3.0 .-ARBOLES ABARCADORES Y COMPONENTES CONEXOS

3.1.- CONCEPTO DE ARBOL ABARCADOR Y SU RELACION CON LOS RECORRIDOS.

3.2.- BOSQUES Y COMPONENTES CONEXOS.

NEXON LENIN CEFERINO POMPOSO

Los árboles son particularmente útiles en informática, pues son empleados en un amplio espectro de algoritmos. Los árboles se pueden emplear para estudiar juegos como las damas, el ajedrez y pueden ayudar a determinar estrategias ganadoras para cada uno de estos juegos.

Los árboles pueden utilizarse para modelar procedimientos que se llevan a cabo mediante una secuencia de decisiones.

Los procedimientos para construir árboles que contengan todos los vértices de un grafo, incluyendo la búsqueda en profundidad y la búsqueda en anchura, pueden usarse para explorar sistemáticamente los vértices de un grado.

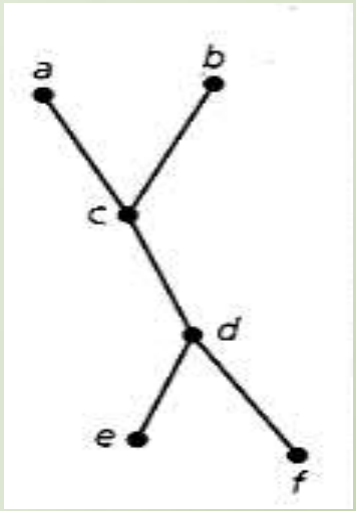
Recorrer los vértices de un grafo haciendo una búsqueda en profundidad, también conocida como vuelta atrás, es la clase de la búsqueda sistemática de soluciones en una amplia variedad de problemas, tales como determinar el modo de colocar ocho reinas en un tablero de ajedrez de modo que dos cualesquiera no se amenacen.

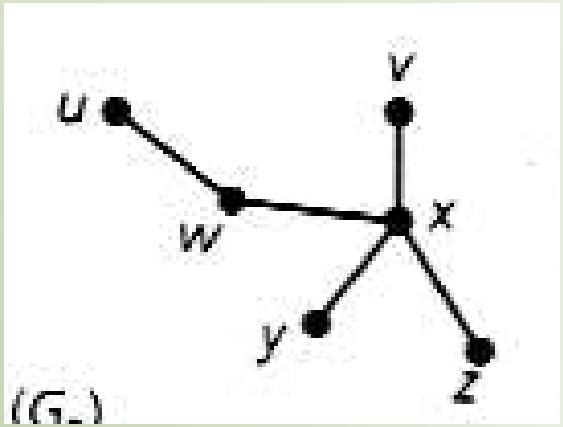
Podemos modelar muchos problemas asignando pesos a los aristas de un árbol. Por ejemplo utilizando árboles ponderados podemos desarrollar algoritmos para construir redes que contengan el conjunto menos costoso de líneas de teléfono que conectan distintos nodos de la red.

3.1.- CONCEPTO DE ÁRBOL ABARCADOR Y SU RELACION CON LOS RECORRIDOS.

Definición: un árbol es un grafo no dirigido, conexo y sin ciclos.

Puesto que un árbol no puede contener ciclos, es un grafo acíclico, tampoco puede tener bucles o aristas múltiples. Por lo tanto, un árbol es necesariamente un grafo simple.





Definición:

Árbol abarcador: sea G un grafo simple. Un árbol abarcador de G es un subgrafo de G que es un árbol y contiene todos los vértices de G.

Teorema 1: si a y b son vértices de un árbol R=(V,A) entonces hay un camino único que conecta estos vértices.

Demostración : primero se observa que en un árbol, todos los caminos son simples (no hay ciclo). Como R es conexo, hay al menos un camino que conecta a con b. Si hubiera mas de un camino que conecta dos vértices se podría formar un ciclo.

Teorema 2: si G=(V,A) es un grafo no dirigido, entonces G es conexo si y solo si tiene un árbol abarcador.

Demostración: si G tiene un árbol abarcador R, entonces para todo para a, b de vértices V, un subconjunto de las aristas de R proporciona un camino único entre a y b de modo que G es conexo. Recíprocamente si G es conexo y no es un árbol, elimínese todos los lazos de G. si el subgrafo resultante G1 no es un árbol entonces G1 debe tener un ciclo mas largo. Para dichos ciclos elimínense de G1 una arista diferente. El subgrafo G2 contiene los vértices de G, es n

conexo y no contiene ciclos. De ahí que G2 sea un árbol abarcador para G.

Teorema 3: si R=(V,A) es un árbol, entonces |V|=|A|+1.

Demostración: por inducción en |A|. Si |A| =0 entonces el árbol que contenga a lo sumo k aristas, k ≥ 0. Considérese ahora un árbol donde |A|=k+1. Eliminemos una arista, obtenemos dios subárboles R1=(V1,A1) y R2=(V2,A2) donde |V| =|V1|+|V2| y |A1|+|A2|+1=|A|. Como 0=< |A1|, |A2|

≤ k, por la hipótesis de inducción |Ai|+1= |Vi|, i=1,2. En consecuencia :

|V|=|V1|+|V2|=(|A1|+1(|A2|+1)= (|A1|+|A2|+1)=|A1|+1

Definición: llamaremos vértices colgantes a los vértices de grado 1.

