3.6 RECONFIGURACIÓN DE UN ALIMENTADOR PRIMARIO

3.6.1 Descripción del problema.

En la figura 3.6.1 vemos el sistema de distribución que será analizado en este ejemplo, el sistema cuenta con tres alimentadores, 14 barras, 13 ramas del árbol y 3 ramas de enlace.

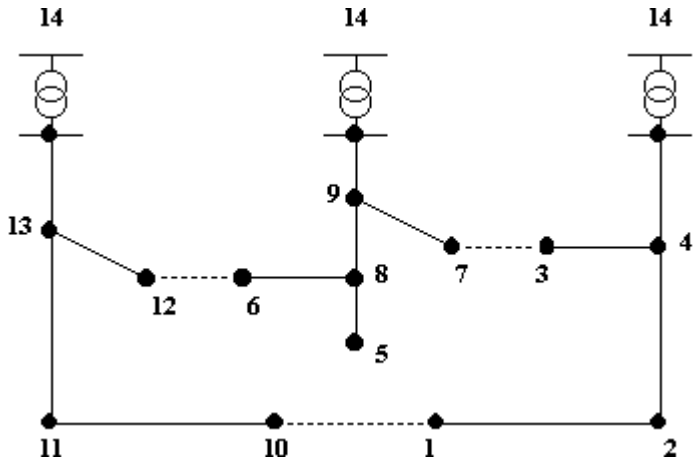

Figura 3.6.1 Sistema de 14 nodos

Cada uno de los nodos representa la carga concentrada como una inyección de potencia, o la división de diferentes ramas y/o ubicación de un seccionador.

Las ramas o líneas continuas interconectan los nodos, transportando la energía demandada y conformando así el árbol; para completar el sistema las líneas punteadas representan líneas desenergizadas formando el coárbol.

Cuando una rama del árbol se intercambia con una rama del coárbol, conservando la estructura radial, la topología del sistema cambia, y a esta maniobra se le llama reconfiguración de la red de distribución.

Bajo condiciones de operación normal la reconfiguración es útil para balancear cargas, evitando sobrecorrientes en las líneas o sobrecargas en los transformadores, mejora los niveles de tensión y reduciendo las pérdidas de potencia activa del sistema; ante aumentos de demanda, la reconfiguración es una herramienta útil en la planeación y diseño de nuevas configuraciones óptimas.

En este ejemplo el objetivo principal es utilizar la reconfiguración como una herramienta para reducción de pérdidas, para ello es necesario correr un flujo de carga antes y después de la maniobra, lo que nos permitirá conocer los valores de las pérdidas de potencia y comprobar así la eficiencia de la operación.

Las pérdidas de potencia activa totales del sistema pueden calcularse como sigue:

$$Pa = \sum_{i=1}^{n} r_i \left[\frac{P_i^2 + Q_i^2}{V_i^2} \right]$$

Donde:

Pa: Pérdidas activas totales del sistema

Pi: Flujo de potencia activa por la rama i

Qi: Flujo de potencia reactiva por la rama i

Vi: Magnitud del voltaje de recibo por la rama i

n :Número de ramas del sistema

El objetivo es encontrar una configuración óptima del sistema de distribución, es decir encontrar todas aquellas posibles configuraciones que con base en la figura 3.6.1 minimicen las pérdidas; nuestra función objetivo (F.O) es entonces:

$$F.O = min\left(\sum_{i=1}^{n} r_i \left[\frac{P_i^2 + Q_i^2}{V_i^2}\right]\right)$$

La reconfiguración óptima debe cumplir con las siguientes restricciones:

- Conservar la topología radial de la red sin que se presenten aislamientos del sistema

$$S_{i\text{-}1} = S_i + S_{Li} \quad (i=2,3,4...n)$$

Donde:

$S_{i\text{-}1}$: Flujo de potencia que sale del nodo i -1

S_i : Flujo de potencia que sale del nodo i

S_{Li}: Potencia de carga en el nodo i-ésimo

n : Número de nodos del sistema

- No sobrepasar las restricciones de voltaje mínimo en la carga

$$V_i \geq V_i^{min} \ (i=1,2,3...,n)$$

Donde:

