Teoría y aplicaciones de la estructuración del conocimiento

Adolfo Guzmán Arenas
Centro de Investigación en Computación, Instituto Politécnico Nacional
aguzman@pollux.cic.ipn.mx aguzman@amiac.org.org.mx

Disertación de ingreso como miembro de la Academia Mexicana de Ingeniería. Octubre de 1999.

RESUMEN. Varios de los métodos empleados, de los modelos construidos y de los problemas resueltos durante mi trabajo profesional, emplean o explotan el acomodo de los datos (o conceptos, o conocimiento, según el área que se trate) no en estructuras simples –como pueden ser matrices o bases de datos– sino en árboles donde hay padres, hijos y tíos, o en gráficas más elaboradas. Estas estructuras –aparentemente más complejas– resultan muy útiles para resolver un sinnúmero de problemas.

En este artículo trato de convencer al lector de la bondad de estas estructuras, y analizo el porqué de tales virtudes. Asimismo, doy ejemplos que demuestran (como era de esperarse) que no son útiles para todo.

1. CONSIDERACIONES GENERALES SOBRE EL USO DE ESTRUCTURAS DEL CONOCIMIENTO

Un programa de cómputo es un *conjunto de instrucciones* en un lenguaje de computadora, que procesa *datos de entrada* para producir *resultados* de salida, a fin de resolver un *problema* dado. Estos datos de entrada varían de un caso al otro. Representan, por ejemplo, los síntomas (temperatura, color de los ojos, datos del examen de orina) de un paciente.

A menudo, un programa se complementa o usa otros datos, más extensos, que no cambian de un caso al otro, de un paciente al otro, sino que describen "parte del entorno" necesario para resolver el problema dado. En el caso de un sistema experto que diagnostica enfermedades (su salida es la *enfermedad* o padecimiento) de pacientes (su entrada es el conjunto de síntomas del paciente), se requiere echar mano de datos generales sobre la diabetes, la tuberculosis, la hepatitis, etc.; de datos sobre el peso, talla, latidos/minuto, etc., de un paciente sano de 10 años, de 15 años, etc.; de datos sobre la probabilidad *a priori* de incidencia de gripe, disentería, amibiasis, etc. Otro ejemplo: es probable que un programa que haga predicciones del crecimiento de ciudades, tenga como datos genéricos el número de habitantes de diversos poblados a través del tiempo.

Esta información que no varía de una aplicación (uso) del programa a otra, se guarda en archivos, bases de datos, bases de conocimientos, de muy diversas formas. El propósito de este trabajo es resaltar la utilidad de los árboles y redes para guardar la información (especialmente la no numérica). Se dan ejemplos derivados de la experiencia del autor.

1.1 Definiciones

Datos del entorno, datos generales. Son los datos sobre el área de aplicación en la que el programa trabaja. No cambian de una aplicación a otra.

Conocimiento. Datos generales procesados u organizados para aumentar su poder de deducción o su utilidad. Por ejemplo, en reglas genéricas. Por ejemplo, por casos de uso frecuente.

Estructura. Conjunto de datos (llamados *campos*) heterogéneos que poseen nombre. Ejemplo: la estructura CLIENTE, con campos NOMBRE, DIRECCION, EDAD, SALDO, FECHA DE INGRESO. CLIENTE posee dos campos alfanuméricos (NOMBRE y DIRECCIÓN), dos numéricos (EDAD y SALDO), y uno tipo fecha. Una estructura reside en memoria principal.

Arreglo, matriz. Arreglo: conjunto de datos homogéneos (llamados *elementos*) que poseen índice. Ejemplo: el arreglo ESTADOS(30), que posee los 30 estados de la república mexicana. Matriz: arreglo bidimensional. A una matriz se le llama a menudo *tabla*, aunque también se usa el nombre tabla para denotar un archivo de una base de datos. Un arreglo yace en memoria principal.

Archivo. Arreglo de estructuras que yace en memoria secundaria (disco duro, disquete). Cada elemento se llama *registro.*

Base de datos. Conjunto de archivos que no contienen información redundante, y con estructura que obedece a ciertas otras reglas, tales que pueden responder correctamente cualquier pregunta formulada en el lenguaje SQL, lenguaje estándar de consulta. Reside en memoria secundaria.

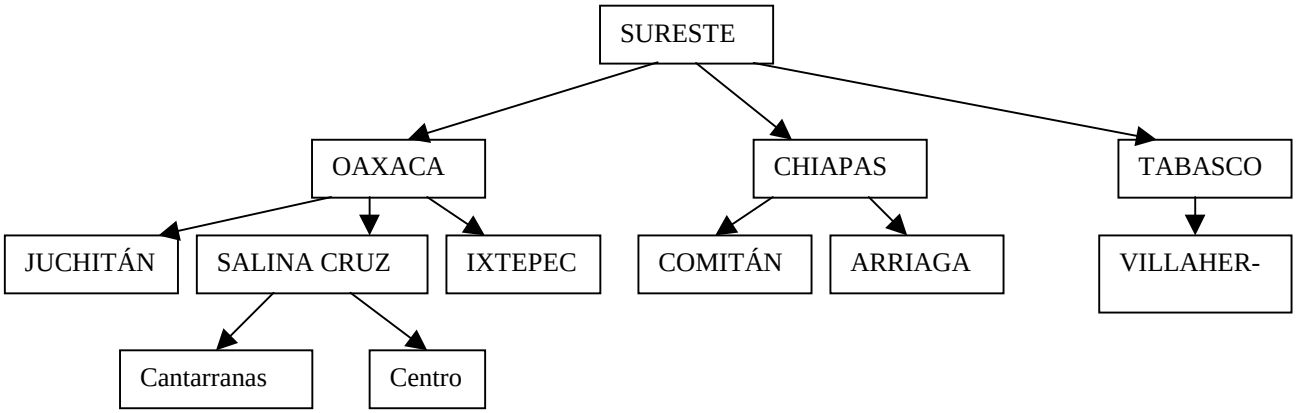
Objeto. Una estructura y los programas que la manipulan, vistos indivisiblemente. Yace en memoria principal.

Bases de objetos. Base de datos cuyo contenido son objetos.

Espacio de direccionamiento. Las celdas de memoria principal que un programa puede accesar.

Objeto distribuidos. Un objeto que puede migrar de un espacio de direccionamiento a otro, por ejemplo, de una computadora a otra.

Árbol. Estructura de datos formada por *nodos,* donde cada nodo puede tener uno o más hijos (numerados) no compartidos (un nodo puede ser hijo de cuando más otro nodo). Cada nodo es una estructura. Ejemplo: el árbol SURESTE.



El nodo SURESTE tiene como primer hijo OAXACA; su tercer hijo es TABASCO. COMITÁN o Cantarranas no tienen hijos –los nodos sin hijos se llaman *hojas.* Hay un solo nodo que no es hijo de nadie, se llama *raiz:* SURESTE en nuestro ejemplo.

Red. Estructura de datos formada por nodos, entre los cuales puede haber apuntadores llamados arcos dirigidos o relaciones. Estos arcos pueden formar *ciclos.* Puede haber distintos tipos de relaciones. Desde este punto de vista, un árbol es una red sin ciclos, con relaciones *mi primer hijo, mi segundo hijo,...*

Cómo aplanar un árbol. Aplanar un árbol es convertirlo en una secuencia de nodos, ninguno repetido, ninguno faltante. Para aplanar un árbol se *recorre* (se visitan los nodos) de alguna forma. Hay dos maneras comunes: (1) *A profundidad primero* («depth first»), donde se visita un hijo, los hijos de éste, etc., antes de pasar con el hermano siguiente. El árbol aplanado a profundidad primero de SURESTE es SURESTE OAXACA JUCHITAN SALINA CRUZ Cantarranas Centro IXTEPEC CHIAPAS COMITÁN ARRIAGA TABASCO VILLAHERMOSA. (2) *A lo ancho primero* («width first»), que consiste en visitar primero a los hijos, luego a los hijos de éstos, etc. El árbol aplanado a lo ancho primero de SURESTE es SURESTE OAXACA CHIAPAS TABASCO JUCHITÁN SALINA CRUZ IXTEPEC COMITÁN ARRIAGA VILLAHERMOSA Cantarranas Centro.

1.2 Usos principales de los árboles

Para representar jerarquías. En el ejemplo SURESTE, es fácil ver que OAXACA y CHIAPAS están al mismo *nivel*, en tanto que VILLAHERMOSA es hijo de TABASCO.

Para búsquedas rápidas. Si tengo que buscar entre 5000 poblaciones, se me ocurre dividirlas en grupos de 30, según el estado a que pertenecen. Así, encontraré más rápidamente a COMITÁN si la busco en CHIAPAS: no tengo que mirar las poblaciones de todo México, solo las de Chiapas.

Para permitir precisión arbitraria. En el ejemplo, la precisión (nivel de detalle de una descripción) llega al nivel de barrio o colonia (Cantarranas) en SALINA CRUZ, pero se queda a nivel de ciudad en VILLAHERMOSA. Si quisiéramos agregar más precisión a Cantarranas, podríamos hacerlo, anexándole hijos (manzanas) y nietos (predios), en tanto que podríamos dejar sin alterar la precisión de VILLAHERMOSA.

Para permitir precisión variable. Se acaba de explicar.

Para delimitar rangos de acción. De alguna manera, CHIAPAS está limitado a COMITÁN y ARRIAGA.

