1. Capitulo 1: Imágenes Digitales

1.1. CONCEPTOS BÁSICOS

El **procesamiento de imágenes** es el área del conocimiento que se ocupa de los algoritmos que permiten realizar transformaciones sobre una imagen digital. Mientras que la **visión computarizada o visión artificial** hace uso de estos algoritmos integrándolos con sistemas apropiados de hardware, introduciendo sus propios algoritmos.

Estos últimos se podrían clasificar en:

- Orientados a aplicaciones prácticas
- Orientados a actividades investigación. Poseen perspectivas de aplicación a corto plazo sujeto a actividades que buscan alcanzar **capacidades de visión avanzadas** (procesamiento de puntos, de áreas, transformaciones geométricas, etc.). Estas últimas introducen el concepto de **extracción de características**, un sistema de reconocimiento bajo condiciones de entorno o estructuradas, lo que significa que el sistema no posee información a priori a cerca de las características de la posición espacial, iluminación etc. Si debe poseer información a cerca de ciertas propiedades visuales del objeto.

El ojo humano percibe imágenes analógicas (considerada como una función continua f(x,y) en un espacio continuo entre dos dimensiones), pero para que la imagen pueda ser procesada por una computadora primero debe ser digitalizada. El proceso resulta de dos operaciones:

- Muestreo de intensidades: correspondiente a una matriz de CxR puntos. Cada vez que se muestrea una señal se da un valor a cada pixel de la imagen, un color determinado, una determinada intensidad.
- Cuantización de niveles: se trabajara con diferentes niveles de grises para facilitar el trabajo, entonces cada pixel va a tener un nivel diferente de oscuridad (escala de grises). Se procesan los diferentes niveles de gris por medio de un convertidor A/D, siendo k el número de bits por muestra de la imagen.

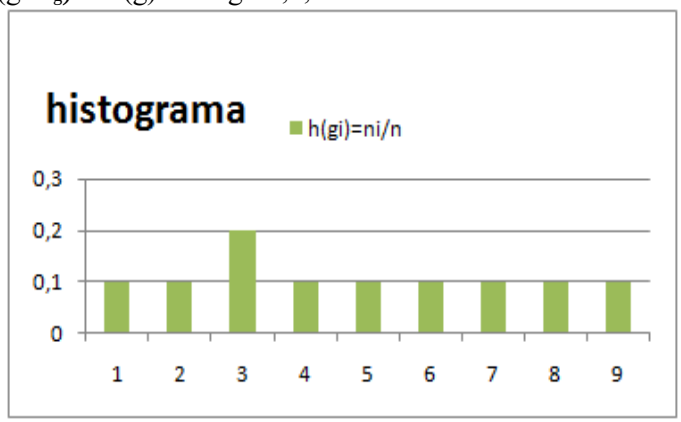
La imagen digital queda representada por RxC pixeles cada uno codificado con k bits.

1.1.1. HISTOGRAMA

Hay algunos métodos de procesamiento que dependen de las propiedades estadísticas de la imagen. Un **histograma** representa la distribución de probabilidades de las frecuencias de intensidades de una imagen. Si tenemos una imagen digital con niveles de grises en el rango [1,L] su histograma es una función discreta con valores $h(g_i) = n_i/n$ (distribución de frecuencia) i = 1,2.L Siendo g_i = el i-esimo nivel de gris, n_i = nro de píxeles en la imagen con ese nivel de gris, n = nro total de pixeles de la imagen. La función h representa la probabilidad de que un pixel elegido al azar pertenezca a un determinado nivel de gris.

A partir del histograma podemos obtener también valores estadísticos como el valor medio de niveles de gris $m_g = \sum g \times h(g)$ y la desviación estándar $\sigma^2 = \sum (g-m_g)^2 \times h(g)$. Con g =1,2,…L

1	2	4	4	8	1
3	3	3	8	5	9
4	2	9	3	5	3
6	6	9	5	7	6
7	1	2	3	7	8



gi	ni	h(gi)=ni/n
1	3	0,1
2	3	0,1
3	6	0,2
4	3	0,1
5	3	0,1
6	3	0,1
7	3	0,1
8	3	0,1
9	3	0,1
	30	1

1.1.1.1. BINARIZACIÓN. UMBRAL OPTIMO

Una imagen binaria es aquella que está formada solamente por 1 y 0. Existe un histograma especial llamado **bimodal** que representa la distribución de probabilidades de este tipo de imagen. Este se caracteriza por tener dos máximos locales claramente separados. Este histograma corresponde a una imagen con aéreas claras y oscuras y muy pocas o ningún nivel de gris. De este tipo de imágenes y de su histograma, se pueden obtener imágenes binarias detectando su umbral óptimo como el valor mínimo entre los dos picos y agrupando los valores hacia un lado u otro del umbral.

PROCESAMIENTO

ECUALIZACIÓN

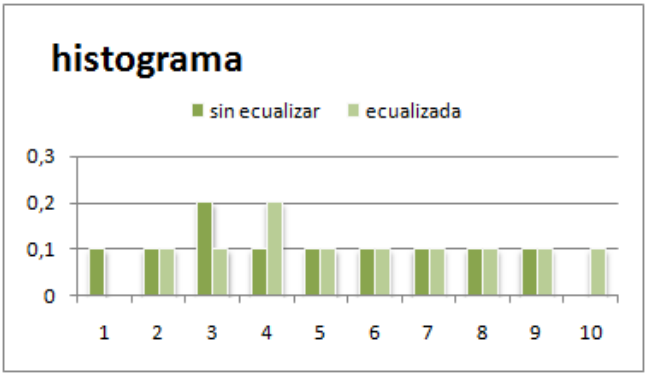
Es un proceso aplicado a imágenes de bajo contraste, es decir con áreas de nivel medio de grises. Este proceso consiste en transformar los niveles de grises: g → g´, se reacomodan las líneas del nuevo histograma con nuevos intervalos Δg de manera de obtener niveles de densidades correspondiente entre todos los valores de g y g´.

Si tenemos g=nivel de gris; h(g)=frecuencia de nivel de gris; Δg=k*h(g) donde k= numero de niveles de gris y g´=g+Δg. el nuevo histograma es h(g´) donde traslado los valores originales de h(g).

	(g)	g	´	(g´)

1	2	4	4	8	1
3	3	3	8	5	9
4	2	9	3	5	3
6	6	9	5	7	6
7	1	2	3	7	8

gi	ni	h(gi)=ni/n	Ag = k*h(gi)	g' = gi + Ag
0	0	0	0	0
1	3	0,1	1	1
2	3	0,1	1	2
3	6	0,2	2	4
4	3	0,1	1	5
5	3	0,1	1	6
6	3	0,1	1	7
7	3	0,1	1	8
8	3	0,1	1	9
9	3	0,1	1	10
	30	1	10	



FILTROS

Sirven para modificar aquellos valores que difieren demasiado de sus vecinos. Calcula un nuevo pixel para la imagen transformada en función de su vecindad. Hay que considerar que además de reducir el ruido, se reducen los detalles pequeños.

- *MEDIANA*(suavisado de imágenes y eliminación de ruidos, se adapta muy bien al ruido impulsivo)
  Recorremos la imagen de un vector con una ventana de filtro predeterminada. Al tomar una imagen de un vector (ancho impar) vemos el valor que posee le pixel central, ordenamos la secuencia de esa ventana, tomamos el valor central y lo reemplazamos en la imagen original. Es decir reemplazamos en valor central de la imagen original por el valor central de la ventana ordenada.
  80 90 200 110 120, primero ordeno (80 90 **110** 120 200) tomo el valor del medio y lo aplico
  Imagen filtrada 80 90 **110** 110 120
- *VALOR MEDIO*(para eliminar ruido gaussiano pero disminuye información del borde)
  Este caso es similar al anterior solo que no se reemplaza por el valor central de la ventana ordenada, sino que se reemplaza por el promedio de los valores de la ventana.
  80 90 200 110 120 valor medio(600/5=120) imagen filtrada 80 90 **120** 110 120

CONVOLUCIÓN

Consiste en re calcular el valor de un pixel en base a los pixeles de su vecindad. Dependiendo de la máscara que se utilice es la nueva imagen que vamos a obtener. El método consiste en definir una máscara cuya dimensión sea igual a la dimensión de la vecindad elegida para procesar la imagen. Consideramos a los vecinos de un pixel representados por la matriz $G(x,y)_{mxn}$ y la máscara a aplicar $F(x,y)_{mxn}$ calculamos los nuevos valores del pixel P como la suma de los productos entre cada uno de los elementos de $G(x,y)_{mxn}$ por cada uno de los elementos de $F(x,y)_{mxn}$ y reemplazar el valor de P por el nuevo valor calculado $P' = \sum\sum g(x,y) \times f(x,y)$.

$$g(x,y)=\begin{bmatrix}1 & 0 & -1\\ 1 & 0 & -1\\ 1 & 0 & -1\end{bmatrix} \qquad f(x,y)=\begin{bmatrix}d & c & b\\ e & P & a\\ f & g & h\end{bmatrix}$$

$$f(x) = \begin{bmatrix}d & c & b\\ e & P' & a\\ f & g & h\end{bmatrix}$$

La convolución discreta consiste en nuestro caso simplemente en realizar la suma de los productos entre los elementos de la misma posición en la imagen y reemplazar el pixel de referencia de la vecindad (central) por el nuevo valor.

