационной системы из «облака» [10]. Но такая система требует значительных капиталовложений в оборудование и развития информационных технологий. По этому возникает проблема создания альтернативной системы, т.е. программного метода хранения информации, не требующего значительных затрат.

РЕЗУЛЬТАТЫ, ПРЕДЛОЖЕНИЯ

Как известно, компьютерная техника делиться на две части – аппаратную и программную. Некоторые устройства могут быть выполнены как аппаратным так и программным способом вплоть до построения программным методом виртуального компьютера в программном пространстве реального компьютера [2].

До сих пор все виды памяти представлялись только аппаратными средствами так как для хранения одного бита информации необходим какой-то физический элемент – конденсатор или триггер на кристалле микросхем или намагниченный участок на магнитном диске или отражающий свет на лазерном диске. Программного метода хранения не существует [6].

Тем не менее существует всем известный пример программного хранения информации – это генератор псевдослучайных чисел (ГПСЧ) в дальнейшем – генератор (ПСЧ).

Генератор (ПСЧ) представляет собой программу выполненную аппаратным способом, хотя возможно его задание программным способом.

Программа его функционирования примитивна и проста, их существует несколько разновидностей. Отличительной особенностью такого генератора является то, что он может на своем выходе выдавать информацию, объём которой намного больше объёма памяти, необходимой для его функционирования. А именно он выдает информацию длинной 2^n , где n – количество функциональных элементов в его устройстве. Так при n=50 длинна будет $2^{50} = 10^{15}$ бит. Под длинной понимается количество двоичных знаков (единиц и нулей) в последовательности выдаваемой генератором (ПСЧ).

Хотя генераторы существуют многоразрядные мы будем рассматривать одноразрядный генератор (ПСЧ). Не взирая на то что информация генератора (ПСЧ) бессмысленная, важно то, что она не хранилась до этого в генераторе, а синтезирована в нем в процессе работы генератора. Она как бы свернута в нем и при работе генератора саморазворачивается. Такое название генератор (ПСЧ) получил потому что через 2^n тактов работы информация выдаваемая генератором повторяется в отличии от настоящего генератора случайных чисел, информация которого не повторяется никогда. Примером такого генератора может служить обычный лототрон с шариками используемый для розыгрыша числовых лоте-

рей или рулетка. Существуют и электронные генераторы (СЧ). В них в изменяемом полупроводниковом диоде сжимается флуктуация колебаний тока возникаемого от бебиового движения электронов. Эти колебания усиливаются усилителем и далее подсчитываются количество колебаний за определенный период времени например за 1 мск, если количество четное ставиться 0, а если нечетное ставиться 1 и таким образом получают двоичную гамму. Далее ее можно преобразовать в любые коды необходимые для использования в компонентах.

Конечно, было бы заманчиво для любой полезной информации заданного объема синтезировать генератор (ПСЧ), который после запуска выдавал бы нам эту информацию. Так как нанотехнологии уже приближаются к своему пределу 10^{-10} м, то есть размеру атома.

Конечно, создавать генератор для наперед заданной информации это неосуществимая утопия, но если использовать генератор (ПСЧ) как источник одноразрядной двоичной гаммы опорного носителя на который мы сможем наложить полезную информацию, именно потому что эту гамму не нужно хранить в памяти компьютера, а синтезировать каждый раз когда это будет необходимо.

Рассмотрим работу генератора (ПСЧ) на конкретном примере генератора построенного на кольцевом регистре сдвига представляющего собой цепочку одноразрядных ячеек памяти в которых может быть записан 0 и 1.

