**План-конспект занятия по информатике.**

Город: **Раменское. МОУ «СОШ № 8»**

Учитель: **Константинова Елена Ивановна**

Класс: **11 «А»**

**Тема учебного занятия**: «Алгоритмы сжатия. Алгоритм построения орграфа Хаффмана»

**Продолжительность учебного занятия**: урок 45 минут

**Тип учебного занятия**: комбинированный (объяснение нового материала +практическая работа)

**Цель урока**: условие рассмотрения алгоритма сжатия на основе построения орграфа Хаффмана.

**Задачи (**образовательная, развивающая, воспитательная):

**Образовательная:**

1. Познакомить учащихся с понятиями алгоритм сжатия, дерево, дуга, орграф, орграф Хаффмана.
2. Рассмотреть пример построения орграфа Хаффмана, и чем он хорош при нахождении коэффициента сжатия.
3. Организовать практическую работу по нахождению частотности букв русского алфавита в тексте, освоению алгоритма построения дерева Хаффмана.

**Развивающая:**

1. Развивать у учащихся познавательный интерес к курсу «Математические основы информатики».
2. Развивать алгоритмическое мышление, память.
3. Развитие практических навыков.

**Воспитательная:**

1. Способствовать воспитанию у учащихся внимательности.
2. Воспитывать аккуратность ведения записей в тетради.
3. Привитие навыка самостоятельности в работе.
4. Воспитание трудолюбия и чувства уважения к науке.

**Оборудование:** АРМ учителя, мультимедийный проектор, интерактивная доска.

**Программное обеспечение**: операционная система WinXP, Microsoft Office, Smart Board.

**Дидактические материалы к учебному занятию**: мультимедийная презентация **«Алгоритмы сжатия. Алгоритм построения орграфа Хаффмана»**, конспект урока, опорный конспект.

**Подготовка к уроку**: подбор материалов для теоретической информации, практической, подбор материалов для презентации, создание опорного конспекта.

**Педагогическая технология**: урок-исследование

***Методы обучения:***

1. Словесные (объяснение)
2. Наглядные (презентация)
3. Практические (упражнения)

**Ключевые слова, опорные понятия:** орграф, дуга (ветвь), корень, дерево, орграф Хаффмана.

**Ход учебного занятия:**

1. Организация начала урока (2 мин)
2. Подготовка учащихся к усвоению(5 мин)
3. Изучение нового материала (15 мин).
4. Закрепление знаний (выполнение практической работы) (15мин)(в перерыве –физкультминутка)
5. Подведение итогов урока. (3 мин)
6. Информация о домашнем задании (5 мин)

**Формы:** *лекция, практикум*

**Ход урока:**

1. Организация начала урока.

Приветствие учителя.

1. Подготовка учащихся к усвоению.

Повтор предыдущего урока.

Вопросы учащимся:

- дать определение префиксному коду;

- дать определение ориентированному графу (орграфу);

- дать определение корню (дереву);

-дать определение дуге (ветви).

1. Изучение нового материала (лекция).

Построить код Хаффмана для фразы «**НА\_ ДВОРЕ\_ ТРАВА,\_ НА\_ ТРАВЕ\_ ДРОВА»**. Определить коэффициент сжатия для данной фразы и сравнить его, если каждый символ кодируется в ASCII.

Сжатие информации - проблема, имеющая достаточно давнюю историю, гораздо более давнюю, нежели история развития вычислительной техники, которая (история) обычно шла параллельно с историей развития проблемы кодирования и шифровки информации.   
  
Все алгоритмы сжатия оперируют входным потоком информации, минимальной единицей которой является бит, а максимальной - несколько бит, байт или несколько байт.   
  
Целью процесса сжатия, как правило, есть получение более компактного выходного потока информационных единиц из некоторого изначально некомпактного входного потока при помощи некоторого их преобразования.   
  