Para permitir recursividad. Podemos diseñar un programa P (como el generador de árboles k-d, de la línea 12, §2.8 y §5.12),[1] cuya entrada es una matriz donde cada renglón describe una cosa u objeto (un cliente, digamos), y cada columna describe una propiedad de ese objeto (v. gr., su edad, el color de sus ojos). De alguna manera, parte del algoritmo es eliminar algunas columnas de esta matriz, dando orígen a una sub-matriz, *a la que volvemos a aplicar P recursivamente.* Resulta, pues, que las matrices de cosas forman un árbol, donde los nodos sucesivos representan matrices con columnas eliminadas.

[1] Estas son las líneas de trabajo en las que ha incurrido el autor, y se detallan en el §5.

2. EJEMPLOS DONDE LOS ÁRBOLES SE EXPLOTAN VENTAJOSAMENTE

No lo aprendí en curso alguno, sino en la práctica: cómo usar árboles, redes y otras estructuras, para representar conocimientos. La exposición que sigue es aproximadamente cronológica.

2.1 Sistemas de información geográfica

En la línea 3,[2] Bases de datos geográficos (§5.3) [las publicaciones que describen los trabajos realizados se encuentran listados en el §5], me encontré en 1973 con el problema de representar una gran cantidad de detalles que existen en las cartas de Cetenal (ahora INEGI). Estábamos concentrados en las de escala 1:50,000. Miramos un arroyo intermitente, y vemos cómo serpentea entre los cerros, conforme baja por una cañada. Nos ensimismamos contemplando la gran cantidad de curvas de nivel (una cada 10 metros de diferencia en elevación) que cubren estos cerros. O la línea ondulada que separa un pastizal de un bosque. Toda esta información resulta muy detallada, sobre todo para capturar (introducir a la computadora). Alguna es relevante. Otra no lo es. ¿Qué hacer?

Inspirados por la misma descomposición que los geógrafos hacen de sus cartas (a distintas escalas), decidimos (Ernesto Bribiesca y yo) que íbamos a representar una superficie dada (limitada por un rectángulo, digamos por dos meridianos y dos paralelos) por un *registro* de un archivo. En este registro íbamos a anotar todo lo que ese rectángulo contenía, *pero sin precisar dónde.* Es decir, el rectángulo era, a ese nivel de precisión, *atómico* o indivisible. El primer nodo, o raíz, de la base de datos geográfica que construimos (inventamos un lenguaje para hacer preguntas generales, que resultó ser un subconjunto de FORTRAN, por lo que nos ahorramos el compilador –pero esa es otra historia), representa, pues, a MÉXICO. En ese registro pusimos cuántos ríos tiene México, cuántas ciudades, su altura máxima, su altura mínima, el porcentaje del territorio mexicano cubierto con matorral espinoso, cuántos faros hay en sus costas, etc. Todas las propiedades que Cetenal (INEGI) usa para describir superficies, líneas o puntos, y que existan en México. Detalles de esta representación en el §2.1.1.

Ahora bien, ¿dónde están localizados los faros? Esta pregunta no se resuelve en el rectángulo MÉXICO. Ahí solo dice: hay 53 faros (o los que sean). Si queremos saber más detalle sobre los faros (en muchas preguntas, no se requiere más detalle), tendremos que *bajar de nivel*, usando otros cuadros que representan a los hijos, nietos,... de MÉXICO. Por consiguiente, la raíz tiene como hijos las cartas 1:50,000, que representan el siguiente nivel de precisión. Cada una de estas cartas está representada por un registro, de estructura igual a la del padre. Para cada carta, registramos el número de "eventos geográficos" que la carta contiene: número de carreteras transitables en todo tiempo, número de poblados, porcentaje de la zona cubierta de manglares, ...

De cada registro que representa una carta 1:50,000 cuelgan sus hijos: son cuadrados de 2 por 2 kilómetros, ya venían marcados en las cartas. Se abre un registro para cada uno de estos cuadrados, y se representa la información de la misma manera.

[2] Estas líneas se encuentran, incluyedo referencias a publicaciones, en el §5, al final de este documento.

Si hay necesidad de representar alguna región con más precisión, se divide ésta en cuatro cuadrados de 500 metros cada uno, y se representa la información en ellos contenida. Y así sucesivamente.

Nótese que:

1. Unas regiones pueden representarse con más precisión (tener nietos y tataranietos) que otras.
2. Conforme bajamos de nivel, tenemos una idea (geográfica) de por dónde está el pastizal, por qué rectángulos pasa la carretera Panamericana, etc. O, si nuestra pregunta no requiere tanta precisión, nos conformamos con la información a menos precisión, respondiendo con rectángulos del tamaño de una carta 1:50,000, digamos.
3. Para muchos casos, si un nivel del árbol no contiene cierta información, es innecesario bajar más de nivel. Si una carta 1:50,000 no contiene islas, ninguna de sus sub-cartas (hijos) contendrá islas.

2.1.1 Detalles de la representación de cada rectángulo geográfico

Una propiedad superficial (bosque de encino, pastizal inducido, reolita –se representaron las cinco cartas 1:50,000 del INEGI: carta topográfica, de uso actual del suelo, de uso potencial, edafológica y geológica) se representa como *porcentaje del cuadro* cubierto por tal propiedad superficial. Si un cuadro está cubierto en un 53% por un lago, se representa como LAGO—53%.

Una propiedad lineal (ríos, carreteras, ferrocarriles,...) se representa viendo cuántas de ellas cortan el perímetro del rectángulo. Digamos que un cierto rectángulo es atravesado por una carretera de cuota. Entra por un lado y sale por otro. Entonces nosotros registramos: CARRETERAS DE CUOTA—2 (aunque en realidad alguien podría argüir que es solo *una* carretera). Si un poblado tiene una carretera que no cruza los lados del rectángulo, no cuenta.

Una propiedad puntual (poblado, faro, escollera, malecón, mina, panteón,...) se representa contando cuántas de ellas aparecen dentro del rectángulo. Si un rectángulo contiene 3 minas de plata, registramos: MINAS DE PLATA—3. También, para las poblaciones, se apuntaba del registro del rectángulo a otro registro –en otro archivo– donde se hallaban los *descriptores no geográficos* de esa población: cuántas escuelas primarias, si tiene calles pavimentadas, ...

Ahora bien, las propiedades que no existían en ese rectángulo *no se representaban.* Si en un rectángulo no existen arrecifes de coral, no decimos ARRECIFES DE CORAL—0, sino que simplemente no existe información en el registro sobre estos arrecifes. Están ausentes. Ausencia de información = no hay = cero (Este truco ahorró enormes cantidades de espacio en disco).

No importa el nivel o tamaño del cuadro representado, su estructura es la misma. Los porcentajes se refieren al total del área del cuadro representado, y para propiedades lineales y superficiales, se usa “número de”, como número de ríos, número de minas de plata,...

2.1.2 Lecciones aprendidas

A. La representación con “porcentajes” y “número de” nos permite acortar drásticamente la cantidad de información a guardar.

B. La estructura (que después se denominó quad-tree, o árbol cuaternario –porque en estos árboles cada nodo tiene cero o cuatro hijos) nos permite precisión variable, y precisión arbitraria.
C. El truco de no guardar las propiedades cuyo valor es cero, ahorró grandes cantidades de espacio en disco.

2.2 Hallando el parecido entre dos formas mediante su *número de forma*

En la línea 6, Descripción de formas mediante números de forma (§5.6), Ernesto Bribiesca y yo (1978; las publicaciones relevantes se encuentran listadas en el §5) queríamos describir formas bidimensionales (siluetas) a distinto grado de precisión, de suerte que las representaciones con una precisión dada dependieran de forma conocida de las representaciones menos precisas. Así, se nos ocurrió que podríamos intentar reducir una silueta arbitraria a una serie de siluetas estereotipadas (formadas por lados verticales y horizontales, todos formando parte de una retícula cuadrada). La idea para capturar el grado de precisión era el tamaño de la retícula.

Una mejor idea fue la siguiente: una vez normalizada la silueta (a llenar un cuadrado de lado 1), tratemos de rodearla de cuatro cerillos (de tamaño conveniente, todos iguales), o de seis, o de ocho, o de diez,..., ya sea horizontales o verticales. Una forma (silueta) estaría representada por *una* forma canónica a nivel 8 (la de los cerillos) , por otra forma canónica a nivel 10, etc. Estas formas forman un árbol. En el primer nivel (raíz) está la única forma de orden 4: un cuadrado. En el nivel siguiente está la única forma de nivel 6: un rectángulo de dos por un cerillos de lado. En el nivel siguiente están tres formas de orden 8: un cuadrado de 2 x 2, un triángulo, y un rectángulo de 3 x 1. En el siguiente nivel están las formas de orden 10, etc. No hay formas con un número impar de cerillos. Los *números de formas* se obtienen de las formas simplemente asignando un 1 a esquinas convexas, un 2 a "esquinas" de 180 grados (o sea, continuaciones, cerillos colineales), y un 3 a esquinas cóncavas.

Sea F una forma. Sea F_n la forma canónica de F, a nivel n. El parecido o *distancia* entre dos formas F y G se obtiene como sigue.

1. F_4 y G_4 son idénticas, pues solo hay una forma de nivel 4. Lo mismo sucede con F_6 y G_6.
2. Hállese el nivel n tal que $F_n = G_n$, pero $F_{n+1} \neq G_{n+1}$. Entonces se dice que la distancia entre F y G es n, y el parecido es 1/n. Es decir, la *distancia entre dos formas* es el "poder resolutivo" n de un lente que apenas las confunde, pero si aumentamos la precisión de ese lente, ya las distingue (las ve distintas).
   La métrica definida en el punto 2 está definida sobre el árbol de niveles.