Цепочка замкнута в кольцо (рис. 1)

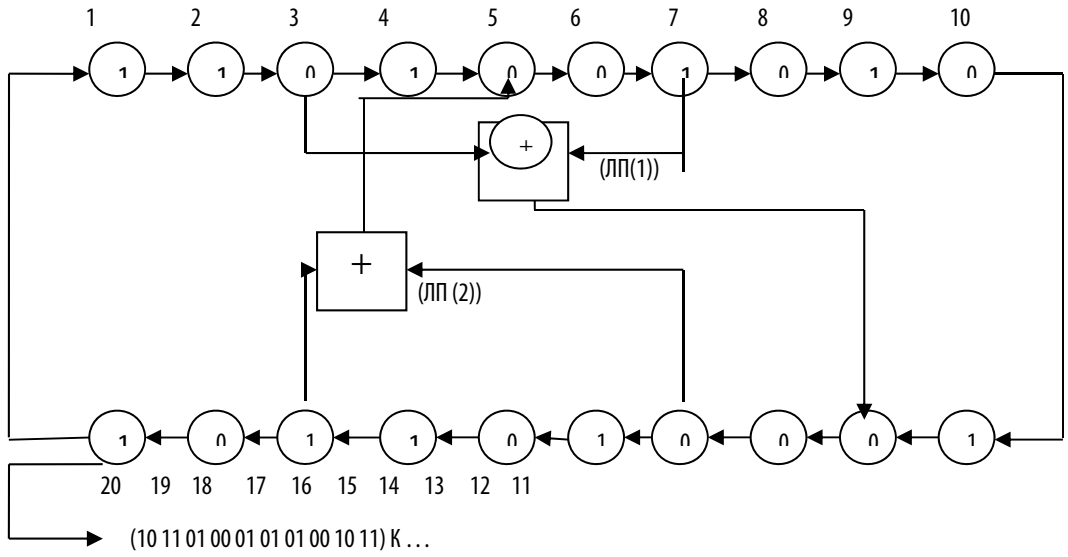

Рис. 1 Генератор (ПСЧ) посторенний на кольцевом регистре сдвига

После каждого такта сдвига информация, которая содержится в ячейке, переписывается в ячейку с номером большим на единицу. Таким образом, информация, предварительно записанная в ячейки этого кольца, будет двигаться по часовой стрелке и через 20 тактов сдвига снова возвращается на свое первоначальное положение. Это похоже на гирлянду бегущих огней, если представить вместо единиц в ячейках светящиеся лампочки. Если будем снимать ячейки №20 информацию, то она будет повторятся через каждых 20 тактов сдвига.

Этот кольцевой регистр сдвига еще не является генератором (ПСЧ) для этого в его схему необходимо ввести так называемые логические перемычки (ЛП) (не менее 2), которые представляют собой устройство которое контролирует информацию в 2 заданных ячейках и в зависимости от этой информации управляют информацией в третьей ячейке например как в Рис.1. ЛП1 между тактами сдвига считывают информацию в ячейках №3 и №7, складываем ее по модулю 2 1+1=0, 1+0=1, 0+1=1, 0+0=0 (то есть двоичное сложение без переноса) и в зависимости от результата управляем информацией в ячейке №12 при этом если результат сложения (1) то информация не изменяется и если результат (0) то информация в ячейке инвертируется, то есть изменяется на противоположную (0) на (1) а (1) на (0). Таким образом, информация записывается перед запуском генератора (ПСЧ) в кольцо регистра (желательно чтобы подряд не шло более трех единиц или нулей) будет в процессе сдвига видоизменятся и на выходе из ячейки №20 будет выходить не повторяющаяся последовательность единиц и нулей длиной $2^{20} \approx 10^6$. Такую последовательность в дальнейшем будем назвать двоичной гаммой

Информация на выходе не повторится через 2^{20} в лучшем случае может случится что она повторится ранее например через 2^{10} тактов, это будет зависеть от информации записанной в кольцо регистра сдвига и от алгоритма работы логических перемычек и каким ячейкам они будут присоединятся. В любом случае 2^{20} тактов могут быть разбиты на несколько периодов которые в сумме будут составлять 2^{20}. Так как в конечном итоге все зависит от кода в кольце, если через какое то количество тактов повторится уже появившийся код то и последовательность повторится именно с этого момента, а кодов длинной 20 может быть 2^{20}. При неблагоприятных обстоятельства в кольце регистра он может быстро вырождаться так как появляется много идущих подряд нулей и /или единиц и гамма быстро повторится. Поэтому необходимо для нашего генератора применять поворотный код, который дает гарантированную длинную двоичную гамму.