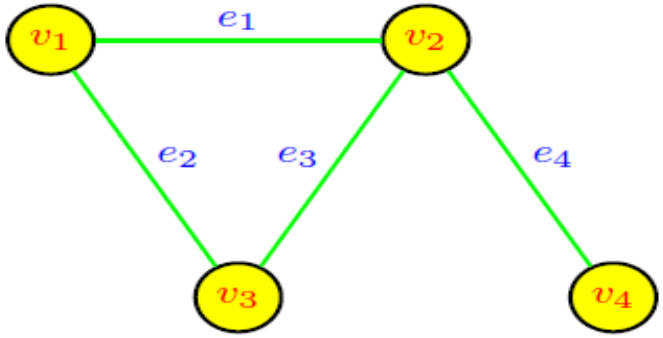
Teorema 4: Para cualquier árbol, R=(V,A), si |V| ≥ 2, entonces R tiene al menos dos vértices colgantes.

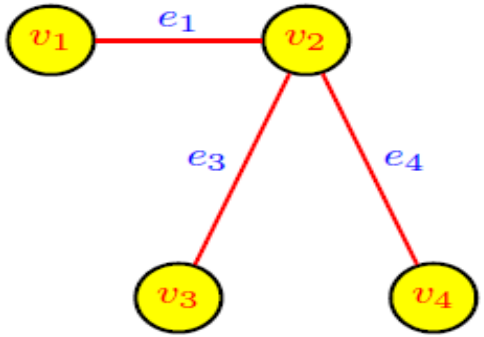
DEMOSTRACION:

Sea $|V| = n \geq 2$. Por el teorema 4, $|A| = n - 1$, por un teorema anterior: $2(n-1) = 2n-2 = 2|A| = \Sigma v \varepsilon V \; grad(v)$. Si $grad(v) \geq 2$ para $v \varepsilon V$, entonces $\Sigma v \varepsilon V \; grad(v) \geq 2|V| = 2n$. De donde $2n - 2 \geq 2n$ lo cual es absurdo, luego hay al menos dos v vértices v para los que $grad(v) < 2$. Como no hay vértices aislados, estos deben ser colgantes.

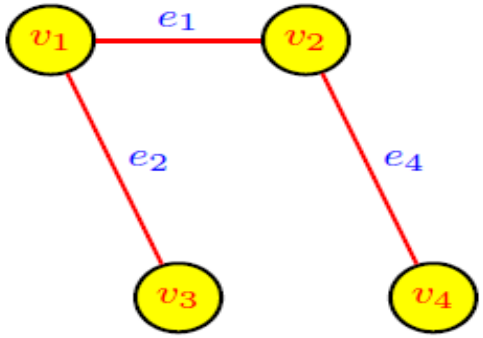
Ejemplos árboles abarcadores para un grafo

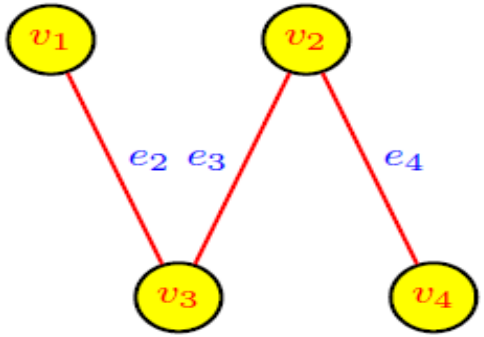
Considere el grafo:





Árboles abarcadores son:





ÁRBOLES CON RAIZ.

DEFINCION: Es un árbol en el que uno de sus vértices ha sido designado como la raíz y todas las aristas están de modo que se alejan de la raíz.

En un árbol con raíz, un vértice *v* con grado de salida igual a 0 se denomina hoja (o vértice terminal).
Los restantes vértices se denominan *nodos de ramificación* (o *vértices internos*).
En la figura 3 el camino de la raíz *r* hasta *s* es de longitud 2, por lo tanto se dice que *s* está en el nivel 2 del árbol.
De manera análoga *x* está en el nivel 3.
A *s* se le denomina *hijo* de *p* y a *p* padre de *s*.
Los vértices *y, x* se consideran *descendientes* de *n, r* y *n, r* se denominan *ancestros* de *y, x*.