1.1.2.DETECCIÓN DE BORDES

Borde es cualquier parte de la imagen que cambie bruscamente de intensidad (mayor frecuencia espacial) y frecuentemente corresponden a partes del contorno de un objeto. La forma en la que se obtienen es por Convolución. Existen mascaras pre definidas para la detección de bordes(reduce la influencia del ruido en la detección de discontinuidades:

Prewit Sobel Prewit Sobel

$$\begin{bmatrix}-1 & 0 & 1\\ -1 & 0 & 1\\ -1 & 0 & 1\end{bmatrix}\begin{bmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{bmatrix} \text{ Verticales } \begin{bmatrix}1 & 1 & 1\\ 0 & 0 & 0\\ -1 & -1 & -1\end{bmatrix}\begin{bmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{bmatrix} \text{ Horizontales}$$

$$\begin{bmatrix}-1 & -1 & 0\\ -1 & 0 & 1\\ 0 & 1 & 1\end{bmatrix}\begin{bmatrix}-2 & -1 & 0\\ -1 & 0 & 1\\ 0 & 1 & 2\end{bmatrix} \text{ -45° } \begin{bmatrix}0 & -1 & -1\\ 1 & 0 & -1\\ 1 & 1 & 0\end{bmatrix}\begin{bmatrix}0 & 1 & 2\\ -1 & 0 & 1\\ -2 & -1 & 0\end{bmatrix} \text{ +45°}$$

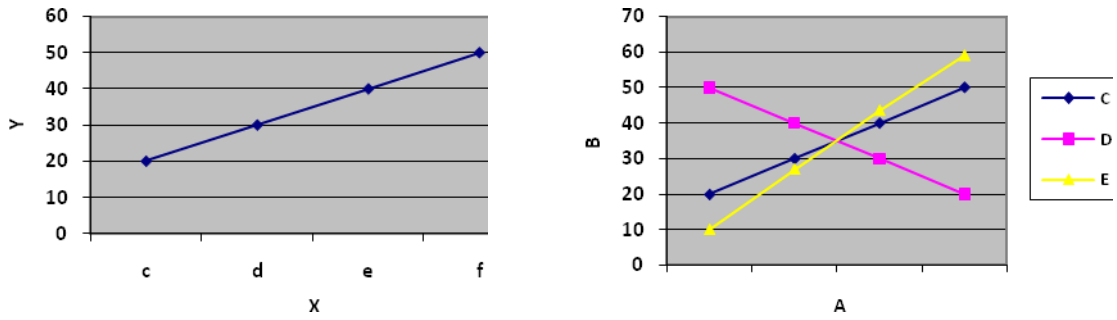
Estas máscaras se pueden combinar para obtener diferentes interpretaciones. Existen además otros tipos en los cuales se pueden detectar bordes en cualquier dirección.

Métodos orientados a líneas (círculos, eclipse y recta)

1.1.3. *TRANSFORMADA DE HOUGH*

Procedimiento utilizado para detectar los puntos relevantes pertenecientes a una recta o curva obteniendo los parámetros que definen a ese objeto en la imagen. Para la detección de rectas se obtienen los parámetros de la forma norma Hessiana y el número de pixeles dispuestos sobre la recta.

Si consideramos una recta y=ax+b → b=-ax+y



Un punto en el plano (x,y) se transforma en una recta en el plano (a,b).

Una recta en el plano (x,y) se transforma en una punto en el plano (a,b)

Así los puntos c, d y e en (x,y) son rectas C,D y E en (a,b).

La robustez del método se evidencia en el hecho de que las líneas rectas son detectadas aun si están interrumpidas o no, todos los pixeles yacen exactamente sobre la línea.

2. Capitulo 2: Reconocimiento de Patrones.

GENERALIDADES

Un patrón es una descripción de un objeto. Podemos distinguir dos categorías de reconocimientos:

- Perceptual: reconocimiento de objetos concretos.
- Conceptual: reconocimiento de objetos abstractos.

El reconocimiento de patrones se basa en el reconocimiento perceptual. A su vez el reconocimiento perceptual, como atributo humano, está asociado a la inferencia inductiva y asocia una percepción con algunos conceptos generales o pistas derivados de su experiencia pasada. Es la estimación del parecido

relativo con que los datos de entrada pueden ser asociados a experiencias pasadas que forman las pistas e información a priori para el reconocimiento.
El problema del reconocimiento puede ser concebido como el hecho de discriminar, clasificar o categorizar la información de entrada, no entre patrones individuales sino entre poblaciones, por medio de la búsqueda de características o atributos invariantes entre los miembros de una población.
Si definimos una **clase** como una categoría determinada por algunos atributos comunes y un **patrón** como una descripción de cualquier miembro de una categoría que representa a una clase de patrones, decimos que la teoría del **Reconocimiento de patrones** se ocupa de buscar la solución al problema general de reconocer miembros de una clase dada en un conjunto que contienen elementos de muchas clases diferentes.

APLICACIONES

Reconocimientos de caracteres, del habla de voz, predicción del clima, diagnósticos médicos, etc.

Tarea de clasificación	Datos de entrada	Salida
Reconocimiento de caracteres	Señales ópticas o líneas	Nombre del caracter
Reconocimiento de voz	Voz	Nombre de quien habla
Diagnostico médico	Síntomas.	Enfermedad.

PROBLEMAS – ETAPAS DEL RECONOCIMIENTO

SENSADO:

Identificación y representación de la información obtenida de un objeto a reconocer mediante algún sensor. Cada cantidad medida describe una característica del objeto. Toda la información obtenida está contenida en un vector de patrones. Cuando el contenido de los vectores patrones son números es fácil representarlos en el plano n-dimensional. En ciertas ocasiones esta información pude agruparse en Cluster, grupos de puntos que poseen características comunes.

EXTRACCIÓN DE CARACTERÍSTICAS (O PREPROCESO):

Consiste en la reducción de dimensión de un vector de patrones, extracción de las características más importantes y significativas del objeto a reconocer. Los métodos utilizados se denominan "análisis de componentes principales" o "transformación de los ejes principales" lo que se logra es reducir la dimensión de los patrones, pero con la posibilidad de perder un bajo porcentaje de información contenida en ellos.

CLASIFICACIÓN:

Es necesario diseñar un mecanismo que dado un conjunto de patrones de entrada, de los cuales no se conoce a priori la pertenencia a la clase, permita determinar a qué clase pertenece ese patrón. Un mecanismo útil puede ser la "función de decisión" o "función discriminante"

CLASIFICACIÓN DE LOS MÉTODOS

- **Heurísticos:** Basados en la intuición y experiencia.
- **Matemáticos:**
  - **Determinísticos:** Estadística no explicita.(clasificadores entrenables)
  - **Estadísticos:** Estadística explicita. (clasificador de Bayes)
- **Sintácticos:** Basados en relaciones entre los objetos.
- Si el sistema de reconocimiento puede autoajustar ciertos coeficientes internos que definen las funciones discriminantes estaremos en presencia de un **sistema adaptativo** o con capacidad de aprendizaje.

REPRESENTACIÓN GRAFICA

Es importante que las personas podamos representar y visualizar claramente las situaciones, solo en el plano y en el espacio tridimensional. Otro método es un vector como los valores de un histograma.

FUNCION DE DECISION LINEAL

Al tomar decisiones a cerca de la pertenencia a una clase dada de los patrones con que se conforma un sistema de reconocimientos es necesario establecer algunas reglas en que basar estas decisiones.
Consideramos $d(x)=w_1x_1+ w_2x_2 +w_3=0$ como la ecuación de una línea de separación donde w es el vector de parámetros y x es el vector de las variables de entrada. La función d(x) puede usarse como una función de

decisión dado un patrón de clasificación desconocida x, para que podamos determinar si pertenece a una clase w1 si cumple una condición o a una clase w2 en caso contrario.

Esta función de decisión se puede generalizar a n-dimensiones: $d(x)=w_1x_1+ w_2x_2+\ldots+w_nx_n+w_{n+1}=0$, $d(x)=\sum w'_0x + w_{n+1}$ donde el vector w'_0 es llamado vector de pesos de parámetros y el vector x el vector de patrones. Además se suele adicionar 1 como última componente de los vectores de patrones (v.p aumentado).

MATRIZ DE COEFICIENTES

Los vectores conocidos permiten calcular la matriz de coeficientes. Sabiendo que se quiere clasificar patrones que corresponden a la clase w_1 o w_2 y que cada clase contiene dos patrones de dimensión $|\, x_1^1, x_2^1 \,|$ y $|\, x_1^2, x_2^2 \,|$ donde los supra índices indican la clase. Si las clases son linealmente separables entonces se debe encontrar un vector w=(w1,w2,w3) tal que se cumpla:

$$w_1x_{11}^1 + w_2x_{12}^1 + w_3 > 0$$
$$w_1x_{21}^1 + w_2x_{22}^1 + w_3 > 0$$
$$w_1x_{11}^2 + w_2x_{12}^2 + w_3 < 0$$
$$w_1x_{21}^2 + w_2x_{22}^2 + w_3 < 0$$

Para calcular w que nos define la función discriminante, partimos de la aproximación del error cuadrático medio. Así W es solución del sistema lineal y se calcula: $w = [X^tX]^{-1}X^tY$, donde Y es el vector de salida del clasificador polinomial, X matriz de coeficientes.

EJEMPLO CONCLUSIÓN

Un sistema de clasificación lineal para clases separables puede ser calculado matemáticamente en forma precisa. Esta posibilidad es teóricamente aplicable a casos de mayor complejidad, tañes como clasificación no lineal, no separable y con dimensión de los patrones mayores que dos. Sin embargo los cálculos necesarios aumentan notablemente. De modo que si bien es teóricamente posible calcular un clasificador cualquiera, generalmente es impracticable.

Este problema da lugar a la aplicación de métodos numéricos de solución para el clasificador polinomial, a métodos recursivos como las redes neuronales, o a métodos de optimización más elaborados como los clasificadores de margen óptimo.

Fases de un clasificador:

- Entrenamiento: estimación, optimización, adaptación, aprendizaje, etc. De acuerdo al tipo de clasificador. Supervisado No supervisado. Cálculo de los coeficientes W
- Prueba: representa la capacidad de generalización de un clasificador.

2.1.1.FUNCIONES DE DECISIÓN GENERALIZADAS

Generalizando aun más la función de decisión se pueden considerar con patrones de entrada funciones f(x) funciones reales simples del patrón x, representa una infinita variedad de funciones dependiendo de la elección de funciones y el numero de términos usados para la expansión.

