Data Structures and its Applications

V R BADRI PRASAD
Department of Computer Science & Engineering

DATA STRUCTURES AND ITS APPLICATIONS

Implementation of TRIE Trees :

- Display Operation
- Deletion Operation
- Search Operation

V R BADRI PRASAD

Department of Computer Science & Engineering

TRIE TREES – Display function

```
void display(struct trienode *curr)
 {
  int i,j;
  for(i=0;i<255;i++)
  {
   if(curr->child[i]!=NULL)
   {
     word[length++]=i;
     if(curr->child[i]->endofword==1)//if end of word
     {
      printf("\n");
      for(j=0;j<length;j++)
        printf("%c",word[j]);
     }
    display(curr->child[i]);
     }
   }
  length--;
  return ;
}
```

Data Structures and its Applications

TRIE Trees – Deletion Operation

```
void delete_trie(struct trienode *root, char *key)
 {
   int i,index,k;
   struct trienode *curr;
   struct stack x;
   curr=root;

   for(i=0;key[i]!='\0';i++)
   {
     index=key[i];
     if(curr->child[index]==NULL)
     {
        printf("The word not found..\n");
        return;
      }
     push(curr,index);
     curr=curr->child[index];
   }
   curr->endofword=0;
   push(curr,-1);
```

TRIE Trees – Deletion Operation

```
while(1)
  {
    x=pop();
    if(x.index!=-1)
      x.m->child[x.index]=NULL;
    if(x.m==root)//if root
     break;
    k=check(x.m);
    if((k>=1)||(x.m->endofword==1))
      break;
    else
      free(x.m);
   }
  return;
 }
```

Data Structures and its Applications

TRIE Trees – Search Operation

```
int search(struct trienode * root,char *key)
 {

 int i,index;
  struct trienode *curr;
  curr=root;

  for(i=0;key[i]!='\0';i++)
  {
    index=key[i];

    if(curr->child[index]==NULL)
       return 0;
     curr=curr->child[index];
   }
   if(curr->endofword==1)
      return 1;
  return 0;
}
```

THANK YOU

V R BADRI PRASAD

Department of Computer Science & Engineering

badriprasad@pes.edu