**Построение алгоритма Хаффмана**.

*Коды или Алгоритм Хаффмана (Huffman codes) — широко распространенный и очень эффективный метод сжатия данных, который, в зависимости от характеристик этих данных, обычно позволяет сэкономить от 20% до 90% объема.*  
Рассматриваются данные, представляющие собой последовательность символов. В алгоритме Хаффмана используется таблица, содержащая частоты появления тех или иных символов.

**НА\_ ДВОРЕ\_ ТРАВА,\_ НА\_ ТРАВЕ\_ ДРОВА**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |

При обычном кодировании мы каждый символ записываем в фиксированном количестве бит, например каждый символ в одном байте, или в двух.

Однако, т. к. некоторые символы встречаются чаще, а некоторые реже − можно записать часто встречающиеся символы в небольшом количестве бит, а для редко встречающихся символов использовать более длинные коды. Тогда суммарная длина закодированного текста может стать меньше.

Алгоритм построения дерева (орграфа Хаффмана) прост.

• Символы входного алфавита образуют список свободных узлов. Каждый лист имеет вес, равный количеству вхождений символа в сжимаемое сообщение.   
• Выбираются два свободных узла дерева с наименьшими весами.   
• Создается их родитель с весом, равным их суммарному весу.   
• Родитель добавляется в список свободных узлов, а двое его детей удаляются из этого списка.   
• Одной дуге, выходящей из родителя, ставится в соответствие бит **1**, другой - бит **0**.   
• Шаги, начиная со второго, повторяются до тех пор, пока в списке свободных узлов не останется только один свободный узел. Он и будет считаться корнем дерева.

Допустим, у нас есть следующая таблица частот:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |

На первом шаге из листьев дерева выбираются два с наименьшими весами - **д** и **,** (запятая)

3

0 1

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |

Они присоединяются к новому узлу-родителю, вес которого устанавливается 2+1 = 3.

Затем узлы **д** и **,** (запятая) удаляются из списка свободных. Узел **д** соответствует ветви **0** родителя, узел **,**(запятая) - ветви **1**.

На следующем шаге то же происходит с узлами **е** и **н**, так как теперь эта пара имеет самый меньший вес в дереве.

*3 4*

*0 1 0 1*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |

Создается новый узел с весом **4**, а узлы **е** и **н** удаляются из списка свободных.

*3 4 4*

*0 1 0 1 0 1*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |

Создается точно такой узел с весом **4**, а узлы **о** и **т**  удаляются из списка свободных. Создается новый узел с весом **7**, а узел **в** удаляется из списка свободных.

7

1

*4 4*

*0 0 1 0 1 0 1*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |

Создается новый узел с весом **8**, а узел **р** удаляется из списка свободных.

7 8

1 0

*4 1 4*

*0 0 1 0 1 0 1*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |

Создается новый узел с весом **9**, а узел **\_** (пробел) удаляется из списка свободных.

7 8 9

1 0 0

*1 1*

*0 0 1 0 1 0 1*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |

Создается очередной новый узел с весом **13**, а узел **а**  удаляется из списка свободных.

13

1

1 8 9

0 0 0 0

*0 1 0 1 1 0 1 1*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |

На следующем шаге "наилегчайшей" парой оказываются узлы **8** и **9**. Для них еще раз создается родитель, теперь уже с весом **17**.

13 17

1 0 1

*1 0 1 0 1*

*0 0 0 1 0 1 0 1*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |

Узел **8** соответствует ветви **0** родителя, **9** - ветви **1**. На последнем шаге в списке свободных осталось только два узла - это **13** и узел **17**. В очередной раз создается родитель с весом **30** и бывшие свободными узлы присоединяются к разным его ветвям.   
Поскольку свободным остался только один узел, то алгоритм построения дерева кодирования Хаффмана завершается.