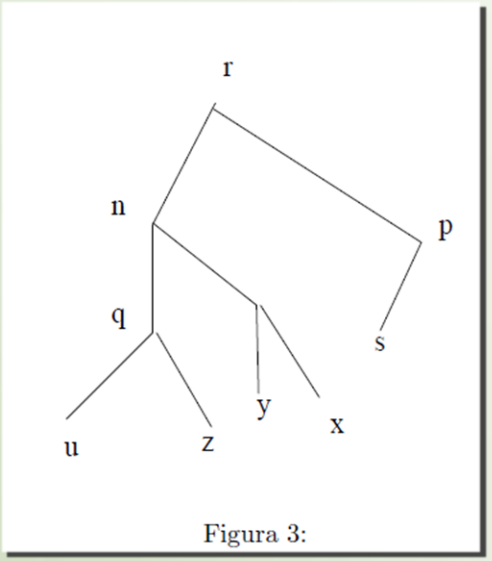
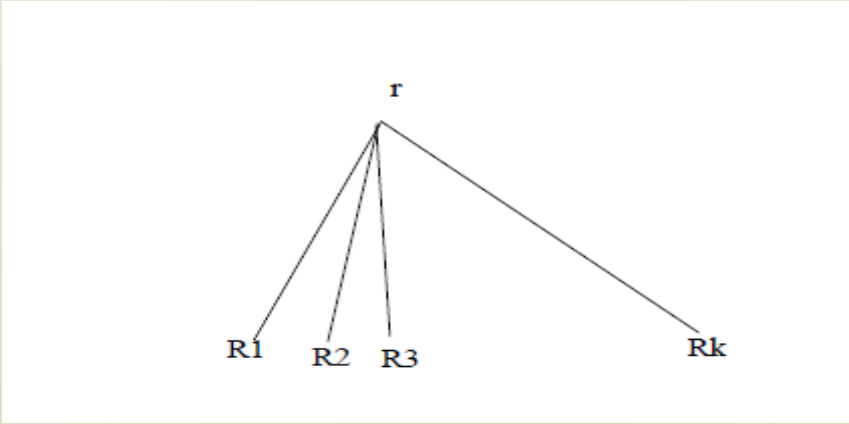


Figura 3:

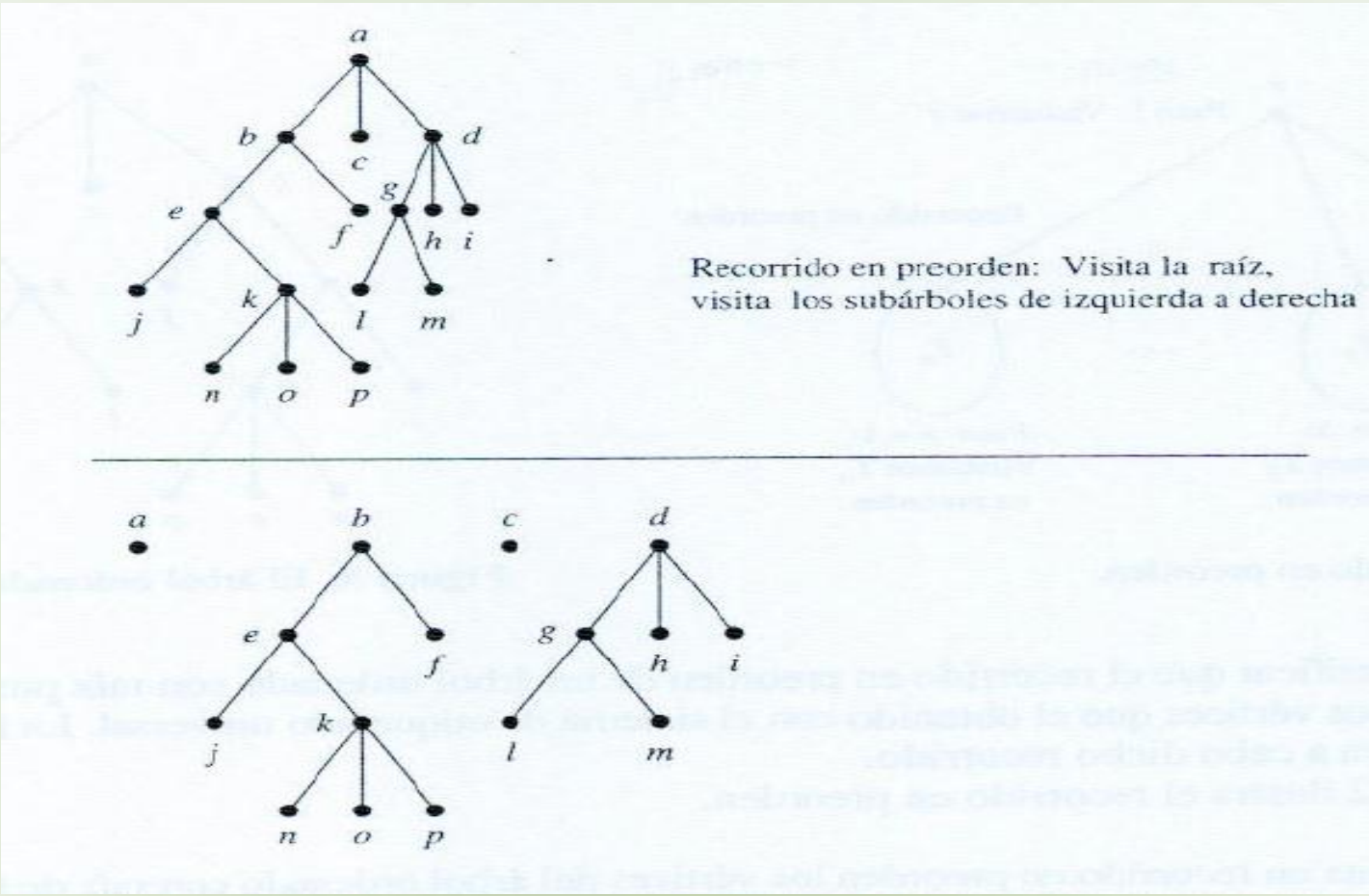
En general si *v*1, *v*2 son vértices de un árbol con raíz, existe un camino de *v*1 a *v*2 y *v*1 tiene menor número de nivel entonces *v*1 es ancestro de *v*2. .Dos vértices con un padre en común se denominan hermanos. Por último i *v*1 es cualquier vértice del árbol, el *subárbol* en *v*1 es el formado tomando *v*1 como raíz y sus descendientes (puede no haber descendientes).