$d(x) = w_1f_1(x) + w_2f_2(x) + \cdots + w_kf_k(x) + w_{k+1}1 = \sum_{i=1}^{k+1} w_if_i(x)$

CLASIFICADOR POLINOMIAL

El clasificador lineal es inadecuado para resolver problemas de clasificación de mayor complejidad, por lo que para poder comprender el clasificador no lineal se introduce el concepto de función de decisión generalizada, lo que significa aplicar una función a cada una de las componentes del vector de entrada, tomándose estas como componentes de un nuevo vector. Si al menos alguna de estas funciones es no lineal se obtiene una función de clasificación no lineal.

Dado un vector $x=(x_1,x_2,x_3,\ldots,x_n)$ podemos transformarlo en $x^*=(f(x_1),f(x_2),f(x_3,)\ldots,f(x_n))$. El caso más simple se da cuando las funciones son lineales donde f(x)=x donde $d(x)=w_0x+w_{n+1}$. Ahora cuando tenemos funciones cuadráticas se plantea lo siguiente: $x^* = (x_1,x_2)^2=(x_1^2,x_2^2,x_1\, x_2,\, x_1\, ,x_2,1)$ obteniendo una d(x*) =

$w_1x_1^2 + w_2x_2^2 + w_3x_1x_2 + w_4x_1 + w_5x_2 + w_6$. Este caso puede generalizarse:

$$d(x) = \sum_{j=1}^{n} w_{jj}x_j^2 + \sum_{j=1}^{n-1}\sum_{k=j+1}^{n} w_{jk}x_jx_k + \sum_{j=1}^{n} w_{jj}x_j + w_{n+1}$$

Donde para cada una de las funciones puede definirse $f(x)=x_p^s x_q^t$ con p y q=1,…n y s y t =1,0 .

3. Capítulo 3: Redes Neuronales.

REDES NEURONALES

El costo de resolver un problema de clasificación de forma exacta es muy elevado. La mejor forma de encarar este problema es utilizar un modelo de redes neuronales artificiales, basado en el funcionamiento de neuronas biológicas, en el cual cuando se alcanza un valor umbral a la salida de una neurona se propaga la señal.

Estos criterios determinan la regla de aprendizaje. Se suelen considerar dos tipos de reglas, las que responden a lo que se conoce como aprendizaje supervisado y aprendizaje no supervisado. Una de las clasificaciones de redes neuronales obedece al tipo de aprendizaje utilizado. Así se pueden distinguir:

- Neuronas con aprendizaje supervisado
- Neuronas con aprendizaje no supervisado

La diferencia fundamental entre ambos tipos es la existencia o no de un agente externo (*supervisor)* que controle el proceso de aprendizaje.

Redes con aprendizaje supervisado: El proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor o maestro) que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor comprueba la salida de la red y en caso de que ésta no coincida con la deseada, se procederá a modificar los pesos de las conexiones, con el fin de que la salida obtenida se aproxime a la deseada.

Se suelen considerar tres formas de llevar a cabo el aprendizaje:

- Aprendizaje por corrección de error: Consiste en ajustar los pesos de las conexiones de la red en función de la diferencia entre los valores deseados y los obtenidos en la salida de la red; es decir, en función del error cometido en la salida.
- Aprendizaje por refuerzo: Es un aprendizaje más lento que el anterior, que se basa en la idea de no disponer de un ejemplo completo del comportamiento deseado; es decir, de no indicar durante el entrenamiento exactamente la salida que se desea que proporcione la red ante una determinada entrada. En el aprendizaje por refuerzo la función del supervisor se reduce a indicar mediante una señal de refuerzo si la salida obtenida en la red se ajusta a la deseada (éxito = +1 o fracaso = -1), y en función de ello se ajustan los pesos basándose en un mecanismo de probabilidades.
- Aprendizaje estocástico: Este tipo de aprendizaje consiste básicamente en realizar cambios aleatorios en los valores de los pesos de las conexiones de la red y evaluar su efecto a partir del objetivo deseado y de distribuciones de probabilidad. Según lo anterior, el aprendizaje consistiría en realizar un cambio aleatorio de los valores de los pesos y determinar la energía de la red. Si la energía es menor después del cambio; es decir, si el comportamiento de la red se acerca al deseado, se acepta el cambio; de lo contrario, se aceptaría el cambio en función de una determinada y preestablecida distribución de probabilidades.

Redes con aprendizaje no supervisado

Las redes con dicho aprendizaje no requieren de influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información por parte del entorno que le indique si la salida generada en respuesta de una entrada es o no correcta. Suele decirse que estas redes son capaces de autoorganizarse.

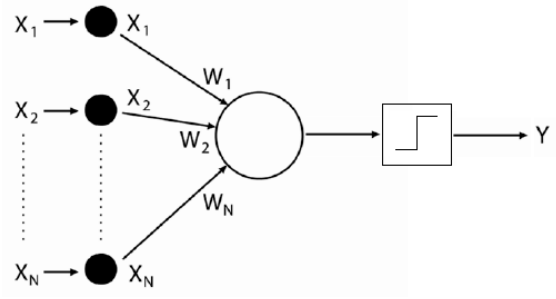
Suelen considerarse dos algoritmos de aprendizaje no supervisado (monocapa):

- Aprendizaje hebbiano (red de hopfield)
- Aprendizaje competitivo y cooperativo. (las neuronas compiten para activarse)

PERCEPTRON

Es una red mono o multicapa y con aprendizaje supervisado.

Un esquema basado en redes neuronales que realiza una función de clasificación lineal.



En lugar de expandir el vector x, se adiciona como entrada un valor unitario (siempre). El componente de w que corresponde a ese valor unitario se denomina b.

El funcionamiento del perceptrón es el siguiente: el perceptrón recibe las entradas, calcula una función de decisión lineal y aplica una función umbral que obliga al perceptrón obtener solo dos salidas (0y1 o -1y1 dependiendo de cómo se defina la función F). La diferencia entre el reconocimiento de patrones y el perceptrón es a la hora en que se realizan las funciones de clasificación lineal en el momento de calcular la matriz de coeficientes o el vector de pesos, ya que no se calcula en forma matricial, sino que mediante un proceso iterativo de aproximación.

LEY DE APRENDIZAJE

Considera como una función iterativa que permite calcular la matriz de coeficientes. El aprendizaje se logra modificando los pesos o la función umbral.

LEY DE HEBB

Es la primera ley de aprendizaje y representa un modelo simplificado del funcionamiento de una neurona biológica. Calcula w como sigue: $\Delta w_{ij} = l_r.x_i.y_j$. Donde los coeficientes se modifican según el producto de la entrada por la salida y un factor que se denomina taza de aprendizaje que permite ajustar la variación de esa magnitud. Los parámetros se modifican según: $w_{ij+1} = w_{ij} + \Delta w_{ij}$ y $b_{ij+1} = b_{ij} + \Delta b_{ij}$

LEY DEL PERCEPTRÓN

Considera que en el producto de Hebb no interviene directamente la salda sino que la diferencia entre la salida obtenida al procesar la información y la salida correcta que debería haberse generado $\Delta wij = xi.(y´j-yj)$ donde y´j es la salida deseada y yj la salida realmente obtenida. Resulta como un premio castigo ya que si x pertenece a una clase se modifica el valor de w, de lo contrario no. Los parámetros se modifican según: $wij+1 = wij + \Delta wij$ y $bij+1 = bij + \Delta bij$

Regla de aprendizaje del perceptron.

Es de tipo supervisado (lo cual requiere que sus resultados sean evaluados) y se realicen las oportunas modificaciones del sistema si fuera necesario

1. Inicialización de los pesos y del umbral
2. Presentación de un nuevo par (entrada, salida esperada)
3. Calculo de la salida actual, $f(\sum_{x=0}^{n} x_i w_i)$
4. Comparo la salida de la red con la salida deseada (calculo el error)
5. Adaptación de los pesos $w_{(t+1)} = w_{(t)} + \alpha * error * entrada$ donde α es la tasa de aprendizaje por defecto vale 1.
6. Volver al paso 2 hasta que para todas las entradas el error es igual a cero.

Ejemplo de aplicación

Operación OR

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	1

ampliación de la matriz

x0	x1	x2	y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

s= x1*w1+x2*w2*w0			
w0=	1,5	f(sumatoria) >=1	1
w1=	0,5	f(sumatoria) <0	0
w2=	1,5		

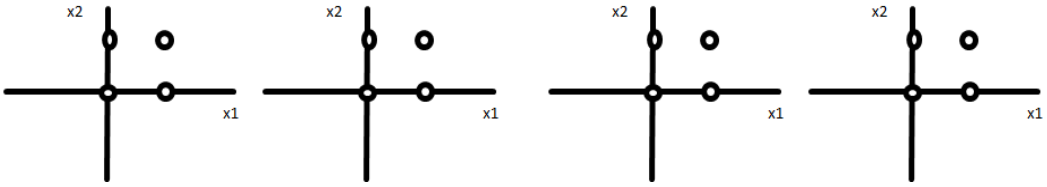
	entrada	w	sumatoria	f(sumatoria)= y	y'	error (y'-y)	A	w(t+1)
x0	1	1,5	1,5	1	0	-1	-1	1,5+(-1)=0,5
x1	0	0,5					0	0
x2	0	1,5					0	0
x0	1	**0,5**	2	1	1	0	0	
x1	0	0,5					0	
x2	1	1,5					0	
x0	1	**0,5**	1	1	1	0	0	
x1	1	0,5					0	
x2	0	1,5					0	
x0	1	**0,5**	2,5	1	1	0	0	
x1	1	0,5					0	
x2	1	1,5					0	

Grafica de la función

Para esto es necesario x1*w1+x2*w2-θ= 0, despejando x2 para la representación grafica del método.