Алгоритм Хаффмана представлен на рисунке

30

1 0 1

*1 0 1 0 1*

*0 0 0 1 0 1 0 1*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |

Чтобы определить код для каждого из символов, входящих в сообщение, мы должны пройти путь от листа дерева, соответствующего этому символу, до корня дерева, накапливая биты при перемещении по ветвям дерева. Полученная таким образом последовательность битов является кодом данного символа, записанным в обратном порядке.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **а** | **в** | **д** | **,** | **е** | **н** | **р** | **о** | **т** | **\_** |
| **00** | **010** | **0110** | **0111** | **1000** | **1001** | **101** | **1100** | **1101** | **111** |
| **6** | **4** | **2** | **1** | **2** | **2** | **4** | **2** | **2** | **5** |

**Нахождение коэффициента сжатия.**

Подсчитаем, сколько двоичных символов окажется в сообщении

**«НА\_ ДВОРЕ\_ ТРАВА,\_ НА\_ ТРАВЕ\_ ДРОВА»**

Для этого надо найти произведение числа символов в коде каждой буквы на количество раз, которое эта буква встречается в сообщении, а затем полученные произведения сложить.

Получаем: 2\*6+ 3\*4+ 4\*2+ 4\*1+ 4\*2+ 4\*2 +3\*4 +4\*2 +4\*2 +3\*5 = 95

Поскольку в сообщении используется 10 различных символов, для их кодирования требуется как минимум четырехбитовые цепочки, поэтому после кодирования данного сообщения получается цепочка объемом 120 бит.

Коэффициент сжатия это отношение объема исходного сообщения к объему сжатого. В нашем случае это отношение равно 120 /95=1,26.

На самом деле данное сообщение в памяти компьютера закодировано с помощью ASCII, поэтому на каждый символ отведено 8 бит, тем самым, объем исходного сообщения 240 бит, а коэффициент сжатия составляет 240/95 = 2,53.

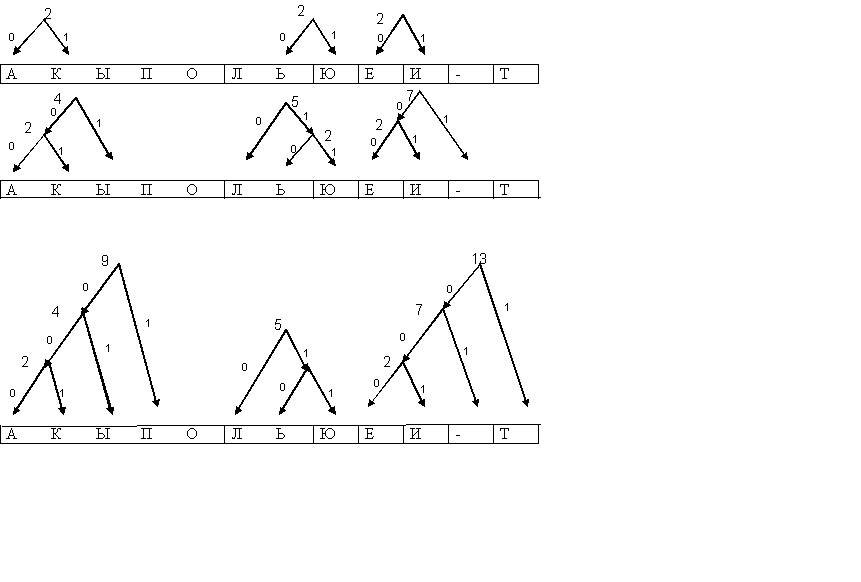
**IV. Закрепление знаний. Практическая работа.**