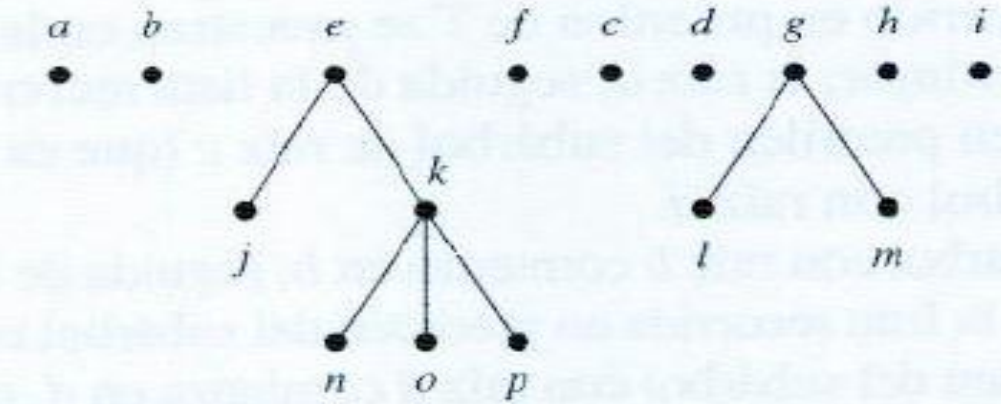
DEFINICION

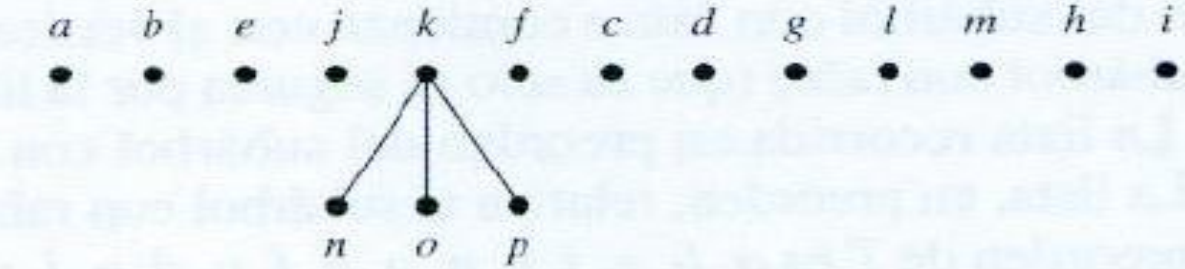
RECORRIDOS: Sea R=(V,A) un árbol con raíz. Supongamos |V|>1. Sean R1,R2,R3…..Rk los subárboles de R según se va de izquierda a derecha.

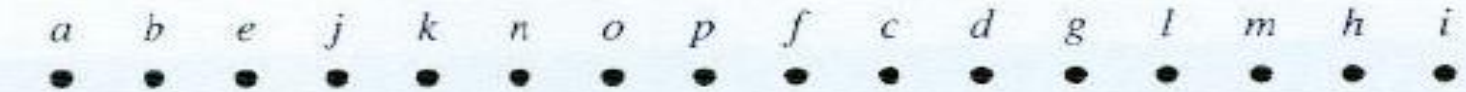


1. El recorrido en orden previo *de R comienza en la raíz r y después pasa por los vértices de R*1 *en orden previo, a continuación por los vértices de R2 en orden previo y así sucesivamente hasta que se pasa por los vértices de Rk en orden previo. Ejemplo:*