V_i : Voltaje en el nodo i

V_i^{min} : Voltaje mínimo permitido para el nodo i

n : Número de nodos del sistema

- Cumplir con las especificaciones de carga máxima en los transformadores y de corriente máxima que circule por cada rama

$$S_i \leq S_i^{max} \quad (i= 1,2,3,...,m)$$

Donde:

S_i : Flujo de carga por la rama i, o potencia demandada para el transformador i

S_i^{max} : Capacidad máxima de la rama i, o capacidad máxima del transformador i

n : Número de ramas del alimentador o número de transformadores

En el proceso de encontrar la configuración óptima, pueden diferenciarse dos técnicas, aquellas exhaustivas que estudian todas las combinaciones de estados

del sistema, y otras que proporcionen una ruta heurística que se acerque de manera rápida al estado objetivo.

Una heurística es una técnica que aumenta la eficiencia de un proceso de búsqueda, en ocasiones pueden causar que una ruta sea pasada por alto, pero en promedio, mejoran la calidad de las rutas que exploran. Al usar buenas heurísticas se esperan soluciones a problemas difíciles, que aunque no siempre son las óptimas, podría decirse que son subóptimas.

Un gran número de las metodologías que se han utilizado para resolver el problema de la reconfiguración de alimentadores primarios, se ha basado en técnicas heurísticas; de estos trabajos se destacan: Cinvalar [7], Baran M.E y Wu F.F [4], Goswami K.S y Basu S.K [20], Chen C.S y Cho M [6] y Gallego R.[18], metodología que se empleará en esta aplicación.

En la figura 3.6.1, las líneas punteadas representan los enlaces del sistema para la configuración inicial. Los datos de la estructura del sistema pueden verse en la tabla 3.6.1, cuyos valores base son: Vb=23Kv y Sb=100MVA,

Nodo de	Nodo de	Resistencia	Reactancia	Demanda nodo de recibo	
Envío	Recibo	de rama (pu)	De rama(pu)	P(MW)	Q(MVAR)
14	13	0.0075	0.1	2	1.6
13	12	0.08	0.11	3	1.5
13	11	0.09	0.18	2	0.8
11	10	0.04	0.04	1.5	1.2
14	9	0.11	0.11	4	2.7
9	8	0.08	0.11	5	3
9	7	0.11	0.11	1	0.9
8	6	0.11	0.11	0.6	0.1
8	5	0.08	0.11	4.5	2
14	4	0.11	0.11	1	0.9
4	3	0.09	0.12	1	0.7
4	2	0.08	0.11	1	0.9
2	1	0.04	0.04	2.1	1
12	6	0.04	0.04		
7	3	0.04	0.04		
10	1	0.09	0.12		

Tabla 3.6.1 Estructura del sistema de catorce nodos

Como no existen curvas de carga que simulen el comportamiento de este sistema, se supondrá un agrupamiento según perfiles de demanda de igual tendencia, de acuerdo con el procedimiento empleado por Baran M.E [4]

Grupo	Nodos que conforman el grupo
Grupo #1	6-10-11-13
Grupo #2	4-8-9-12
Grupo #3	1-2-3-5-7

Tabla 3.6.2 Clasificación de las cargas según su tipo

Cada grupo representa un perfil de demanda típico, el grupo #1 Residencial, grupo #2 Comercial y grupo #3 Industrial. Para discretizar las curvas de carga en niveles, aunque estas no se conozcan se pueden utilizar una discretización por

niveles representativos que son un porcentaje de la demanda pico, este criterio es utilizado principalmente por N.I SANTOSO y O.T TAN [39] quienes consideraron que los niveles de carga que deben tenerse en cuenta ocurren para el 50%, 70%, 85% y 100% de la demanda pico. Según lo anterior para el nodo #1, las variaciones de la demanda a tenerse en cuenta son:

Valor de la demanda al 100%	2.0MW+j1.6MVAR
Valor de la demanda al 85%	1.7MW+j1.36MVAR
Valor de la demanda al 70%	1.4MW+j1.12MVAR
Valor de la demanda al 50%	1.0MW+j0.8MVAR

Tabla 3.6.3 Niveles representativos de la demanda pico para el nodo #1

Por lo tanto para tres grupos de demanda, cada uno de ellos con cuatro niveles de carga, el número total de combinaciones de carga es $\mathbf{4^3 = 64}$, que serán los datos con los cuales se entrenará la red, este criterio de discretización puede verse ampliamente en FLOREZ O. y SALAZAR H [11], de donde se extrajo la siguiente tabla que ilustra las 64 combinaciones posibles:

Combinación	Nivel grupo 1	Nivel grupo 2	Nivel grupo3
1	1,00	1,00	1,00
2	1,00	1,00	0,85
3	1,00	1,00	0,70
4	1,00	1,00	0,50
5	1,00	0,85	1,00
6	1,00	0,85	0,85
7	1,00	0,85	0,70
8	1,00	0,85	0,50
9	1,00	0,70	1,00
10	1,00	0,70	0,85

11	1,00	0,70	0,70
12	1,00	0,70	0,50
13	1,00	0,50	1,00
14	1,00	0,50	0,85
15	1,00	0,50	0,70
16	1,00	0,50	0,50
17	0,85	1,00	1,00
18	0,85	1,00	0,85
19	0,85	1,00	0,70
20	0,85	1,00	0,50
21	0,85	0,85	1,00
22	0,85	0,85	0,85
23	0,85	0,85	0,70
24	0,85	0,85	0,50
25	0,85	0,70	1,00
26	0,85	0,70	0,85
27	0,85	0,70	0,70
28	0,85	0,70	0,50
29	0,85	0,50	1,00
30	0,85	0,50	0,85
31	0,85	0,50	0,70
32	0,85	0,50	0,50
33	0,70	1,00	1,00
34	0,70	1,00	0,85
35	0,70	1,00	0,70
36	0,70	1,00	0,50
37	0,70	0,85	1,00
38	0,70	0,85	0,85
39	0,70	0,85	0,70
40	0,70	0,85	0,50
41	0,70	0,70	1,00
42	0,70	0,70	0,85
43	0,70	0,70	0,70
44	0,70	0,70	0,50
45	0,70	0,50	1,00
46	0,70	0,50	0,85
47	0,70	0,50	0,70
48	0,70	0,50	0,50
49	0,50	1,00	1,00
50	0,50	1,00	0,85
51	0,50	1,00	0,70
52	0,50	1,00	0,50
53	0,50	0,85	1,00
54	0,50	0,85	0,85
55	0,50	0,85	0,70
56	0,50	0,85	0,50
57	0,50	0,70	1,00
58	0,50	0,70	0,85
59	0,50	0,70	0,70

60	0,50	0,70	0,50
61	0,50	0,50	1,00
62	0,50	0,50	0,85
63	0,50	0,50	0,70
64	0,50	0,50	0,50

Tabla 3.6.4 Número total de combinaciones

El procedimiento para determinar la topología que disminuye las pérdidas para los 64 datos de entrada es [11]:

1. Leer los datos del sistema original
2. Leer una fila de los niveles de carga, es decir una fila de la tabla 3.6.4
3. Multiplicar la demanda pico de cada barra, por el nivel correspondiente leído en el numeral anterior.
4. Ejecutar el algoritmo de Gallego R [18], con los valores de carga establecidos en tres.
5. Ejecutar un flujo de carga radial, con la configuración encontrada en cuatro para determinar la efectividad del reconfigurador.
6. Repetir los pasos anteriores para cada una de las 64 combinaciones

Mediante este procedimiento se encontró que las dos topologías que garantizaban una disminución en las pérdidas de potencia activa son las ilustradas en la tabla 3.6.5:

Número de Configuración	**Ramas del Coárbol**		
1	9 - 7	8 - 6	10 - 1
2	11 - 10	9 - 7	8 - 6

Tabla 3.6.5 Configuraciones óptimas después de evaluar las 64 combinaciones

Para formar el set de entrenamiento los niveles de carga para cada nodo se representan por un vector de cuatro elementos el cual tendrá un uno en la posición que corresponda al nivel de carga, por consiguiente el set de entrenamiento estará conformado por una matriz de 54 filas, a las 52 primeras corresponden los niveles de carga en forma binaria y los dos últimos a la configuración que disminuye las pérdidas; cada columna es un patrón de entrenamiento. De esta forma las entradas a la red serán siempre 0 o 1, los patrones de entrenamiento de la red se presentan en la tabla 3.6.6