2.2.1 Lecciones aprendidas

D. Se puede usar un árbol para medir distancia o similitud. Dos nodos siempre tendrán un ancestro común. Mientras más larga sea la trayectoria (se suman los arcos viajados) hacia el ancestro común, más distantes o disímiles serán los nodos. Ejemplo: la distancia entre Cantarranas y Centro es 2 (de Cantarranas a Salina Cruz hay un arco, de Salina Cruz hacia Centro hay otro, son dos en total); entre Cantarranas e IXTEPEC es 3, y entre Cantarranas y Comitán es 5.
E. Dos formas del mismo nivel son iguales o diferentes, y dos formas de distinto nivel no se pueden comparar. Ejemplo: las formas F_8 y G_{10} son inconmensurables, según la definición de *distancia entre dos formas* dada arriba. El método evita tener que hacer comparaciones numéricas entre dos formas, por ejemplo, usando mínimos cuadrados.

2.3 Representación de una superficie mediante una aproximación digital

En la línea 7, Modelos digitales del terreno (§5.7), quise aplicar (1978) la descomposición recursiva de una superficie en otras más pequeñas, y así sucesivamente, hasta lograr un cierto objetivo. Se trata de representar mediante un mosaico de triángulos planos pero no necesariamente paralelos con el plano X-Y, una superficie tridimensional dada (la superficie de una región de México), de tal manera que la diferencia entre la realidad y la representación no exceda un umbral o tolerancia (1 metro de altura, digamos).

Supongamos que la superficie inicial es un rectángulo, digamos el contenido por una carta 1:50,000.[3] Divídase en cuatro triángulos, trazando sus dos diagonales. Los vértices de cada uno de estos cuatro triángulos tienen coordenadas x, y definidas. No así la coordenada z de cada vértice, la que podemos ajustar de tal manera (esto equivale a inclinar los triángulos o separarlos del plano X-Y) que los triángulos aproximen la superficie terrestre dentro de la tolerancia dada. Esto podría ser si la carta contiene una superficie bastante plana, como un lago o un desierto. Si la superficie tiene cañadas y serranías, ninguna inclinación de los triángulos obtendrá la tolerancia deseada. Entonces, es menester dividir alguno(s) de los cuatro triángulos *en otros cuatro,* mediante la unión de los puntos medios de sus lados. Cada triángulo ahora tiene cuatro hijos. El proceso se repite. Ahora debemos inclinar los hijos tratando de que se ajusten al terreno (por encima o por debajo del mismo) dentro de la tolerancia especificada. Si un triángulo se ajusta, el proceso termina y el triángulo no engendra hijos. Los triángulos mal ajustados procrean hijos (cuatro triángulos menores), los que a su vez buscan ajustarse o procrear. Es de imaginarse que las zonas relativamente planas (aunque sean inclinadas) pronto quedarán cubiertas (descritas) por triángulos relativamente grandes, en tanto que en las cañadas y montes veremos triángulos pequeños aglomerados alrededor de los cerros y desfiladeros. El proceso es equivalente a tratar de ajustar la superficie con una hoja de papel arrugado, cuyos pliegues o arrugas forman triángulos de distinto tamaño.

Con esto hemos conseguido una representación jerárquica, que adquiere una precisión (constante, o fija en este caso) a costa de un mosaico de elementos –triángulos– de tamaño variable.

[3] Los conocedores dirán que una de estas cartas, limitadas por dos meridianos y dos paralelos, no contiene un rectángulo, ni es plana. Esto no afecta mi algoritmo, solo mi descripción de él.

2.3.1 Lecciones aprendidas

B. La estructura (que después se denominó quad-tree, o árbol cuaternario –porque en estos árboles cada nodo tiene cero o cuatro hijos) nos permite precisión arbitraria. (Esta lección ya había aparecido en §2.1, bajo la letra B).

2.4 Jerarquías de computadoras para procesar en paralelo

En la línea 8, Máquinas jerárquicas para procesamiento en paralelo (§5.8), Ed Krall y yo (1986) tuvimos una idea simple: agrupar varias de estas máquinas en un grupo («cluster»), aquéllas que comparten memoria –nuestra estructura es de memoria compartida–. Los grupos comparten a su vez memoria con otros grupos, y esto forma otro nivel en el árbol. Esta estructura ofrece la ventaja de que la mayoría de los accesos de un procesador son a su misma memoria, o a la memoria inmediatamente compartida –la del grupo–, en tanto que los accesos a las memorias lejanas son menos frecuentes. Esta misma organización de memorias se usó en la máquina CM* de Carnegie Mellon.

2.5 Representando el conocimiento común. El Proyecto CYC.

En la la línea 9, Representación del conocimiento en CYC (§5.9), el Proyecto CYC (encabezado por Douglas Lenat) trataba de construir el *árbol del conocimiento común* (1978). El conocimiento común es aquél que posee una persona cuando se le quita el conocimiento de su especialidad. Es también el que posee una persona que no tiene especialidad alguna, un niño de 10 años, digamos. Es la *intersección* de los conocimientos individuales de muchas personas. Se estima que hay entre un millón y diez millones de nodos (conceptos) del conocimiento común. Contiene los sustantivos, objetos o nombres; los verbos, acciones o procesos; los adjetivos (amarillo, caliente,...); las emociones; los polinomios,... No hay información muy especializada (que algunos sepan y otros no). Estos nodos están ligados por varias relaciones como parte de, habitante de, hijo de, dueño de,..., y sus *inversas:* mi parte es, habito, mi progenitor(a) es, mi dueño es,... Estas relaciones también son conceptos que están representados en el árbol (no es un árbol, sino una red ligada por muchas relaciones). La jerarquía en esta red no es obvia, pero en general podemos decir que va de conceptos generales a conceptos particulares.

Los usos de este árbol del conocimiento común eran varios:

(a) para evitar la *fragilidad* de los sistemas expertos, que saben mucho de algo y nada de todo lo demás –en contraste con un ser humano, que sabe (quizá) mucho de algo y *un poquito* de todo lo demás–, y por consiguiente, *sabe cuándo no sabe;*
(b) para poder entender textos en lenguaje natural, ya que el problema principal es la desambiguación, la que requiere de conocimientos del mundo que nos rodea;
(c) una vez logrado (b), poderle dar a la computadora órdenes en lenguaje natural;
(d) una vez logrado (b), poder crear una máquina que aprendera *leyendo libros* escritos en lenguaje natural.

El proyecto, después de 10 años de trabajo, fue terminado. Aunque no se logró construir el árbol, influyó en otros trabajos míos: línea 15, Clasitex (§2.11 y §5.15), que usa un arbolito simplificado del árbol de CYC, y línea 17, Interciencia (§2.12 y §5.17).

2.5.1 Lecciones aprendidas

En el proyecto aprendí una lección que no se aplica tan solo a árboles:

F. Si el objetivo de un proyecto es la construcción de X, *hay que dedicarse a hacer X*. El Proyecto CYC no logró construir el árbol (se hizo quizá un décimo del mismo), después de diez años de trabajo por un grupo de expertos.

2.6 Lenguaje visual para programar en paralelo

En la línea 10, Lenguaje visual para programar procesos paralelos (§5.10), la jerarquía viene a través de los nodos de una red. Resulta que estoy usando (1990) una red para representar programas. Los nodos representan otros programas (otras redes, o programas escritos en un lenguaje de tercera generación, como C o Ada). Luego, la abstracción es a nivel nodo. Un nodo puede estar constituido por otra red, y así sucesivamente.

2.6.1 Lecciones aprendidas

Las lecciones no tienen que ver con árboles para representar el conocimiento.

G. Los flujogramas o diagramas de flujo representan programas paralelos muy fáciles de paralelizar, y donde el régimen de ejecución no importa. Se obtienen los mismos resultados si, de todos los nodos que ya pueden ejecutarse, se ejecutan en cualquier orden, ya bien sea secuencial o simultáneamente. De hecho, es posible construir hardware que los paraleliza. Tal es la interpretación que hoy le doy a la computadora AHR (§3.4 y §5.5) que se construyó en el IIMAS-UNAM.
H. Los trámites, procesos de negocio y procesos administrativos son susceptibles de ser analizados por la computación, pues son procesos distribuidos. Ver §2.7.

2.7 Flujo de trabajos y trámites

En la línea 11, Flujo de trabajo de documentos, procesos de negocio, tramitología (§5.11), se usan los flujogramas del §2.6 para representar trámites o procesos de negocio, y en general procesos distribuidos en el tiempo y en el espacio, cuyos ejecutores son personas. En 1993 diseñé uno de estos sistemas. En 1999, en su tesis de maestría, Cecilia Palomino construyó un *ejecutor* de trámites que usa correo electrónico. La jerarquía de la representación es la misma que la del §2.6.

2.7.1 Lecciones aprendidas

La lección no tiene que ver con árboles para representar el conocimiento.

H. Los trámites, procesos de negocio y procesos administrativos son susceptibles de ser analizados por la computación (ya aprendida en el §2.6.1).