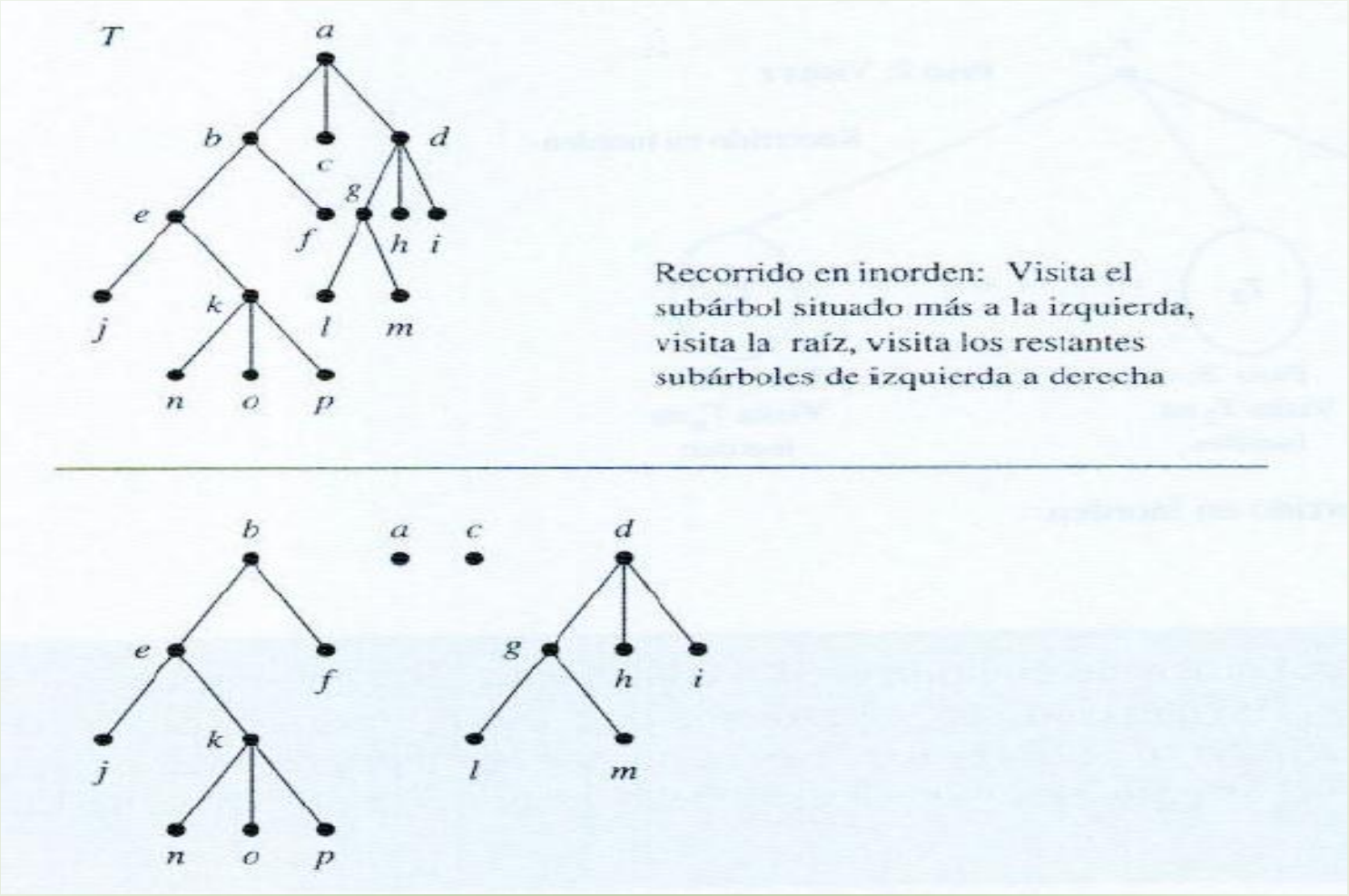


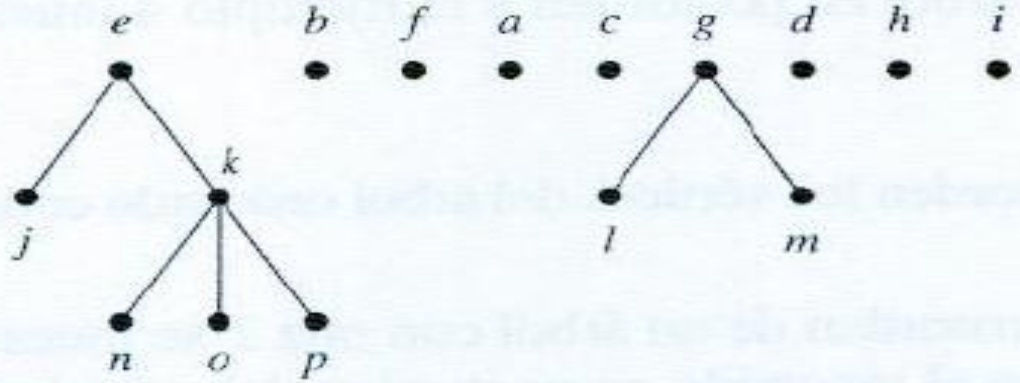

a b e f c d g h i
k
j
l m
n o p

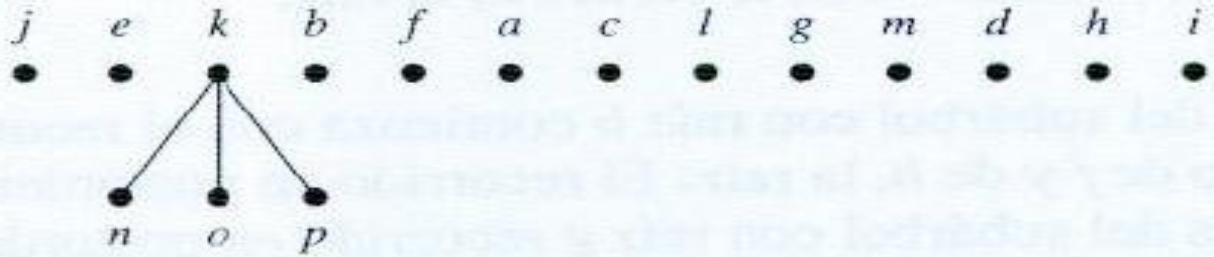

a b e j k f c d g l m h i
n o p

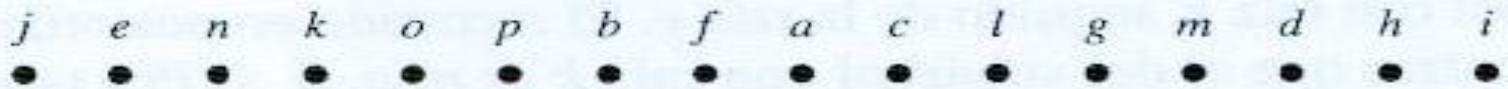

a b e j k n o p f c d g l m h i

2. El recorrido en orden simétrico *de R primero pasa por los vértices de R*1 *en orden simétrico, después por la raíz r y a continuación por los vértices de los subárboles R2 …… Rk en orden simétrico.*