		Ejemplo 1										Ejemplo 64							
Fila1 →	Barra #1	1	1	1	1	1	1	1	1	.....		0	0	0	0	0	0	0	0
Fila2 →		0	0	0	0	0	0	0	0	.....		0	0	0	0	0	0	0	0
Fila3 →		0	0	0	0	0	0	0	0	.....		0	0	0	0	0	0	0	0
Fila4 →		0	0	0	0	0	0	0	1	.....		1	1	1	1	1	1	1	1
Fila5 →	Barra #2	1	1	1	1	1	1	1	1	.....		0	0	0	0	0	0	0	0
Fila6 →		0	0	0	0	0	0	0	0	.....		0	0	0	0	0	0	0	0
Fila7 →		0	0	0	0	0	0	0	0	.....		0	0	0	0	0	0	0	0
Fila8 →		0	0	0	0	0	0	0	0	.....		1	1	1	1	1	1	1	1
Fila9 →	Barra #3	1	1	1	1	1	1	1	1	.....		0	0	0	0	0	0	0	0
Fila10 →		0	0	0	0	0	0	0	0	.....		0	0	0	0	0	0	0	0
Fila11 →		0	0	0	0	0	0	0	0	.....		0	0	0	0	0	0	0	0
Fila12 →		0	0	0	0	0	0	0	0	.....		1	1	1	1	1	1	1	1
Fila41 →	Barra #11	0	1	0	0	0	1	0	1	0	.....	0	1	0	0	0	1	0	0
Fila42 →		0	0	1	0	0	0	0	0	1	.....	0	0	1	0	0	1	0	0
Fila43 →		0	0	0	1	0	0	0	0	0	.....	0	0	0	1	0	0	0	1
Fila44 →		1	0	0	0	1	0	0	0	0	.....	1	0	0	0	1	0	0	0
Fila45 →	Barra #12	0	1	0	0	0	1	0	0	0	.....	0	1	0	0	0	1	0	0
Fila46 →		0	0	1	0	0	0	1	0	0	.....	0	0	1	0	0	0	1	0
Fila47 →		0	0	0	1	0	0	0	1	0	.....	0	0	0	0	1	0	0	1
Fila48 →		1	0	0	0	1	0	0	0	1	.....	1	0	0	0	1	0	0	0
Fila49 →	Barra #13	1	0	0	0	1	0	0	0	0	.....	0	1	1	0	0	0	1	0
Fila50 →		0	0	1	0	0	0	1	0	0	.....	0	0	1	0	0	0	1	0
Fila51 →		0	0	1	0	0	0	0	1	0	.....	0	0	0	1	0	0	0	1
Fila52 →		1	1	0	0	1	0	0	0	1	.....	1	1	1	1	1	1	1	0
Configuración #1		1	1	1	1	1	1	1	1	1	.....	0	0	0	0	0	0	0	0
Configuración #2		0	0	0	0	0	0	0	0	0	.....	1	1	1	1	1	1	1	1

Tabla 3.6.6 Matriz de entrenamiento

3.6.2 Justificación del tipo de red.

El objetivo es entrenar una red neuronal competitiva y más explícitamente una red LVQ que para cada una de las 64 combinaciones generadas encuentre la mejor configuración, esta configuración consistirá en una de las dos especificadas en la tabla 3.6.5.

El entrenamiento de la primera capa de la red LVQ, la cual es una capa competitiva y por lo tanto de aprendizaje no supervisado, determinará las subclases dentro de las cuales se ubicará cada una de las combinaciones, estas subclases son cuatro (determinadas arbitrariamente).

El entrenamiento de la segunda capa de la red LVQ, la cual es una capa lineal de aprendizaje supervisado, requiere conocer las clases en las cuales se ubicará cada una de las salidas de la primera capa, estas clases son las dos topologías encontradas en el set de entrenamiento.