$X2=\frac{-w1}{w2} * x1 + \frac{\theta}{w2}$

Planteando la grafica en un eje cartesiano



3.1.1. CLASIFICACIÓN NO LINEAL

El perceptrón no puede resolver clasificaciones no lineales, para ello se debe utilizar más de un perceptrón obteniendo una red. Esta red puede ir creciendo hasta alcanzar un perceptrón multicapa, el cual se caracteriza por tener tres tipo de capas: la de entrada (1), las ocultas (2) y la de salida (3).

ADALINE

A pesar de que el perceptrón multicapa pueda resolver problemas no lineales, existe aún una importante mejora que puede introducirse sobre el mismo y es sencillamente reemplazar la función de salida abrupta por una rampa. Esto posibilita generalizar el perceptrón a entradas – salidas continuas y utilizar la técnica de gradiente o descenso iterativo. Las unidades de procesamiento se denominan Adalinas.
La ley de aprendizaje que corresponde a este modelo se denomina regla delta o ley de Widrow/Hoff:
Perceptrón → $\Delta w_{ij} = x_i.(y'_j - y_j)$
Ley de Hebb → $\Delta w_{ij} = l_r.x_i.y_j$ Regla delta: $\Delta w_{ij} = l_r.x_i.(y'_j - y_j) = l_r.x_i.\delta = \alpha*(y-\sigma)*x_i$
Para comprender el sentido de la búsqueda por iteración descendente del valor de W se considera el cuadrado del error cuadrático medio: LMSE = $\frac{1}{2L} * \sum_{i=1}^{L}(d_k - s_k)^2$ donde la salida deseada d_k y la salida obtenida s_k, cuando se introduce el patrón o entrada k-esima, donde L es el número de vectores de entrada (patrones) que forman el conjunto de entrenamiento.
Y además la función para calcular el error de la red (error en una neurona): $s_k=Net_k=\sum_{j=0}^{N} w_i x_{ik}$

La función de error es una función matemática definida en el espacio de pesos multidimensional para un conjunto de patrones dados. Es una superficie que tendrá muchos mínimos (global y locales), y la regla de aprendizaje va a buscar el punto en el espacio de pesos donde se encuentra el mínimo global de esta superficie.
Esto se considera cuando es un caso unidimensional donde la grafica del error al cuadrado en función de w es una parábola, en la cual la regla delta permite modificar en cada paso el valor de w hasta alcanzar el óptimo. El gradiente debe ser descendiente.
Lo que se debe hacer es calcular la derivada de la función para que nos oriente en la búsqueda de un mínimo relativo, derivamos el erro con respecto a los distintos pesos:

$$\Delta w_i = -l_r \frac{\partial E_k}{\partial w_i}\ ;\ \frac{\partial E}{\partial w_i} = \frac{\partial E_k}{\partial \sigma_k}\frac{\partial \sigma_k}{\partial Net_k}\frac{\partial Net_k}{\partial w_i}$$

Resolviendo cada una:

$$\frac{\partial E_k}{\partial \sigma_k} = \frac{1}{2}2(y_k - \sigma_k)(-1) = -(y_k - \sigma_k)$$

$$\frac{\partial Net_k}{\partial w_i} = \frac{\partial w_i x_{ik}}{\partial w_i} = x_i$$

$\frac{\partial \sigma_k}{\partial Net_k} = 1$ Porque la salida es igual a una función lineal de la sumatoria, solo para Adaline.

Reemplazando: $\Delta w_{ij} = -\alpha(-(y_k - \sigma_k))x_i$
En el caso en el que no es un plano, debemos conformarnos con encontrar un mínimo local lo suficientemente bueno, que pueda efectuar la clasificación que exige el problema. Si la taza de aprendizaje es alta difícil de converger al resultado, si la tasa es baja se hace más largo el proceso de resolución.
Diferencias con el perceptron.

1) La función de activación es una función rampa.
2) Determinación del error. Establezco el límite del error aceptable $\frac{1}{2L} * \sum_{i=1}^{L}(d_k - s_k)^2$

Algoritmo.

1) Fijar el error aceptado.
2) Aplicar vector de entrada x_1, x_1,…, x_k
3) Salida de la red.

   $S_k=\sum_{j=0}^{N} x_{kj} w_j$ $\varepsilon_k = (d_k - s_k)$
4) Actualizamos los pesos
   $w_{(t+1)} = w_{j(t)} + \alpha*\varepsilon_k x_{kj}$

5) Repetir pasos del 1 al 3 con todos los valores de la entrada hasta que el $\frac{1}{2L} * \sum_{i=1}^{L}(d_k - s_k)^2$ sea menor al error establecido

RETROPROPAGACION

Consiste en entrenar redes multicapas. Está basado en la generalización de la regla delta con conexiones hacia adelante y cuyas células tienen funciones de activación continuas, dando lugar al algoritmo de retropropagación. Estas funciones continuas son no decrecientes y derivables. El algoritmo es una regla de aprendizaje que se puede aplicar en modelos de redes con más de dos capas.

El funcionamiento de una red backpropagation consiste en un aprendizaje de un conjunto predefinido de pares de entradas-salidas dados, empleando un ciclo propagación-adaptación de dos fases: primero se aplica un patrón de entrada como estímulo para la primera capa de neuronas de la red, se va propagando a través de todas las capas superiores hasta generar una salida, se compara el resultado obtenido en las neuronas de salida con la salida que se desea obtener y se calcula un valor del error para cada neurona de salida. A continuación, estos errores se transmiten hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de la capa intermedia que contribuyan directamente a la salida, recibiendo el porcentaje de error aproximado a la participación de la neurona intermedia en la salida original. Este proceso se repite, capa por capa, hasta que todas las neuronas de la red hayan recibido un error que describa su aportación relativa al error total. Basándose en el valor del error recibido, se ajustan los pesos de conexión de cada neurona, de manera que la siguiente vez que se presente el mismo patrón, la salida esté más cercana a la deseada; es decir, el error disminuya.

La importancia de la red backpropagation consiste en su capacidad de autoadaptar los pesos de las neuronas de las capas intermedias para aprender la relación que existe entre un conjunto de patrones dado como ejemplo y sus salidas correspondientes.

La solución viene dada por la posibilidad de calcular el error a la salida de cada capa en función del error de salida de la capa de salida. Para explicar de qué modo se logra esto, se plantea el siguiente algoritmo de aprendizaje en el que se consideran una capa oculta:

0) Establecer error aceptado para cada entrada.
1) Inicializar la taza de aprendizaje y los pesos de toda la red.
2) Ingresar una entrada aleatoria $x_p = x_{p1}, x_{p2}, ..., x_{pn}$, y la salida deseada para cada entrada
3) Calcular la salida de la red.

   Capas ocultas

   Función de activación $net_{pj}^{h} = \sum_{i=1}^{N} w_{ji}^{k} x_{pi} + \theta_j^h, y_{pj} = f_j^h(net_{pj}^h)$ h=hidden (oculto)

   Capa de salida

   $net_{px}^{o} = \sum_{i=1}^{N} w_{kj}^{o} y_{pj} - \theta_k^o, y_{pk} = f_k^o(net_{pk}^o)$ o=output (salida)

4) Calcular el error

   Capa de salida $\delta_{px}^{o} = (d_{px} - y_{px}) * f_k^o(net_{pk}^o)$ Capas ocultas $\delta_{pj}^{h} = f_{pj}^{h}(net_{pj}^h) * \sum_{k=1}^{N} \delta_{pk}^{o} w_{kj}^{o}$

5) Actualizar los pesos.

   Capa de salida $w_{kj}^{o}(t+1) = w_{kj}^{o}(t) + \propto \delta_{pk}^{o} y_{pj}$ Capas ocultas $w_{ji}^{h}(t+1) = w_{ji}^{h}(t) + \propto \delta_{pj}^{h} x_{pi}$

   Se puede incluir un valor β que se llama momento para ayudar a la convergencia de la red

   Capa de salida $\beta = (w_{xj}^{o}(t) - w_{kj}^{o}(t-1))$ Capas ocultas $\beta = (w_{ji}^{h}(t) - w_{ji}^{h}(t-1))$

6) $E_p = \frac{1}{2}\sum_{k=1}^{M}(\delta_{pk})^2$ donde M es la cantidad de neuronas de la capa de salida.

Se itera hasta que el E_p sea menor que el fijado al principio, voy al punto 2

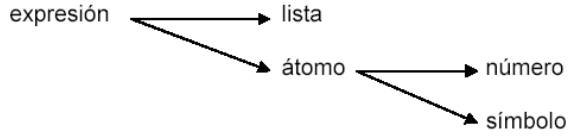
	PERCEPTON	ADALINE	BACKPROPAGATION
Topología	Monocapa o multicapa		Multicapa
Aprendizaje	Supervisada, es decir, necesita conocer los valores esperados para cada una de las entradas presentadas		
Tipo inf E/S	Heteroasociacitiva, es decir, el tipo de inf. de Entrada es distinto al tipo de inf. de Salida.		
Inf. E/S	E: Analógica - S: Binaria	E: Analógica - S: Binaria y Analógica	
Rep. de la inf E/S	discreta	Continua	
Tipo de Función de activ.	Escalón	lineal con umbral - rampa	Tangente hiperbólica o signoidal
Función de transferencia	posee un limitador fuerte (hardlim)	función lineal (purelin)	signoidal (logsig):
Desventaja	Incapacidad para solucionar problemas que no sean linealmente separables		
CONCEPTO	Es una red que realiza una función de clasificación lineal. Recibe el vector de entrada, calcula la función de decisión lineal y le aplica una función umbral que fuerza a la red a obtener solo dos valores posibles a la salida	Puede resolver problemas no lineales y solo remplazo la función de salida por una función rampa. Esta permite generalizar el perceptor para E/S continuas y permite aplicar la técnica del gradiente o descenso iterativo	Emplea un ciclo de propagación - adaptación de 2 fases. Al aplicar un patrón de entrada a la red este se propaga desde la 1a capa a través de las capas superiores hasta generar una salida que se compara con la deseada y se calcula el error por cada una. Las salidas de error se propagan hacia atrás partiendo de la capa de salida hacia todas las neuronas de la capa oculta que constituyen la salida.
Entrenamiento	1er paso: Inicializar los pesos (para adaline y back-propagation fijo el error aceptable)		
	2do paso: Presentar patrón de entrada (y salidas aceptables en back-propagation)		
	3er paso: Calcular salida	Tercero Paso: Obtener la salida lineal (función rampa) de la red	3er paso: Calcular la salida de la red (de la capa de salida y de las capas ocultas)
	4to paso: Calcular error	4to paso: Calcular error	4to paso: Calculo el error(de la capa de salida y de las capas ocultas)
	5to paso: Ajuste de pasos (debido a que hay error)	5to paso: Ajuste de pesos	5to paso: Actualizar los pesos(de la capa de salida y de las capas ocultas)
		6to Paso: Para cada entrada se repite nuevamente los pasos con los pesos ajustados y luego se calcula el error cuadrático medio, que en caso de ser menor al "error medio cuadrático aceptable" se finaliza, caso contrario comienzo de nuevo con la 1er entrada, y así sucesivamente. Calculamos el error cuadrático medio	6to paso: Calcular el error cuadrático medio considerado al final de cada ciclo, volver el paso 2 hasta que el error sea menor que el deseado