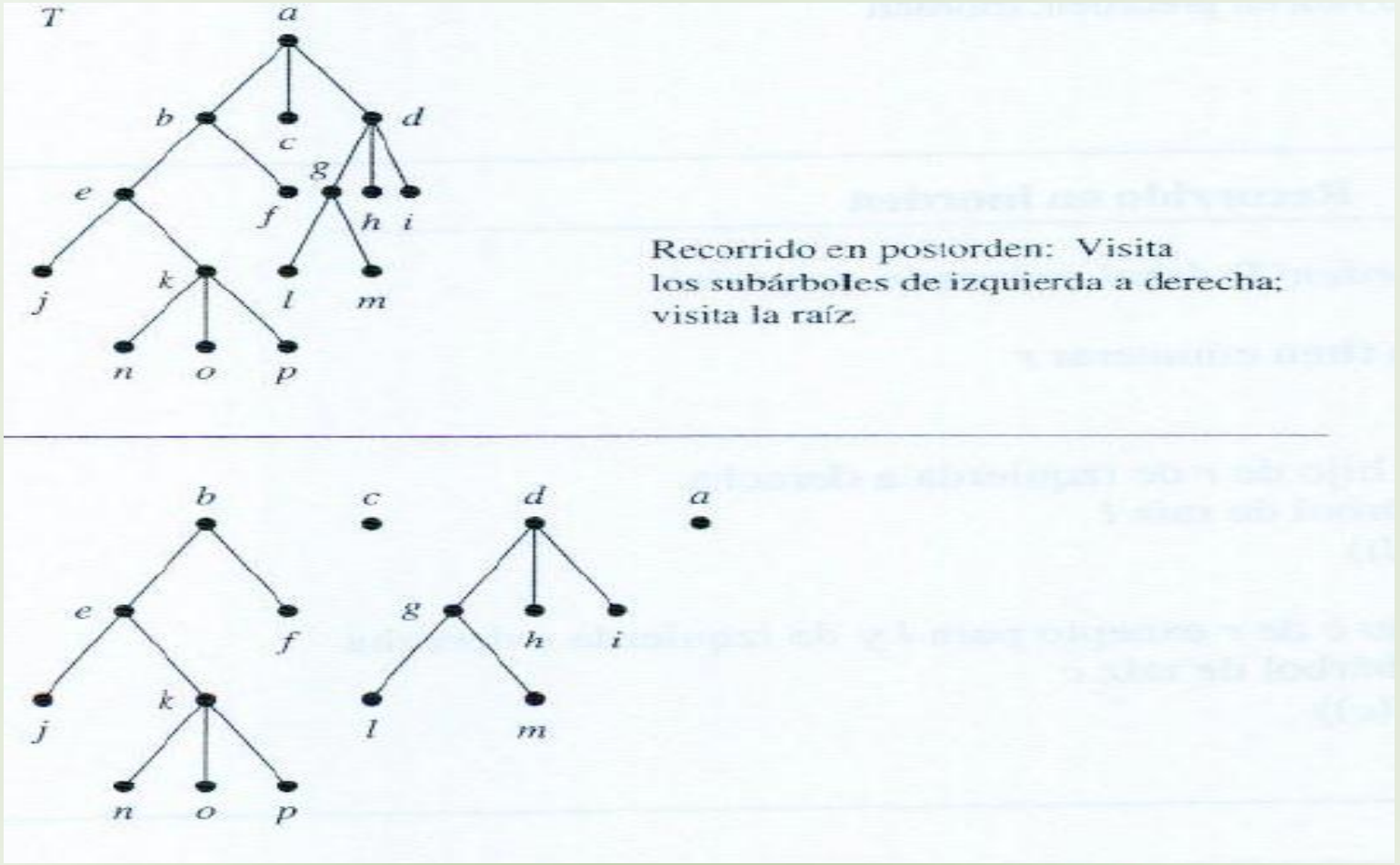


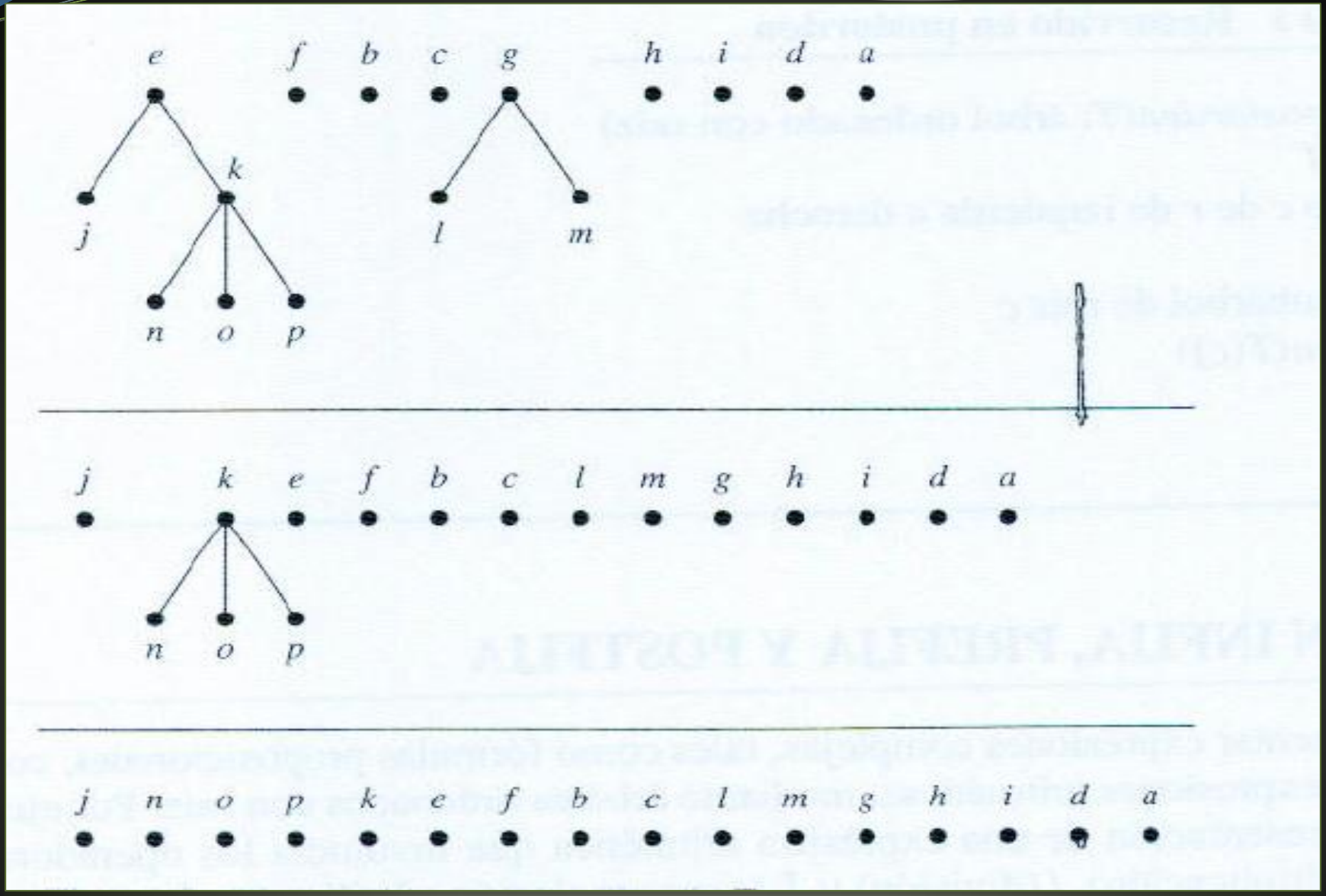

e
b
f
a
c
g
d
h
i
k
j
l
m
n
o
p


j
e
k
b
f
a
c
l
g
m
d
h
i
n
o
p


j
e
n
k
o
p
b
f
a
c
l
g
m
d
h
i

3. El recorrido en orden posterior *de R pasa por los vértices de los subárboles R1;R2……Rk en orden posterior y a continuación por la raíz.*