Выбрана именно такая схема генератора ПСЧ потому что он обладает одним важным качеством необходимым нам – это обратимость то есть такой генератор (ПСЧ) рекуррентный. При изменении направления вращения в кольцевом регистре сдвига он будет выдавать двоичную гамму в обратном направлении то есть когда информация из ячеек будет переписываться с номером меньшим на 1 от данной то вращение информации в кольце будет против часовой стрелки а логические перемычки будут возвращать информацию в первоначальное состояние и коды в кольце будут меняться в обратном направлении. Кроме того код в кольце является идентификатором места на двоичной гамме, поэтому ненужно ставить ни каких счетчиков если нужно отметить какое то место на гамме чтобы потом быстро вернутся на это место то не нужно прогонять генератор (ПСЧ) до этого места достаточно обнулить информацию в кольце регистра сдвига и записать туда код соответствующий этому месту на гамме и генератор начнет выдавать гамму с этого места.

Обратить гамму в противоположном направлении нам необходимо будет когда будем отображать двоичную гамму полезной информации на двоичной гамме генератора (ПСЧ) или будем начинать с конца полезной двоичной гаммы, а чтобы восстановить полезную гамму необходимо будет двигаться от конца гаммы

генератора к это началу а так как эту гамму мы в памяти компьютера хранить не будем а просто запустим генератор в обратную сторону.

Так как уже было сказано выше мы будем двоичную гамму полезной информации отображать на двоичной гамме генератора (ПСЧ). После чего полезную информацию мы удалим из памяти компьютера, двоичную гамму генератора также хранить не нужно, остается лишь код кольца регистра сдвига на котором мы закончили работу.

Как же мы можем восстановить полезную двоичную гамму и куда же она делась? В этом и заключается отображение полезной двоичной гаммы на двоичной гамме генератора (ПСЧ).

Назовем двоичную гамму генератора для краткости опорной гаммы. В выбранном начале этой опорной гаммы установим курсор в виде подчеркивания длинной два двоичных разряда установленных сверху над гаммой. Под курсором всегда будет два разряда гаммы. Курсор на полезной гамме сдвигается на соседнюю дистанцию и процедура отображения повторяется пока не кончится полезная гамма. При этом курсор на опорной гамме уходит все дальше по опорной гамме в право. Процесс восстановления происходит в обратном порядке, информация под курсором на опорной гамме складывается с информации соответствующей длине шага и получаем два разряда полезной гаммы. К сожалению нам не известно длина шага и у нас осталось только место положения курсора на опорной гамме. Для того чтобы предположить предыдущие места курсора на опорной гамме мы используем избыточность то есть будем отображать информацию полезной гаммы одновременно на двух разных гаммах тогда при восстановлении полезной гаммы не зная предыдущее место курсора на верхней гамме мы сделаем шаги всех четырех длин и каждый раз информацию под курсором сложим по модулю 2 с кодом длины шага и получим таким образом четыре варианта. Такой же курсор установим на полезной гамме (Рис.2).