2.8 Árboles k-d

En la línea 12, árboles k-d (§5.12), un programa P analiza (1995) una matriz de objetos (llamada matriz de aprendizaje, o MA) a fin de construir un clasificador supervisado CS (un clasificador es un programa que asigna *clases* a su argumento de entrada. Se llama supervisado cuando conoce de antemano las clases a asignar), que toma la forma de un árbol de

decisiones. Este árbol es de hecho un programa en C con muchos CASE (muchos condicionales) anidados. El objeto del programa P es producir el programa CS. El programa CS clasifica su argumento de entrada según lo aprendido de la matriz de aprendizaje MA. Para construir CS, P se fija en la columna de la MA que *menos confusión* introduce,[4] y produce una pregunta (una condición) sobre esa variable. El programa generado se ve como

```
SWITCH (edad) {
CASE 1: ... más código 1;
CASE 2: ... más código 2;
.....
CASE 99: .. más código 99;
ELSE ...  más código
  }
```

Acto seguido, el programa considera de MA solo aquéllos objetos con EDAD=1, ignora la edad (puesto que no discrimina ya, todos tienen edad=1), y –de la nueva matriz resultante MA', muy reducida de tamaño–, busca a su vez cuáles de *sus* columnas causa menor confusión, etc. Este código va donde dice "más código 1." Aplica este proceso a los objetos de MA con EDAD=2, para generar el código que va donde dice "más código 2", etcétera. Es decir, la matriz MA pronto se ve despedazada en muchas matricitas que dan lugar (usando el programa P recursivamente) a condicionales del tipo SWITCH que se ensamblan unos dentro de otros.

2.8.1 Lecciones aprendidas

Esta lección es muy útil, pues aprendí cómo hacer recursiones con matrices y arreglos.

I. El árbol se representa como un programa en C, que contiene un nido de condicionales dentro de condicionales. Es un árbol de decisión clásico, donde el orden en que aparecen las preguntas (si primero pregunto por la edad, y luego por el peso, ...) viene dado por el poder de discriminación relativo de cada variable.

2.9 ANASIN, una bodega de datos con replicación automática

En la línea 13, Anasin, analizador-sintetizador inteligente, bodega de datos (§5.13), construí una serie de programas (1993), distribuidos en una región del territorio nacional, que envían de manera automática datos a otros programas. Los programas forman un árbol, el mismo árbol que forman los centros de trabajo de la empresa. Generalmente, una empresa tiene una oficina corporativa o nacional, luego varias oficinas regionales, de las cuales dependen los centros de trabajo. Esta es una empresa de tres capas. En cada oficina (nacional o regional) y en cada centro de trabajo se instala un módulo de ANASIN, para bombear

[4] Una columna de MA es un rasgo o variable de un objeto (un objeto es un renglón de MA). Por ejemplo, si los objetos son clientes, entonces la primer columna puede ser NOMBRE, la segunda, EDAD, la tercera, PESO, la cuarta, COLOR DE OJOS,... La *confusión* introducida por una variable es el número de pares de objetos en MA que tienen el mismo valor de esa variable, pero que pertenecen a clases distintas. Por ejemplo, si MA tiene un objeto K de la clase *militar* con EDAD=30, un objeto L de la clase *médico con* EDAD=30, y tres objetos M, N, O de la clase *torero* con EDAD=30, entonces la confusión introducida por EDAD = número de pares de objetos con EDAD=1 pero en clases distintas + ... + número de pares de objetos con EDAD=99 pero en clases distintas. Si (para simplificar nuestro ejemplo) la única edad es 30, entonces la confusión introducida por EDAD = par (K, L) + par (K, M) + par (K, N) + par (K, O) + par (L, M) + par (L, N) + par (L, O) = 7. Nótese que los pares (M, N), (M, O) y (N, O) no cuentan, por ser ambos objetos de la misma clase. No contribuyen a la confusión.

datos “corriente arriba” (hacia la raíz del árbol, es decir, hacia oficinas nacionales) y recibir datos (resumidos) de los nodos “corriente abajo”.

2.9.1 Lecciones aprendidas

La lección aprendida es muy importante:

J. Usa árboles isomorfos con la estructura de la empresa. Guíate por el cliente, por la organización que él le dio a su negocio, a sus productos, a sus clientes.

2.10 Mineros de datos

En la línea 14, Minería de datos (§5.14), el autor busca (1995) cómo hallar automáticamente situaciones interesantes, anomalías y desviaciones, en un mar de datos (en una bodega de datos, como la que constituyen los datos recolectados por ANASIN, del §2.9). Los mineros de ANASIN usan un cubo de dimensión k (generalmente k=3), donde cada eje es un árbol aplanado (Cf. §1.1 Definiciones), por ejemplo, un eje es el eje de productos que la empresa vende (ferretería, papelería, abarrotes,...); otro eje es el eje geográfico donde están sus expendios o centros de trabajo (Jalisco, Guadalajara, Colonia La Libertad, Tlaquepaque,...); otro eje es el eje del tiempo.

2.10.1 Lecciones aprendidas

J. Usa árboles isomorfos con la estructura de la empresa. Esta lección ya se había aprendido en el §2.9.1.

2.11 CLASITEX, o de qué habla un texto en español

En la línea 15, Clasitex, hallando los temas principales de un artículo en español (§5.15), A. Guzmán analiza (1994) artículos en español, para encontrar de qué temas o tópicos hablan. Se usa un árbol parecido al de CYC, aunque muy simplificado. El árbol de CLASITEX solo tiene una relación: “sugiere el tema”, o “habla de.” Los nodos del árbol representan conceptos.

2.11.1 Lecciones aprendidas

K. Es útil representar conocimientos extensos utilizando conceptos como nodos. Esta era la premisa de CYC (§2.5 y §5.9).

2.12 La Interciencia

En la línea 17, Interciencia (§5.17), A. Guzmán discute (1999) cuándo dos agentes o personas con taxonomías del conocimiento distintas se pueden comunicar, y concluye de una manera algo pesimista.

2.12.1 Lecciones aprendidas

L. La comunicación es posible solo entre dialectos (pequeñas variantes) de un mismo lenguaje. Para que dos agentes (o dos personas, o una persona y un agente) se comuniquen,

deben tener un número suficientemente grande de conceptos comunes, organizados análogamente –o de la misma manera.

2.13 Colaboración e interacción entre agentes con múltiples hebras

En la línea 18, Interacción dirigida entre agentes con propósito (§5.18), Jesús Olivares, Araceli Demetrio, María del Carmen Ortega y yo, tratamos (1999) de hacer que varios agentes interactúen e intercambien información con propósitos y fines dados. Para esto, los agentes participan en *interacciones*, que son scripts o frames (según Marvin L. Minsky) u "obras de teatro", por ejemplo, *en un restaurante*, *en una subasta*, etc. Cada agente tiene metas, ambiciones, recursos, y posee varias hebras de ejecución (puede hacer varias cosas a la vez, por ejemplo, comer y pensar, o ir de vacaciones y –dentro del script *en vacaciones*– adquirir el papel de "comensal" en el script *en un restaurante*). La cosa se complica porque (a) los agentes no hablan exactamente el mismo idioma, de manera que los intercambios de información entre ellos tienen que ser analizados por un programa que compara y maneja *ontologías mixtas;* y (b) el mundo en que viven sufre de la aparición de *eventos inesperados*, como terremotos, huracanes, visitas de la suegra, me encuentro con un amigo entrañable en la calle, me encuentro con un billete de 100 pesos tirado,... Por cierto, hay un número infinito de estos eventos, no todos los cuales podemos percibir (o nos interesan).

En cuanto a árboles, el *comparador de ontologías* tiene que usarlos para resolver qué conceptos de un árbol se mapean a cuáles conceptos del otro. Por ejemplo, uno de los agentes quiere comprar calefactores, pero el que vende estas cosas tiene hornillas, calentadores, lanzallamas, hornos, equipos de calefacción, estufas, anafres y braseros, y tienen que ponerse de acuerdo. Otro ejemplo: yo quiero melocotones, tú vendes chabacanos. Y él, albaricoques.

2.13.1 Lecciones aprendidas

No llevamos mucho tiempo en este proyecto.

L. La comunicación es posible solo entre dialectos (pequeñas variantes) de un mismo lenguaje. Ver §5.17 Interciencia, donde se filosofa sobre esto.

M. Frente a un desconcierto, se recurre a un nodo más arriba (más cerca de la raíz, por ejemplo, al padre p del nodo que desconcierta) para ver si ahí ya no hay desconcierto. Y luego se procede hacia abajo, tratando de fijarse cuidadosamente cuál de los hijos h_1 h_2 ... h_k del nodo padre p en el árbol 1 es el mismo que alguno de los hijos i_1 i_2 ... i_m del nodo padre p en el árbol 2 (para simplificar el ejemplo, suponemos que él árbol 1 y el 2 tienen el mismo nodo p, *coinciden* en el concepto p). Es decir, se trata de hallar r y s tales que $h_r = i_s$.

3. EJEMPLOS EXITOSOS QUE NO USAN ESTRUCTURAS DE REDES

Arreglos. Se utilizan para guardar sobre todo datos numéricos, accesables mediante un índice. Las matrices de aprendizaje del §5.12 son un ejemplo exitoso.

Archivos. Idem.

Bases de datos. Gran parte de la información es guardada actualmente en bases de datos, las que no poseen una estructura jerárquica implícita.

3.1 CONVERT, lenguaje para manipular símbolos

En la línea 1, Convert, lenguaje para manipulación simbólica (§5.1), se manejan (1966) listas.

3.2 SEE. Hallando los cuerpos que componen una escena

En la línea 2, descomposición de una escena visual en cuerpos tri-dimensionales (§5.2), se manejan dibujos constituidos por líneas rectas (1968).

3.3 Percepción remota: análisis de fotografías de satélite del territorio nacional

En la línea 4, P. R., Percepción Remota: análisis de imágenes de satélite (§5.3), se manipulan imágenes (1976) en tres dimensiones: X, Y y B (la banda del satélite).