Un esquema general de la red que solucionará la aplicación es:

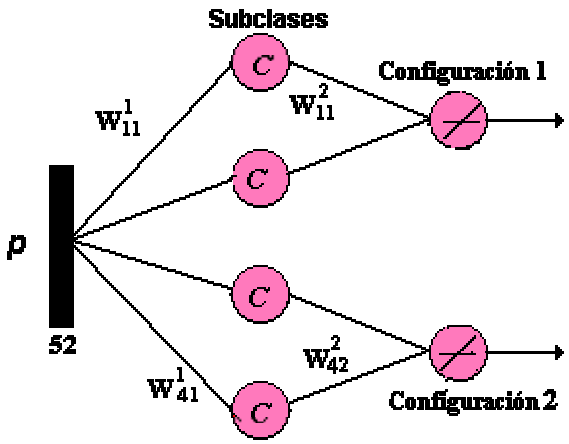

Figura 3.6.2 Esquema general de la red LVQ para reconfiguración

El vector de entrada ***p*** corresponde a cada una de las columnas de la tabla 3.6.6, las que conforman un vector de entrada de dimensiones 1x52, debido a que la reconfiguración se realizará para 13 nodos, cada uno con cuatro niveles de carga.

La función de transferencia *compet* retorna un vector de salida formado por ceros a excepción de la neurona ganadora, aquella cuyos pesos están más cercanos al vector de entrada y que ocasiona una salida de uno, esto es, las neuronas de esta capa compiten por ser la subclase ganadora. El vector de salida de la segunda capa, el cual depende de la subclase ganadora de la primera capa, determina la configuración óptima para cada patrón de entrada, de tal manera que si se obtiene un uno en la primera fila, la configuración óptima será la configuración número uno correspondiente a las ramas de coárbol 9-7, 8-6, 10 -1 y si el uno es obtenido en la segunda fila, la configuración óptima corresponde al coárbol 11-10, 9 -7, 8 – 6.

La disposición de las dos capas del algoritmo LVQ en notación compacta es la que se muestra en la figura 3.6.2.

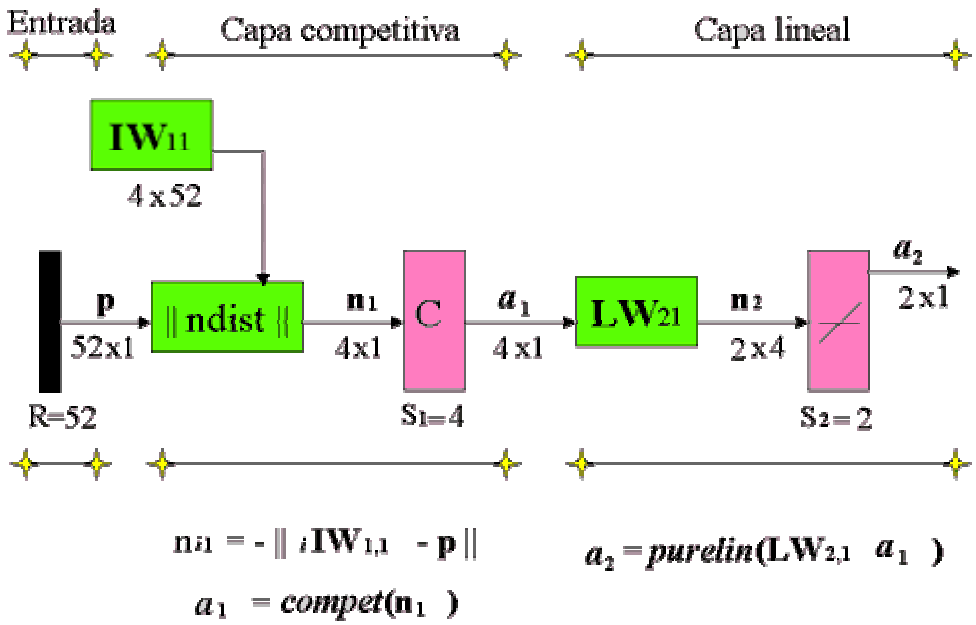

Figura 3.6.3 Red LVQ

La salida de la capa competitiva a_1 determina una posible configuración, la que red sospecha que es la configuración indicada a través del aprendizaje no supervisado, sin embargo la decisión de la mejor configuración es determinada en la capa dos, mediante un aprendizaje supervisado.

De esta forma la red LVQ ofrece una gran ventaja, puesto que en su primera capa la red tiene la libertad de determinar la "mejor" configuración, la cual es corroborada por la segunda capa, esto presenta dos mejoras, primero, un menor tiempo computacional en su entrenamiento y segundo, ofrece la flexibilidad para

que la red pueda generar sus propias representaciones (configuraciones) en la capa competitiva, la cual en el mejor de los casos puede ser una configuración completamente desconocida.