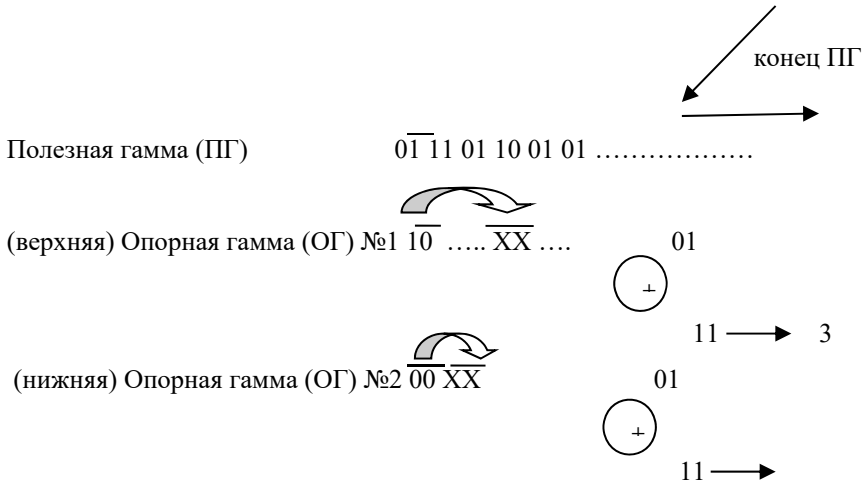

Рис. 2 Отображение информации полезной гаммы одновременно на двух разных гаммах

Полезная информация будет управлять перемещением курсора по опорной гамме, при этом перемещение курсора будет зависеть и он информации под курсором на опорной гамме.

Для этого информация под курсором на полезной гамме складывается (по модулю 2) с информацией под курсором на опорной гамме в результате сложения возможно 4 варианта 01, 10, 11, 00, что будет соответствовать длине шага курсора по опорной гамме на 1, 2, 3, 4 дистанции (1 дистанция – 2 разряда) в приведенном примере на Рис.2 это 3 дистанции.

После этого курсор на опорной гамме перемещается в установленную позицию неопределенности и с каждым шагом в обратном направлении по восстановлении полезной гаммы дерево неопределенности 0 будет увеличивается. Для борьбы с неопределенностью будем применять многоуровневую избыточность.

Первый уровень мы уже применили у нас была полная неопределенность теперь есть частичная ситуативная.

Второй уровень будет заключатся в следующем: каждый ряд полезной информации перекодируем двумя разрядами 1→01, 0→ 10. Тогда 2 кода 00 и 11 у нас будут избыточными и когда при восстановлении на опорной гамме будут ситуативно появляется эти коды мы их воспринимать не будем и неопределенность существенно уменьшиться, но это будет зависеть от ситуации.

Поэтому применим еще прежний уровень избыточности каждый разряд полезной информации перекодированный двумя разрядами мы будем отображать на опорной гамме по 3 раза подряд. Тоже самое мы проделаем и на нижней опорной гамме в результате чего мы отбираем информацию только ту что совпадает на верхней и нижней опорных гаммах так как мы одну и туже информацию под курсором на полезной гамме отображаем на опорных гаммах. А определив информацию мы определим и предыдущие позиции курсоров (Рис.3).

(верхняя) Опорная гамма (ОГ) $\overline{01}.00.\overline{01}.10$
нижняя) Опорная гамма (ОГ) $11.\underline{01}.\overline{11}.11$

Рис. 3. Управление перемещением курсора по опорной гамме

В приведенном примере верхней и нижней опорных гаммах предлагаемая информация совпадает только 01. При этом предыдущее место курсора на нижней опорной гамме определенно абсолютна, а на верхней 2 варианта, то есть возникла неопределенность. В дальнейшем при восстановлении необходимо будет на верхней опорной гамме использовать оба варианта, а там снова будет возникать неустранимая неопределенность так как эти ситуации будут устранятся еще в процессе отображения. Естественно что это только теория, на практике будут возникать трудности которые необходимо будет преодолевать но эти проблемы будут относится к программированию. Так например необходимо иметь 2 опорные гаммы значит необходимо создавать 2 генератора (ПСЧ), но можно обойтись и одним тогда нужно будет расслоить опорную гамму одного генератора взяв в ней разряды через 1 нечетное для первой а четное для второй. Но при этом уменьшится быстродействие, оно и так будет очень низким из за большой избыточности перекодирования, утроенных две части + незначительная информация (2*3*2=12). Для текстовой информации это не существенно, а вот для аудио и видео это значит что для восстановления 1 часа аудио информации нужна неопределенность на тех ветках, где информация не повториться три раза и более или менее эти ветки можно отследить как не достоверные.