3.4 AHR: computadora formada por varios microprocesadores

En la línea 5, AHR: computadora paralela para procesar lenguajes simbólicos (§5.4) no se manejan (1980) estructuras jerárquicas significativas.

3.5 EVA. Espacios virtuales de aprendizaje

En la línea 16, EVA, espacios virtuales de aprendizaje – Software para educación no presencial y asíncrona (§5.5), no se manejan (1998) estructuras jerárquicas significativas.

4. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

Los árboles pueden usarse fructíferamente para representar mediante una gran variedad de formas, datos extensos, sobre todo, datos simbólicos (no numéricos). El documento provee varios ejemplos.

Algunos tipos de conocimiento pueden ser representados de manera más sencilla, con arreglos o bases de datos.

Ver también las *lecciones aprendidas* A a M, en el cuerpo de este documento.

4.2 Recomendaciones

No tema representar conocimiento simbólico, o conocimiento extenso. Use primero estructuras simples: arreglos, archivos. Si esto no funciona, use bases de datos. Luego, si

lo anterior no resuelve el problema, use árboles. Finalmente, use estructuras más complejas.

Conserve simples sus estructuras en disco (memoria secundaria): cada apuntador toma 30 milisegundos en leer lo apuntado (el disco tiene que mover su cabeza móvil). No tema usar estructuras complejas en memoria, aquí seguir un apuntador toma unos 50 nanosegundos.

4.3 Sugerencias para trabajos futuros

Haga programas que formen automáticamente los árboles. Es decir, que aprendan.

Despliegue gráficamente los árboles que forma (sobre todo si los construye manualmente), y dote a su programa de despliegue de la habilidad para manipular los árboles: mover sub-árboles de lugar, mover un nodo, borrarlo, agregar uno nuevo, etc.

5. APÉNDICE. ÁREAS DE INVESTIGACIÓN DE A. GUZMÁN, ORGANIZADAS POR TEMAS

Aquí, a manera de apéndice, presento los temas principales en los que he trabajado, organizados por temas. Esto permite "ver el bosque en vez de los árboles."

5.1 CONVERT, lenguaje para manipulación simbólica

Se creó el lenguaje CONVERT [4, 6,7], para manipular símbolos. Está implementado como un intérprete sobre Lisp. Esta fue la tesis de licenciatura de A. Guzmán, dirigida por Harold V. McIntosh. CONVERT trabaja con patrones contenidos en una expresión, la que compara contra otra expresión (el "espacio de trabajo") que contiene solamente constantes. Los patrones contienen variables libres que pueden tomar valores, a fin de casar. Un patrón casa con una expresión si hay una asignación válida de valores a las variables del patrón.

El lenguaje fue empleado por McIntosh para álgebras de grupos aplicadas a problemas de física. También, Raymundo Segovia usó Convert para hacer compiladores, y Gerardo Cisneros para un compilador de REC. Otras aplicaciones de CONVERT son para analizar escenas conteniendo dibujos con líneas (en "modo vector", se diría hoy) [8, 9, 10, 11, 18], y para hallar trayectorias en una gráfica [13].

3. McIntosh, H.V., Barberan, J. and Guzmán, A. *LISP Conversion.* Program Note #3, CINVESTAV, I. P. N.. February 1965.

4. Guzmán, A. *CONVERT-Diseño de un lenguaje para manipulación de símbolos, y su correspondiente intérprete.* (En español, inglés y Braille). Tesis de Licenciatura. ESIME, I. P. N.. Cd. de México. Agosto 1965.

* 5. Guzmán, A. TRACE Y HUSMEA: dos funciones LISP para depurar programas. *Ciencias de la Información y Computación 1,* Junio 1966.

6. Guzmán, A. and McIntosh, H.V. *A Program Feature for CONVERT.* Memorandum MAC M 305 (AI Memo 95). Project MAC, M. I. T April 1966.

* 7. Guzmán, A. and McIntosh H.V. "CONVERT" *Communications of the ACM 9, 8.* August 1966. 604-415. Also available as Memo MAC M 316 (AI Memo 99), M. I. T. June 1966.
8. Guzmán, A. *Polybrick: Adventures in the domain of parallelepipeds.* Memorandum Project MAC, MAC M 308 (AI Memo 96) M. I. T. Mayo 1966.
9. Guzmán, A. *Scene Analysis Using the Concept of Model.* Report 67-0133; Computer Corporation of America. Cambridge, Mass., USA January 1967. AD-652-017.
10. Guzmán, A. *A primitive Recognizer of Figures in a Scene.* Memorandum Project MAC, MAC M 342 (AI Vision Memo 119) M. I. T., January 1967.
11. Guzmán, A. *Some Aspects of Pattern Recognition by Computer.* Master's Thesis, Department of Electrical Engineering, M. I. T., February 1967. AD-656-041. Also available as a Project MAC Technical Report, MAC TR 37.
12. McIntosh H.V.and Guzmán, A. *A Miscellany of CONVERT Programming.* Project MAC Memorandum MAC M 346 (AI Memo 130) April 1967.
* 13. Guzmán, A. and McIntosh H.V. "Comments on ´All Paths Through a Maze´". *Proceedings of the IEEE.* Vol. 55 No. 8, 1525-27. August 1967.
* 14. Guzmán, A. and McIntosh, H.V. "Patterns and Skeletons in CONVERT". In *The Programming Language LISP: Its Operations and Applications,* Vol. II; Berkeley, E. C. y Bobrow, D. G. (eds). MIT Press, Cambridge, Mass. 1969.
* 18. Guzmán, A. Object Recognition: Discovering the Parallelepipeds in a Visual Scene. *Proceedings of the Second Hawaii International Conference on Systems Sciences.* Universidad de Hawaii, Honolulu, Hawaii, January 1969, 479-482, Western Periodicals Co.

5.2 Descomposición de una escena visual en cuerpos tri-dimensionales

Un programa, llamado SEE [15, 16, 17, 19] analiza una escena y la descompone en los cuerpos que la forman. Estos cuerpos son poliedros arbitrarios. Se usa un método llamado de propagación de restricciones.

15. Guzmán, A. *Decomposition of a Visual Scene into Bodies.* Project MAC Memoramdum MAC M 357 (AI Memo 139). MIT, September 1967.
* 16. Guzmán, A. Decomposition of a Visual Scene into Three-dimensional Bodies. *Proceedings of the AFIPS Fall Joint Computer Conference.* Vol. 33, First Part, 291-304. December 1968. Also available as Project MAC Memorandum MAC M 391 (AI Memo 161).
* 17. Guzmán, A. Analysis of Scenes by Computer: Recognition and Identification of Objects. In *Interpretation and Classification of Images.* Grasselli, Antonio (ed) Académic Press, 1969. Chapter 12.
19. Guzmán, A. *Computer Recognition of Three-dimensional Objects in a Visual Scene.* Ph. D. Thesis, Electrical Engineering Department, M. I. T., December 1968. Also available as Project MAC Technical Report MAC TR 59. AD-692-200.
* 20. Guzmán, A. Analysis of Curved Line Drawings Using Context and Global Information. In *Machine Intelligence VI.* D. Michie y B.Meltzer (eds.), Univ. of Edinburg Press, 1970. 325-375.

5.3 Base de datos geográficos

Se utiliza un árbol de cuadrados o rectángulos (después llamado árbol quad, o quadtree) para representar regiones geográficas con precisión arbitraria y variable. Se construyó con esta representación una base de datos que puede contener (describir) cualquier región, con atributos superficiales (un lago) o de área, de línea (un río) y puntuales (un faro).

* 23. Guzmán, A. y Bribiesca, E. Uso de una Base de Datos Geográfica. Memorias del Primer Congreso Panamericano y del III Congreso Nacional de Fotointerpretación, Fotogrametría y Geodesia. Cd. de México. Julio 7-12, 1974.
24. Bribiesca, E., y Guzmán, A. Manual de usuario para la explotación de una Base de Datos Geográfica. Reporte CCAL 74 17, Centro Científico IBM de América Latina. Cd. de México. Diciembre 1974.
25. Guzmán, A. Uso de computadoras en la Planeación Regional. *Memorias de la 1ª. Reunión Anual de la Academia Nacional de Ciencias (México).* Cd. de México. 15-24 julio 1975. Describe nuestras experiencias en la planeación regional utilizando la base de datos geográfica de CETENAL (ver referencia 24).

5.4 P. R. Percepción Remota: análisis de imágenes de satélite

Se analizan espectralmente imágenes del territorio mexicano tomadas por el satélite LANDSAT. Se hicieron aplicaciones a detección de cosechas, cuerpos de agua, y cubrimiento de terreno (uso del suelo).