4. Capitulo 5: LISP

LISP

Programa interprete basado en el lenguaje funcional, no genera daños colaterales, es un hibrido ya que me permite realizar bucles y asignar variables. El nombre de LISP surge de abreviar "List Procesing" o Procesamiento de Listas. Fue propuesto por John Mc Carthy en 1959 en el MIT. En LISP toda la información (inclusive las sentencias) se expresan como listas o como los componentes de las mismas.

Un código de LISP está compuesto de formas o expresiones; el intérprete LISP lee una expresión, la evalúa e imprime el resultado. Este procedimiento se denomina ciclo de lectura, evaluación, impresión (read – eval – print loop).



Una **expresión o forma** es una lista o un átomo (símbolo,

entero o cadena).
Un átomo es un símbolo o un número. Si la forma es una lista LISP trata el primer elemento como el nombre de una función, y evalúa los restantes elementos recursivamente, y luego llama a la función con los valores de los elementos restantes.
Un **número** se puede escribir en cualquiera de las notaciones habituales.
Un **símbolo** se representa por un nombre que está formado por caracteres alfanuméricos que no puedan interpretarse como números.
Una **lista** es una sucesión de cero o más expresiones entre paréntesis. El blanco (no la coma) es el separador de expresiones dentro de una lista. Cada una de estas expresiones se denomina elemento de la lista.
Cuando LISP evalúa una expresión, devuelve un valor (que será otra expresión) o bien señala un error.

- Un número se evalúa a sí mismo.
- Un símbolo se evalúa al valor que tiene asignado.
- Una lista se evalúa independientemente una por una del siguiente modo, siempre en función del primer elemento y en función de este considera como emplear los elementos siguientes:

1) **El primer elemento** de la lista debe ser un símbolo "s" cuyo significado funcional "F" esté definido.

2) Se evalúan los restantes elementos de la lista, obteniendo los valores v1,…,vn. Si el número o tipo de los valores obtenidos no es coherente con los argumentos requeridos por F, se produce un error.

3) La lista se evalúa a F(v1,…,vn).

La forma en la que se evalúan las listas corresponde a una notación pre fija, donde primero se escribe el operador y luego los parámetros.
FUNCIONES: Existen dos símbolos (T y NIL) que se evalúan a sí mismos. Estos se utilizan como TRUE y FALSE o como TRUE y NULL en otros lenguajes. Un símbolo "s" puede estar ligado, es decir, contener una referencia o indicación a otra expresión "V". Llamamos ligadura al par (s,V). Un símbolo "s" (salvo T o NIL) se evalúa al valor "V" al que está ligado. Recordar que forma es una expresión que se puede evaluar.
QUOTE: Es una forma que tiene un solo argumento. (QUOTE expresión) El valor de (QUOTE expresión) es precisamente "expresión", sin evaluar. La abreviatura de QUOTE es " ' ". (QUOTE expresión) = 'expresión Ejemplo: (+ 3 4) 7, '(+ 3 4) (+ 3 4), (1 2 3) error, '(1 2 3) (1 2 3)

PREDICADOS

Son las funciones que devuelven T o NIL.
(EQ e1 e2) es T cuando e1 y e2 son el mismo símbolo
(EQL e1 e2) es T cuando e1 y e2 son EQ o números iguales de igual tipo
(EQUAL e1 e2) es T cuando e1 y e2 son EQL o listas iguales
(= e1 e2 … en) es T cuando e1, e2, …, en son números todos iguales de cualquier tipo
(NULL e) es T cuando e es NIL
(ATOM e) es T cuando e es un átomo
(SYMBOLP e) es T cuando e es un símbolo
(NUMBERP e) es T cuando e es un número
(LISTP e) es T cuando e es una lista
(ENDP e) es T cuando e es NIL (lista vacía). Si e no es una lista da error
(ZEROP e) es T cuando e es cero
(PLUSP e) es T cuando e > 0
(MINUSP e) es T cuando e < 0

IF

Tiene como argumentos: Una condición. Una o dos expresiones. Si la condición es T, se evalúa la primera condición, en caso contrario se evalúa la segunda.

5. Capitulo 6: Sistemas Expertos

¿QUE SON LOS SISTEMAS EXPERTOS?

Es un programa destinado a generar inferencias en un área específica del conocimiento en forma similar a la que se espera de un experto humano. Deben resolver problemas por aplicación de conocimiento en un dominio específico. Este conocimiento es adquirido a través de la intervención de expertos humanos y almacenado en lo que se denomina Base de Datos de Conocimiento. Son aplicaciones a mundos reducidos, ya que no pueden operar en situaciones llamadas de sentido común por ser muy extenso el dominio de conocimiento que debe tener el sistema.

Se utiliza SE cuándo:

- No existe algún algoritmo para resolver un problema
- El problema es resuelto satisfactoriamente por expertos humanos
- Existe algún experto humano que pueda colaborar en el desarrollo de un SE
- El conocimiento del dominio debe ser relativamente estático

Símbolos: Las diferentes definiciones que se van a utilizar en el SE, afirmaciones, elementos, etc.
Reglas: Se aplican sobre los símbolos, se deben obtener del conocimiento de los expertos. Son heurísticas dado que no es posible demostrar su validez general.
Hechos: son todos los predicados que se suponen verdaderos.

FUNDAMENTOS

Los sistemas expertos básicos se basan en la lógica de predicados. La diferencia entre esta y la proposicional es que la lógica de predicados emplea variables y permite expresar una premisa en una sola proposición mientras que en la otra deberíamos usar una proposición por cada una de las variables que estemos analizando. La representación mediante predicados es una forma de representar la estructura del conocimiento.

Para que la lógica de primer orden nos ayude a resolver problemas y garantice inferencias validas, el significado de los símbolos a emplear debe ser preciso no deben existir ambigüedades. En el caso de los SE se pretende aplicar la lógica a mundos que solo conocen en profundidad los expertos humanos, en que ellos expresan su conocimiento en lenguaje natural. Por lo que para que un SE basado en el formalismo lógico requiere definir los símbolos con los cuales se operará y su significado preciso, además de traducir el conocimiento del experto del lenguaje a natural a la expresiones lógicas correspondientes (reglas).

Los SE se basan en la posibilidad de aplicar las nociones de la lógica formal para incrementar una base de conocimientos y solo pueden aplicarse a mundos de muy baja complejidad. El modo más inmediato consiste en aplicar reglas del tipo si-entonces e instanciando los antecedentes de la misma obtener una instancia válida para el consecuente.

En lugar de instanciar el consecuente para ver si es soportado por sus antecedentes se podría trabajar a la inversa:

Encadenamiento hacia adelante el SE toma una serie de afirmaciones de entrada y ensaya todas las reglas disponibles, una y otra vez, incorporando nuevas afirmaciones, hasta que ninguna de las reglas pueda producir nuevas afirmaciones.

Encadenamiento hacia atrás se comienza con una hipótesis y se trata de verificarla haciendo uso de las afirmaciones disponibles.

LÓGICA DE PREDICADOS

LÓGICA

La **lógica** permite la posibilidad de utilizar un conjunto de reglas para gestionar inferencias creíbles. Una **inferencia** es la extracción de conclusiones a partir de premisas más o menos explicitas, puede ser resultante del sentido común o de la aplicación de reglas muy detallistas o cálculos estadísticos. La **epistemología** estudia el origen y características del conocimiento, en especial la forma en como aparece y sus limitaciones.
Los **formalismos** a usar para representar los hechos que ocurren en el mundo, deben ser lo suficientemente adecuados como para representar la información disponible.
El **silogismo** es una formula lógica con premisas seguidas de una conclusión.
La **lógica de primer orden** es uno de los formalismos más utilizados para representar el conocimiento. Cuenta con un lenguaje formal mediante el cual es posible representar formulas llamadas **axiomas** que permiten describir fragmento del conocimiento y consta de un conjunto de **reglas de inferencia** que aplicadas a los axiomas permiten derivar nuevo conocimiento.

El alfabeto

Posee dos símbolos:

- Lógicos: constantes, operadores proposicionales y cuantificación, lógicos y auxiliares.
- No lógicos: variables individuales, funciones n-arias y relaciones n-arias.

A partir de los símbolos se construyen las expresiones:

- Términos: una constante, palabra, letra, variable y una expresión de la forma.
- Formulas: son expresiones de la forma R(t1….tn) donde R es un símbolo de relación y t1…tn son los términos.

Los términos permiten nombrar objetos del universo, mientras que las formulas permiten afirmar o negar propiedades de estos o bien establecen relaciones entre los objetos del universo.