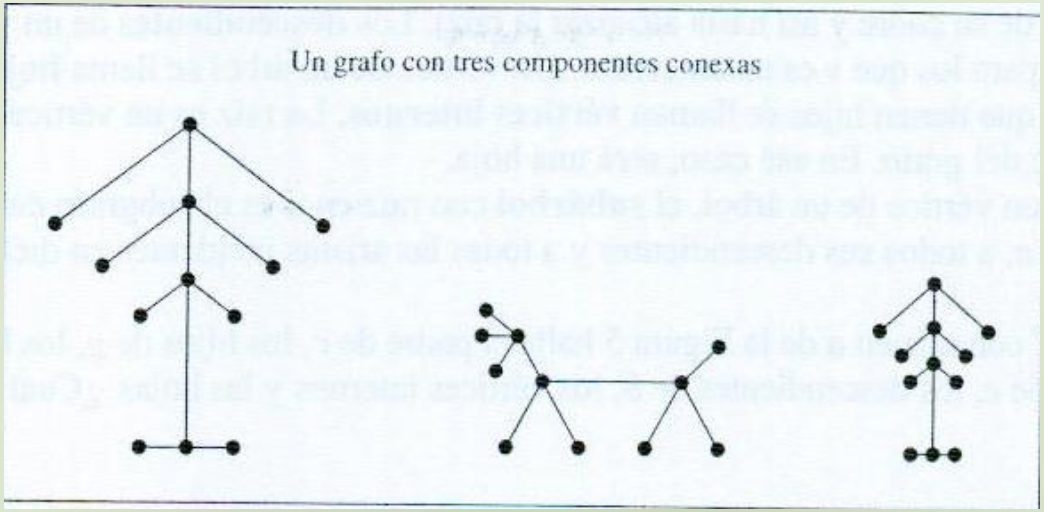



e f b c g h i d a
k
j
l m
n o p
j k e f b c l m g h i d a
n o p
j n o p k e f b c l m g h i d a

3.2 BOSQUES Y COMPONENTES CONEXOS

DEFINICION BOSQUE: Un bosque es un conjunto de árboles o de otra manera podemos decir que un bosque es un grafo acíclico, de dice que el grafo es acíclico si no se tiene ningún ciclo simple.