26. Guzmán, A. *Proyecto P.R. Reporte de Actividades y Resultados. Etapa cero.* Reporte Técnico P.R. 72 2A, Octubre 1975. IIMAS, Universidad Nacional de México. En este reporte describo la realización del Proyecto de Percepción Remota.
27. Guzmán, A. Percepción Remota a través de computadoras: *Equipo, programas y aplicaciones.* Reporte Técnico PR 75 2, noviembre 1975. IIMAS-UNAM.
29. Guzmán, A. Aplicaciones del Reconocimiento de Patrones al análisis por computadora de Imágenes Aéreas. *Memorias de MEXICON 76, IEEE*, 7-9 de julio, 1976. Cd. de México.
* 31. Guzmán, A. Seco, Rosa and Sánchez, Víctor Germán. Analysis of LANDSAT Images for Crop Identification in Mexico. *Proceedings of the International Conference on Information Science and Systems,* 19-24, August 1976. University of Patras, Greece. 361-366. Se describen en este documento los resultados de la identificación de trigo, algodón y otros cultivos encontrados en el noreste de México en la Etapa Uno del proyecto P.R. También disponible como Reporte Técnico Vol. 7 No. 135, 1976. IIMAS-UNAM.
32. Guzmán, A. Percepción Remota en la UNAM. *Memorias de la II Reunión para el uso de datos de satélites.* Noviembre 1976. Comisión Nacional del Espacio Exterior, SCT. Cd. de México. Noviembre 1976. Se describen los principales esfuerzos realizados en la investigación en Percepción Remota y Procesamiento de Imágenes en la UNAM.

5.5 AHR: Computadora paralela para procesar lenguajes simbólicos

Se diseñó, simuló y construyó una computadora (AHR), que tiene varios procesadores, no posee sistema operativo en hardware, y usa Lisp como lenguaje principal.

AHR significa Arquitecturas Heterárquicas Reconfigurables. Heterárquico es: no jerárquico, pero tampoco anárquico. Aunque no haya jerarquía entre las computadoras, éstas se organizan para desarrollar trabajo útil. Compare con §1.8 (computadoras jerárquicas).

También se hizo el diseño para convertir una computadora (PS-2000) soviética, de tipo SIMD, en una computadora de tipo MIMD, como la AHR, *sin cambiar el hardware ni la arquitectura de la máquina.*

* 30. Guzmán, A. and Segovia, Raymundo. A Parallel Reconfigurable LISP Machine. *Proceedings of the International Conference on Information Science and Systems,* 19-24, August 1976. University of Patras, Greece. 207-211. Este artículo describe la estructura y características de un nuevo tipo de computadora que utiliza LISP como su lenguaje máquina y realiza tanto procesamiento en paralelo como es posible. También disponible como Reporte técnico Vol. 7, No. 133, 1976. IIMAS-UNAM.
* 36. Guzmán, A. Arquitecturas Heterárquicas Reconfigurables para Procesamiento Digital en Paralelo con lenguajes de alto nivel. Presentado en la Conferencia Internacional de Informática y Equipo de Oficina. México, 1979.
38. Guzmán A. Arquitecturas heterárquicas para Procesamiento en Paralelo de Imágenes Digitales. Reporte Técnico AHR-79-3 (PR-79-23), IIMAS UNAM. También: presentado en el II Seminario Internacional del uso de Percepción Remota. 1979. Cd. de México.
45. Guzmán A., Lyons, L. , et. al. La computadora AHR: Construcción de un procesador con LISP como su lenguaje principal. Reporte Técnico AHR 80 10, IIMAS, UNAM. 1980.
46. Guzmán A. y Rosenblueth, D. (eds.). Estructuras de Computadoras Digitales. Reporte Técnico AHR 80 11, IIMAS UNAM.
* 47. Guzmán A. A Parallel Heterarchical Machine for High Level Language Processing. In *Languages and Architectures for Image Processing.* M. J. B. Duff y S. Levialdi (eds). 1981. Academic Press. 230-244. Also in: *Proc. 1981 International Conference on Parallel Processing,* 64-71.
* 48. Guzmán A. A Heterarchical Multi-microprocessor Lisp Machine. *Proceedings of the 1981 IEEE Workshop on Computer Architecture for Pattern Analysis and Image Database Management.* Hot Springs, VA 1981. 309-317. IEEE Publ. #81CH-1697-2.
* 49. Guzmán A., Lyons, L. *et. al.* Construcción de un Multiprocesador que utiliza Lisp. Parte I: Teoría de la operación de una máquina AHR. *Memorias del VII Congreso de la Academia Nacional de Ingeniería* . Oaxaca, México, 1981.
* 50. Guzmán A., Lyons, L. *et. al.* Construcción de un Multiprocesador que utiliza Lisp. Parte II: Arquitectura de la máquina AHR. *Memorias del VII Congreso de la Academia Nacional de Ingeniería* . Oaxaca, México, 1981.
* 51. Guzmán, A. Computadora Mexicana de Procesamiento en Paralelo. *Información Científica y Tecnológica,* Vol. 3, No. 48, páginas 42-43.

* 52. Guzmán, A. A multi-microprocessor that executes pure Lisp in parallel. *Proc. 15th Hawaii International Conference on System Sciences,* 1982, Vol. 1, pages 368-377.
53. Norkin, Kemer y Guzmán, Adolfo. Diseño y Construcción de una Máquina Paralela Heterárquica: Reporte final del proyecto AHR. Reporte técnico AHR 82 21, Laboratorio AHR, IIMAS, UNAM. 1982.
54. Guzmán, A.; Gerzso, J. Miguel; Norkin, Kemer B. y Kuprianov, Boris. La computadora PS-2000 SIMD: Descripción Funcional y Conjunto de Instrucciones. Reporte técnico AHR 82 23, Laboratorio AHR, IIMAS UNAM. 1982. (En inglés).
* 56. Guzmán A., Gerzso, M., Norkin, K. B., and Vilenkin, S. Y. The Conversion via Software of a SIMD Processor into a MIMD Processor. PS2000, an Array Processor, becomes AHR, a general purpose LISP machine. In *Computer Architectures for Spatially Distributed Data,* H. Freeman y G. G. Pieroni (eds), Springer-Verlag. 1985. 121-137.
61. Norkin, K. B., y Guzmán A. Especificaciones Funcionales y Diseño Preliminar de HECTOR, una Heterarquía de Microprocesadores. Reporte Técnico. Departamento de Ingeniería Eléctrica, CINVESTAV I. P. N. 1985.
63. Guzmán A.; Gerzso, M.; Norkin, K.; Logunova, N.; Vilenkin, S.; y Kuprianov, B. Diseño Funcional de un Intérprete Lisp para la Computadora PS-2000 SIMD. Reporte Técnico AHR 83 24, IIMAS, UNAM. 1983.
64. Guzmán A. Tutorial de computadoras de quinta generación. Reporte Técnico AM 28, Departamento de Ingeniería Eléctrica, CINVESTAV I. P. N.. 1985.
* 66. Guzmán A. A Multiprocessor that executes Pure Lisp in Parallel. *Proc. 15th Hawaii International Conference on System Sciences,* 1982, V. 1, 368-377.
* 67. Guzmán A. AHR: A Parallel Computer for Pure Lisp. In *Parallel Computation and Computers for Artificial Intelligence,* J. S. Kowalik (ed.), 201-222. Kluwer. 1988.Also: MCC Technical Report PP 355 86, Microelectronics and Computer Technology Corporation, Austin, Tx.

5.6 Descripción de formas mediante números de forma

Se inventaron los números de forma, que permiten describir, con cadenas de los dígitos 1, 2 y 3, cualquier forma o silueta bidimensional, y medir el parecido entre dos formas así descritas.

* 33. Bribiesca. E., y Guzmán, A. Shape Description and Shape Similarity Measurement for Two-dimensional Regions. *Proceedings of the 4th International Conference on Pattern Recognition,* Kyoto, Japón, 1978. 608-612. Also available as Technical Report PR 78 18 (Orange Series 9, 166), IIMAS UNAM, Also in: *Journal of Geoprocessing,* Vol. 1, No. 2, 129-144 (1980).
34. Bribiesca, E. y Guzmán, A. *Números de Forma: una notación para describir formas puras y para medir semejanzas y diferencias en formas.* Reporte Técnico PR 78 20 (Serie Naranja 178), IIMAS UNAM, 1978.
* 39. Bribiesca, E. y Guzmán A. How to Describe Pure Forms and how to Measure Differences in Shapes using Shape Numbers. Invited paper to the *IEEE Conference on Pattern Recognition and Image Processing.* Chicago, USA. Also in *Pattern Recognition,* Vol 12, No. 2, 1980, 101-112. This article won the Seventh Annual Pattern Recognition

Award, awarded by the Pattern Recognition Society in November 1981, as the best article of the year.

5.7 Modelos digitales del terreno

Se usó una estructura jerárquica (árboles cuaternarios, o árboles quad) para representar superficies tridimensionales (como la superficie de un país), construyendo un mosaico de triángulos, a fin de poder representar tal superficie con cierta precisión dada.

* 35. Gómez, Dora y Guzmán, A. A digital Model for Three-dimensional Surface Representation. *Journal of Geoprocessing* 1, 1979, 53-70. Elsevier Publishing Co. Also in: *Proceedings of an International Conference "Computer Mapping for Resource Analysis"*, a CoGeoData Conference. Kansas Geological Survey, University of Kansas, and Instituto de Geografía de la UNAM. Mexico, 1978, pages 183-204.

5.8 Máquinas jerárquicas para procesamiento en paralelo

Se diseñó y simuló una arquitectura de un procesador en paralelo, formada por jerarquías de procesadores compartiendo memoria común.