FORMULAS BIEN FORMADAS

La lógica proposicional permite el razonamiento mediante un mecanismo que primero evalúa sentencias simples y luego sentencias complejas mediante el uso de conectivos proposicionales. Una preposición es una sentencia simple con un valor asociado. La lógica proposicional permite la asignación de un valor v o f para la sentencia compleja, no tiene la facilidad para analizar las palabras individuales que componen la sentencia. Una FBF es una proposición simple o compuesta de significado completo cuyo valor de veracidad puede ser determinado, basado en los valores de veracidad de las proposiciones simples y en la naturaleza de los conectores lógicos involucrados.

El **Silogismo categórico** dice que si es verdadera una instancia cualquiera de las siguientes premisas: Si M es A y B es M entonces B es A. la **resolución** se basa en que una afirmación no puede ser simultáneamente verdadera y falsa.

ELEMENTOS BÁSICOS

- Átomos: proposiciones simples.
- Conectivos lógicos: expresiones que sirven para formar preposiciones compuestas a partir de preposiciones simples.
- Sentencias: átomos o clausulas formadas por la aplicación de conectivos a sentencias.
- Cuantificadores: universal $\forall$ y existencial $\exists$
- Símbolos de puntuación y delimitación.

Silogismos

Hipotético: A es B y B es C → A es C
Disyuntivo: $\neg A \vee B$ es A → B

Función de Skolem

Cuando un cuantificador existencial esta en ámbito de uno universal, la variable cuantificada debe ser reemplazada con una función de skolem: $\forall$ x $\exists$ y (x<y) entonces $\forall$ y (y-1<y).

Leyes de Morgan

$-(A \wedge B) = -A \vee -B$ y $-(A \vee B) = -A \wedge -B$

FBF se pueden expresar:

- Forma normal conjuntiva $(A \vee B) \wedge (A \vee B)$
- Forma normal disyuntiva $(A \wedge B) \vee (A \wedge B)$
- Clausula de Horn: Es una conjunción de disyunciones con no más de un literal positivo (no negado). Una expresión es valida satisfacible si alguna combinación de valores v o f de sus átomos hacen verdadera la expresión.

REDUCCIONA LA FORMA CLAUSAL

Reducir los predicados a un conjunto de clausulas (afirmaciones) para que puedan ser procesadas por un mecanismo lógico

1. Eliminar implicaciones
2. Reducir la aplicación de negaciones
3. Eliminar cuantificadores
4. Aplicar propiedad disyuntiva y asociativa
5. Que cada clausula tenga nombre de variable único.

RESOLUCIÓN

La demostración se lleva a cambo mediante la reducción al absurdo. El problema puede surgir con un gran número de combinaciones posibles. Es en esencia un procedimiento iterativo cuya finalidad es evaluar la verdad o falsedad de una premisa. En cada paso dos clausulas padres son resueltas para obtener una tercera denominada resolvente. Para este procedimiento se definen las siguientes propiedades:

- Para clausulas P y –P el resolvente es la clausula vacía.
- Si un conjunto de sentencias contiene ambos P y –P el conjunto es insatisfacible.
- Si ambos antecedentes son verdaderos luego el resolvente es verdadero.
- La resolución verifica la refutación, "si el conjunto de clausulas implica a P luego es contradictorio con el mismo conjunto que además contenga a –P"

UNIFICACIÓN Y LIGADURA

Permite razonar dentro de un esquema formal sin relación con la codificación de los programas. En el caso de un código, este se encarga de reducir las formas clausales a listas asociadas más simples de manipular denominadas ligaduras, esta fluye a través de los filtros del programa y al final del proceso pueden ser analizadas para extraer nuevas afirmaciones o demostrar hipótesis.

CORRESPONDENCIA DE PATRONES

Relacionado con la codificación de SE es la correspondencia de patrones simbólicos. La correspondencia con patrones se basa en identificar si dos vectores son similares es decir se corresponden. La forma simple de entenderlo es calcular la distancia entre los mismos y si esta es lo suficientemente pequeña de acuerdo a valores preestablecidos para un problema dado, entonces puede decirse que ambos vectores se corresponden.

CORRESPONDENCIA SIMBÓLICA

Busca similitudes entre un conjunto de nombres de variables y no entre los valores numéricos de estas. Se consideran dos procedimientos:

Correspondencia: encadenamiento hacia adelante, comparando una expresión común (dato) con un patrón. A diferencia de los datos los patrones pueden contener variables patrón, para facilitar el reconocimiento de estas variables se asocian a un símbolo.

Unificación: encadenamiento hacia atrás. Hace corresponder dos patrones en lugar de un patrón y un dato.

CORRESPONDENCIA SUB SIMBOLICA

Correspondencia numérica.

ENCADENAMIENTO.

PROGRESIVO

- **Correspondencia**: Busca afirmaciones en la base de datos que correspondan con los antecedentes de una regla.
- **Resolución:** En caso de que se cumplan los antecedentes instanciar el consecuente produciendo una nueva afirmación.

REGRESIVO

- **Correspondencia:** Busca afirmaciones en la base de datos que correspondan a una hipótesis.
- **Unificación:** buscar una regla cuyo consecuente concuerde con la hipótesis. La hipótesis y el consecuente pueden tener variables de patrón.
- **Resolución:** para cada regla cuyo consecuente concuerde con la hipótesis, se intenta verificar de manera recursiva cada antecedente de la regla, considerando a cada una como una hipótesis.

6.

6.1. HEURISTICA

El tipo de función de membrecía de los adjetivos empleados en las reglas debe ser propuesta por un diseñador del sistema, los valores particulares también, el método de defuzzificacion ha de ser elegido por el diseñador, etc. Todo esto no solo muestra una aplicación particular de la heurística, sino que evidencia su importancia lo que hace que muchas veces se cuestionen los fundamentos para el diseño y construcción.

7. Capitulo 8: Búsqueda

CONCEPTO

Procedimiento cuya finalidad es encontrar datos dentro de un conjunto de estos, pero donde el objetivo del proceso es más bien encontrar cierto camino recorrido hasta encontrar en dato antes de corroborar la existencia del dato. Esto se debe a que estos procesos se usan en planificación, optimización, resolución de problemas, juegos, etc. Para realizar las búsquedas se utilizan arboles y grafos para conocer la secuencia de estados que deben alcanzarse sucesivamente para lograr un estado objetivo.

7.1. METODOS NO INFORMADOS

PRIMERO EN ANCHURA BPA

Se generan para cada nodo todas las posibles situaciones resultantes de la aplicación de todas las reglas adecuadas. Se continúa:

1. Se crea una pila (abierta) y se le asigna el estado inicial.
2. Mientras la pila no esté vacía.
    a. Extraer el primer nodo de la pila llamarlo m.
    b. Expandir m. Se generan las entradas sucesoras del estado actual, para cada operador aplicable hacer:
        i. Aplicar el operador al nodo obteniendo un nuevo estado.
        ii. Si el nuevo estado es el objetivo termina el proceso.
        iii. Incluir el nuevo estado el final de la pila volver a 2ª

Ventajas: No entra en callejones sin salida, si existe una solución garantiza alcanzarla, si existen varias soluciones garantiza alcanzar la optima.

Desventajas: utiliza mucha memoria, es lento.

PRIMERO EN PROFUNDIDAD BPP

Continúa por una sola rama del árbol hasta encontrar la solución o hasta que se tome la decisión de terminar la búsqueda por esa dirección, se puede tomar por que: se llego a un callejón sin salida, se produce un estado ya alcanzado o la ruta se alarga más de lo especificado.

1. Se crea una pila (abierta) y se le asigna el estado inicial.
2. Mientras la pila no esté vacía.
    a. Extraer el primer nodo de la pila llamarlo m.

b. Si la profundidad de m es igual al límite de profundidad regresar a A, en caso contrario continuar.
c. Expandir m. Se generan las entradas sucesoras del estado actual, para cada operador aplicable hacer:
    i. Aplicar el operador al nodo obteniendo un nuevo estado.
    ii. Si el nuevo estado es el objetivo termina el proceso.
    iii. Incluir el nuevo estado principio de la pila volver a 2ª

Ventajas: Si existe una solución garantiza alcanzarla, no recorre todo el árbol para alcanzar la solución, pero es por azar.

RAMIFICACIÓN Y ACOTACIÓN

Comienza generando rutas completas, manteniéndose la ruta más corta encontrada hasta ese momento. Deja de explorar tan pronto la distancia total sea mayor a la marcada como la ruta más corta.

7.2. HEURÍSTICA

Se refiere a un conocimiento que intuitivamente poseemos, que podría parecer experimental debido a la inspiración de quien lo propone. La diferencia entre lo heurístico y lo científico es que este último es un conocimiento debido a la observación, una regla empírica se cumple aunque no sepamos sus causas. En la regla heurística se espera que se cumpla, en casos particulares, y no en todos los casos en que podría aplicarse; no puede ser demostrada teóricamente, ni comprobada experimentalmente para todos los casos no se generaliza; puede resolver perfectamente un problema en particular, pero puede arrojar resultados erróneos con un problema similar.

7.3. METODOS INFORMADOS O DE BUSQUEDA HEURISTICA

La heurística es una técnica que aumenta la eficiencia de un proceso proporcionando una guía para este y posiblemente sacrificando demandas de complejidad. La búsqueda heurística resuelve problemas complicados con eficacia, garantizando una estructura de control que encuentra una buena solución.

Porque usar heurísticas:

- No enredarnos en una explosión combinatoria.
- Buscar una solución adecuada.
- Los peores casos, raramente ocurren.
- Profundizar nuestra comprensión del dominio del problema.

PRIMERO EL MEJOR

Combina las ventajas de BPA y BPP sigue solo un camino a la vez y cambia cuando alguna ruta parezca más prometedora. Estrategia:

1. Se selecciona el nodo más prometedor según la función heurística apropiada
2. Se expande el nodo elegido de acuerdo a las reglas de expansión adecuadas
3. Si alguno de los nodos es el optimo se termina, sino:
    a. Se añaden nuevos nodos a la lista
    b. Se vuelve al paso 1

*A**

Se modifica el anterior empleando una función heurística para estimar el costo desde un nodo actual al nodo objetivo. Realiza una estimación de cada uno de los nodos que se van agregando, esto permite examinar los caminos más prometedores.