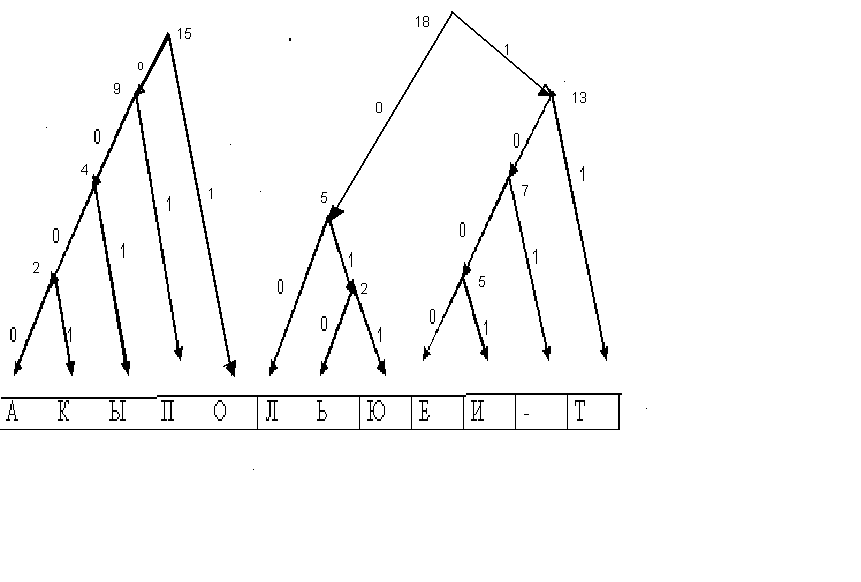
1)построить код Хаффмана для фразы «**ОТ ТОПОТА КОПЫТ ПЫЛЬ ПО ПОЛЮ ЛЕТИТ».**

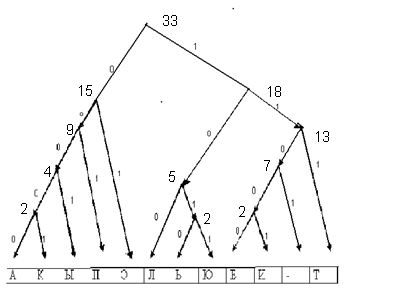
2)определить коэффициент сжатия для данной фразы.

**«ОТ ТОПОТА КОПЫТ ПЫЛЬ ПО ПОЛЮ ЛЕТИТ».**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 2 | 5 | 6 | 3 | 1 | 1 | 1 | 1 | 5 | 6 |
| А | К | Ы | П | О | Л | Ь | Ю | Е | И | - | Т |







б) Определите коэффициент сжатия для данной фразы, если каждый символ кодируется в ASCII.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 00000 | 00001 | 0001 | 001 | 01 | 100 | 1010 | 1011 | 11000 | 11001 | 1101 | 111 |
| А | К | Ы | П | О | Л | Ь | Ю | Е | И | - | Т |

1 1 2 5 6 3 1 1 1 1 5 6

5\*1+5\*1+4\*2+3\*5+2\*6+3\*3+4\*1+4\*1+5\*1+5\*1+4\*5+3\*6=5+5+8+15+12+9+4+4+5+5+20+18=10+20+15+17+48=30+32+48=62+48+110

33\*8=264

264/110=2,4 – коэффициент сжатия

1. **Подведение итогов урока.**

Из этого видно, какой выигрыш мы получили, если это сообщение нужно было бы передать по каналу связи или сохранить на каком-либо носителе.

Для декодирования сжатого сообщения вместе с ним обычно пересылают не коды исходных символов (т.е. первые две строки), а сам орграф Хаффмана (без указания веса корня и разметки на дугах, ибо она стандартна: дуга, идущая влево, размечается 0, а идущая вправо -1).

На этом, оказывается, то же можно сэкономить.

Математики доказали, что среди алгоритмов кодирующих каждый символ по отдельности и целым количеством бит алгоритм Хаффмана обеспечивает наилучшее сжатие.

**VI. Домашнее задание**: 1) конспект урока;

2)построить код Хаффмана для фразы **«ШЛА САША ПО ШОССЕ И СОСАЛА СУШКУ».**

3)определить коэффициент сжатия для данной фразы.

**Список литературы:**

А.Г. Гейн. Математические основы информатики.2008. Педагогический университет «Первое сентября».