68. McGehearty, P., and Guzmán A. Hierarchical Parallel Architectures for Symbolic Processing. MCC. Technical Report PP-387-86.
69. Guzmán A., Krall, E. J., McGehearty, P. F., and Bagherzadeh, N. *Measurement of Symbolic Applications* on a Parallel Architecture. Technical Report PP-076-87, MCC. Austin, Texas.
* 70. Guzmán A., Krall, E. J., McGehearty, P. F., and Bagherzadeh, N. Performance of Symbolic Applications on a Parallel Architecture. Technical Report PP-163-87., MCC. Also in *International Journal of Parallel Programming*, 16, 3, June 1987.
* 71. Guzmán A., and Hermenegildo, M. Constructs and Evaluation Strategies for Intelligent Speculative Parallelism –Armageddon revisited. Technical Report PP-220-87., MCC. Also in *Proceedings of the ACM 1988 Sixteen Annual Computer Science Conference*, Atlanta, Ga. February 1988.
* 75. Guzmán A., Krall, E. J., McGehearty, P.F. y Bagherzadeh, N. The Effect of Application Characteristics on Performance in a Parallel Architecture.Technical Report ACA-262-87., MCC. Also in *Proceedings of the Twenty First Hawaii International Conference on System Sciences*, January 1988. Honolulu, Hawaii.

5.9 Representación del conocimiento en CYC

El proyecto CYC buscaba representar el conocimiento común, formando un árbol de conceptos, ligados por múltiples relaciones (las que a su vez eran nodos en otra parte del árbol).

72. Guzmán A. *Canonical Shape description for 3-D Stick Bodies*. Technical Report ACA-254-87., MCC. Austin, Tx.

73. Guzmán, A. *Shape Description in CYC.* Technical Report AI-x-87(P), Artificial Intelligence Laboratory, MicroElectronics and Computer Technology Co. Austin, Tx, 1987. MCC/ACA Confidential and Proprietary.
74. Guzmán, A. *Existing and Projected Functionalities of CYC.* Technical Report ACA-AI-331-87(P). Artificial Intelligence Laboratory, MicroElectronics and Computer Technology Co., Austin, Tx., 1987. MCC/ACA Confidential and Proprietary.

5.10 Lenguaje visual para programar procesos paralelos

Se diseñó y construyó (intérprete y compilador) un lenguaje donde los programas se expresan como redes dirigidas. Cada nodo es otro programa visual o un programa en C (o Ada). Los arcos llevan información (resultados de salida, que se convierten en datos de entrada para el siguiente nodo corriente abajo) representada por objetos. Este lenguaje permite expresar cálculos en paralelo. El régimen de ejecución es el de flujo de datos: un nodo se dispara (y se ejecuta sin interrupción) cuando todos sus datos de entrada están presentes en los arcos de entrada. El compilador genera código distribuido, pudiendo especificarse la máquina objeto (una arquitectura paralela o distribuida) para la cual se desea el código.

*76. Ramón D. Acosta and Adolfo Guzmán. An environment for functional and performance prototyping of parallel programs. In *Collected Papers of the 1991 Workshop on Hardware/Software CoDesign*, XIII, International conference on Software Engineering, Technical Report No. MCC-CAD-156-91, Microelectronics and Computer Technology Corporation, Austin, Tx, May 1991.
77. Guzmán, A. *Unified Step Language (USL).* Technical Report, International Software Systems, Austin, Tx. 1991.
78. Guzmán, A. *Process supervision, control and evaluation using an augmented SDDS.* Technical Report, International Software Systems, Austin, Tx. 1991.
79. Guzmán, A. *Compilation of schema objects into flat Ada structures.* Technical Report, International Software Systems, Austin, Tx. 1991.
80. Guzmán, A. *Flexibility and efficiency via operations' particularization.* Technical Report ISSI-91-005, International Software Systems, Austin, Tx. 1991.
81. Guzmán, A. *Graphic queries and updates for 2-step.* Technical Report ISSI-91-003, International Software Systems, Austin, Tx. 1991.
*82. Guzmán A., and Yin, Weiping. Interconnection of software tools or applications, considered as black boxes. *Fourth International Symposium on Artificial Intelligence*, Cancún, Mexico, November 1991.
81. Guzmán, A. *Graphic queries and updates for 2-step.* Technical Report ISSI-91-003, International Software Systems, Austin, Tx. 1991.
*82. Guzmán A., and Yin, Weiping. Interconnection of software tools or applications, considered as black boxes. *Fourth International Symposium on Artificial Intelligence*, Cancún, Mexico, November 1991.

5.11 Flujo de trabajo, de documentos, procesos de negocio, tramitología.

El lenguaje visual descrito anteriormente se aplica al formalismo de representar procesos de negocio, o trámites. Los ejecutores de cada nodo pueden ser computadoras o personas. Esto da pie a procesamiento "mixto."

*82. Guzmán A., and Yin, Weiping. Interconnection of software tools or applications, considered as black boxes. *Fourth International Symposium on Artificial Intelligence*, Cancún, Mexico, November 1991.

86bis. Guzmán, A. *Diseño de una Sistema General de Seguimientos*. Simposium Internacional de Computación , CENAC-IPN, Nov. 10-13, 1993, México, D.F.

Guzmán, A. Tramitología, Simplificación de procesos administrativos y reingeniería de negocios. A presentarse en el *Congreso Internacional de Mujeres sobre* ... 1999. También: enviado a *Academia,* Instituto Politécnico Nacional.

Cecilia Palomino. Construcción de un sistema de flujo de documentos (work flow) con múltiples servidores. Tesis de M. en C., Centro de Investigación en Computación, 1999.

5.12 Árbokes k-d

Se construyó un programa que aprende de una matriz de aprendizaje, y cuya salida es un programa en C. Éste programa en C representa un árbol de decisiones. Este árbol (y el programa) clasifican objetos según lo aprendido de la matriz de aprendizaje. Es decir, a partir de una matriz de aprendizaje, se genera automáticamente un clasificador (un programa en C) que asigna clases a objetos.

*89. Guzmán, A. Árboles k-d como clasificadores supervisados y para la substitución de sistemas expertos. Congreso Internacional sobre Reconocimiento de Patrones. *ICIMAF*, Habana, Cuba. 1995.

109. García, A., Guzmán Arenas, A., Martínez Luna, G., y Núñez Esquer, G. Clasificación supervisada. Inducción de árboles de decisión. Algoritmo K-D. *Simposium Internacional de Computación CIC 98* "La computación: investigación, desarrollo y aplicaciones. Noviembre de 1998. Págs. 602-614. México, D. F. ISBN 970-18-1916-0.

* 100. Ruiz Shulclóper, José; Guzmán, A. y Díaz de León, Juan Luis. *Enfoque lógico combinatorio al Reconocimiento de Patrones: Clasificación Supervisada.* Editorial Politécnica, 1999.

5.13 ANASIN, Analizador-sintetizador inteligente. Bodegas de datos

Se construyeron varios programas, que recolectan automáticamente datos de archivos que yacen en computadoras existentes en centros de trabajo. De manera asíncrona, se extraen datos (haciendo resúmenes) de tales archivos, y se envían a una base de datos "corriente arriba", a oficinas regionales o nacionales, por ejemplo. El ANASIN actúa como una "bomba de datos" que periódicamente extrae información operacional de los centros de trabajo, la convierte (al resumirla) en datos estratégicos, y se "bombean" o impulsan a bases de datos relacionales en centros regionales y corporativo, para la toma de decisiones. La

operación del ANASIN es totalmente automática, y los datos se extraen y envían fuera de horas pico, típicamente en las noches y fines de semana.

*95. Guzmán A. Herramientas de Software para la Organización Distribuida. II Congreso Nacional sobre NElectrónica y Comunicaciones, Colegio de Ingenieros en Comunicaciones y Electrónica. Octubre 28-30, 1996, México, D.F., páginas 1-14.
110. Guzmán, A. Herramientas para la empresa distribuida. *Foro "Computación, de la teoría a la práctica."* Págs. 5-14. 26-28 de mayo de 1999. México, D. F. ISBN 970-18-3012-1
113. Guzmán, A. Minería y bodega de datos. *Simposium Nacional de Computación SICOM 99*, 7-11 de junio de 1999, Villahermosa, Tabasco. Págs. 5-13. ISBN 970-18-3046-6.

5.14 Minería de datos

Sobre el ANASIN se construyó un software para *minería de datos*, que encuentra situaciones interesantes, desviaciones, tendencias y anomalías, en un mar de datos. Los mineros trabajan fuera de horas pico, de manera autónoma, sin intervención humana, y dejan sus hallazgos en un archivo de "situaciones interesantes encontradas", para su posterior revisión por personas.

*92. Guzmán A. Mineros de Datos. En *Soluciones Avanzadas*, México D.F., 1996. También en: *ARCHIPIÉLAGO*. Vol. 2 núm. 9, página 19. Nov.-Dic. 1996.
*97. Guzmán A. Estado del Arte y de la Práctica en Minería de Datos, Análisis y Crítica. *Memorias del II Taller Iberoamericano de Reconocimiento de Patrones*. Marzo 24-28, 1997. La Habana Cuba, páginas 367-376.
107. Adolfo Guzmán, Gilberto Martínez Luna. Minería de datos con búsqueda de patrones de comportamiento. *Boletín de Política Informática,* Año XXII, **2,** 1999, 13-30. INEGI, Aguascalientes, México.
112. García, A., Guzmán Arenas, A., y Martínez Luna, G. Anasin: Minería de datos con búsqueda de patrones de comportamiento. *Foro "Computación, de la teoría a la práctica."* Págs. 15-28. 26-28 de mayo de 1999. México, D. F. . ISBN 970-18-3012-1
113. Guzmán, A. Minería y bodega de datos. *Simposium Nacional de Computación SICOM 99*, 7-11 de junio de 1999, Villahermosa, Tabasco. Págs. 5-13. ISBN 970-18-3046-6.