Utiliza la siguiente función: f´=g+h´. Donde f´=estimación del costo desde el nodo inicial al nodo objetivo, g =nivel del árbol en que se encuentra el nodo, h´=estimación desde el nodo actual al nodo objetivo.

Se emplean dos listas. Abierta: los nodos que se les ha aplicado la función heurística. Cerrada: nodos que ya han sido expandidos.

Se toma el nodo más prometedor y que no haya expandido. Se generan sus sucesores y se le aplica la función heurística y se los añaden a la lista de nodos abiertos

GENERACIÓN Y PRUEBA

La generación de la posible solución puede realizarse de formas diferentes. Consiste en generar la posible solución, verificar si es una solución, si no se hallo la solución volver a empezar de lo contrario devolverla y terminar.

ESCALADA

Simple: se usa para ayudar al generador a decidirse por cual dirección moverse. La función de prueba se amplía con una función heurística que ayuda a evaluarlos estados y proporciona una estimación de lo cerca que nos encontramos del objetivo. Necesita más tiempo para alcanzar la solución.

Máxima Pendiente: es una variación del anterior, que considera todos los movimientos posibles a partir del estado actual y elige el mejor de ellos como nuevo estado. Tarda más en seleccionar un movimiento. Puede caer en un máximo local (mejor q todos pero no el mejor), meseta (estados vecinos produce el mismo valor, no se sabe a donde moverse) o cresta (no tengo operadores para ubicar la cima)

ENFRIAMIENTO SIMULADO

El objetivo es disminuir la probabilidad de caer en una meseta, máximo local o cresta. Por lo que se realiza una exploración con saltos amplios al principio que se van reduciendo paulatinamente. Se plantea minimizar la función objetivo, para alcanzar un estado final de mínima energía.

MEDIANTE AGENDA

Consideramos a cada nodo como una tarea que un sistema debe realizar. Al expandir una tarea van a existir diversos caminos. Cada camino que recomienda una tarea proporciona una razón por la cual debe ser realizada. Cuantas más razones haya, aumenta la probabilidad de que la tarea lleve a algo adecuado. Para almacenar las tareas se propone el concepto de agenda, que es en definitiva el conjunto de tareas que el sistema debe realizar.

REDUCCIÓN DE PROBLEMAS

Consiste en dividir el problema en sub problemas hasta llegar al nivel de módulos. En este caso, un nodo no es mejor por sí mismo, sino por pertenecer a una ruta más prometedora.

VERIFICACIÓN DE RESTRICCIONES

Define los estados como un conjunto de restricciones que deben satisfacerse para resolver completamente el problema, de este modo lo que se busca es un estado que satisfaga un conjunto de restricciones impuestas por el problema.

ANÁLISIS DE MEDIOS Y FINES

Permite razonar tanto hacia adelante como para atrás, permite resolver las partes más importantes del problema y después volver atrás y resolver los pequeños problemas. Se centra en la detección de diferencias entre el estado del nodo actual y el nodo objetivo.

ALGORITMO DE RUTEO

No solo busca que existan las rutas a conectar, ni que sean las rutas más cortas, sino que la ruta encontrada para cada par de puntos sea la ruta que menos interfiera con el menor número de rutas.

8. Capitulo 9: Planificación

8.1. INTRODUCCION

Se refiere al proceso de computar varios pasos de un procedimiento de resolución de un problema antes de ejecutar alguno de ellos.

Cuando nos enfrentamos con el problema de planificar y ejecutar una secuencia de pasos en un mundo que no es completamente predecible nos enfrentamos al problema de planificación en IA.

8.2. CLASIFICACION

Planificación en IA:

- Numérica
- Lógica: Por pila de objetivos, No lineal, Jerárquica
- Reactiva

Planificación en robótica

- Reactiva
- De trayectoria de manipuladores

Planificación industrial: métodos basados en simulación discreta

8.3. EL PROBLEMA

El problema de clasificación se caracteriza porque se puede aplicar a diversos dominios además de que no se pueden explotar todas las opciones.

Entradas: Descripción del estado del mundo, descripción del objetivo, conjunto de acciones.

Salida: una secuencia de acciones que pueden ser aplicadas al estado, hasta alcanzar la descripción del estado final.

La lógica nos ofrece una manera de reducir los pasos de búsqueda, representando los estados por predicados aplicando reglas lógicas para pasar de uno a otro, dicha aplicación debería modificar el estado anterior definiendo nuevos estados. La regla en si nos dice que está permitido pasar de un estado a otro pero no efectúa el cambio necesario para generar este paso. Para que se modifique el estado actual es necesario aplicar procedimientos (operadores).

8.4. MUNDO DE LOS BLOQUES

Para poder hacer un estudio sistemático de los métodos y poder compararlos usamos el mundo de los bloques. Se basa en una superficie palana en la que se colocan bloques (de forma cubica únicamente). Los bloques se pueden ubicar unos sobre otros y se pueden llevar a cabo acciones que pueden manipular los bloques (operadores asociados a las reglas).

8.5. CODIFICACION DE LOS OPERADORES

Para poder pasar de un estado a otro modificando predicados usamos operadores, cada operador puede describirse mediante una lista de nuevos predicados que el operador provoca que sean ciertos y una lista de los viejos predicados que el operador provoca que sean falsos.

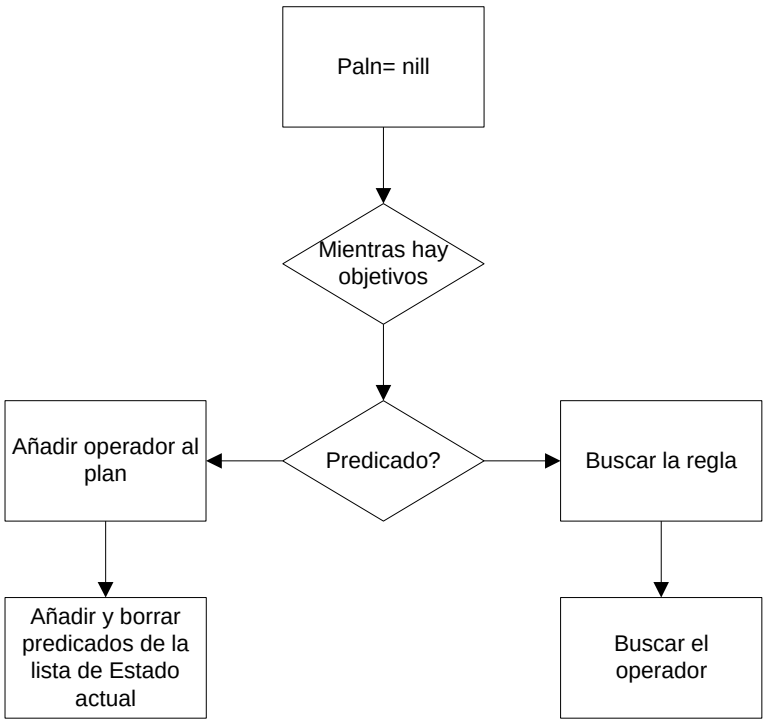
8.6. PLANIFICACION MEDIANTE UNA PILA DE OBJETIVOS

Elementos: Bloques, superficie y brazo operador.

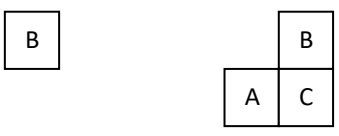
Estados de los elementos: estado del bloque y del brazo.

Operadores: acciones a realizar. Se componen de tres listas: precondición (lo que se debe cumplir para aplicar el operador), Borrado (lo que se debe eliminar de la pila de objetivos), Adición (lo que se agrega al plan de tareas)

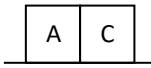
Pilas: en todo momento se cuanta con una pila de objetivos y operadores, una de estado actual y la del plan a seguir.



Procedimiento: se tratan los objetivos como sub problemas a resolver. Se pregunta si el primer objetivo se cumple. Si no se cumple se debe aplicar el operador que lo transforme, por lo que se debe reemplazar al predicado objetivo por el operador y luego agregar a la pila las precondiciones que deben cumplirse para ese operador. Para resolver el objetivo que ha quedado arriba de la pila empleamos el operador reemplazando y agregando las precondiciones. Al ir extrayendo los operadores de esa pila, se incorporan a la lista plan y se actualiza el estado actual.



EI: sobremesa(A), sobremesa(C), sobre(B,A), despejado(C), despejado(B)
EF: sobremesa(A), sobremesa(C), sobre(B,C), despejado(A), despejado(B)



Estado inicial Estado final desapilar(B,A) precondic: sobre(B,C), despejado(B), brazolibre
postcondic: agarrado(B), despejado(A)

~~despejado(B)~~
~~sobremesa(C)~~
~~sobremesa(A)~~
sobre(B,C)
despejado(A)
sobremesa(A), sobremesa(C), sobre(B,C), despejado(A), despejado(B)

despejado(B)
sobre(B,C)
sobre(B,C), despejado(B)
desapilar(B,A)
sobre(B,C)
despejado(A)
sobremesa(A), sobremesa(C), sobre(B,C), despejado(A), despejado(B)

plan
~~**despejado(B)**~~ **desapilar(B,A)**
~~**sobre(B,C)**~~
~~**sobre(B,C), despejado(B)**~~ **Estado actual**
~~**desapilar(B,A)**~~ **agarrado(B)**
sobre(B,C)
despejado(A)
sobremesa(A), sobremesa(C), sobre(B,C), despejado(A), despejado(B)

agarrado(B)
despejado(C)
despejado(C), agarrado(B)
apilar(B,C)
sobre(B,C)
despejado(A)
sobremesa(A), sobremesa(C), sobre(B,C), despejado(A), despejado(B)

plan
~~**agarrado(B)**~~ **desapilar(B,A)**
~~**despejado(C)**~~ **apilar(B,C)**
~~**despejado(C), agarrado(B)**~~
~~**apilar(B,C)**~~
sobre(B,C)
despejado(A)
sobremesa(A), sobremesa(C), sobre(B,C), despejado(A), despejado(B)