Но главное что в процессе отображения проделав некоторое количество шагов отображения мы будем возвращаться или проверять как глубоко уходит дерево неопределенности если оно не исчезает совсем, то есть не остается 100% достоверная информация на определенном количестве шагов применяются дополнительные меры избыточности, небольшие блоки информации например числа имеют номера и они меняются местами. В самых блоках на определенных местах находятся участки с наперед известной незначащей информацией, которую можно менять по заданному алгоритму, тем самым находить ситуации когда неопределенность устраняется. Таким образом при восстановлении информации не будет возникать тупиковых ситуаций неопределенности 12 часов, что неприемлемо. Это можно устранить только параллельной работой 12 процессоров или создавать специальные процессоры содержащие устройство отображения- восстановления в необходимых количествах. Процесс отображения будет еще медленнее из-за постоянных возвратов для проверки определенности восстановления. Как уже говорилось для поиска необходимых мест начал глав и разделов необходимо отмечать эти места запоминая коды кольца регистра сдвига и формируя каталоги с такими кодами.

РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЯ

Но не взирая на все трудности мы получим принципиально новый вид памяти не требующий носителя, где можно хранить колоссальные массивы информации. Можно сразу ожидать возражения что информация не сжимается и на одной и той же гамме нельзя записать множество различных видов информации. На что можно возразить что действительно это так, но мы не будем отображать на одной и той же гамме если возьмем генератор $2^{100} \approx 10^{30}$ бит. Этого хватит на многие годы что бы отображать каждый раз на новом участке, кроме того мы можем применять смешанные опорные гаммы, когда к гамме генератора через каждые 10 разрядов будем подмешивать один разряд полезной информации из участка далеко стоящего вправо от курсора, тогда каждой отображаемой информации будет соответствовать единственная на структуре гамма и противоречие будет устранено.

ЛИТЕРАТУРА:

1. Akulov L. G., Naumov, V. Yu. Storage and protection of computer information.– Volgograd: Volgograd GTU, 2015. – 260 p.
2. Beshenkov S. A., Lyskova Y. V., Rakitina E. A. Information and information processes.- Omsk, 2010.- 180 p.
3. Borisov N. A. Lukin A. A. computer Information network. – M.: EMPA them. A. S. Griboedov, 2012. – 63 p.
4. Bykov A. He who owns the information, or the Staff as a weak element of the security system company // Kadrovik. - 2014. - N 5 .- P. 12-14.
5. Dorokhov V. E. computer network. – M.: EMPA them. A. S. Griboedov, 2011. – 120 p.
6. Kaplunova N. I., Sharygin V. V., Khmel'nyts'kyi S. V. The Concept of development of information resources. // Under the editorship of S. B. Khmelnitsky. - SPb.: European University at Saint-Petersburg, 2011. – 280 p.
7. Sokolov A. V., Stepanyuk O. M. Methods of information security of objects and computer networks.- M., 2011.- 200 p.
8. Topilin Ya. The regulations on licensing system of access to the information resources of the organization that contain personal data (employees, customers, citizens). - 2013 .- N 1 .- S. 18-24
9. Shubin A. S. Our Secret crying out loud.... // Protection inform. Insider. - 2012 .- N 1 .- S. 19-27
10. http://www.aq.ru/about/press-center/news/udobnee-hranit-i-bystree-schityvat
11. http://www.citforum.ru/cfin/articles/problcorpinf.shtml

Рецензент: д.т.н., доц. Бандура В.М.