5.15 CLASITEX. Hallando los temas principales de un artículo en español

CLASITEX analiza documentos escritos en español, y nos dice cuáles son los temas principales de los que habla tal documento. Hace un análisis por *conceptos*, no por palabras clave, de suerte que puede identificar que un artículo habla de fútbol sóccer, aunque no contenga tales palabras, si habla de defensa izquierdo, tiro de esquina, saque de meta, Chivas Rayadas, Jorge Campos, ...

*98. Guzmán A. Hallando los temas principales en un artículo en español. *Soluciones Avanzadas*. Vol. 5, núm. 45, pág. 58. I parte, 15 de Julio de 1997, II parte vol. 5, núm.

49, pág. 66, 15 de septiembre de 1997. También en : *Simposium Internacional de Computación.* Centro de Investigación en Computación. Instituto Politécnico Nacional. Noviembre 12-14, 1997. México, D.F., páginas 36-51.

* 99. Guzmán, A. Finding the main themes in a Spanish document. *Journal Expert Systems with Applications*, Vol. 14, No. 1/2, Jan./Feb. 1998, pages 139-148.

*101. Beatriz Beltrán Martínez, Adolfo Guzmán Arenas, Francisco Martínez Trinidad, José Ruiz Shulcloper. Clasitex++: una herramienta para el análisis de textos. Memorias del *Tercer Taller Iberoamericano de Reconocimiento de Patrones*, TIARP-98, Centro de Investigación en Computación, Instituto Politécnico Nacional, México, D. F. Marzo 1998. Páginas 369-379.

108. Alexander Gelbukh, Grigori Sidorov, Adolfo Guzman. Text categorization using a hierarchical topic dictionary. Submitted to *IJCAI-99* (International Joint Conference on Artificial Intelligence).

5.16 EVA: Espacios virtuales de aprendizaje – Software para educación no presencial y asíncrona

EVA es un software que (mediante un examen de conocimientos) determina el conocimiento inicial de un estudiante sobre un cierto tema, le pregunta su estado final deseado (a dónde quiere llegar en sus estudios), y le diseña su *plan de estudios individual*, que se llama "trayectoria de aprendizaje." A continuación, le envía (por la red) material educativo de acuerdo a tal plan de estudios. Este material (llamado "polilibros") se ensambla exprofesso para cada alumno. Contiene módulos que al final hacen un examen –por computadora– al alumno, y de esta manera EVA sabe en qué punto de su trayectoria de aprendizaje se encuentra cada alumno. Mediante comparación de trayectorias, EVA encuentra *asesores* para el alumno, y *compañeros de estudio*, y los pone en contacto. Si el alumno proporciona su horario de actividades, EVA le sugiere actividades síncronas (una teleconferencia, una plática presencial) congruentes con su tiempo disponible y su etapa de aprendizaje.

EVA mitiga la necesidad de que el alumno *esté presente* o en el mismo lugar que el profesor, y *al mismo tiempo* que él. Trata de que la educación sea no presencial (remota) y asíncrona (el profesor escribe su polilibro en cierto día y el estudiante aprende en otro).

102. Gustavo Núñez, Leonid Sheremetov, Jesús Martínez, Adolfo Guzmán, Álvaro Albornoz. The EVA Teleteaching Project - the concept and the first experience in the development of Virtual Learning Spaces. *15th IFIP World Computer Congress 'The Global Information Society on the way to the next Millennium', Vienna and Budapest, 31 Aug. - Sept. 4, 1988*, organized by the Austrian Computer Society and John von Neumann Computer Society on behalf of the International Federation for Information Processing.

*105. Adolfo Guzmán and Gustavo Núñez. Virtual Learning Spaces in distance education; tools for the EVA Project. Accepted for publication in *Journal Expert Systems with Applications*, Elsevier.

5.17 Interciencia

La Inter-Ciencia puede verse como la comunicación e inter-relación efectiva entre varias ciencias –Por ejemplo, Ciencias Sociales, Ciencias Naturales, Humanidades. La Inter-Ciencia surge porque hay problemas que trascienden una especialidad. Estudia áreas que quedan a horcajadas entre dos o más divisiones actuales (disciplinas) de la Ciencia.

La Inter-Ciencia es un fenómeno artificial. No importa cómo se parta o divida la Ciencia, un ser humano solo podrá abarcar una pequeña parte de ella: lo que pueda aprender en digamos 20 años de estudio, de primaria a doctorado.

¿Cuáles son las limitaciones o ventajas del ser humano frente a la Interciencia? ¿Qué papel jugará en ella? ¿Y las mismas preguntas, pero aplicadas a las computadoras? El artículo brinda algunas respuestas.

103. Adolfo Guzmán. La Computación en la Interciencia. Artículo a ser publicado en un libro en homenaje al Dr. Marco Murray Lasso. 1998. También: *Simposium Internacional de Computación CIC 98* "La computación: investigación, desarrollo y aplicaciones. Noviembre de 1998. 41-56. México, D. F. ISBN 970-18-1916-0. También: Informe Técnico del CIC, No. xx. Instituto Politécnico Nacional. 1998

5.18 Interacción dirigida entre agentes con propósito

Estudio de entes autónomos, llamados *agentes.* Nuestros agentes poseen un propósito o fin, y poseen múltiples hebras de ejecución. La interacción con otros agentes tiene lugar en unos *escenarios* (llamados interacciones, o *scripts* en inglés) como por ejemplo *en un restaurant*, *en una subasta,* etc. Además, los agentes usan representaciones de conocimiento algo distintas: sus ontologías no coinciden, por lo que tienen que *dialogar* para entenderse plenamente. También se modela la reacción de los agentes frente a *eventos inesperados* que los hacen desviarse de sus trayectorias o planes.

We propose the development of a model that enables agents that use different ontologies to interact and interchange information among them.

The behavior of each agent will be defined in a high-level language that we will design and develop, with the following features:

(1) Each agent and each interaction can be described by several sequences of instructions that can be executed concurrently (multi-threaded). As part of our research, we will deliver an executor (perhaps an interpreter) for above language.
(2) The model allows communications between agents that do not share the same data dictionary (ontology), but rather need a conversion or matching among the primitives they use.
(3) Some of the scripts can be partially executed, thus giving rise to the idea of a "degree of satisfaction" of a given behavior or interaction.

As part of our research, we will deliver an executor (an interpreter) for above language.

We will validate the model using test cases based on real situations like electronic commerce, product delivery, and automatic filling of databases that use different ontologies.

111. Olivares, J, Demetrio Aguirre, A., Domínguez Ayala, María, y Guzmán Arenas, A. Computación dirigida entre agentes con propósito. *Foro "Computación, de la teoría a*

la práctica." Págs. 210-219. 26-28 de mayo de 1999. México, D. F. . ISBN 970-18-3012-1

5.19 Relación entre áreas de investigación

	Inteligencia Artificial			Sistemas de información	Computación distribuida y paralela			
	Conocimiento común	Manipulación Simbólica	Visión, reconocimiento de patrones	Software Geo	Bases de datos	Hardware paralelo, proc. distribuido	Educación a distancia	Tecnología de software, herramientas
1. Convert, lenguaje de manipulación simbólica		■	■					■
2. Descomposición de escenas			■					
3. Base de datos geográficos				■	■			■
4. P. R. Percepción Remota			■	■				
5. AHR computadora paralela		■				■		
8. Máquinas jerárquicas para proceso paralelo		■				■		
6. Números de forma			■	■				
7. Modelos digitales del terreno				■				
9. CYC – sentido común	■				Base de conocimientos			
10. Lenguaje visual		■	■			■		■
11. Flujo de trabajo		■				■		
12. Árboles k-d			■					■
13. Anasin, bodega de datos					■	■		
14. Minería de datos			■		■			
15. Clasitex, temas de artículos en español	■							
18. Agentes con propósito	■					■		■
16. EVA, educación a distancia						■	■	

17. Interciencia	

5.20 Temas que utilizan la estructuración del conocimiento

	Utiliza estructuración del conocimiento
1. Convert, lenguaje de manipulación simbólica	
2. Descomposición de escenas	
3. Base de datos geográficos	Divide una región en cuatro (o más) regiones, y así recursivamente. Permite precisión arbitraria, variable.
4. P. R. Percepción Remota	
5. AHR computadora paralela	
8. Máquinas jerárquicas para proceso paralelo	Las máquinas se organizan en grupos (“clusters”) que comparten memoria.
6. Números de forma	Se forma un árbol de formas elementales según el número de elementos (precisión)
7. Modelos digitales del terreno	Trata de ajustar un triángulo a una porcion de la superficie tridimensional. Si el ajuste no tiene la precisión adecuada, divide el triángulo en cuatro sub-triángulos, y así sucesivamente.
9. CYC – sentido común	Los conceptos se organizan en una jerarquía del sentido común.
10. Lenguaje visual	Un programa es una red dirigida, cuyos nodos pueden ser otras redes, o programas en un lenguaje de tercera generacion.
11. Flujo de trabajo	Un trámite es una red dirigida, cuyos nodos pueden ser otros sub-trámites, o tareas elementales.
12. Árboles k-d	Un clasificador es una estructura arborescente de decisiones.
13. Anasin, bodega de datos	La colección de bases de datos forma un árbol isomorfo a la organización de la empresa.
14. Minería de datos	Se utiliza un cubo cuyos ejes son árboles aplanados.
15. Clasitex, temas de artículos en español	Se organizan los conceptos en un árbol, bajo la relación “sugiere el tema de”
18. Agentes con propósito	Usa ontologías parecidas a las de CYC (tema 9).
16. EVA, educación a distancia	
17. Interciencia	Usa ontologías parecidas a las de CYC (tema 9).