~~sobre(B,C)~~
~~despejado(A)~~
~~sobremesa(A), sobremesa(C), sobre(B,C), despejado(A), despejado(B)~~

9. Capitulo 9: Complejidad

9.1. INTRODUCCIÓN

Se asocia la complejidad con los recursos, una rama de la ciencia de la computación, la teoría de la complejidad computacional se ocupa de estudiar estos recursos que normalmente son tiempo y espacio.
La teoría de la complejidad computacional siempre aparece vinculada a la teoría de la computabilidad, aunque sus objetivos son esencialmente distintos. El de la teoría de la computabilidad está en la existencia, o no, de procedimientos que permitan resolver cierto problema. La complejidad computacional siempre fue motivo de interés. En los comienzos de la era de la computación esto fue una consecuencia natural de los magros recursos disponibles. Sin embargo paralelamente, los objetivos de los sistemas informáticos son cada vez más ambiciosos y el interés por la complejidad computacional mantiene así plena vigencia.
La inteligencia artificial no es ajena a esta realidad. Renovados objetivos y el permanente descubrimiento, desarrollo y aplicación de nuevas técnicas y herramientas han conducido progresivamente a un incremento de la complejidad de los sistemas involucrados. Entre otros, los juegos de animación, algoritmos genéricos, autómatas celulares, minería de datos. En estos casos la complejidad computacional es unos de los principales factores a ser considerados.

9.2. PROCEDIMIENTO Y ALGORITMO

Procedimiento: todo conjunto finito y ordenado de instrucciones, de forma tal que puedan ser ejecutadas por una persona o una máquina sin necesidad de conocimiento adicional a lo ya expresado en las propias sentencias.
Algoritmo: procedimientos que permiten alcanzar siempre una solución en un número finito de pasos, son definidos como algoritmos o procedimientos efectivos.
Aquí debe anticiparse que habrá problemas para los cuales no es posible encontrar un procedimiento y otros problemas para los que existen procedimientos pero no algoritmos. Más aun, que la solución de un problema quede representada por un procedimiento o por un algoritmo puede ser intrínseco del propio problema y en muchos casos tan solo depender de los datos.
Debe ahora reconocerse que un procedimiento (efectivo o no) puede presentarse de muy diversas maneras: en su forma más general una secuencia mecánica de instrucciones es una máquina de turing.
Hopcroft y Ullman dicen que “un procedimiento es una secuencia finita de instrucciones que pueden ser cumplidas en forma mecánica, como es el caso de un programa de computadora. Un procedimiento que siempre termina lleva el nombre propio de algoritmo”.

9.3. CONCEPTO DE COMPLEJIDAD

Se pretenderá buscar algún indicador de la complejidad intrínseca de los problemas, procurando prescindir de la habilidad de la persona que deba resolverlos, de los algoritmos que deba resolverlos, de los algoritmos que puedan ser utilizados y de las características de los medios de cálculo disponibles. Y finalmente, cuando no sea posible prescindir de estos factores, poder establecer las bases que permitan definir métricas y hacer comparaciones.

a. Identificar los parámetros primitivos que serán medidos.
b. Definir las propias métricas.
c. Establecer las condiciones en que se harán las mediciones.

9.4. MEDIDAS Y METRICAS DE LA COMPLEJIDAD

El enfoque que se propone es el de determinar indirectamente la complejidad, no se evalúa la complejidad del propio problema, sino más bien los recursos demandados correspondientes al procedimiento empleado.
De acuerdo a esta idea, se seleccionan al tiempo y espacio como los indicadores más representativos.

9.4.1.COMPLEJIDAD TEMPORAL

Es uno de los indicadores de complejidad más utilizados. Se refiere a la cantidad de intervalos o unidades elementales que demanda completar la ejecución de un proceso.

Es así que se procurara conocer la razón de crecimiento entre el indicador tiempo y la dimensión de los datos, que representa el parámetro medible.

El límite en el crecimiento de esta medida se denomina complejidad temporal asintótica y es finalmente lo que determina el tamaño del problema que puede ser resuelto con cierto algoritmo.

La selección del tiempo como indicador representativo de la complejidad no requiere mayores justificaciones. La búsqueda de la reducción de los tiempos de proceso es tan antigua como la misma ciencia de la computación.

A medida que los computadores son más rápidos se pretende resolver problemas de mayor tamaño en un contexto de nuevas exigencias y la necesidad de algoritmos eficientes mantienen plena vigencia o cobra aún más importancia que antes.

Algoritmo	Orden de complejidad temporal T(n)	Máximo Tamaño de lote de datos "n"			Incremento en el tamaño "n" del problema que es procesable en 1seg
		1seg	60seg	3.600seg	
A_1	n	1.000	60.000	3.600.000	x 10
A_2	n log n	140	4.893	200.000	x 10 (aprox.)
A_3	n^2	31	244	1.897	x 3,16
A_4	n^3	10	39	153	x 2,15
A_5	2^n	9	15	21	+ 3,30

9.4.2.COMPLEJIDAD ESPACIAL

El otro indicador que se emplea con frecuencia es la cantidad de espacio de almacenamiento requerido para resolverlo a través de cierto algoritmo. Esto se apoya en la idea de que al aumentar la complejidad de un proceso aumenta también el espacio que demanda y este parámetro es reconocido como la complejidad espacial del problema. El límite en el crecimiento de esta medida se denomina complejidad espacial asintótica y determinará finalmente el tamaño del problema que puede resolver cierto algoritmo.

9.5. MEDICIÓN DE LA COMPLEJIDAD Y MÁQUINAN DE TURING

La complejidad temporal y la espacial por la resolución de un problema dependerán del soporte físico donde el proceso es realizado. La complejidad temporal de un problema medida sobre un computador dependerá de su arquitectura, rapidez del microprocesador, disponibilidad de coprocesador aritmético para operaciones con punto flotante, rapidez de los canales de I/O, etc. Similarmente la espacial también dependerá de la arquitectura del equipo, forma de codificación de los datos entre otros factores.

Para eliminar esta dependencia entre las mediciones de los indicadores de complejidad y el medio de cálculo utilizado las comparaciones deben ser realizados en un contexto fijo, por medio de la máquina de Turing, que no solo proporciona un ambiente bien definido donde trabajar sino que además las conclusiones que se obtengan son fácilmente transferibles a otros sistemas computacionales.

Teorema 1 de Shannon: cualquier máquina de Turing con m símbolos y n estados puede ser simulada por otra máquina de Turing con exactamente dos estados y 4.m.n+m símbolos. Existe una máquina universal de Turing de solo dos estados.

Teorema 2 de Shannon: cualquier máquina de Turing con m símbolos y n estados puede ser simulada por con exactamente dos símbolos y menos de 8.m.n estados.

Definición de complejidad en la máquina de Turing

Complejidad temporal: número de pasos requeridos para completar el cálculo.

Complejidad espacial: al número de celdas requeridas de la cinta de E/S.

No son totalmente diferentes uno del otro ya que en "n" pasos de una MT tiene acceso a un máximo de "n+1" celdas.

Una máquina de Turing se dice acotada superiormente en tiempo por T(n) si para toda entrada "x" de tamaño "n" se verifica $T(x) \leq T(n)$

Una máquina de Turing se dice acotada superiormente en el espacio por E(n) si para toda entrada "x" de tamaño "n" se verifica $E(x) \leq E(n)$

Un lenguaje "L" se dice de complejidad T(n) y E(n) si existe una MT acotada superiormente en tiempo y espacio que acepta "L". Es decir que para toda cadena $x \in L$ que verifica que $T(x) \leq T(n)$ y $E(x) \leq E(n)$, siendo $n = |x|$

9.6. MEDICIÓN DE LA COMPLEJIDAD Y COMPUTADORES REALES

Para determinar las complejidades temporales y espaciales sobre una aplicación en un computador real deben considerarse los tiempos requeridos por las instrucciones ejecutadas y los registros utilizados.

Una opción es el determinado "criterio de costo uniforme" que se asigna a cada instrucción el consumo de una unidad de tiempo y a cada registro utilizado una unidad de espacio. Por este criterio, T(x) es el número de instrucciones ejecutadas por el computador cuando lee cierta cadena de datos "x" y E(x) la cantidad de registros de memoria utilizados, conduciendo a valores aproximados y es solo recomendable para obtener una primer aproximación.

Un criterio más realista y especifico es el de "costo logarítmico". Con este criterio se considera la incidencia del largo de los números representados en los operandos en el tiempo de ejecución de cada operación. Así, para el caso de un número "n" la cantidad de dígitos de su representación está dada por $L(n) = \log_2 |n| + 1$ y el costo de ejecutar una sentencia de programación será proporcional a esta cifra

9.7. CLASES DE PROBLEMAS

Todos los problemas matemáticos imaginables pueden ser divididos en dos grupos: los que admiten un algoritmo para su solución y los demostrablemente irresolubles.

A la solución algorítmica a su vez dividir en dos grupos, los de complejidad temporal polinómica (tienen su tiempo de ejecución acotada por una función de este tipo y son considerados eficientes) y los de complejidad exponencial (carecen de interés práctico, los problemas denominados intratables solo pueden alcanzarse una solución cuando la dimensión de datos "n" es pequeña)

En la segunda clase se encuentran los problemas que pueden ser resueltos pero no admiten un método general, es decir no disponen de un procedimiento. En esta segunda clase los problemas de solución iterativa y los de solución probabilística.

No siempre es factible establecer a cuál de las dos subclases pertenece un cierto problema, ya que existe una situación intermedia, las mejores soluciones que se conocen requieren de tiempos exponenciales pero sin embargo nadie pudo demostrar que no tienen una solución de tiempo polinómico, los denominados No Deterministicamente Polinómicos (NP) y un subconjunto de estos son los NP-completos e incluyen problemas clásicos de particular interés en lógica y matemática.