3.6.3 Entrenamiento de la red.

El código del algoritmo para esta aplicación fue desarrollado en Matlab 5.3, por medio de comandos generales sin utilizar la herramienta de redes neuronales, esto con el fin de utilizar los datos de la tabla 3.6.5 sin transformaciones; es necesario aclarar que para el entrenamiento de la red se emplearon las funciones de transferencia adecuadas para una red tipo competitiva.

Como datos de entrenamiento se tomaron 36 columnas de las 64 disponibles en la tabla 3.6.5, los valores restantes fueron escogidos como patrones de prueba para comprobar la capacidad de generalización de la red.

Los datos iniciales de los pesos fueron generados aleatoriamente con valores entre –1 y 1, por medio de la función *rands*

```
W1i=rands(52,36);
dim=size(W1);
cw=dim(2);
```

Los valores de W_2 (dos últimas filas de la tabla 3.6.5) representan las clases a las que corresponden los vectores de entrada en el algoritmo LVQ; en este caso, cada una de ellas muestra la configuración apropiada que debe adoptarse dependiendo de las condiciones impuestas por los datos de entrada.

La rata de aprendizaje, se varió hasta encontrar un valor óptimo de 0.45

```
alfa=0.45;
```

Los comandos mediante los cuales se realiza la validación y actualización de los pesos son los siguientes:

```
for i=1:cw
   for j=1:cw
      n(j,1)=-norm(W1(:,j)-p(:,i));
   end
  a1=compet(n);
    a2=W2*a1;
    e(:,i)=a2-W2(:,i);

    if e(:,i)==0
       W1(:,i)=W1(:,i)+alfa*(p(:,i)-W1(:,i));
       else
       l=find(a1);
       W1(:,l)=W1(:,l)-alfa*(p(:,i)-W1(:,l));
    end
end
```

Después de ejecutar este algoritmo para *seis iteraciones* la red alcanzó convergencia, obteniendo los siguientes valores para la matriz de pesos de la capa competitiva:

W1=net.IW{1,1}

	1	**2**	**3**	**4**	.....	**48**	**49**	**50**	**51**	**52**
N1	1,1229	0,8898	1,1379	1,052	.....	0,0232	-0,111	0,0271	-0,0448	0,0603
N2	-0,0713	0,0111	-0,353	-0,2442	.....	-0,0773	-0,1375	-0,2203	0,0168	-0,0654
N3	-0,1078	-0,1675	-0,0779	-0,1233	.....	0,9166	-0,0185	0,0619	0,9126	-0,0326
N4	-0,0037	0,0525	0,1002	0,0298	.....	-0,0532	0,9624	1,0164	-0,0679	0,9667

Tabla 3.6.7 Pesos de la capa competitiva

Los pesos para la segunda capa se visualizan en la tabla 3.6.7, los cuales como se esperaba, corresponden a la salida deseada de la red.

W2=net.LW{2,1}

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
N1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
N2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0

	19	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**
N1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
N2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Tabla 3.6.8 Pesos de la capa lineal

El algoritmo LVQ no tiene ganancias, aún así la red alcanza un eficiente desempeño clasificando correctamente todos los patrones, es decir encontrando la configuración más adecuada para cada una de las condiciones del sistema, esta

evaluación del desempeño de la red se hace con base en los flujos de carga ejecutados para las diferentes combinaciones y comparándolo con la configuración óptima determinada por la red, ya que al ser éste tipo de red un híbrido entre aprendizaje supervisado y competitivo no es posible determinar un error en su entrenamiento, pues el vector de salida para la primera capa no está especificado.

Esta red tiene grandes ventajas con respecto a otras que podrían también realizar esta aplicación, una de ellas es el tiempo de cómputo, el cual es más corto para una red LVQ, comparado con el tiempo que invierte por ejemplo una Backpropagation en alcanzar convergencia para el mismo número de patrones de entrenamiento [11].

Otra importante característica es la configuración de la red, pues el número de neuronas de cada una de las dos capas queda determinado por las mismas condiciones del problema. Además su capacidad de generalización no se ve afectada por la cantidad de patrones de entrada a la red.