Ejemplo de bosque.



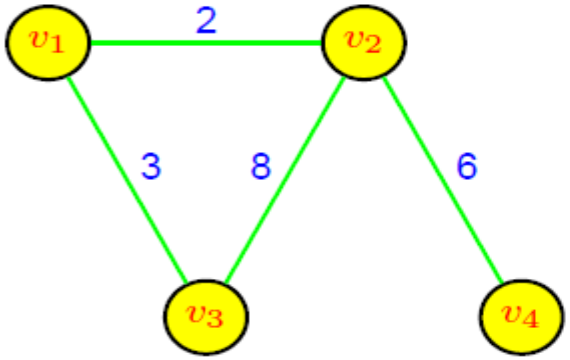
Un componente conexo es cada árbol que conforma al bosque.

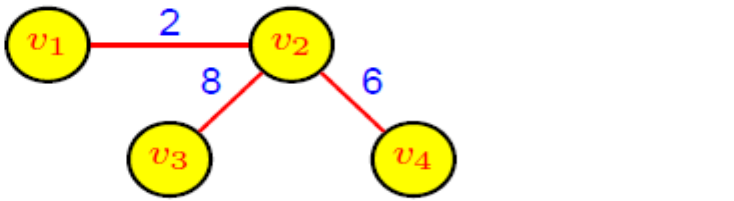
Sea G un grafo un árbol abarcador de G es un grafo conexo que tienen los mismos vértices que G y no tiene ciclos.

Árbol Abarcador Mínimo (Minimum Spanning Tree)

Un *árbol abarcador mínimo de un grafo ponderado es un árbol generador tal* que la suma de los pesos de sus aristas es l mínima de entre de todos los árboles generadores.

Considere el grafo:





Peso árbol abarcador=16

ALGORITMOS PARA CONSTRUIS ÁRBOLES GENERADORES MINIMOS.

ALGORITMO DE PRIM

El algoritmo de Prim es un algoritmo de la teoría de los grafos para encontrar un árbol de expansión mínimo en un grafo conexo, no dirigido y cuyas aristas están etiquetadas.

En otras palabras, el algoritmo encuentra un subconjunto de aristas que forman un árbol con todos los vértices, donde el peso total de todas las aristas en el árbol es el mínimo posible. Si el grafo no es conexo, entonces el algoritmo encontrará el árbol de expansión mínimo para uno de los componentes conexos que forman dicho grafo no conexo.

El algoritmo de Prim construye un árbol de recubrimiento mínimo sobre un grafo ponderado no dirigido. Se considera el coste del árbol la suma de los pesos de las aristas que lo componen. La construcción del árbol sigue los siguientes pasos.

Algoritmo:

La idea básica consiste en añadir, en cada paso, una arista de peso mínimo a un árbol previamente construido:

1. Empezar en un vértice arbitrario v. El árbol consta inicialmente sólo del nodo v.
2. Del resto de vértices, buscar el que esté más próximo a v, es decir, con la arista (v, w) o (w, v) de coste mínimo. Añadir w y la arista al árbol.

3. Buscar el vértice más próximo a cualquiera de estos dos. Añadir ese vértice y la arista al árbol de expansión.
4. Repetir sucesivamente hasta añadir los n vértices.

El siguiente link te mostrará ejemplos del uso del algoritmo de PRIM.

http://www.youtube.com/watch?v=O8XEOz8FCDQ

Este link te lleva a una aplicación de ejercicios del algoritmo PRIM.

http://students.ceid.upatras.gr/~papagel/project/prim.htm

ALGORITMO DE KRUSKAL

El objetivo del algoritmo de Kruskal es construir un árbol (subgrafo sin ciclos) formado por arcos sucesivamente seleccionados de mínimo peso a partir de un grafo con pesos en los arcos.

El siguiente link te mostrará ejemplos del uso del algoritmo de KRUSKAL.

http://www.youtube.com/watch?v=AR_Y88kkh58&feature=related

Este link te lleva a una aplicación de ejercicios del algoritmo PRIM.

http://students.ceid.upatras.gr/~papagel/project/kruskal.htm

- BIBLIOGRAFIA.

P. GRIMALDI Ralph. Matemáticas discreta y combinatoria. 3ra. Ed. Wilmington, Delaware, E.U.A: Addison-wesley Iberoamericana.

H. ROSEN Kenneth. Matemáticas discretas sus aplicaciones. 5ª. ed. McGraw-HILL.

CALDERON VILCA Hugo David. Matemáticas discretas para le ciencia de la computación. Abril 2008. Puno, Perú: Pacifico.